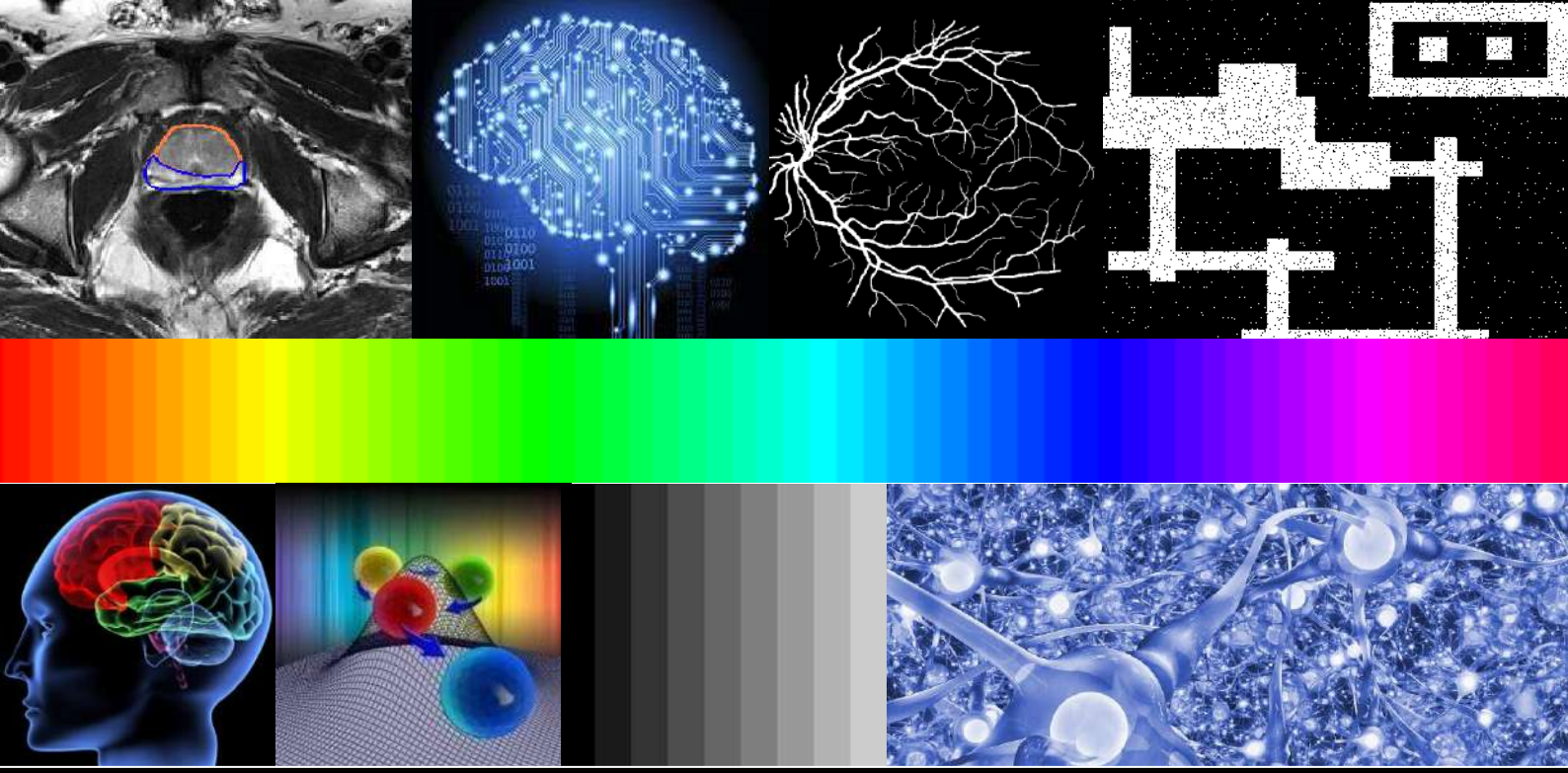


APRENDIZAJE COMPUTACIONAL y MORFOLOGÍA MATEMÁTICA

aplicados al

PROCESAMIENTO de IMÁGENES BIOMÉDICAS

Msc. Marco E. Benalcázar Palacios

Tesis de Doctorado en Ingeniería Electrónica

Facultad de Ingeniería

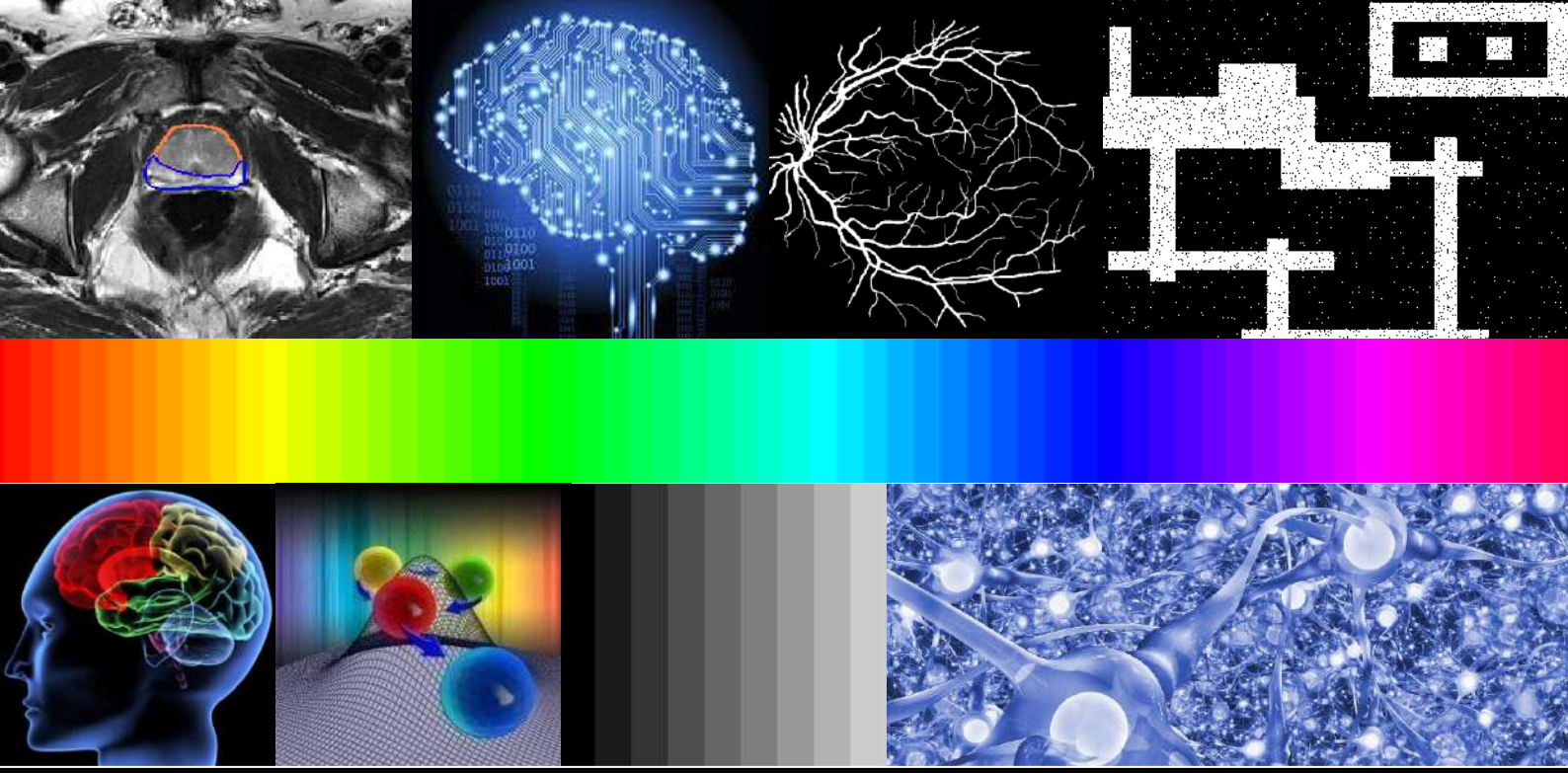
Universidad Nacional de Mar del Plata

Director: Dr. Marcel Brun

Codirectores: Dra. Virginia L. Ballarin

Dr. Robin G. Álvarez Rueda

- 2014 -



APRENDIZAJE COMPUTACIONAL y MORFOLOGÍA MATEMÁTICA

aplicados al

PROCESAMIENTO de IMÁGENES BIOMÉDICAS

Msc. Marco E. Benalcázar Palacios

Tesis de Doctorado en Ingeniería Electrónica

Facultad de Ingeniería

Universidad Nacional de Mar del Plata

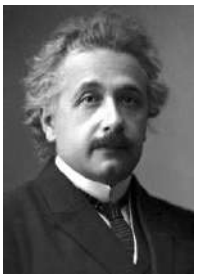
Director: Dr. Marcel Brun

Codirectores: Dra. Virginia L. Ballarin

Dr. Robin G. Álvarez Rueda

- 2014 -

COLECCIÓN DE FRASES CÉLEBRES Y CITAS FAVORITAS



- Albert Einstein -
1879 – 1955
Premio Nobel de Física 1921.

"A medida que se expande el círculo del conocimiento, también lo hace la circunferencia de obscuridad que lo rodea."

- Niels Bohr -
1885 – 1962
Premio Nobel de Física 1922.

"Hacer predicciones es difícil, especialmente si es acerca del futuro."

- Richard Feynman -
1918 – 1988
Premio Nobel de Física 1965.

Respecto de un modelo científico "...es cuestión de si la teoría da predicciones que concuerdan con los experimentos. No se trata de si una teoría es filosóficamente encantadora, fácil de entender, o perfectamente razonable desde el punto de vista del sentido común..."

QED: The Strange Theory of Light and Matter (1985, pp. 10)

- George Box -
1919 – 2013
Reconocido Profesor de Estadística.

"...El hecho de que un polinomio sea una aproximación no necesariamente va en detrimento de su utilidad ya que todos los modelos son aproximaciones. Esencialmente, todos los modelos están equivocados, pero algunos son útiles. Sin embargo, la naturaleza aproximadora de un modelo debe tenerse siempre en cuenta..."

Empirical Model-Building and Response Surfaces (1987, pp. 424)

DEDICATORIA

A mi familia y amigos.

A la gente que sueña y se esfuerza por hacer realidad sus sueños. A aquellos que con su trabajo tesonero, e incluso su vida misma, han contribuido y contribuyen para que este mundo sea un mejor lugar para las presentes y futuras generaciones.

Marco E. Benalcázar

AGRADECIMIENTO

La gratitud es uno de los sentimientos más nobles que llenan de mucha satisfacción al ser humano. Con el mayor respeto y humildad del caso me permito dar las gracias a los países, instituciones, y personas que han formado parte y contribuido de diferentes maneras para mi formación doctoral y realización de esta tesis:

Un agradecimiento muy profundo a la Argentina, y su pueblo, por su calidez, hermandad, y por ser mi hogar durante el periodo doctoral. Al Ecuador, y su pueblo pujante, por luchar incansablemente y anhelar un futuro mejor. Mi gratitud imperecedera para el Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET - Argentina) y la Secretaría Nacional de Educación Superior, Ciencia, Tecnología, e Innovación (SENESCYT - Ecuador) por el financiamiento brindado. A la Universidad Nacional de Mar del Plata (UNMDP), Alma Máter pública y gratuita de la Argentina, por abrirme sus puertas y acogerme cálidamente como su doctorando. Al Grupo de Procesamiento Digital de Imágenes de esta universidad por el honor de permitirme ser uno de sus miembros. A la Universidad de Texas A&M de los Estados Unidos por darme la oportunidad para intercambiar experiencias y realizar investigación por un periodo de seis meses. Mi enorme gratitud va acompañada del compromiso permanente de, a través de lo aprendido y desarrollado, servir y ser útil a la sociedad.

Gracias infinitas a los Doctores Marcel Brun y Virginia Ballarin, director y codirectora de esta tesis, por la oportunidad de trabajar juntos, por su tiempo, guía, y apoyo permanente. Por su intermedio, también expreso mi agradecimiento a mis compañeros y colegas del Grupo de Procesamiento Digital de Imágenes de la UNMDP por todas las experiencias y conocimientos compartidos. Al Dr. Robin Álvarez, codirector ecuatoriano de esta tesis, por su apoyo y guía permanente. A los miembros de la Comisión de Seguimiento de mi trabajo doctoral, Doctores Gustavo Meschino y Juan Ignacio Pastore, por su tiempo y sugerencias brindadas. Vaya también mi agradecimiento a todas las personas de la UNMDP y amigos de Ecuador que me han colaborado en todo momento para mi llegada y estancia en la Argentina.

Mi gratitud imperecedera a los Doctores Oscar Bustos, Juan Pablo Graffigna, y Adriana Scandurra por su valioso tiempo y paciencia dedicada a la revisión de esta tesis.

De igual forma, un agradecimiento especial al Dr. Edward Dougherty por la apertura y guía brindada durante mi visita al Laboratorio de Procesamiento de Señales Genómicas de la Universidad de Texas A&M. También expreso un agradecimiento especial a los demás miembros de este laboratorio por el apoyo y amistad brindada, especialmente a los Doctores Ulisses Braga-Neto e Ivan Ivanov. Muchas gracias a las personas y amigos de Estados Unidos que en todo momento me hicieron sentir como en casa.

Mi eterna admiración, respeto, y agradecimiento a Clara y Manuel, mis queridos papás. También a Freddy, mi hermano y amigo incondicional, a Myriam, mi hermana política, a Freddy Israel, Nataly, y Lisbeth, mis queridos sobrinos. Su apoyo permanente ha sido vital y a pesar de la distancia siempre hemos estado unidos por el amor de familia. A todas mis amigas y amigos mil gracias por estar a mi lado compartiendo su vida. Sepan que uno de los descubrimientos más importantes que he realizado es que la vida y la Madre Naturaleza son maravillosas. Por lo tanto, no sólo hay que tratar de entenderlas, sino también disfrutarlas en todo momento.

¡Muchas gracias a todos! Espero que al leer estas cortas líneas sepan lo agradecido que estoy.

Marco E. Benalcázar

TABLA DE CONTENIDO

Página

Página

Página

Página

ÍNDICE DE FIGURAS

Página

Página

Página

Página

ÍNDICE DE TABLAS

Página

RESUMEN

El Procesamiento Digital de Imágenes (PDI) es una subdisciplina aplicada del procesamiento digital de señales. La morfología matemática es una técnica no lineal de PDI que sirve para el procesamiento y análisis de imágenes. Esta técnica se compone de dos operaciones fundamentales que son la erosión y la dilatación. En base a secuencias de estas operaciones se pueden diseñar algoritmos morfológicos de manera heurística. El principal problema del diseño heurístico es que los resultados están altamente condicionados a la experiencia del diseñador de naturaleza subjetiva. En esta tesis se propone un nuevo paradigma para el diseño automático de operadores morfológicos invariantes ante traslaciones y localmente definidos por medio de una ventana, llamados W-operadores. El paradigma propuesto consiste en definir y representar a un W-operador para clasificación y segmentación mediante un sistema de reconocimiento de patrones.

Esta tesis está compuesta por siete capítulos. En el capítulo I se presentan las definiciones necesarias y se formula a nivel teórico el problema del diseño automático de W-operadores. En el capítulo II se realiza una revisión bibliográfica exhaustiva y un análisis teórico de los métodos propuestos en la literatura científica para el diseño de W-operadores. En el capítulo III se propone y analiza el nuevo paradigma para el diseño de W-operadores basado en el uso de teoría de reconocimiento de patrones. En el capítulo IV se propone y testea un nuevo método para el procesamiento de imágenes binarias basado en redes neuronales tipo feed-forward. Luego, en el capítulo V se extiende el método propuesto para imágenes binarias al caso donde las imágenes a procesar son imágenes en escala de grises. En este capítulo se aplica, evalúa, y realizan comparaciones del método propuesto en la segmentación de imágenes médicas. En el capítulo VI se extienden el método propuesto al caso donde las imágenes a procesar son imágenes color RGB. En este capítulo también se aplica el método propuesto a un problema de segmentación de imágenes médicas. Finalmente, en el capítulo VII se presenta una discusión, conclusiones, y trabajo futuro.

PUBLICACIONES

Durante el periodo doctoral se han realizado las siguientes publicaciones científicas:

1. Revistas

L.A. Dalton, M.E. **Benalcázar**, M. Brun, y E.R. Dougherty, "Bayes Labeling and Bayes Clustering Operators for Random Labeled Point Processes," *IEEE Transaction on Signal Processing*, submitted, 2014.

M.E. **Benalcázar**, M. Brun, V.L. Ballarin, y R.M. Hidalgo, "Automatic Design of Ensembles of Window Operators for Ocular Image Segmentation," *Latin America Transactions, IEEE (Revista IEEE America Latina)*, vol. 12, pp. 935–41, 2014.

M.E. **Benalcázar**, M. Brun, y V.L. Ballarin, "Artificial neural networks applied to statistical design of window operators," *Pattern Recognition Letters*, vol. 34, pp. 970–9, 2013.

2. Conferencias Nacionales e Internacionales con Referato

M.E. **Benalcázar**, M. Brun, y V.L. Ballarin, "Automatic Design of Window Operators for the Segmentation of the Prostate Gland in Magnetic Resonance Images," *Proceedings of the Latin American Conference on Biomedical Engineering CLAIB 2014*, Paraná Argentina, 2014.

M.E. **Benalcázar**, M. Brun, y V.L. Ballarín, "Automatic Design of Aperture Filters Using Neural Networks Applied to Ocular Image Segmentation," *Procedings of the European Siganl Processing Conference EUSIPCO 2014*, Lisboa, 1–5, 2014.

M.E. **Benalcázar**, I.A. Pagnuco, D.S. Comas, P.M. Corva, G.J. Meschino, M. Brun, y V.L. Ballarin, "Classification of Cattle Coat Color Based on Genotype Using Pattern Recognition Methods," *Proceedings of the Latin American Conference on Biomedical Engineering CLAIB 2014*, Paraná Argentina, 2014.

D.S. Comas, M.E. **Benalcázar**, I.A. Pagnuco, P.M. Corva, G.J. Meschino, M. Brun, y V.L. Ballarín, "Classification of Bovine Coat Color based on Genotype," *Proceedings of the V Argentinian Congress of Bioinformatics and Computational Biology (VCAB2C)*, San Carlos de Bariloche Argentina, 21–3, 2014.

M.E. **Benalcázar**, M. Brun, y V.L. Ballarín, "Automatic Segmentation of Exudates in Ocular Images using Ensembles of Aperture Filters and Logistic Regression," *Proceedings of the 19th Argentinean Bioengineering Society Congress (SABI 2013)*, Tucumán Argentina, 2–11, 2013.

M. Brun, M.E. **Benalcázar**, I.A. pagnuco, y V.L.Ballarín, "New Balanced Logistic Regression Algorithm with application to Phenotype Classification," *Proceedings of the 4ta. Conferencia Internacional de la Sociedad Iberoamericana de Bioinformática (SolBio)*, Rosario Argentina, 2013.

L.A. Dalton, M.E. **Benalcázar**, M. Brun, y E.R. Dougherty, "Bayes clustering operators for known random labeled point processes," *Proceedings of the Conference on Signals, Systems, and Computers, 2013 Asilomar*, 893–7, 2013.

M.E. **Benalcázar**, M. Brun, y V.L. Ballarin, "Segmentación de Vasos Sanguíneos en Angiogrfías Retinales usando Ensambles de Filtros Aperture," *Proceedings of the International Symposium on Innovation and Technology ISIT2012*, Perú, 125–9, 2012.

M.E. **Benalcázar**, J.I. Pastore, M. Brun, y V.L. Ballarin, "Filtros Aperture para Clasificación de Imágenes Color," *Torneo Regional de Inteligencia Computacional - TRICV*, Córdoba - Argentina, 88, 2012.

M.E. **Benalcázar**, M. Brun, V.L. Ballarin, I. Passoni, G. Meschino, y L. Pra, "Automatic Design of Binary W-Operators Using Artificial Feed-Forward Neural Networks Based on the Weighted Mean Square Error Cost Function," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. vol. 7441, L. Alvarez, M. Mejail, L. Gomez, and J. Jacobo, Eds.: Springer Berlin Heidelberg, 2012, pp. 495-502.

M.E. **Benalcázar**, J. Padín, A. Bouchet, M. Brun, y V. Ballarin, "Diseño Automático de Operadores Morfológicos Aplicado a la Segmentación de Angiografías Retinales," *Proceedings of the Congreso Argentino de Informática y Salud* (*CAIS 2011*), Córdoba, Argentina, 137–47, 2011.

M.E. **Benalcázar**, J. Padín, M. Brun, J.I. Pastore, V.L. Ballarin, L. Peirone, y G. Pereyra, "Measuring Leaf Area in Soy Plants by HSI Color Model Filtering and Mathematical Morphology," *Proceedings of the 8th Argentinean Bioengineering Society Conference* (*SABI 2011*) *and 7th Clinical Engineering Meeting*, Mar del Plata Argentina, 2011.

CAPÍTULO I

Procesamiento Digital de Imágenes y Morfología Matemática

Resumen: En este capítulo se inicia describiendo, a nivel general, los objetivos del procesamiento digital y análisis de imágenes. Se menciona una breve historia de sus orígenes y el por qué de su continua y rápida evolución en los últimos años. Posteriormente, se describe de manera sucinta su importancia y utilidad en el área médica, y en particular, en el campo de la imagenología médica. Luego se aborda sobre una de las principales técnicas no lineales del procesamiento y análisis de imágenes y que es foco de esta tesis: la morfología matemática. En esta sección comienza con una breve reseña histórica de sus orígenes y continúa con la definición de los diferentes tipos de imágenes que utilizarán a lo largo de esta tesis. Posteriormente, se definen las dos operaciones morfológicas fundamentales como son la erosión y dilatación. En base a estas definiciones se plantea y analiza luego el problema de diseño heurístico de operadores morfológicos. En lo subsiguiente, se propone el diseño automático como una alternativa al diseño heurístico de operadores morfológicos, lo cual constituye el tema central de esta tesis. En este contexto se presenta la definición de operador de ventana, o W-operador. Luego se describen sus propiedades de definición local por medio de una ventana e invariancia ante traslaciones. Las dos secciones finales de este capítulo tratan sobre el diseño automático de operadores morfológicos en base a ejemplos de entrenamiento analizando sus ventajas y desventajas frente al diseño heurístico.

1.1. Introducción

El *procesamiento digital de imágenes* (PDI) es una subdisciplina del procesamiento de señales dedicada a la manipulación y análisis de imágenes y video. El resultado de manipular, procesar, transformar, restaurar, o mejorar una imagen es una nueva imagen. El resultado de analizar una imagen es un conjunto de características o atributos que representan dicha imagen. El procesamiento y el análisis de imágenes se basan en el uso de funciones lineales y no lineales llamadas operadores de imágenes y transformadas, respectivamente. En la Figura 1.1 se ilustran ambos conceptos usando una imagen ocular de la base de datos DRIVE [Staal *et al.*, 2004]. En este caso, el procesamiento consiste en extraer o segmentar los vasos sanguíneos oculares; mientras que el análisis consiste en el conteo del número de bifurcaciones de los vasos sanguíneos y la localización del centro de masa del disco óptico.

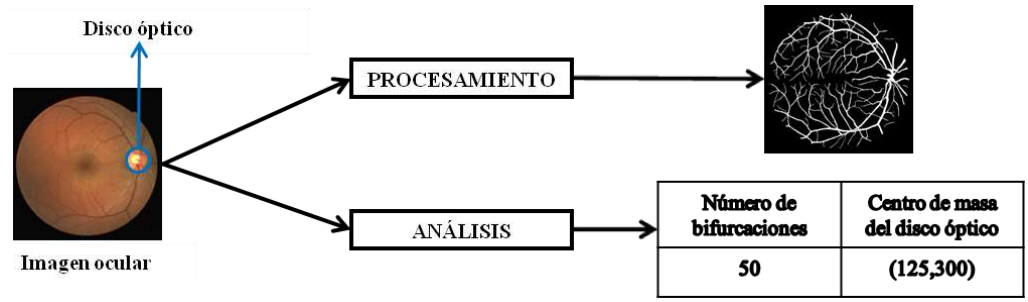


Figura 1.1. Ilustración del procesamiento y análisis de una imagen ocular.

1.2. Historia y Evolución del Procesamiento de Imágenes

Desde su origen alrededor del año 1960, el procesamiento digital de imágenes, o simplemente procesamiento de imágenes, ha experimentado un notable crecimiento. Las primeras aplicaciones de PDI se remontan al campo espacial con el procesamiento de imágenes lunares, y al campo médico con la invención de la tomografía axial computarizada [Gonzalez y Woods, 2008]. Este crecimiento se ha visto influenciado

notablemente por el incremento exponencial de la capacidad de cálculo de los computadores. Adicionalmente, la aplicación de las diferentes técnicas de PDI en otras disciplinas científicas para resolver nuevos problemas es otro factor responsable de su desarrollo. El deseo humano de que los computadores emulen el sistema visual humano, conocido como *visión artificial*, también ha demandado el desarrollo de nuevas y mejores técnicas de PDI.

El continuo desarrollo de nuevas tecnologías de la información y comunicación (TICs) y su aplicación en diferentes dominios científicos y tecnológicos, o su uso regular en la vida cotidiana, genera diariamente grandes volúmenes de imágenes y/o videos. A nivel científico y tecnológico, se requiere que estos datos sean procesados y/o analizados para generar información relevante que contribuya al desarrollo de nuevas teorías científicas y aplicaciones tecnológicas. Para aplicaciones de la vida cotidiana se requiere, por ejemplo, el desarrollo de métodos de compresión y realce de la calidad visual de imágenes y videos, reconocimiento de objetos, entre otros. Estas tareas de análisis y procesamiento exigen un continuo desarrollo de nuevas técnicas de PDI, o la mejora de las existentes, con sólidos fundamentos teóricos, y capaces de procesar estos grandes volúmenes de información eficientemente.

1.3. Procesamiento de Imágenes Médicas

Es de especial interés para esta tesis el desarrollo de nuevas técnicas de PDI para *aplicaciones médicas*. En particular, en el campo médico las técnicas de PDI se usan, principalmente, para la detectar o segmentar zonas o regiones de interés, así como también para realzar la calidad visual de las imágenes provenientes de resonancias magnéticas, tomógrafos, ecógrafos, endoscopios, cámaras de fondo ocular, entre otras técnicas de adquisición de imágenes médicas. La segmentación de imágenes médicas y el filtrado de ruido serán los principales problemas que se abordarán como aplicaciones de este trabajo. El objetivo final de la segmentación de imágenes médicas es obtener información relevante para que el especialista médico pueda diagnosticar, evaluar, y tratar diferentes patologías con mayor exactitud y menor subjetividad. Adicionalmente, las técnicas de PDI aplicadas al campo médico también contribuyen al desarrollo e implementación de nuevos métodos de diagnóstico asistidos por computador, la automatización de los procedimientos manuales o como herramienta ayuda a la investigación médica [Dougherty, 2011].

1.4. Morfología Matemática

La *morfología matemática* es una técnica *no lineal* de procesamiento y análisis de imágenes. Su función es la caracterización geométrica de formas, de ahí su nombre, y el análisis de texturas basado en la teoría de conjuntos.

1.4.1. Historia

La morfología matemática remonta su origen a 1964 cuando Georges Matheron investigaba las relaciones entre la geometría de medios porosos y su permeabilidad. En paralelo a este trabajo Jean Serra, alumno de Matheron, trabajaba en cuantificar la composición y estructura petrográfica de minerales de hierro. Durante este trabajo, Serra desarrolló los conceptos de elemento estructurante y la transformada hit or miss. Posteriormente, el análisis de teórico de esta transformada llevó a Matheron a formular

y analizar los conceptos de erosión y dilatación que, posteriormente, constituirán la base para la formulación de operadores morfológicos más complejos que, en conjunto, constituirán la denominada morfología matemática [Serra, 1982], [Shih, 2009]. Desde su origen hasta la actualidad, el éxito de la morfología matemática se debe a aspectos prácticos y teóricos. Desde la perspectiva práctica, los resultados de muchos operadores morfológicos se pueden explicar en términos de características geométricas y topológicas de las imágenes a procesar. Desde un punto de vista formal, la morfología matemática se basa en un marco teórico sólido para el estudio de las propiedades algebraicas de los operadores.

1.4.2. Modelos Matemáticos de Imágenes

En esta sección se definen los modelos de imágenes que se utilizarán en esta tesis. Sean $\boldsymbol{E}$ un subconjunto finito y no vacío de $\mathbb{Z}^2$ y $L = \{l_{min},\ldots,l_{max}\}$ un conjunto con $l = |L|$ niveles de gris. Una *imagen digital en escala de grises* es una función O: $\boldsymbol{E} \rightarrow L$ con dominio en $\boldsymbol{E}$ y rango en L. Un punto arbitrario $t = (x,y)$ del dominio de O se denomina *pixel*. El tamaño de la imagen O se define como el número de píxeles que forman su dominio. A nivel computacional, una imagen digital se representa por una matriz **O** compuesta por M filas y N columnas, donde el valor de **O** en la coordenada (i,j) representa la *intensidad*, o *nivel de gris*, de la imagen O en el píxel t indexado con (i,j), con $i = 1,\ldots,M$ y $j = 1,\ldots,N$ (Figura 1.2). El conjunto de todas las imágenes con dominio $\boldsymbol{E}$ y rango L se denota mediante $L^{\boldsymbol{E}}$. Sea una *imagen binaria,* o *imagen blanco y negro*, una función I: $\boldsymbol{E} \rightarrow \{0,1\}$ que es un caso particular de la definición presentada para las imágenes en escala de grises cuando el conjunto de niveles de gris es $L = \{0,1\}$. El conjunto de todas las imágenes binarias con dominio en $\boldsymbol{E}$ y rango en el conjunto $\{0,1\}$ se denota mediante $\{0,1\}^{\boldsymbol{E}}$.

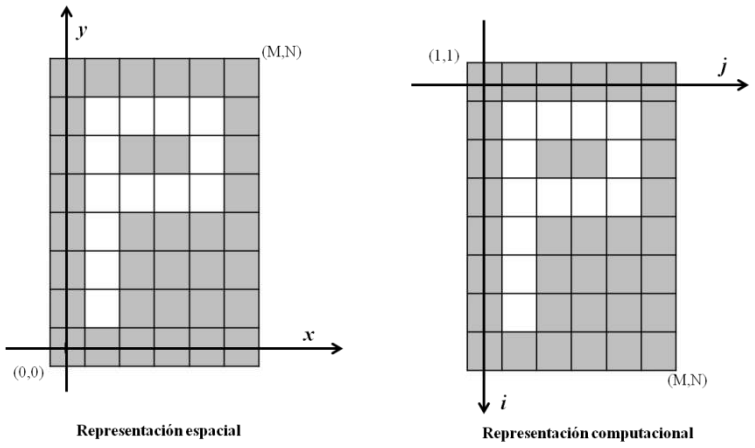


Figura 1.2. Representación espacial y computacional de una imagen binaria.

Una *imagen multicanal*, o *multiespectral*, es una función $\underline{O}$: $\boldsymbol{E} \rightarrow (L_1 \times L_2 \times \ldots \times L_m)$ tal que $\underline{O} = (O_1,O_2,...,O_m)$, donde las imágenes $O_i \in L_i^{\boldsymbol{E}}$ son los canales de $\underline{O}$ con $i = 1,\ldots,m$. En muchos casos se tiene que $L_1 = L_2 = \ldots = L_m = L$. Para este tipo de imágenes, un píxel t de $\underline{O}$ es una coordenada en $\boldsymbol{E}$ que toma el vector $(O_1(t),O_2(t),...,O_m(t))$. El conjunto de todas las imágenes multicanal con dominio en $\boldsymbol{E}$ y rango en $(L_1 \times L_2 \times \ldots \times L_m)$ se denota mediante $(L_1 \times L_2 \times \ldots \times L_m)^{\boldsymbol{E}}$. Las *imágenes color* son un caso particular de las imágenes multicanal con $m = 3$ canales. Para esta clase de imágenes, cada canal representa la información de algún atributo del color. Por ejemplo, para el caso de *imágenes color*

RGB, las imágenes O_1, O_2, y O_3 representan la información de los colores primarios rojo, verde, y azul, respectivamente. Del mismo modo, para el caso de *imágenes color HSI*, las imágenes O_1, O_2, y O_3 representan la información de tono H (color puro), saturación S (grado en que un color está diluido con el blanco), e intensidad I (nivel de brillo), respectivamente. Nótese también que las imágenes en escala de grises y las imágenes binarias son casos particulares de las imágenes multicanal con $m = 1$.

Una *imagen volumétrica*, o simplemente *volumen*, o *imagen tridimensional* (3D) es una función $\ddot{\mathbf{V}} : \boldsymbol{D} \rightarrow L$, donde el domino $\boldsymbol{D}$ es un subconjunto no vacío y finito de $\mathbb{Z}^3$. Un vóxel $r = (x,y,z)$ es una coordenada en $\boldsymbol{D}$ del volumen $\ddot{\mathbf{V}}$ que toma un valor del conjunto de niveles de gris L. Un *corte frontal del volumen* $\ddot{\mathbf{V}}$ es una imagen en escala de grises $O = \ddot{\mathbf{V}}(x, y, z = z_0)$ que se obtiene al fijar el valor de la coordenada z en z_0 mientras que las dos coordenadas restante, x e y, son variables. De la misma manera se pueden obtener cortes sagitales y transversales fijando los valores de x e y, respectivamente (Figura 1.3). *Cortes oblicuos* se obtienen fijando los valores de dos componentes y variando el valor de la componente restante. El conjunto de todas las imágenes volumétricas con dominio $\boldsymbol{D}$ y rango L se denota mediante $L^{\boldsymbol{D}}$.

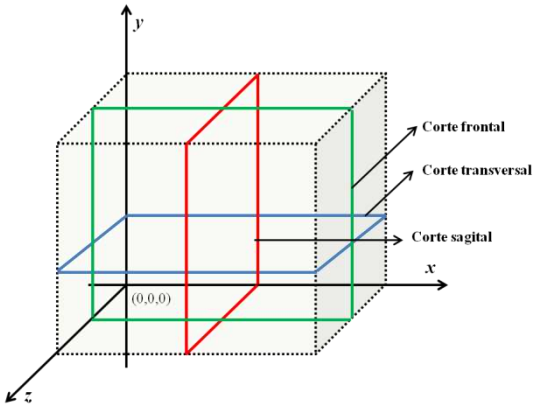


Figura 1.3. Cortes de una imagen volumétrica.

Un *operador de imágenes binarias* es una función $\boldsymbol{\Psi}$: $\{0,1\}^{\boldsymbol{E}} \rightarrow \{0,1\}^{\boldsymbol{E}}$ que transforma una imagen binaria I en otra imagen binaria $\boldsymbol{\Psi}(I)$. Un *operador de imágenes en escala de grises* es una función $\boldsymbol{\Psi}$: $L_1^{\boldsymbol{E}} \rightarrow L_2^{\boldsymbol{E}}$ que transforma una imagen en escala de grises en otra imagen en escala de grises, donde no necesariamente se cumple que $L_1 = L_2$. Un ejemplo de operador de imágenes en escala de grises donde se cumple que $L_1 = L_2$ es cuando $\boldsymbol{\Psi}$ es un operador para realce de contraste. Un ejemplo donde $L_1 \neq L_2$ es cuando $\boldsymbol{\Psi}$ es un operador para segmentación de imágenes, donde la entrada puede ser una imagen en escala de grises O y la salida $\boldsymbol{\Psi}(O)$ puede ser una imagen binaria. Un *operador de imágenes multicanal* es una función $\boldsymbol{\Psi}$: $(L_1 \times \ldots \times L_m)^{\boldsymbol{E}} \rightarrow (L'_1 \times \ldots \times L'_{m'})^{\boldsymbol{E}}$ cuya entrada es una imagen multicanal y cuya salida puede ser también una imagen multicanal, una imagen en escala de grises, o una imagen binaria. Finalmente, un *operador de imágenes volumétricas* es una función $\boldsymbol{\Psi}$: $L^{\boldsymbol{D}} \rightarrow L^{\boldsymbol{D}}$ cuya entrada y salida son volúmenes. Un operador de imágenes volumétricas puede estar compuesto por uno o más operadores en escala de grises aplicados a cada uno de sus cortes.

1.4.3. Operaciones Morfológicas Fundamentales

A continuación se definen las operaciones morfológicas fundamentales de erosión y dilatación para imágenes binarias, en escala de grises, y color. Adicionalmente, también se presenta la definición del operador hit or miss para imágenes binarias.

Para la definición clásica de las operaciones morfológicas binarias, la imagen a procesar se modela como un subconjunto de $\mathbb{Z}^2$. El conjunto finito $B = \{b_1,\ldots,(0,0),\ldots,b_n\} \subset \mathbb{Z}^2$ se denomina *elemento estructurante*. La *reflexión* del elemento estructurante *B*, denotada mediante B^r, se define como la rotación de *B* en un ángulo de 180º alrededor de su origen (0,0): $B^r = \{-b_i \mid \forall b_i \in B\}$. La *traslación espacial* de *B* por *t*, denotada mediante B_t, se define como $B_t = \{b_i + t \mid \forall b_i \in B\}$.

***Erosión Binaria**:* La erosión de la imagen binaria *I* modelada como un conjunto por el elemento estructurante *B*, denotada mediante $I \ominus B$, se define como

$$I \ominus B = \{t \mid B_t \subseteq I\}. \tag{1.1}$$

***Dilatación Binaria**:* La dilatación de la imagen binaria *I* modelada como un conjunto por el elemento estructurante *B*, denotada mediante $I \oplus B$, se define como

$$I \oplus B = \{t \mid (B^r)_t \cap I \neq \varnothing\}. \tag{1.2}$$

Ejemplo 1.1. En la Figura 1.4 se muestran el resultado de erosionar y dilatar una imagen binaria. En este caso la erosión *contrae* el lado derecho de las líneas horizontales en un píxel. Por otro lado, la dilatación *estira* el lado derecho de las líneas horizontales en un píxel.

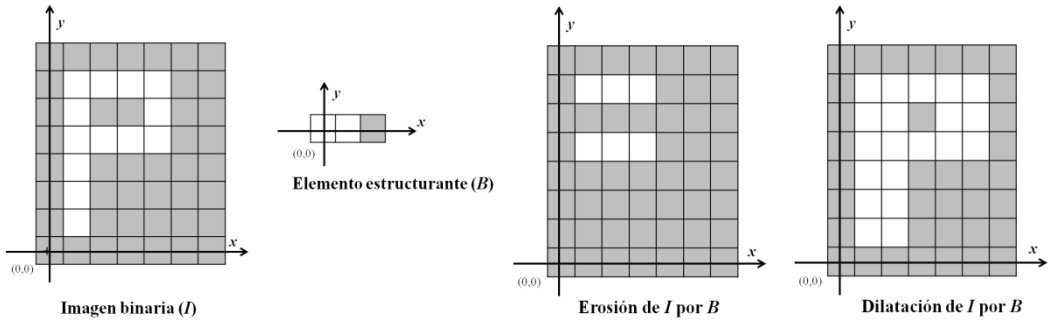


Figura 1.4. Erosión y dilatación de una imagen binaria. El blanco denota 1 y el negro denota 0.

***Operador Hit or Miss**:* Sea $B = \{B_1,B_2 \mid B_1 \cap B_2 = \varnothing\}$ un *elemento estructurante compuesto* formado por los elementos estructurantes B_1 y B_2, la operación de hit or miss de la imagen binaria *I* por *B* se define como

$$I \otimes B = \underbrace{(I \ominus B_1)}_{\text{hit (acierto)}} \cap \underbrace{(\sim I \ominus B_2)}_{\text{miss (falla)}}, \tag{1.3}$$

donde $\sim I$ denota el complemento del conjunto *I*: $\sim I = \{t \mid t \notin I\}$. El operador hit or miss se utiliza en la detección de formas, o patrones, específicos.

Para las siguientes definiciones se utiliza la imagen en escala de grises $O \in L^{\boldsymbol{E}}$ y el elemento estructurante G: $\boldsymbol{E}' \rightarrow L_1$, con $\boldsymbol{E}'$ un subconjunto pequeño y no vacío de $\mathbb{Z}^2$.

Erosión en Escala de Grises: La erosión de la imagen en escala de grises O por el elemento estructurante G se define como

$$(O \ominus G)(t) = inf_{(t' \in \boldsymbol{E}')}\{O(t + t') - G(t')\}. \quad (1.4)$$

En palabras, para un punto arbitrario $t \in \boldsymbol{E}$, el resultado de la erosión de O por G se obtiene trasladando espacialmente G por t: $G_t(t') = G(t' + t)$ $\forall t' \in \boldsymbol{E}'$, y luego calculando el ínfimo del conjunto de valores que se obtienen al substraer, píxel a píxel, G_t de O. El ínfimo de un conjunto ordenado corresponde al máximo de las cotas inferiores de dicho conjunto, para lo cual se usa la relación de orden usual $<$.

Dilatación en Escala de Grises: La dilatación de la imagen en escala de grises O por el elemento estructurante G se define como

$$(O \oplus G)(t) = sup_{(t' \in \boldsymbol{E}')}\{O(t - t') + G(t')\}. \quad (1.5)$$

En palabras, para un punto arbitrario $t \in \boldsymbol{E}$, el resultado de la dilatación de O por G se obtiene calculando la reflexión de G: $G^r(t') = G(-t')$ $\forall t' \in \boldsymbol{E}'$ y luego obteniendo el supremo del conjunto de valores de la suma, píxel a píxel, entre O y G^r trasladado espacialmente por t. El supremo de un conjunto ordenado corresponde al mínimo de las cotas superiores de dicho conjunto para lo cual nuevamente se emplea la relación de orden $<$.

Es posible que el ínfimo de la ecuación 1.4 o el supremo de la ecuación 1.5 retornen valores que están fuera el rango de la imagen a procesar. En este caso, una solución es truncar dichos valores al mínimo o máximo, respectivamente, del rango de la imagen a procesar O. Para evitar este problema, en PDI es común el uso de erosiones y dilataciones de imágenes en escala de grises mediante el empleo de un *elemento estructurante plano* $B = \{b_1,\ldots,(0,0),\ldots,b_n\} \subset \mathbb{Z}^2$. Por lo tanto, la erosión en escala de grises de la imagen O por el elemento estructurante plano B se calcula mediante

$$(O \ominus B)(t) = inf_{(t' \in B)}\{O(t + t')\}, \quad (1.6)$$

donde el resultado en el punto $t \in \boldsymbol{E}$ de la erosión de O, usando un elemento estructurante plano B, consiste en calcular el ínfimo del conjunto de píxeles de O en la vecindad definida por B trasladado espacialmente por t.

De manera similar, la dilatación en escala de grises de la imagen O por el elemento estructurante plano B se calcula mediante

$$(O \oplus B)(t) = sup_{(t' \in B)}\{O(t - t')\}. \quad (1.7)$$

Es decir, el resultado en el punto $t \in \boldsymbol{E}$ de la dilatación de O, usando un elemento estructurante plano B, consiste en calcular el supremo del conjunto de píxeles de O en la vecindad definida por la reflexión de B trasladada espacialmente por t.

Ejemplo 1.2. En la Figura 1.5 se muestra el resultado de erosionar y dilatar la *imagen Lena* usando un elemento estructurante plano. En este caso el elemento estructurante es un cuadrado de 3×3 píxeles. Nótese como la erosión agranda las partes obscuras de la imagen original y elimina las partes brillantes cuyos tamaños son menores que el tamaño del elemento estructurante. Adicionalmente, la imagen resultante de la erosión es un tanto más obscura que la imagen original. Mientras tanto, la dilatación produce un efecto contrario al de la erosión; es decir, la imagen resultante es un tanto más brillante que la imagen original.

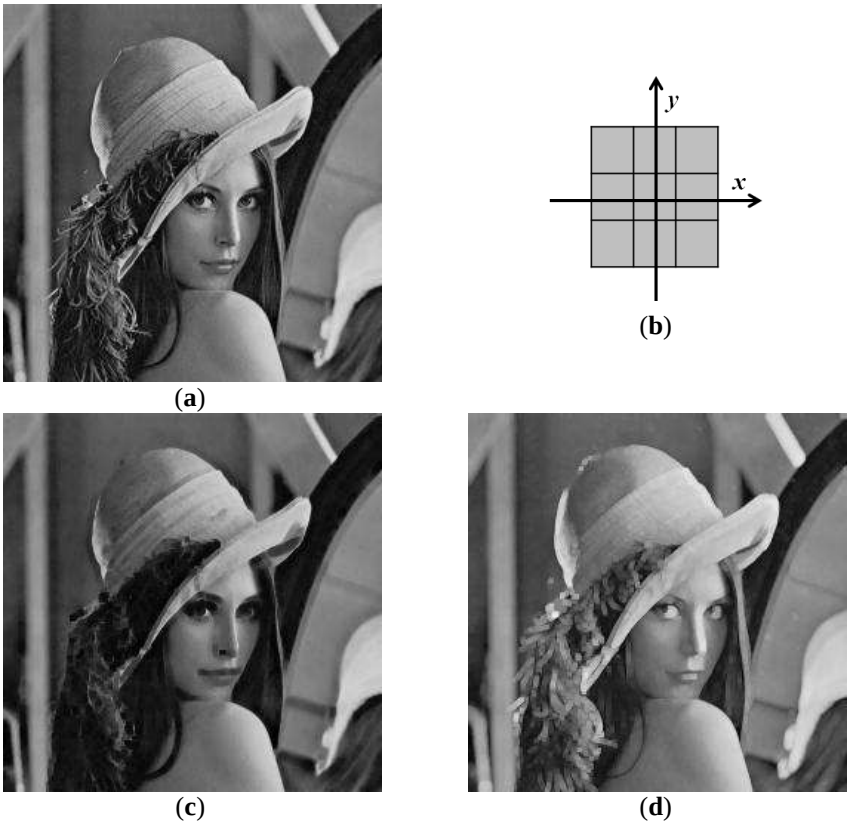


Figura 1.5. Erosión y dilatación de una imagen en escala de grises usando un elemento estructurante plano: (**a**) imagen original, (**b**) elemento estructurante plano, (**c**) resultado de la erosión y (**d**) resultado de la dilatación.

Las definiciones de erosión y dilatación morfológicas presentadas en las ecuaciones 1.4 y 1.6, y 1.5 y 1.7, respectivamente, pueden ser también aplicadas al caso de imágenes multicanal, y en particular a imágenes color. Para esto se requiere la definición de una relación de orden entre vectores. Múltiples nociones de orden para conjuntos formados por vectores han sido propuestas en la literatura científica [Astola y Haavisto, 1990], [Serra, 1992], [Trahanias y Venetsanopoulos, 1992], [Angulo, 2007], [Aptoula y Lefèvre, 2007], [Guo y Guo, 2007]. Sin embargo, hasta el momento no existe un criterio unánimemente aceptado. Otra alternativa consiste en procesar cada canal de una imagen multicanal de manera independiente llamado procesamiento marginal. Es decir, cada canal de una imagen multicanal se trata como si fuera una imagen en escala de grises. El principal problema de este enfoque es que, usualmente, los canales de una imagen multicanal están altamente correlacionados. Por lo tanto, aplicar erosiones o dilataciones a cada canal, de manera independiente, puede ocasionar que el resultado global contenga artefactos artificiales introducidos durante el procesamiento. Para el caso de imágenes color, este enfoque puede generar los denominados *falsos colores*.

Ejemplo 1.3. En la Figura 1.6 se muestra el resultado de erosionar y dilatar cada canal de una imagen color RGB de manera independiente y usando un elemento estructurante plano de 15×15 píxeles. En los resultados se puede notar el encogimiento y ensanche de objetos que producen la erosión y dilatación, respectivamente. Adicionalmente, el hecho de procesar cada canal independientemente hace que en las imágenes resultantes aparezcan colores que no estaban contenidos en la imagen original (falsos colores).

(**a**) (**b**) (**c**)

Figura 1.6. Erosión y dilatación marginal de una imagen color RGB: (**a**) imagen original, (**b**) imagen erosionada, y (**c**) imagen dilatada por un elemento estructurante cuadrado de 15×15 píxeles.

Los volúmenes se pueden procesar usando las definiciones de erosión y dilatación para imágenes en escala de grises. Aquí la única salvedad es el hecho de que los elementos estructurantes y las imágenes a procesar tienen dominio en $\mathbb{Z}^3$ [Lin et al., 2003], [Luengo *et al.*, 2012]. Otra alternativa consiste en procesar cada corte de manera independiente [Gonzalez, 2008], [Dougherty, 2009].

Varios algoritmos han sido desarrollados para implementar las operaciones de erosión y dilatación, especialmente, para imágenes binarias y en escala de grises [Haralick y Shapiro, 1992], [Boomgard y Balen, 1992], [Vincent, 1993], [Gonzalez y Woods, 2008], [Shih, 2009]. Para el caso de imágenes en escala de grises se puede, por ejemplo, aplicar una descomposición binaria, usando operaciones de umbralamiento, a la imagen a procesar así como también al elemento estructurante en el caso de que éste no sea plano. Luego, erosiones y/o dilataciones binarias son aplicadas en paralelo a todas las imágenes resultantes de esta descomposición. El resultado en escala de grises se obtiene al combinar las imágenes binarias procesadas en paralelo. Este resultado es similar al resultado de aplicar los operadores de erosión y dilatación definidos en las ecuaciones 1.4 y 1.6, y 1.5 y 1.7, respectivamente, para imágenes en escala de grises. La ventaja radica en la disminución del costo computacional en lo referente al tiempo del procesamiento. Esto debido a que la erosión y dilatación binaria se ejecutan en paralelo usando las operaciones lógicas de *AND* y *OR* [Shih y Mitchell, 1989].

1.4.4. Diseño Heurístico de Operadores Morfológicos

El diseño heurístico de operadores morfológicos consiste en la implementación de algoritmos y/o filtros morfológicos usando la experiencia del diseñador y el método de prueba y error para resolver un problema de análisis o procesamiento de imágenes. Los *algoritmos morfológicos* son usados para extracción de características, descripción de formas y reconocimiento de patrones [Gonzalez y Woods, 2008], [Shih, 2009]. Estas tareas de análisis de imágenes se basan fundamentalmente en la información de

tamaños, formas, y orientaciones de los objetos de interés. Los *filtros morfológicos* se emplean para realce, compresión, restauración, y segmentación. Estas tareas de procesamiento se basan tanto en la remoción de artefactos no deseados y/o realce del nivel de brillo o contraste de los objetos de interés [Shih, 2009]. Un *objeto* se define como un conjunto de píxeles, o vóxeles, espacialmente conectados y con atributos "similares" de intensidad, color, textura, entre otros. En el caso de imágenes binarias, un objeto es un conjunto de píxeles, con valor 1, que están espacialmente conectados. Los valores de conectividad pueden ser 4 y 8 para imágenes binarias, en escala grises, e imágenes multicanal. Para imágenes volumétricas la conectividad puede ser de 6, 18, o 26 píxeles [Gonzalez y Woods, 2008], [Gomes y Velho, 1997].

Planteado de otra forma, el diseño heurístico de operadores morfológicos consiste en responder a la pregunta ¿*Cuál es la mejor secuencia de operaciones morfológicas y elementos estructurantes para resolver un problema determinado de análisis o procesamiento de imágenes*? Como es de suponer, la complejidad de la respuesta está directamente relacionada con la complejidad del problema en cuestión. Por ejemplo, a nivel general la extracción de los bordes internos de una imagen binaria sin ruido es un problema más simple que segmentar un tumor en una imagen de rayos X (imagen en escala de grises) conteniendo ruido. El diseñador es el encargado de dar respuesta a esta pregunta en base a su conocimiento técnico y la información *a priori* que disponga sobre el problema en cuestión [Gonzalez y Woods, 2008], [Shih, 2009].

Desde el inicio de la morfología matemática hasta la actualidad se han desarrollado una serie de operadores morfológicos para tareas generales y problemas o aplicaciones específicas de análisis y procesamiento de imágenes [Serra, 1982], [Shih, 2009]. Muchos de estos operadores se basan en una combinación de los operadores morfológicos fundamentales de erosión y dilatación. Por ejemplo, el *gradiente morfológico por dilatación* consiste en la diferencia puntual, o píxel a píxel, entre la imagen original después y antes de ser dilatada. El *gradiente morfológico por erosión* consiste en la diferencia puntual entre la imagen original antes y después de ser erosionada. Usualmente, estos operadores morfológicos son usados para la extracción de bordes [Gonzalez y Woods, 2008].

Uno de los aspectos que ha hecho posible que los operadores morfológicos ganen popularidad es su capacidad para resolver problemas donde las técnicas lineales de procesamiento de imágenes fallan [Gomes y Velho, 1997], [E.R. Dougherty, 1999], [Gonzalez y Woods, 2008], [Dougherty, 2009], [Shih, 2009]. Por ejemplo, para el caso de restauración de imágenes, el filtrado clásico usando filtros pasabajos no produce buenos resultados si los artefactos a remover son provenientes de *ruido tipo sal y pimienta*. Para este caso, el uso de operadores no lineales como el filtro mediana o una combinación de erosión y dilatación de la imagen a restaurar usando el mismo elemento estructurante producen buenos resultados [Gonzalez y Woods, 2008]. Esta secuencia de erosión y dilatación usando el mismo elemento estructurante se denominada *apertura*. Secuencias de erosiones y dilataciones con elementos estructurantes de tamaño creciente son también una poderosa herramienta de análisis de imágenes [Shih, 2009]. Una aplicación de esta secuencia de operaciones consiste en el análisis de la distribución de tamaños de objetos denominada *granulometría* [Serra, 1982].

Segmentación de imágenes: La segmentación de una imagen *O* consiste en obtener una partición $\mathcal{P}_O = \{S_1,\ldots,S_q\}$ de su conjunto de píxeles/vóxeles tal que los subconjuntos

resultantes $S_i \in \mathcal{P}_O$ sean disjuntos y se cumpla que $O = \mathrm{U}_{i=1}^{q} S_i$. Los píxeles de cada subconjunto S_i, con $i = 1,\ldots,q$, tienen características de intensidad, color, textura, y otros atributos "similares" entre sí. Como resultado de la segmentación, se obtiene una nueva imagen I conteniendo q tipos diferentes de objetos o regiones de interés, incluyendo el fondo. Para el caso particular de la *segmentación binaria*, $q = 2$, donde se extrae un sólo tipo de objeto de interés, la imagen resultante I es una imagen binaria $I \in \{0,1\}^{\boldsymbol{E}}$, donde 1 denota los píxeles pertenecientes a las regiones u objetos de interés y 0 representa el fondo.

Para la segmentación de imágenes en escala de grises usando operadores morfológicos se propuso el uso de la *transformada watershed* [Meyer y Beucher, 1990], [Vincent y Soille, 1991]. La definición de esta transformada fue extendida posteriormente para ser usada con imágenes color [Koschan y Abidi, 2008]. Otra alternativa para segmentar imágenes en escala de grises consiste en aplicar primero un preprocesamiento a la imagen en cuestión. Este preprocesamiento está basado, usualmente, en la aplicación de erosiones y dilataciones para el filtrado de ruido y otros artefactos no deseados. Posteriormente, se aplica un umbralamiento a la imagen preprocesada, obteniendo de esta manera una imagen binaria I. Como etapa final, o pos-procesamiento, se puede aplicar un nuevo filtrado a la imagen segmentada I usando, por ejemplo, una apertura morfológica binaria para eliminar ruido o artefactos espurios que pudieron ser creados en las etapas previas [Gonzalez y Woods, 2008], [Dougherty, 2011], [Shih, 2009]. Para la segmentación de imágenes multicanal, y en particular de las imágenes color, un enfoque consiste en la aplicación de una transformación de color a nivel de grises. En este caso la segmentación se puede realizar usando el procedimiento antes descrito aplicado a la imagen en escala de grises resultante de la transformación.

1.4.5. Anotaciones y Críticas sobre el Diseño Heurístico

En la práctica, el uso de las técnicas lineales y no lineales de PDI, como el caso de morfología matemática, no son mutuamente excluyentes. Por el contrario, es común que, dada una aplicación, se combinen operadores de ambos tipos para formar un *algoritmo* que permita alcanzar el objetivo deseado [Gomes y Velho, 1997], [E.R. Dougherty, 1999], [Gonzalez y Woods, 2008], [Dougherty, 2009]. A pesar del continuo desarrollo de nuevos operadores de imágenes evidenciado por los ejemplos antes citados, el responder a la pregunta planteada sobre el diseño heurístico de operadores morfológicos no es una tarea trivial. El resultado en este caso está altamente condicionado en la experiencia del diseñador.

En la *etapa de diseño* heurístico de operadores morfológicos, el procedimiento a seguir para un problema determinado usualmente consiste en probar varias secuencias de operaciones morfológicas sobre un *conjunto de imágenes de diseño*. Este conjunto contiene imágenes que son una muestra del problema a resolver. Las secuencias de operaciones a probar se evalúan en función de alguna medida de error que refleja el objetivo del diseño. Luego, una vez que el diseñador ha encontrado una secuencia apropiada de operaciones, entre todas las secuencias testeadas, en la *etapa de testeo* se procede a estimar la capacidad predictiva de dicha secuencia usando otro conjunto de imágenes, denominado *conjunto de imágenes de testeo*.

El conjunto de imágenes de testeo está compuesto por imágenes muestra del problema en cuestión y que no fueron usadas en ninguno de los pasos de la etapa de diseño. Es común asumir que tanto las imágenes de entrenamiento como las imágenes de testeo son generadas por un mismo proceso estocástico. Como resultado de la etapa de testeo se obtiene una estimación del desempeño del algoritmo diseñado. *Para que la estimación del desempeño de un algoritmo no sea sobreoptimista, o sesgada, es crucial que el conjunto de testeo no sea usado en ninguna parte de la etapa de diseño*. Adicionalmente, nótese que los operadores que conforman las secuencias candidatas para el diseño están restringidos, usualmente, a los operadores que el diseñador conoce o es más afín. Dicha familia de operadores no necesariamente puede resultar en la mejor secuencia de operaciones, o algoritmo óptimo, para resolver un problema determinado. Adicionalmente, el tiempo que toma probar cada combinación de operadores puede ser muy largo, por lo cual el diseñador puede llegar probar sólo un número reducido de secuencias.

1.5. Operadores de Ventana o W-operadores

En esta parte se definen los conceptos de operador de ventana o W-operador, ventana espacial, configuración de ventana, y función característica. Luego se detallan los aspectos teóricos que subyacen el diseño automático de operadores morfológicos. Finalmente se presenta, a nivel general, el enfoque propuesto en esta tesis para el diseño de esta familia de operadores de imágenes en base a muestras, ejemplos, o datos de entrenamiento.

1.5.1. Definiciones

Sin pérdida de generalidad, para las siguientes definiciones consideremos una imagen en escala de grises $O \in L_1^{\boldsymbol{E}}$ y un operador de imágenes en escala de grises $\boldsymbol{\Psi}: L_1^{\boldsymbol{E}} \rightarrow L_2^{\boldsymbol{E}}$. Un operador $\boldsymbol{\Psi}$ es *invariante ante traslaciones espaciales* si $\boldsymbol{\Psi}(O_t) = \boldsymbol{\Psi}(O)_t$ para cualquier $t \in \boldsymbol{E}$, donde O_t denota la traslación de O por t: $O_t(t') = O(t' + t)\ \forall t' \in \boldsymbol{E}$. Sea una *ventana espacial* $W = \{w_1,\ldots,(0,0),\ldots,w_n\}$ un subconjunto usualmente pequeño relativo al tamaño de $\boldsymbol{E}$ y no vacío de $\mathbb{Z}^2$ centrado en (0,0). Si se asocian los puntos de W a un vector $(w_1,\ldots,(0,0),\ldots,w_n)$, entonces un operador $\boldsymbol{\Psi}$ es *localmente definido por* W si $\boldsymbol{\Psi}(O)(t) = \boldsymbol{\Psi}(O_t(W))$ para cualquier $t \in \boldsymbol{E}$. Esto implica que el valor del operador $\boldsymbol{\Psi}$, en un punto arbitrario $t \in \boldsymbol{E}$, depende únicamente de los valores de los píxeles de O_t en la vecindad en torno al píxel (0,0) definida por W [Maragos, 1989], [Heijmans, 1990, 1994], [Barrera y Banon, 1992], [Barrera y Brun, 1998], [Barrera *et al.*, 2005], [Hirata *et al*., 2006].

Si el operador de imágenes $\boldsymbol{\Psi}$ está definido mediante una ventana W y es invariante ante traslaciones espaciales, entonces $\boldsymbol{\Psi}$ es un *operador de ventana*, o de aquí en adelante llamado *W-operador* [Barrera y Brun, 1998], [Barrera *et al.*, 2005], [R. Hirata *et al*., 2006]. Para cuando la ventana W está compuesta por un sólo píxel, $W = \{(0,0)\}$, el operador $\boldsymbol{\Psi}$ se denomina *operador puntual* de imágenes, o píxel a píxel. Para el caso contario, cuando $W = \boldsymbol{E}$, el operador $\boldsymbol{\Psi}$ es llamado un *operador global* de imágenes. El caso que concierne a esta tesis es aquel donde $\boldsymbol{\Psi}$ es un *operador local*, donde W es un

conjunto pequeño relativo al tamaño de $\boldsymbol{E}$: $|W| << |\boldsymbol{E}|$. La expresión $|a|$ denota el número de elementos, tamaño, o cardinal del conjunto a. Si $W = \{w_1,\ldots,(0,0),\ldots,w_n\}$ con $n = |W|$, y si se asocian los puntos de W a un vector $(w_1,\ldots,(0,0),\ldots,w_n)$, entonces una *configuración de ventana*, u *observación* es una función de W en L_1 que, para un punto arbitrario t, se obtiene al trasladar la imagen O por t y luego observar los valores de O dentro de la vecindad definida por W centrada en (0,0):

$$O_t(W) = (O(w_1 + t),\ldots,O(w_n + t)). \tag{1.8}$$

Si se define $\mathbf{o}_t = (O(w_1 + t),\ldots,O(w_n + t))$, entonces $\mathbf{o}_t$ es un vector n-dimensional que contiene los valores de la observación para el píxel t de la imagen O a través de W.

Ejemplo 1.4. En la Figura 1.7 se ilustra la obtención de una configuración de ventana para una imagen en escala de grises. Para esto se utiliza una ventana cuadrada W compuesta por $n = 3\times3$ píxeles, cuyo centro es $w_5 = (0,0)$. Para esto, primero se traslada la imagen $O(t_0)$ por $t = (2,3)$ obteniendo $O(t_0 + t)$. Luego se copian en el vector $\mathbf{o}_t$ los valores de $O(t_0 + t)$ observados a través de W.

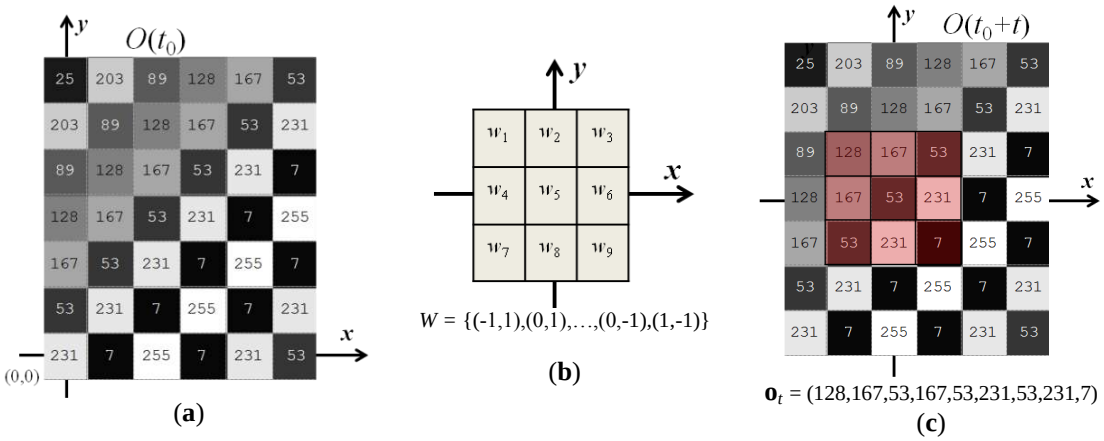


Figura 1.7. Ilustración de la obtención de una configuración de ventana: (**a**) Imagen original O, (**b**) ventana W, y (**c**) configuración de ventana registrada en el vector $\mathbf{o}_t$ para el píxel $t = (2,3)$ de la imagen O.

Sea una ventana W de n píxeles y los conjuntos de niveles de gris L_1 para la imagen de entrada O y L_2 para la imagen de salida $\boldsymbol{\Psi}(O)$ del procesamiento, respectivamente. La función ψ: ${L_1}^W \rightarrow L_2$ cuya entrada es una configuración de ventana y cuya salida es un nivel de gris del conjunto L_2 es llamada *función característica*. Un W-operador $\boldsymbol{\Psi}$ se define y representa a través de una función característica ψ (Figura 1.8) [Barrera y Brun, 1998], [Barrera *et al.*, 2005], [R. Hirata *et al.*, 2006]. Esta función característica procesa configuraciones de ventana observadas a través de W, con lo cual se tiene que

$$\boldsymbol{\Psi}(O)(t) = \psi(O(w_1 + t),\ldots,O(w_n + t)) = \psi(\mathbf{o}_t). \tag{1.9}$$

Por lo tanto, existe una biyección entre el *espacio de W-operadores* $\{\boldsymbol{\Psi}\}$ y el *espacio de funciones características* $\{\psi\}$ que operan sobre las configuraciones vistas a través de la ventana W. Consecuentemente, *diseñar un W-operador consiste en encontrar su función característica* [Barrera *et al.*, 2005], [R. Hirata *et al.*, 2006]. Nótese además que la propiedad de invariancia de los W-operadores garantiza que la función característica es la misma para cualquier píxel a procesar.

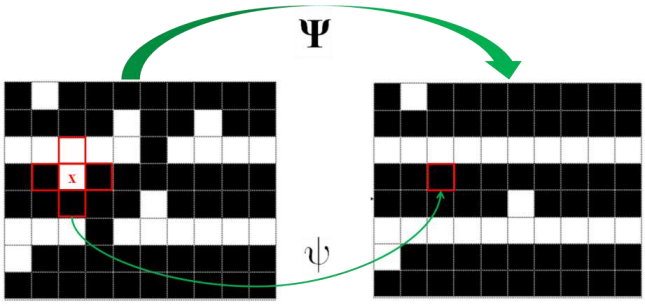


Figura 1.8. Ilustración de un W-operador y su función característica. La entrada y salida de un W-operador son imágenes. La entrada de una función característica es una configuración de ventana y la salida es un nivel de gris.

1.5.2. Diseño Automático de W-operadores: Teoría

Sean las imágenes en escala de grises $O \in L_1^{\boldsymbol{E}}$ e $I \in L_2^{\boldsymbol{E}}$ y la ventana W. En el contexto de los W-operadores, la imagen O será llamada imagen observada y la imagen I será llamada imagen ideal. La *imagen observada* O es una muestra del problema a resolver; mientras que la *imagen ideal* I representa la salida deseada o imagen de gold estándar del procesamiento de la imagen O. Para el diseño automático de W-operadores, las imágenes O e I se modelan como realizaciones de dos procesos estocásticos conjuntos que generan imágenes observadas e imágenes ideales.

Proceso Estocástico: Un *proceso estocástico*, *proceso aleatorio*, o *imagen aleatoria*, se define como un conjunto de variables aleatorias $\{\mathbf{O}(\omega,t)\}$, donde el píxel $t = (x,y)$ pertenece al conjunto $\boldsymbol{E}$ y ω un evento que pertenece al espacio muestral S. Para la representación computacional de imágenes (Sección 1.4.2), el píxel t es un elemento del conjunto de índices $T = \{(i,j) \mid i = 1,\ldots,M \text{ y } j = 1,\ldots,N\}$. Dado un píxel arbitrario y fijo t_0 y ω variable, entonces el valor de $\mathbf{O}$ en el píxel t_0, $\mathbf{O}(\omega,t_0)$, es una variable aleatoria con dominio S y rango L. Por el contrario, dado un evento fijo $\omega_0 \in S$ y t variable, entonces $\mathbf{O}(\omega_0,t)$ define una imagen determinista, denominada *realización* del proceso aleatorio [Dougherty, 1999]. Para simplificar la notación, de aquí en adelante un proceso aleatorio se escribirá como $\mathbf{O}$ en lugar de $\mathbf{O}(\omega,t)$, teniendo en cuenta que existe una distribución $\Pr(\mathbf{O}(\omega,t))$ subyacente a su conducta. Adicionalmente, una realización del proceso $\mathbf{O}$ se denotará mediante O.

Alternativamente a lo anterior, un proceso estocástico $\mathbf{O}$ puede ser visto como una función que mapea cada resultado de un experimento probabilístico a una imagen O con probabilidad $\mathbb{P}(O)$. La colección de todas las imágenes que pueden ser producidas por $\mathbf{O}$ se conoce como conjunto de imágenes del proceso aleatorio. Cada imagen de este conjunto es determinista, pero el proceso es probabilístico, o aleatorio, porque no se conoce *a priori* qué imagen será generada por el experimento [Oppenheim y Verghese, 2010]. En consecuencia, antes de obtener el resultado del experimento, muchos aspectos de la imagen a generarse son impredecibles (por ejemplo media, varianza, posición, cantidad de objetos, entre otras). Esto se debe a que existe una incertidumbre asociada

respecto a la imagen que será producida. Después del experimento, o *a posteriori*, el resultado del experimento, o imagen, queda totalmente determinada.

Ejemplo 1.5. Consideremos un conjunto de *N* personas (hombres y mujeres) que serán sometidos a un escaneo ocular de su ojo derecho usando una cámara de fondo [Staal *et al.*, 2004]. En este grupo existen personas que no presentan ninguna patología ocular, otros tienen exudados y hemorragias como consecuencia de que son diabéticos o adolecen de hipertensión arterial. El experimento consiste en fotografiar la parte interna del ojo derecho de una persona seleccionada aleatoriamente con probabilidad uniforme (1/*N*). En cada ensayo de este experimento, se obtiene una imagen ocular color. La morfología del árbol arterial, su ubicación dentro de la imagen, y la presencia o no de exudados o hemorragias son determinados después observar la imagen resultante del escaneo ocular (Figura 1.9). En este caso, el conjunto de imágenes del proceso aleatorio está formado por las *N* imágenes obtenidas al examinar a cada paciente. Nótese también que, aparte de los componentes anatómicos del fondo retiniano, las imágenes podrían contener artefactos introducidos por la cámara de fondo ocular como ruido u objetos artificiales creados por la dispersión de luz ocasionada por la retina [Fraz *et al.*, 2012].

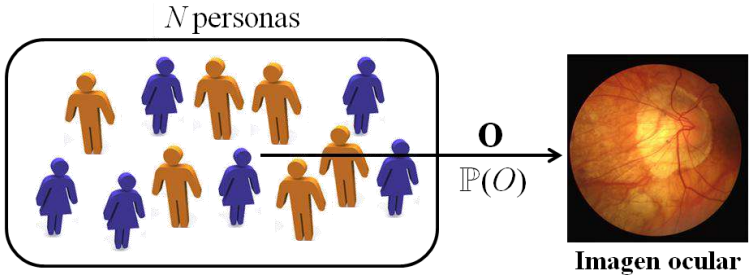


Figura 1.9. Proceso estocástico de generación de imágenes oculares.

La segunda definición de proceso estocástico mencionada anteriormente es fácil de comprender desde el punto de vista conceptual. Sin embargo, es importante mencionar que, a nivel general, resulta impráctico representar un proceso estocástico mediante el conjunto de todas las imágenes y sus probabilidades debido al gran tamaño que podría tener este conjunto. Por esta razón, de aquí en adelante se considera la definición de proceso estocástico que involucra a un conjunto indexado de variables aleatorias $\{O(t_1),\ldots,O(t_m)\}$ con distribución conjunta $\Pr(O(t_1),\ldots,O(t_m))$.

Consideremos un par de procesos estocásticos conjuntos (**O**,**I**) definidos sobre el mismo espacio muestral *S*, donde **O** genera imágenes observadas e **I** genera imágenes ideales. Usualmente, tanto **O** e **I** pueden ser considerados como procesos estocásticos resultantes de aplicar las transformaciones $\mathbb{T}_1$ y $\mathbb{T}_2$ a un proceso estocástico primario **H**, respectivamente. El par de procesos estocásticos conjuntos (**O**,**I**) queda completamente caracterizado a través de la distribución conjunta Pr(**O**,**I**). Por ejemplo, para el caso de restauración de imágenes, se puede tomar una realización de un proceso que genera imágenes no deterioradas **H** y aplicar a dicha realización una transformación de degradación $\mathbb{T}_1$ para producir una realización del proceso de degradación **O**. En este caso, el proceso que genera imágenes ideales **I** resulta de aplicar la transformación identidad $\mathbb{T}_2$ a la realización de **H** [Dougherty, 1999].

Ejemplo 1.6. Consideremos un proceso estocástico **H** que genera imágenes binarias con 9 objetos: 4 diamantes y 5 cuadrados. La distancia entre los vértices y el centro de los

diamantes y la longitud de cada lado de los cuadrados son seleccionados aleatoriamente, con probabilidad uniforme, del conjunto {11,13,15,17,19}. Se define a $\mathbb{T}_1$ como una transformación que añade ruido tipo sal y pimienta, con densidad 0.05, a cada realización de **H**. El proceso que genera imágenes observadas es **O** = $\mathbb{T}_1$(**H**). Se define a $\mathbb{T}_2$ como una transformación que extrae los bordes de todos los objetos contenidos en cada realización de **H**. Para esto se emplea el gradiente morfológico por dilatación con un elemento estructurante que es un disco de radio 2 píxeles. El proceso que genera imágenes ideales es **I** = $\mathbb{T}_2$(**H**).

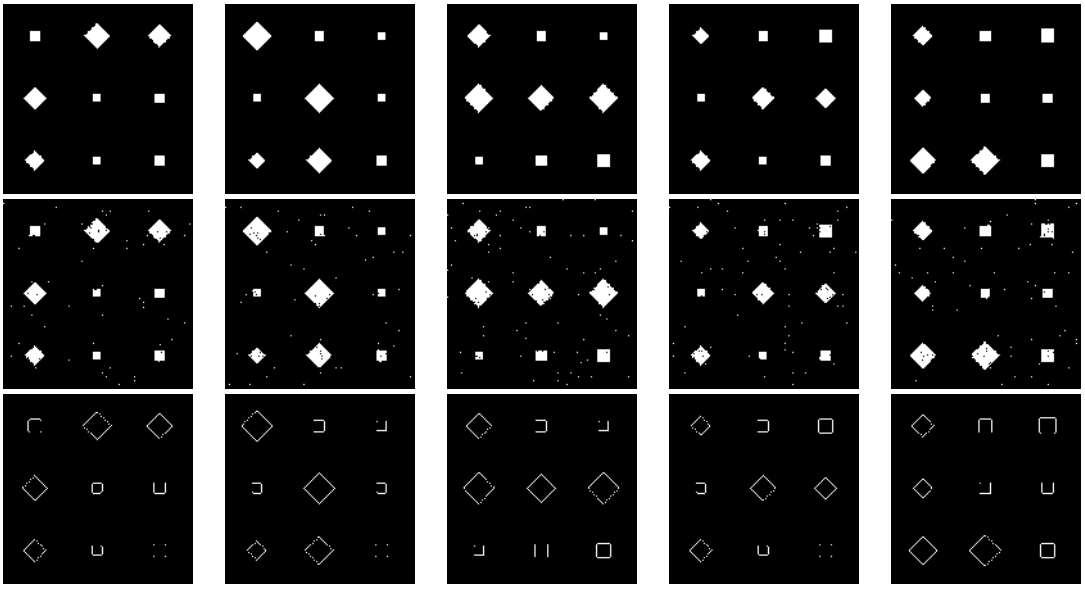


Figura 1.10. Realizaciones de 3 procesos estocásticos **H**, **O** e **I** de generación de imágenes. La primera, segunda, y tercera filas muestran cinco realizaciones de **H**, **O** = $\mathbb{T}_1$(**H**), e **I** = $\mathbb{T}_2$(**H**), respectivamente. $\mathbb{T}_1$ es una transformación que añade ruido tipo sal y pimienta con densidad 0.05, mientras que $\mathbb{T}_2$ es una transformación que extrae bordes.

En la primera fila de la Figura 1.10 se muestran 5 realizaciones del proceso aleatorio **H**. La segunda fila de esta figura muestra las imágenes obtenidas a partir de las imágenes de la primera fila usando $O = \mathbb{T}_1(H)$ (imágenes observadas). La tercera fila muestra las imágenes obtenidas a partir de las imágenes de la primera fila usando $I = \mathbb{T}_2(H)$ (imágenes ideales). Nótese que en este caso las imágenes ideales son aleatorias debido a que las imágenes a partir de las cuales fueron generadas son también aleatorias.

Formulación del Problema de Diseño Automático de W-operadores: Dado el par de procesos estocásticos conjuntos (**O**,**I**) y la ventana *W*, el diseño automático de operadores de ventana o W-operadores consiste en encontrar el operador **Ψ** que, con mínimo costo o error, estime las imágenes ideales a partir de las imágenes observadas para todas las realizaciones (*O*,*I*) de (**O**,**I**). La calidad de la estimación, o error, del operador **Ψ** se obtiene mediante

$$\varepsilon(\mathbf{\Psi}) = \mathbb{E}[R(\mathbf{\Psi}(\mathbf{O}),\mathbf{I})], \qquad (1.10)$$

donde $\mathbb{E}$ es la esperanza calculada sobre la distribución conjunta Pr(**O**,**I**). La función de costo R: $L_1^{\boldsymbol{E}} \times L_2^{\boldsymbol{E}} \rightarrow [0,1]$ se define como

$$R(\boldsymbol{\Psi}(\mathbf{O}),\mathbf{I}) = \sum_{\forall t \in \boldsymbol{E}} r(\boldsymbol{\Psi}(\mathbf{O})(t),\mathbf{I}(t)), \tag{1.11}$$

donde r: $L_1 \times L_2 \rightarrow [0,1]$ es una métrica que mide, píxel a píxel, la discrepancia entre las imágenes O e I. Reemplazando la ecuación 1.11 en 1.10 se obtiene:

$$\varepsilon(\boldsymbol{\Psi}) = \mathbb{E}[\sum_{\forall t \in \boldsymbol{E}} r(\boldsymbol{\Psi}(\mathbf{O})(t),\mathbf{I}(t))]. \tag{1.12}$$

Dado que el W-operador $\boldsymbol{\Psi}$ se define y representa a través de la función característica ψ, entonces la ecuación 1.12 resulta en

$$\varepsilon(\boldsymbol{\Psi}) = \varepsilon(\psi) = \mathbb{E}[\sum_{\forall t \in \boldsymbol{E}} r(\psi(\mathbf{o}_t),\mathbf{I}(t))], \tag{1.13}$$

donde $\mathbb{E}$ es la esperanza calculada sobre la distribución conjunta del par ($\mathbf{o}_t$,$\mathbf{I}(t)$). En este caso $\mathbf{o}_t$ es un proceso estocástico que genera configuraciones de ventana para el píxel t de la imagen **O**. $\mathbf{I}(t)$ es una variable aleatoria asociada con el proceso $\mathbf{o}_t$.

Si se considera que el par de procesos estocásticos conjuntos (**O**,**I**) son *estrictamente estacionarios* [Dougherty, 1999], [Oppenheim y Verghese, 2010], entonces la distribución conjunta del par ($\mathbf{o}_t$,$\mathbf{I}(t)$) no cambia ante traslaciones espaciales de $\mathbf{o}_t$ e $\mathbf{I}(t)$: $\Pr(\mathbf{o}_t,\mathbf{I}(t)) = \Pr(\mathbf{o}_{t+t'},\mathbf{I}(t+t'))$ para cualquier t y t' pertenecientes a $\boldsymbol{E}$. Por lo tanto, si se define el vector aleatorio $\mathbf{X} = (X_1,\ldots,X_n) = \mathbf{o}_t$ y la variable aleatoria $Y = \mathbf{I}(t)$, entonces la ecuación 1.13 resulta en

$$\varepsilon(\psi) = \mathbb{E}[(\sum_{\forall (\mathbf{X},Y) \in \{\mathcal{X} \times \mathcal{Y}\}} r(\psi(\mathbf{X}),Y)], \tag{1.14}$$

donde $\mathbb{E}$ es la esperanza calculada sobre la distribución conjunta Pr(**X**,Y) de los pares (**X**,Y). Esta distribución es inducida por la distribución conjunta Pr(**O**,**I**). $\mathcal{X}$ denota el espacio de todas las posibles configuraciones de ventana que se pueden observar en **O** a través de la ventana de n píxeles W. $\mathcal{Y}$ denota el conjunto de todos los niveles de gris que se pueden observar en **I**. Finalmente, la ecuación 1.14 es equivalente a

$$\varepsilon(\psi) = \sum_{\forall (\mathbf{X},Y) \in \{\mathcal{X} \times \mathcal{Y}\}} r(\psi(\mathbf{X}),Y)\mathbb{P}(\mathbf{X},Y), \tag{1.15}$$

donde $\mathbb{P}(\mathbf{X},Y)$ es la probabilidad conjunta del par $(\mathbf{X},Y) \in \{\mathcal{X} \times \mathcal{Y}\}$.

Para una ventana W de tamaño $n = |W|$, y realizaciones $O \in L_1^{\boldsymbol{E}}$ del proceso estocástico **O**, $\mathcal{X}$ es un conjunto formado por $|L_1|^n$ posibles configuraciones de ventana **X**. Nótese la dependencia exponencial que existe entre el tamaño del conjunto $\mathcal{X}$ y el número n de píxeles de la ventana W. Adicionalmente, es usual que las componentes X_i del vector **X**, con $i = 1,\ldots n$, estén correlacionadas. Así mismo, dado que cada realización I del proceso **I** pertenece al conjunto $L_2^{\boldsymbol{E}}$, entonces $\mathcal{Y}$ es un conjunto formado por $|L_2|$ valores o niveles de gris. Por lo tanto, el tamaño del espacio de W-operadores $\{\boldsymbol{\Psi}\}$ para una ventana W, de n píxeles, y las imágenes $O \in L_1^{\boldsymbol{E}}$ e $I \in L_2^{\boldsymbol{E}}$ es $|L_2|^{|L_1|^n}$.

Si se asume el *costo* 0-1 donde $r = \mathbb{I}(\psi(\mathbf{X}) \neq Y)$, siendo $\mathbb{I}$: $L_1 \times L_2 \rightarrow \{0,1\}$ la *función indicador* igual a 1 si se cumple que $\psi(\mathbf{X}) \neq Y$ y 0 en el caso contrario, entonces la ecuación 1.15 puede expresarse como:

$$\varepsilon(\psi) = \sum_{\forall \mathbf{X} \in \mathcal{X}} \left(\sum_{\forall i \in \mathcal{Y}} \mathbb{I}(\psi(\mathbf{X}) \neq i) \mathbb{P}(Y = i \mid \mathbf{X}) \right) \mathbb{P}(\mathbf{X}), \tag{1.16}$$

donde $\mathbb{P}(Y = i|\mathbf{X})$ denota la probabilidad condicional de que la variable aleatoria Y tome el valor $i \in \mathcal{Y}$ dada la configuración de ventana $\mathbf{X} \in \mathcal{X}$, y $\mathbb{P}(\mathbf{X})$ es la probabilidad marginal con que se observa la configuración $\mathbf{X}$ a través de W.

En estas condiciones, la *función característica óptima* ψ_{opt}, u *operador de Bayes*, que minimiza la ecuación 1.16, para toda configuración $\mathbf{X} \in \mathcal{X}$, está dada por

$$\psi_{\text{opt}}(\mathbf{X}) = argmax_{i \in \mathcal{Y}} \mathbb{P}(Y = i|\mathbf{X}). \tag{1.17}$$

Por lo tanto, para un proceso estocástico, conjunto y estacionario ($\mathbf{O}$,$\mathbf{I}$), y la ventana W, el W-operador óptimo $\boldsymbol{\Psi}_{\text{opt}}$ queda definido a través de la función característica óptima ψ_{opt}. Esta función asigna a cada configuración de ventana $\mathbf{X} \in \mathcal{X}$ el nivel de gris $Y \in \mathcal{Y}$ que tiene la máxima probabilidad condicional $\mathbb{P}(Y|\mathbf{X})$. Finalmente, el error de la función característica óptima, o *error de Bayes*, es $\varepsilon(\psi_{\text{opt}})$.

Para el caso particular donde las imágenes ideales son imágenes binarias pertenecientes al conjunto $\{0,1\}^{\boldsymbol{E}}$, la ecuación 1.16 puede escribirse como

$$\varepsilon(\psi) = \sum_{\{\mathbf{X}:\, \psi(\mathbf{X}) = 1\}} \mathbb{P}(Y = 0 \mid \mathbf{X}) \mathbb{P}(\mathbf{X}) + \sum_{\{\mathbf{X}:\, \psi(\mathbf{X}) = 0\}} \mathbb{P}(Y = 1 \mid \mathbf{X}) \mathbb{P}(\mathbf{X}), \tag{1.18}$$

donde $\mathbb{P}(Y = 1|\mathbf{X})$ y $\mathbb{P}(Y = 0|\mathbf{X}) = 1 - \mathbb{P}(Y = 1|\mathbf{X})$ son las probabilidades condicionales de que, en la imagen ideal, se observe un 1 o un 0, respectivamente, dada la configuración $\mathbf{X}$ registrada en la imagen observada. La ecuación 1.18 es equivalente a la siguiente ecuación:

$$\varepsilon(\psi) = \sum_{\{Y:(\psi(\mathbf{X})=0, Y=1)\}} \mathbb{P}(\mathbf{X} \mid Y = 1) \mathbb{P}(Y = 1) + \sum_{\{Y:(\psi(\mathbf{X})=1, Y=0)\}} \mathbb{P}(\mathbf{X} \mid Y = 0) \mathbb{P}(Y = 0), \tag{1.19}$$

donde $\mathbb{P}(\psi(\mathbf{X}) = 0|Y = 1) = \sum_{\{Y:(\psi(\mathbf{X})=0, Y=1)\}} \mathbb{P}(\mathbf{X} \mid Y = 1)$ es la tasa de falsos negativos (FNR del inglés *false negative rate*) y $\mathbb{P}(\psi(\mathbf{X}) = 0|Y = 1) = \sum_{\{Y:(\psi(\mathbf{X})=1, Y=0)\}} \mathbb{P}(\mathbf{X} \mid Y = 0)$ es la tasa de falsos positivos (FPR del inglés *false positive rate*) de la función característica ψ, respectivamente. Los términos $\mathbb{P}(Y = 0)$ y $\mathbb{P}(Y = 1)$ denotan las probabilidades *a priori* de observar 0 o 1 en las imágenes ideales, respectivamente. Nótese también que en este caso particular, las ecuaciones 1.18 y 1.19 son equivalentes al error cuadrático medio de ψ, $\mathbb{E}[(\psi(\mathbf{X}) - Y)^2]$.

En estas condiciones, la función característica se define mediante la siguiente regla:

$$\psi(\mathbf{X}) = \begin{cases} 1 & \text{si } \mathbb{P}(Y = 1 \mid \mathbf{X}) \geq \tau \\ 0 & \text{en el caso contrario} \end{cases}. \tag{1.20}$$

En la ecuación 1.20, si $\tau = 0.5$ entonces la función característica resultante es la función óptima ψ_{opt}, u operador de Bayes, que minimiza cualquiera las ecuaciones 1.18 y 1.19. Alternativamente, el diseñador puede seleccionar un valor de umbral τ diferente a 0.5 en base a un balance entre los valores de FNR y FPR de la función ψ.

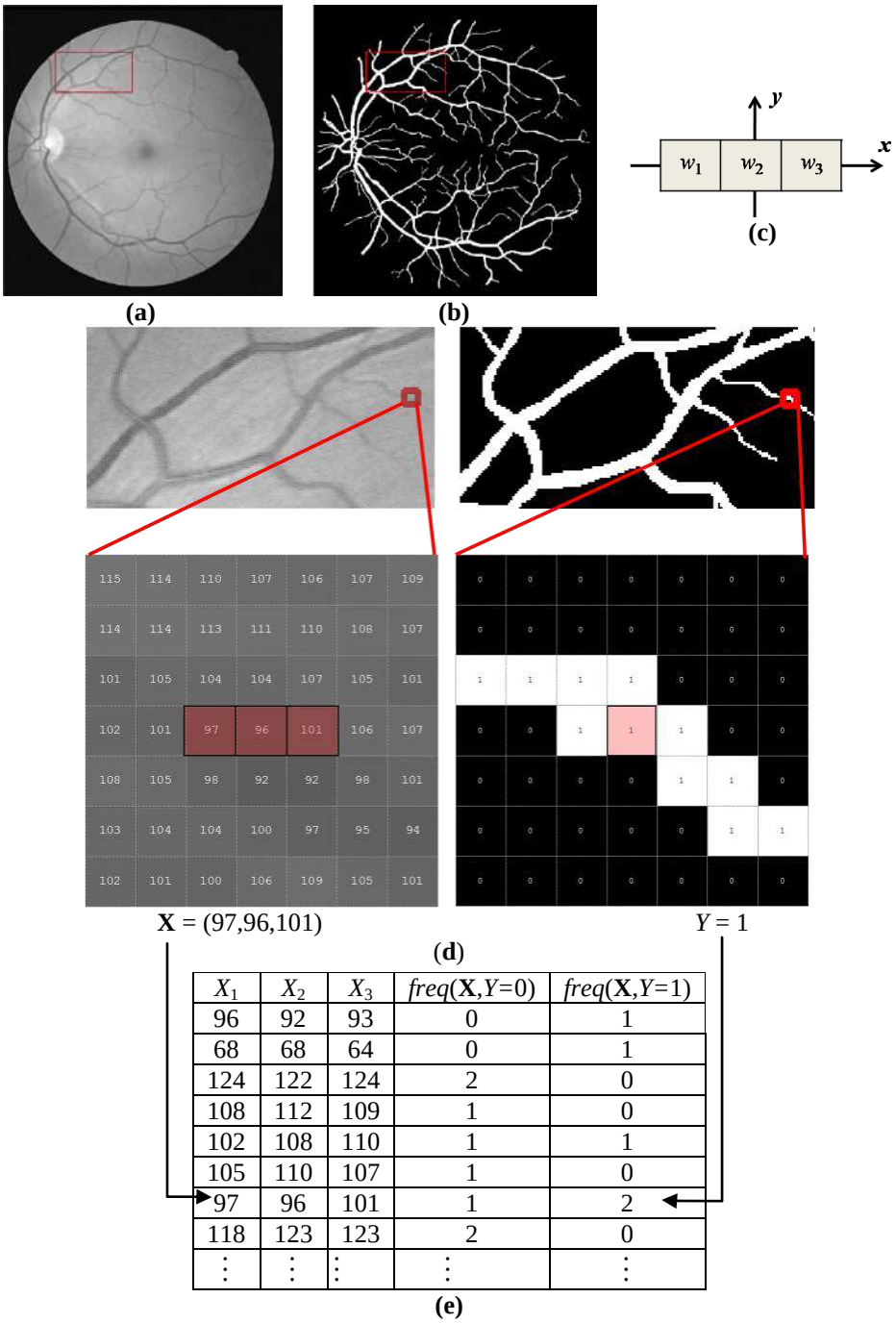
El error $\varepsilon(\psi_{opt})$ del operador óptimo ψ_{opt}, calculado mediante las ecuaciones 1.18 o 1.19, es igual a la expresión $\mathbb{E}[min\{\mathbb{P}(Y = 0|\mathbf{X}),\mathbb{P}(Y = 1|\mathbf{X})\}]$, donde $\mathbb{E}$ es la esperanza calculada sobre la distribución del espacio $\mathcal{X}$. Aplicando la desigualdad de Jensen, se puede mostrar que $\varepsilon(\psi_{opt}) \leq min\{\mathbb{E}[\mathbb{P}(Y = 0|\mathbf{X})],\mathbb{E}[\mathbb{P}(Y = 1|\mathbf{X})]\}$. Por lo tanto, si la probabilidad condicional $\mathbb{P}(Y = 0|\mathbf{X})$, o $\mathbb{P}(Y = 1|\mathbf{X})$, es uniformemente pequeña, lo cual implica que uno de los posibles valores de Y es más probable que el otro, entonces el error del operador óptimo $\varepsilon(\psi_{opt})$ es necesariamente pequeño [Braga-Neto y Dougherty, 2005].

1.5.3. Diseño Automático de W-operadores a partir de Ejemplos de Entrenamiento

Las definiciones presentadas a través de las ecuaciones 1.17 y 1.20 dependen del valor de la probabilidad condicional $\mathbb{P}(Y|\mathbf{X})$ para cada configuración de ventana $\mathbf{X}$. En la práctica, usualmente, se desconoce tanto la distribución $\Pr(\mathbf{O},\mathbf{I})$ subyacente al proceso estocástico conjunto $(\mathbf{O},\mathbf{I})$ como la distribución conjunta $\Pr(\mathbf{X},Y)$ de los pares $(\mathbf{X},Y)$. Sin embargo, es posible contar con un *conjunto de imágenes de entrenamiento* $\{(O_1,I_1),...,(O_m,I_m)\}$, donde cada par (O_i,I_i), denominado *par de imágenes de entrenamiento*, es considerado una *realización independiente* del proceso estocástico conjunto $(\mathbf{O},\mathbf{I})$, con $i = 1,\ldots,m$. Las imágenes observadas del conjunto de entrenamiento son muestras del problema a resolver. Las imágenes ideales se consiguen, usualmente, aplicando un procesamiento manual de las imágenes observadas con la asistencia de expertos en el área del cual provienen las imágenes [Dougherty, 1999]. Por ejemplo, en el caso de las imágenes oculares son los oftalmólogos los que guían la segmentación manual de los vasos sanguíneos o patologías para obtener las imágenes ideales [Fraz *et al.*, 2012].

En base al conjunto de imágenes de entrenamiento $\{(O_1,I_1),...,(O_m,I_m)\}$ y una ventana W, seleccionada usualmente *a priori*, el diseñador puede obtener el *conjunto de ejemplos de entrenamiento* $\mathcal{D} = \{(\mathbf{X}_i,freq(\mathbf{X}_i,Y = 0),...,freq(\mathbf{X}_i,Y = |\mathcal{Y}| - 1))\}$ mediante un escaneo de las imágenes del conjunto de entrenamiento con la ventana W (Ejemplo 1.7). En este caso se asume que cada par $(\mathbf{X},Y)$ observado durante el escaneo es una *realización independientemente* de la distribución conjunta $\Pr(\mathbf{X},Y)$, con $i = 1,\ldots,N$. La cantidad de veces, o frecuencia, con que una configuración de ventana $\mathbf{X}$ es observada asociada con el nivel de gris Y se denota por $freq(\mathbf{X},Y)$. En la práctica, el conjunto de ejemplos de entrenamiento se representa computacionalmente por medio de una *tabla de frecuencias*, o matriz, de por N filas y $(n + |\mathcal{Y}|)$ columnas, donde $|\mathcal{Y}| = |L_2|$. Cada fila de esta matriz contiene una configuración de ventana distinta con sus correspondientes frecuencias $freq(\mathbf{X},Y = i)$, con $i = 0,\ldots,|\mathcal{Y}| - 1$. Las tablas de frecuencias para el diseño

automático de W-operadores son similares a las tablas de verdad para el diseño de circuitos electrónicos digitales.



X_1	X_2	X_3	*freq*(**X**,*Y*=0)	*freq*(**X**,*Y*=1)
96	92	93	0	1
68	68	64	0	1
124	122	124	2	0
108	112	109	1	0
102	108	110	1	1
105	110	107	1	0
97	96	101	1	2
118	123	123	2	0
⋮	⋮	⋮	⋮	⋮

Figura 1.11. Ilustración del procedimiento de escaneo para el diseño automático de W-operadores: (**a**) Imagen observada, (**b**) imagen ideal, (**c**) ventana, (**d**) configuración de ventana, y (**e**) tabla de frecuencias.

Ejemplo 1.7. En la Figura 1.11 se ilustra el proceso de escaneo y construcción de una tabla de frecuencias para diseño automático de W-operadores. Par esto se utiliza una ventana *W* rectangular formada por $n = |W| = 1\times3$ píxeles, una imagen ocular *O* (imagen observada) en escala de grises, y una imagen binaria *I* con los vasos sanguíneos manualmente segmentados (imagen ideal). Estas imágenes provienen de la base de datos pública DRIVE [Staal *et al.*, 2004]. El píxel a procesar corresponde al centro de la ventana *W*. Tanto la imagen en escala de grises como la imagen binaria están

compuestas por 584×565 píxeles. Cada píxel de la imagen observada puede tomar un valor del conjunto $L = \{0,\ldots,255\}$ compuesto por $|L| = 256$ niveles de gris.

Para este ejemplo, la cantidad máxima de configuraciones de ventana que la tabla de frecuencias puede contener es igual al número de píxeles de las imágenes a procesar, en este caso 584×565 = 3329960. Dado que existen configuraciones que se repiten durante el escaneo, el número de filas de la tabla de frecuencias es usualmente menor que la cantidad de píxeles de las imágenes a procesar. La cantidad máxima de posibles configuraciones de ventana que se pueden observar a través de W es $256^3 = 16777216$. Para facilitar la visualización de la obtención de configuraciones de ventana y sus valores asociados en la imagen ideal se utilizan las regiones señaladas con un rectángulo en las imágenes observada e ideal. En la práctica, sin embargo, se escanea todo el dominio de las imágenes de entrenamiento fila por fila (o columna por columna), y píxel a píxel, siempre y cuando W esté totalmente contenida en dicho dominio.

El asumir que los pares $(\mathbf{X},Y)$ son independientes es una fuerte restricción que facilita enormemente el análisis teórico del diseño de W-operadores, pero que en la práctica no siempre es posible de conseguir. Por ejemplo, al escanear las imágenes de entrenamiento fila por fila (o columna por columna), y píxel a píxel, se puede advertir que existe un solapamiento entre las configuraciones de ventana de los píxeles alrededor del píxel actualmente observado (Figura 1.12). Esto implica que, el observar una configuración para un píxel determinado restringe el número de posibles configuraciones que se pueden observar para sus píxeles vecinos. Dicho de otro modo, la observación de una configuración de ventana para un píxel dado aporta algo de información sobre las configuraciones de ventana para los píxeles que están a su alrededor.

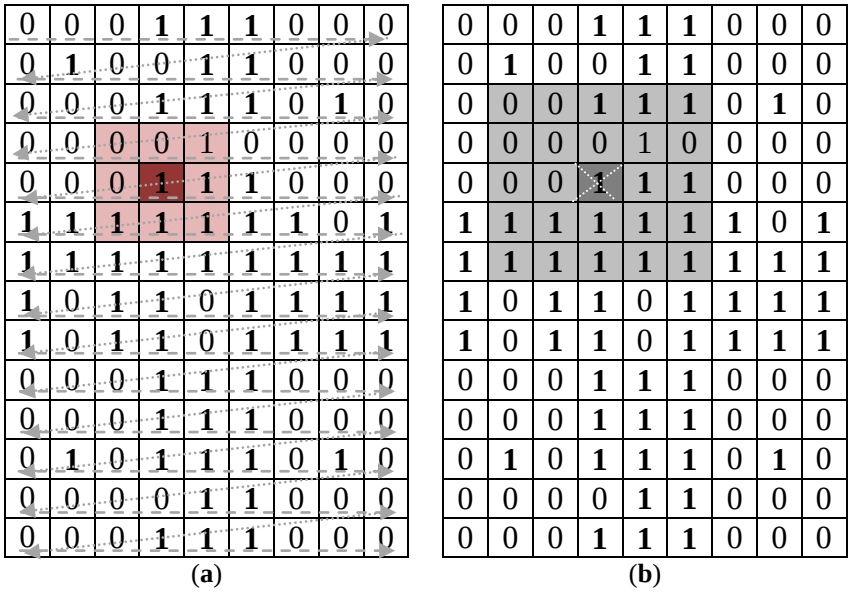


Figura 1.12. Ilustración del solapamiento que hay entre configuraciones de ventana para un escaneo fila por fila y píxel a píxel de una imagen binaria con una ventana de 3×3 píxeles. (**a**) Imagen binaria y configuración de ventana. (**b**) Región de píxeles cuyas configuraciones de ventana se solapan con la configuración para el píxel marcado con una X.

Por lo tanto, se puede ver que en el caso analizado en el párrafo anterior no existe una completa independencia entre los pares $(\mathbf{X},Y)$. Sin embargo, existe evidencia de un

comportamiento similar entre operadores diseñados a partir de conjuntos de entrenamiento donde los pares ($\mathbf{X}$,Y) son ligeramente dependientes y conjuntos donde la independencia entre los pares ($\mathbf{X}$,Y) es total [Devroye *et al.*, 1996]. En el análisis y definiciones sucesivas presentadas en esta tesis se mantiene el criterio de *independencia e idéntica distribución probabilística* de los pares ($\mathbf{X}$,Y). Esto facilita notablemente el análisis teórico sin afectar los resultados empíricos, en base a los cuales se evaluará y se comparará la performance de los modelos testeados dado un problema concreto.

El tamaño y forma de la ventana W se selecciona, usualmente, de forma heurística en función de las características del problema en cuestión. Por ejemplo, si el W-operador se diseña para segmentación o filtrado de ruido, entonces la ventana W debe tener una forma y tamaño que permitan capturar suficiente información para discriminar el objeto a segmentar o eliminar los artefactos de ruido de los demás objetos contenidos en las imágenes a procesar. A nivel general, la ventana W se selecciona siguiendo los mismos criterios con que se seleccionan tanto el tamaño como la forma del elemento estructurante para el diseño heurístico de operadores morfológicos [Gonzalez y Woods, 2008], [Shih, 2009].

En base al conjunto de ejemplos de entrenamiento $\boldsymbol{\mathcal{D}}$, o tabla de frecuencias, y una *regla de estimación*, o *algoritmo de aprendizaje supervisado* $\mathcal{A}$: $\{\boldsymbol{\mathcal{X}}\times\mathcal{Y}\}^N\times\boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$, en la *etapa de entrenamiento*, o *aprendizaje*, el diseñador obtiene la función ψ_N que es un estimador de la función característica óptima ψ_{opt} [Duda *et al.*, 2001], [Murphy, 2012]. Para evaluar la calidad de la función estimada ψ_N, se puede usar un *conjunto de imágenes de testeo* $\{(O_1,I_1),...,(O_p,I_p)\}$, donde ninguno de sus pares de imágenes fueron utilizados la etapa de entrenamiento. En este conjunto, cada par (O_i,I_i), denominado *par de imágenes de testeo*, es considerado una realización independiente del par ($\mathbf{O}$,$\mathbf{I}$), donde $i = 1,\ldots,p$. En base a este conjunto, en la *etapa de testeo* se puede obtener una estimación $\varepsilon_M(\psi_N)$ del error verdadero del W-operador diseñado $\varepsilon(\psi_N)$ mediante la siguiente relación empírica:

$$\varepsilon_M(\psi_N) = (1/M)\sum_{k=1}^{p}\sum_{(i,j)\in\{1,...,A\}\times\{1,...,B\}} \mathbb{I}(\psi_N(\mathbf{X}_{i,j}^{(k)}) \neq Y_{i,j}^{(k)}) . \qquad (1.21)$$

Para la ecuación 1.21 se asume que cada imagen de testeo (observada e ideal) tiene un tamaño de A×B píxeles, donde $M = p$×A×B. La función indicador $\mathbb{I}(\psi_N(\mathbf{X}_{i,j}^{(k)}) \neq Y_{i,j}^{(k)})$ es 1 si se cumple que $\psi_N(\mathbf{X}_{i,j}^{(k)}) \neq Y_{i,j}^{(k)}$ y 0 en el caso contrario. $\mathbf{X}_{i,j}^{(k)}$ es la configuración de ventana observada en el píxel (i,j) de la imagen O_k e $Y_{i,j}^{(k)}$ es el valor del píxel (i,j) de la imagen I_k. En este caso tiene que $\varepsilon_M(\psi_N) \rightarrow \varepsilon(\psi_N)$ cuando $M \rightarrow \infty$.

Es importante anotar que, el error verdadero $\varepsilon(\psi_N)$ del operador diseñado ψ_N es mayor o a lo sumo igual que el error del operador óptimo ψ_{opt}: $\varepsilon(\psi_N) \geq \varepsilon(\psi_{\text{opt}})$. Por lo tanto, existe un *costo de estimación*, o *costo de diseño*, $\Delta(\psi_N,\psi_{\text{opt}})$, de la función ψ_N usando una determinada regla de estimación $\mathcal{A}$ y el conjunto de ejemplos de entrenamiento $\boldsymbol{\mathcal{D}}$. Este costo está dado por $\Delta(\psi_N,\psi_{\text{opt}}) = \varepsilon(\psi_N) - \varepsilon(\psi_{\text{opt}})$. Por lo tanto, el error verdadero

$\varepsilon(\psi_N)$ del operador estimado ψ_N, en función del costo de estimación $\Delta(\psi_N,\psi_{opt})$ y el error $\varepsilon(\psi_{opt})$ del operador óptimo ψ_{opt}, se calcula como

$$\varepsilon(\psi_N) = \Delta(\psi_N,\psi_{opt}) + \varepsilon(\psi_{opt}). \quad (1.22)$$

Nótese que para una regla de estimación determinada $\mathcal{A}$, es posible obtener una función característica ψ_N diferente por cada conjunto de ejemplos de entrenamiento $\mathcal{D}$ formado por N tuplas de la forma $(\mathbf{X},freq(\mathbf{X},Y = 0),...,freq(\mathbf{X},Y = |\mathcal{Y}| - 1))$. Por lo tanto, para fines de comparación de dos reglas de estimación $\mathcal{A}_1$ y $\mathcal{A}_2$ para un problema determinado, es conveniente comparar el error esperado, sobre todos los conjuntos $\mathcal{D}$, de las funciones características diseñadas a partir de cada regla. Para esto se calcula el valor esperado de los dos lados de la ecuación 1.22 con respecto a la distribución probabilística subyacente a la generación de conjuntos $\mathcal{D}$, con N tuplas, obteniendo

$$\mathbb{E}[\varepsilon(\psi_N)] = \mathbb{E}[\Delta(\psi_N,\psi_{opt})] + \varepsilon(\psi_{opt}), \quad (1.23)$$

donde el error $\varepsilon(\psi_{opt})$ es una cantidad constante.

Resumen: el diseño automático de W-operadores consiste de una etapa de entrenamiento y una etapa de testeo. En la etapa de entrenamiento se realiza la estimación, o diseño, de una función característica, u operador, ψ_N. Para esto se emplea algún algoritmo de aprendizaje computacional supervisado $\mathcal{A}$ y un conjunto de ejemplos de entrenamiento $\mathcal{D}$. Este conjunto se obtiene a partir del conjunto de imágenes de entrenamiento realizando un escaneo de sus píxeles con una ventana W. Esta ventana es seleccionada *a priori* en base a las características del problema en cuestión.

El algoritmo de aprendizaje se selecciona teniendo en cuenta la cantidad de datos disponibles para el diseño, entre otras características del problema en cuestión. La estimación de la función característica se realiza de tal modo que, el costo de estimación o diseño sea el más bajo posible, lo cual asegura que el error del operador diseñado sea también bajo tal como se evidencia a través de la ecuación 1.22. En la etapa de testeo se estima el error del operador diseñado mediante el cálculo del error empírico definido a través de la ecuación 1.21. Este error permite evaluar la capacidad predictiva, o futuro desempeño, del operador diseñado cuando se aplica a nuevas imágenes del problema en cuestión que no fueron registradas en los conjuntos de entrenamiento y testeo.

1.6. Ventajas y Desventajas del Diseño Automático de Operadores de Imágenes

Debido a la subjetividad y usual limitación de experiencia y conocimientos del diseñador, el tiempo de diseño heurístico y sus resultados están altamente condicionados en el diseñador y no en el problema en cuestión. Una de las principales ventajas del diseño automático radica en la reducción de su dependencia en la experiencia y conocimientos del diseñador. En el enfoque automático, la solución para un problema determinado depende mayoritariamente de la cantidad y calidad de los datos de entrenamiento y el algoritmo de aprendizaje seleccionado [Devroye *et al.*, 1996], [Barrera et al., 2000], [Duda *et al.*, 2001], [R. Hirata *et al.*, 2002, 2006], [Bishop, 2006],

[Abu-Mostafa *et al.*, 2012], [Mohri *et al.*, 2012], [Murphy, 2012],[Benalcázar *et al.*, 2013, 2014]. Naturalmente, si la calidad de los datos es mala; es decir, si éstos no son una buena muestra de la distribución probabilística subyacente al problema en cuestión, o tienen un exceso de ruido, es esperable que la solución a obtener también sea mala: *basura a la entrada, basura a la salida*. De igual manera, si la cantidad de datos disponibles es baja y el algoritmo de aprendizaje es demasiado complejo, es muy probable que los resultados que se obtengan tampoco sean satisfactorios: *pocos datos y algoritmos demasiado complejos, usualmente, arrojan malos resultados* [Devroye *et al.*, 1996], [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012], [Murphy, 2012].

El conocimiento o experiencia del diseñador respecto de un problema determinado no quedan excluidos del diseño automático. Por el contrario, estos pueden ser incorporados en el diseño, o etapa de entrenamiento, en forma de conocimiento *a priori*. El conocimiento *a priori* también puede servir para elegir el algoritmo que mejor se adapte a la distribución probabilística subyacente al problema en cuestión [Dougherty y Barrera, 2002], [Barrera *et al.*, 2000, 2005]. Si el conocimiento *a priori* disponible es bueno, éste ayudará a reducir el costo de estimación o diseño, y por ende el error del operador a diseñar.

Sin embargo, si el conocimiento *a priori* no es bueno, éste podría aumentar el error del operador diseñado [Dougherty y Barrera, 2002]. De cualquier modo, no es un requisito indispensable para el diseño automático el disponer de conocimiento *a priori*. En contraste a estas ventajas, para el diseño automático de operadores morfológicos, el diseñador debe disponer de conocimiento sobre estimación estadística y aprendizaje computacional. Esto le permitirá seleccionar el algoritmo de aprendizaje a emplearse en función del problema en cuestión y la cantidad de datos disponibles.

Por otro lado, el diseño automático de operadores morfológicos tiene también desventajas como cualquier otro enfoque propuesto para PDI. La principal desventaja consiste en el requerimiento no sólo aquellas imágenes a procesar, sino también de sus imágenes resultado o de gold estándar. Sin embargo, en la mayoría de los problemas se cuenta con imágenes gold estándar, incluso si el enfoque de diseño es heurístico. Esto debido a que las imágenes gold estándar permiten cuantificar la performance o poder predictivo de un modelo científico y hacer comparaciones con otros modelos aplicados a la misma problemática. Después de todo, la validez científica de un modelo no se juzga de manera visual, sino evaluando qué tan exactas son sus predicciones [Feynman, 1985], [Dougherty, 2008], [Bittner y Dougherty, 2012]. Una excepción para esta regla es, por ejemplo, cuando se aborda el problema de realce de la calidad visual de imágenes donde la validación visual es, usualmente, el principal y en algunos casos el único criterio de validación de un modelo.

1.7. Anotaciones Finales

En este capítulo se ha planteado, en el contexto de procesamiento digital de imágenes, al problema de diseño automático de operadores morfológicos. Se han presentado algunas definiciones que se utilizarán en los capítulos posteriores de esta tesis. Adicionalmente, se ha formulado el problema de diseño automático de W-operadores en base a muestras o ejemplos de entrenamiento. En el capítulo siguiente se presentarán y analizarán los principales métodos propuestos en la literatura científica para el diseño automático de operadores morfológicos. El análisis se centrará en base a su idoneidad para resolver

problemas de procesamiento de imágenes médicas. En particular, se abordará sobre los algoritmos o métodos de estimación de la función característica de W-operadores. Se analizará también la influencia de la complejidad del algoritmo de aprendizaje y la cantidad de ejemplos disponibles sobre el desempeño del operador diseñado. En base a este análisis se propondrán nuevos métodos de diseño de W-operadores con aplicaciones a problemas reales de procesamiento de imágenes médicas.

1.8. Referencias

[1] J. Angulo, "Morphological colour operators in totally ordered lattices based on distances: Application to image filtering, enhancement and analysis," *Computer Vision and Image Understanding*, vol. 107, pp. 56-73, 2007.

[2] J. Astola, P. Haavisto y Y. Nuevo, "Vector median filters," *Proceedings of the IEEE*, vol. 78, pp. 678–689, 1990.

[3] J. Serra, "Anamorphoses and function lattices (multivalued morphology)," en *Mathematical Morphology in Image Processing*, E.R. Dougherty ed., pp. 483-523, 1992.

[4] E. Aptoula y S. Lefèvre, "A comparative study on mutivariate mathematical morphology," *Pattern Recognition*, vol. 40, pp. 2914–2929, 2007.

[5] P.E. Trahanias y A.N. Venetsanopoulos, "Color edge detectors based on multivariate ordering," *Proceedings of Visual Communications and Image Processing*, pp.1396-1407, 1992.

[6] X. Guo y B. Guo, "Color image morphology based on distances in the HSI color space", *ISECS International Colloquium on Computing, Communication, Control, and Management*, pp. 264-267, 2009.

[7] G. Dougherty, Digital image processing for medical applications, Cambridge Press, Cambridge, 2011.

[8] R.M. Haralick y L.G. Shapiro, Computer and robot vision, vol. 1, Addison-Wesley, 1992.

[9] R. Van Den Boomgard y R. Van Balen, "Methods for fast morphological image transforms using bitmapped images," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, vol. 54, pp. 254-258, 1992.

[10] L.Vincent, "Morphological grayscale reconstruction in image analysis: applications and efficient algorithms," *IEEE Transactions on Image Processing*, vol. 2, pp. 176-201, 1993.

[11] F.Y. Shih, Image Processing and Mathematical Morphology, CRC Press, New York, 2009.

[12] R.C. Gonzalez y R.E.Woods, Digital Image Processing, Prentice Hall, 2008.

[13] M.A. Luengo, D. Pastor, E. Faure, T. Savy, B. Lombardot, J.L. Rubio, L. Duloquin, M.J. Ledesma, P. Bourgine, N. Peyrieras, A. Santos, "3D+t morphological processing: applications to embryogenesis image analysis," *IEEE Transactions on Image Processing*, vol. 21, pp. 3518-3530, 2012.

[14] G. Lin, U. Adiga, K. Olson, J.F. Guzowski, C.A. Barnes y B. Roysam, "A hybrid 3D watershed algorithm incorporating gradient cues and object models for automatic segmentation of nuclei in confocal image stacks," *Cytometry A*, vol. 56, pp. 23-36, 2003.

[15] F.Y. Shih y O.R. Mitchell, "Threshold decomposition of grayscale morphology into binary morphology," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 31-42, 1989.

[16] F. Meyer y S. Beucher, "Morphological segmentation," *Visual Communications and Image Representation*, vol. 1, pp. 21-46, 1990.

[17] L. Vicent y P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 583-598, 1991.

[18] J. Serra, Image Analysis and Mathematical Morphology, Academic Press, New York, 1982.

[19] E.R. Dougherty, Random Processes for Image and Signal Processing, IEEE Press, New York, 1999.

[20] A.V. Oppenheim y G.C. Verghese, Signals, Systems, and Inference, MIT, Cambridge, 2010.

[21] J. Staal, M. Abramoff, M. Niemeijer, M. Viergever y B. Van Ginneken, "Ridge-based vessel segmentation in color images of the retina," *IEEE Transactions on Medical Imaging*, vol. 23, pp. 501-509, 2004.

[22] J.Gomes y L. Velho, Image Processing for Computer Graphics, Springer, New York, 1997.

[23] R. Hirata, M. Brun, J. Barrera y E.R. Dougherty, "Aperture filters: theory, application, and multiresolution analysis," en *Advances in Nonlinear Signal and Image Processing*, Hindawi Publishing Corporation ed., pp. 15-16, 2006.

[24] J. Barrera y M. Brun, "Translation invariant transformations of discrete random sets," *Anais do XI SIBGRAPI; International Symposium on Computer Graphics, Image Processing, and Visio*n, 1998.
[25] J. Barrera, G.J.F. Banon y E.R. Dougherty, "Automatic design of morphological operators," en *Space, Structure and Randomness*, Springer, pp. 257-278, 2005.
[26] P. Maragos y A Representation, "Theory for morphological image and signal processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 586-599, 1989.
[27] J. Barrera y G.J.F. Banon, "Expressiveness of the morphological language," en *Image Algebra and Morphological Image Processing III*, vol.1769, pp.264-274, 1992.
[28] H.J.A.M. Heijmans y C. Ronse, "The algebraic basis of mathematical morphology - part I: dilations and erosions," *Computer Vision, Graphics and Image Processing*, vol. 50, pp. 245-295, 1990.
[29] H.J.A.M. Heijmans, Morphological Image Operators, Academic Press, Boston, 1994.
[30] M.M. Fraz, P. Remagnino, A. Hoppe, B. Uyyanonvara y A.R. Rudnicka, "Blood vessel segmentation methodologies in retinal images - a survey," *Computer Methods and Programs in Biomedicine*, vol. 108, pp. 407-433, 2012.
[31] L. Devroye, L. Györfi y G. Lugosi, A Probabilistic Theory of Pattern Recognition, Springer-Verlag, New York, 1996.
[32] M.E. Benalcázar, M. Brun y V.L. Ballarin, "Artificial neural networks applied to statistical design of window operators," *Pattern Recognition Letters*, vol. 34, pp. 970-979, 2013.
[33] M.E. Benalcázar, M. Brun, V.L. Ballarin y R.M. Hidalgo "Automatic design of ensembles of window operators for ocular image segmentation," *IEEE Latin America Transactions*, vol.12, pp.935-941, 2014.
[34] J. Barrera, E.R. Dougherty y M. Brun, "Hybrid human-machine binary morphological operator design. An independent constraint approach," *Signal Processing*, vol. 80, pp.1469-1487, 2000.
[35] Y. Abu-Mostafa, M. Magdon-Ismail y H.T. Lin, Learning from data, AML Press, Pasadena, 2012.
[36] K.P. Murphy, Machine Learning a Probabilistic Perspective, MIT, 2012.
[37] R.O. Duda, P.E. Hart y D.G. Stork, Pattern Classification, Wiley, New York, 2001.
[38] C. Bishop, Pattern Recognition and Machine Learning, Springer, Cambridge, 2006.
[39] M. Mohri, A. Rostamizadeh y A. Talwalkar, Foundations of Machine Learning, MIT, 2012.
[40] R. Hirata, M. Brun, J. Barrera y E.R. Dougherty, "Multiresolution design of aperture filters," *Mathematical Imaging and Vision*, vol. 16, pp. 199-222, 2002.
[41] E.R. Dougherty y J. Barrera, "Pattern recognition theory in nonlinear signal processing," *Mathematical Imaging and Vision*, vol. 16, pp. 181-197, 2002.
[42] R. Feynman, QED The Strange Theory of Light and Matter, Princeton University Press, Princeton, 1985.
[43] E.R. Dougherty "On the epistemological crisis in genomics," en *Current Genomics*, vol. 9, pp. 69-79, 2008.
[44] M.L. Bittner y E.R. Dougherty, "Newton, Laplace, and the epistemology of systems biology," en *Cancer Informatics*, vol. 5, pp. 185-190, 2012.

CAPÍTULO II

Métodos de Diseño Automático de W-operadores: Revisión y Análisis

Resumen: En este capítulo se realiza una revisión bibliográfica y un análisis exhaustivo de los métodos de diseño automático de W-operadores propuestos en la literatura científica. Se inicia con un breve resumen de la teoría desarrollada en el capítulo anterior y la descripción de los dos métodos no paramétricos más utilizados para el diseño de W-operadores: la regla plug-in y la regla *k*NN. Posteriormente se analizan, en el contexto de W-operadores, las implicaciones de los teoremas de no free lunch para optimización y aprendizaje computacional supervisado. Luego se analiza la representación computacional y morfológica de la función característica de un W-operador. También se describen y analizan los métodos de diseño basados en la representación morfológica. A continuación se formula el diseño de W-operadores bajo la definición de restricciones. Finalmente, se describen y analizan las restricciones propuestas en la literatura científica. Se presta especial atención al tipo de imágenes al que se aplican los métodos propuestos y los tamaños de ventana para los que son factibles desde el punto de vista computacional.

2.1. Formulación del Problema de Diseño Automático de W-operadores

En esta sección se realiza un breve resumen de la formulación del problema de diseño de W-operadores presentado en el capítulo anterior. Este recuento se presenta para facilitar el análisis que se realiza a lo largo de este capítulo.

La función característica óptima para el caso donde las imágenes observadas son imágenes en escala de grises y las imágenes ideales toman valores de un conjunto de etiquetas $\mathcal{Y}$ es

$$\psi_{\text{opt}}(\mathbf{X}) = argmax_{i\in\mathcal{Y}}\, \mathbb{P}(Y = i|\mathbf{X}), \tag{2.1}$$

donde $\mathbb{P}(Y = i|\mathbf{X})$ es la probabilidad condicional de que la variable aleatoria Y tome el valor $i\in\mathcal{Y}$ dada la configuración de ventana $\mathbf{X}\in\boldsymbol{\mathcal{X}}$. $\boldsymbol{\mathcal{X}}$ es el espacio de todas las posibles configuraciones que se pueden observar a través de una ventana W. Esta definición de la función característica $\psi(\mathbf{X})$ es la que minimiza el error

$$\varepsilon(\psi) = \sum_{\forall \mathbf{X}\in\boldsymbol{\mathcal{X}}}\left(\sum_{\forall i\in\mathcal{Y}} \mathbb{I}(\psi(\mathbf{X}) \neq i)\mathbb{P}(Y = i \mid \mathbf{X})\right)\mathbb{P}(\mathbf{X}), \tag{2.2}$$

donde $\mathbb{P}(\mathbf{X})$ denota la probabilidad con la que se observa la configuración $\mathbf{X}$ e $\mathbb{I}(\psi(\mathbf{X}) \neq i)$ es la función indicador igual a 1 si se cumple que $\psi(\mathbf{X}) \neq i$ y 0 en el caso contrario.

Para el caso donde las imágenes observadas son imágenes en escala de grises y las imágenes ideales son imágenes binarias, la función característica se define como

$$\psi(\mathbf{X}) = \begin{cases} 1 & \text{si } \mathbb{P}(Y = 1 \mid \mathbf{X}) \geq \tau \\ 0 & \text{en el caso contrario} \end{cases}, \tag{2.3}$$

donde $\psi(\mathbf{X})$ se obtiene a partir de la definición de error

$$\varepsilon(\psi) = \sum_{\{\mathbf{X}:\, \psi(\mathbf{X}) = 1\}} \mathbb{P}(Y = 0 \mid \mathbf{X})\mathbb{P}(\mathbf{X}) + \sum_{\{\mathbf{X}:\, \psi(\mathbf{X}) = 0\}} \mathbb{P}(Y = 1 \mid \mathbf{X})\mathbb{P}(\mathbf{X}). \tag{2.4}$$

En la ecuación 2.4, si $\tau = 0.5$ entonces la función característica resultante es la función óptima ψ_{opt} para la distribución conjunta $\Pr(\mathbf{X},Y)$. Sin embargo, en aplicaciones prácticas el diseñador puede seleccionar un valor de umbral diferente a 0.5 en base a un balance entre

las tasa de falsos positivos y la tasa de falsos negativos de la función característica resultante.

Dado que generalmente se desconoce la distribución conjunta Pr($\mathbf{X}$,Y), en la práctica el diseño, o entrenamiento, de W-operadores se realiza en base a un conjunto de N ejemplos de entrenamiento $\mathcal{D} = \{(\mathbf{X}_i, freq(\mathbf{X}_i, Y = 0), ..., freq(\mathbf{X}_i, Y = |\mathcal{Y}| - 1))\}$, donde se asume que cada par ($\mathbf{X}$,Y) es generado a partir de Pr($\mathbf{X}$,Y) de manera independiente. El término $|\mathcal{Y}|$ denota el número de elementos, o cardinal, del conjunto $\mathcal{Y}$. Usando un algoritmo de aprendizaje computacional supervisado $\mathcal{A}$: $\mathcal{D} \rightarrow \psi$ y el conjunto $\mathcal{D}$, se obtiene el operador ψ_N, donde el subíndice N enfatiza el hecho de que este operador depende del conjunto $\mathcal{D}$. Usualmente, ψ_N se selecciona de tal manera que minimice el error empírico sobre los elementos del conjunto $\mathcal{D}$:

$$\varepsilon_N(\psi_N) = (1/N) \sum_{i=1}^{N} \mathbb{I}(\psi_N(\mathbf{X}_i) \neq Y_i). \tag{2.5}$$

El error real $\varepsilon(\psi_N)$ del operador diseñado ψ_N en función tanto del costo de estimación o diseño $\Delta(\psi_N,\psi_{opt})$ como del error $\varepsilon(\psi_{opt})$ del operador óptimo ψ_{opt} para la distribución conjunta Pr($\mathbf{X}$,Y) está dado por:

$$\varepsilon(\psi_N) = \Delta(\psi_N,\psi_{opt}) + \varepsilon(\psi_{opt}). \tag{2.6}$$

El error esperado de diseñar W-operadores en base al algoritmo $\mathcal{A}$ y conjuntos de N ejemplos de entrenamiento $\mathcal{D}$ está dado por:

$$\mathbb{E}[\varepsilon(\psi_N)] = \mathbb{E}[\Delta(\psi_N,\psi_{opt})] + \varepsilon(\psi_{opt}), \tag{2.7}$$

donde la esperanza $\mathbb{E}$ se calcula con respecto a la distribución que genera los conjuntos de entrenamiento $\mathcal{D}$ con N tuplas.

Para un problema determinado, el error $\varepsilon(\psi_{opt})$ del operador óptimo ψ_{opt} es una cantidad constante que depende únicamente de la distribución conjunta Pr($\mathbf{X}$,Y). Esto se evidencia a través del ejemplo 2.1. Esta distribución depende a su vez de la estructura y comportamiento del par de procesos estocásticos, conjuntos, y estacionarios ($\mathbf{O}$,$\mathbf{I}$) que generan las imágenes observadas e ideales. Por otro lado, el error $\varepsilon(\psi_N)$ del operador estimado ψ_N depende también de la complejidad del algoritmo de aprendizaje $\mathcal{A}$ y la cantidad y calidad de ejemplos de entrenamiento disponibles. Aquí el término calidad se refiere a qué tan bien reflejan los elementos de $\mathcal{D}$ a la distribución conjunta Pr($\mathbf{X}$,Y). Idealmente, con una cantidad infinita de ejemplos de entrenamiento y usando un algoritmo de aprendizaje consistente se podría obtener el operador ψ_N con un bajo, o incluso nulo, costo de diseño $\Delta(\psi_N,\psi_{opt})$, con lo cual $\varepsilon(\psi_N) \rightarrow \varepsilon(\psi_{opt})$. Sin embargo, en la práctica la cantidad de ejemplos de entrenamiento disponibles es finita y limitada, y en su lugar una tarea clave dentro del diseño automático de W-operadores es elegir un algoritmo que permita obtener un operador ψ_N con un bajo costo de diseño o estimación, lo cual a su vez permite que el error verdadero $\varepsilon(\psi_N)$ de ψ_N sea también bajo.

Ejemplo 2.1. Sea el modelo Gausiano clásico para la generación de los pares $(\mathbf{X},Y)$. Este modelo lo constituyen dos distribuciones Gausianas multivariable $\mathcal{N}(\boldsymbol{\mu}_0,\Sigma_0)$ y $\mathcal{N}(\boldsymbol{\mu}_1,\Sigma_1)$ para las probabilidades condicionales *a priori* $\mathbb{P}(\mathbf{X}|Y = 0)$ y $\mathbb{P}(\mathbf{X}|Y = 1)$, respectivamente. Se considera que las probabilidades marginales son iguales: $\mathbb{P}(Y = 0) = \mathbb{P}(Y = 1) = 0.5$. Las medias $\boldsymbol{\mu}_0$ y $\boldsymbol{\mu}_1$ de las Gausianas están localizadas en $\alpha\mathbf{a}$ y $-\alpha\mathbf{a}$, con $\alpha > 0$, respectivamente, donde $\mathbf{a} = (a_1,a_2,\ldots,a_n)$ y $\|\mathbf{a}\| = 1$, donde $\|\mathbf{a}\|$ denota la norma del vector $\mathbf{a}$. Para las covarianzas se asume que $\Sigma_0 = \Sigma_1 = \mathbf{I}$, donde $\mathbf{I}$ es la matriz identidad de $n \times n$ elementos. Se asume también que los vectores n-dimensionales $\mathbf{X}$ se generan luego de discretizar el espacio $\mathbb{R}^n$ en hipercubos de lado r. En estas condiciones, se puede mostrar que el error $\varepsilon(\psi_{\text{opt}})$ del operador óptimo ψ_{opt} es $\varepsilon(\psi_{\text{opt}}) = 1 - \Phi(\alpha)$, donde $\Phi(\alpha)$ es la función de distribución normal estándar acumulada. En la situación extrema donde las Gausianas se solapan completamente, $\alpha = 0$, se tiene que $\varepsilon(\psi_{\text{opt}}) = 0.5$. Para el caso contrario, donde las dos Gausianas están completamente separadas, $\alpha \to \infty$, el error del operador óptimo tiende a ser nulo: $\varepsilon(\psi_{\text{opt}}) \to 0$. En la Figura 2.1 se presenta una gráfica de la variación del error $\varepsilon(\psi_{\text{opt}})$ de ψ_{opt} en función del valor del parámetro α que controla la distancia entre las Gausianas $\mathcal{N}(\boldsymbol{\mu}_0,\Sigma_0)$ y $\mathcal{N}(\boldsymbol{\mu}_1,\Sigma_1)$.

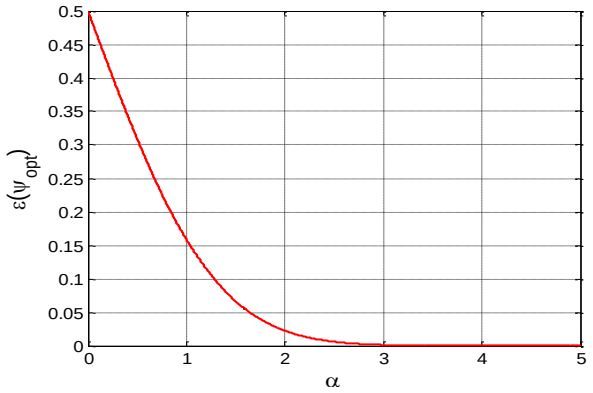


Figura 2.1. Error $\varepsilon(\psi_{\text{opt}})$ del W-operador óptimo ψ_{opt} para el modelo Gausiano clásico compuesto por dos distribuciones normales estándar en función de la distancia entre sus medias.

2.2. Algoritmo Aleatorio

En esta sección se analiza la performance del peor algoritmo que se puede diseñar para un problema determinado: el algoritmo aleatorio. Para el caso general donde las imágenes observadas e ideales son imágenes en escala de grises, si la estimación de las probabilidades condicionales $\mathbb{P}(Y = i|\mathbf{X})$ se realiza de manera aleatoria, entonces el error del operador diseñado es $\varepsilon(\psi_N) = (|\mathcal{Y}| - 1)/|\mathcal{Y}|$. En este caso se asume una distribución uniforme entre los elementos de $\mathcal{Y}$ y $|\mathcal{Y}|$ denota el número de niveles de gris de las imágenes ideales. En general, se deben evitar las situaciones donde el error del operador diseñado es muy cercano, igual, o peor que el error del algoritmo aleatorio. Para el caso particular donde las

imágenes observadas son imágenes en escala de grises y las imágenes ideales son imágenes binarias, el error del algoritmo aleatorio es $\varepsilon(\psi_N) = 0.5$. En este caso, al igual que en el caso anterior, se deben evitar las situaciones donde el error del operador diseñado es mayor o igual que el error del algoritmo aleatorio.

2.3. Regla Plug-in

En esta parte se presenta un análisis teórico y experimental de una de las reglas no paramétrica más utilizada para el diseño de W-operadores: la regla plug-in.

2.3.1. Análisis Teórico

La regla plug-in consiste en primero estimar las probabilidades condicionales $\mathbb{P}(Y|\mathbf{X})$ para cada configuración de ventana $\mathbf{X}\in\boldsymbol{\mathcal{X}}$ y todos los valores de $Y\in\mathcal{Y}$. Luego se reemplazan estos valores estimados en las ecuaciones 2.1 o 2.3 para determinar el valor de la función característica ψ para la configuración de ventana $\mathbf{X}$. En inglés este procedimiento se denomina *plug-in* y de aquí proviene el nombre de esta regla. La estimación del valor de la probabilidad condicional $\mathbb{P}(Y = i|\mathbf{X})$, con $i = 0,\ldots,|\mathcal{Y}| - 1$, se realiza en base al conjunto de ejemplos de entrenamiento $\boldsymbol{\mathcal{D}} = \{(\mathbf{X}_j,freq(\mathbf{X}_j,Y = 0),...,freq(\mathbf{X}_j,Y = |\mathcal{Y}| - 1))\}$, o tabla de frecuencias para el problema en cuestión, con $j = 1,\ldots,N$. Para esto se calcula el cociente entre la frecuencia $freq(\mathbf{X},Y = i)$ y el número total de veces que $\mathbf{X}$ fue observada en las imágenes de entrenamiento:

$$\hat{\mathbb{P}}(Y = i \mid \mathbf{X}) = freq(\mathbf{X},Y = i) / \sum_{j=0}^{|\mathcal{Y}|-1} freq(\mathbf{X},Y = j). \qquad (2.8)$$

Nótese que en este caso el número total de configuraciones de ventana cuyas probabilidades condicionales se deben estimar es $|L_1|^n$, recordando que L_1 es el rango, o conjunto de niveles de gris, de las imágenes observadas, y $n = |W|$ es el tamaño de la ventana W. El número total de probabilidades condicionales a estimar es $(|\mathcal{Y}| - 1)|L_1|^n$. Para cada configuración $\mathbf{X}$ se estiman $(|\mathcal{Y}| - 1)$ valores de probabilidades condicionales en lugar de $|\mathcal{Y}|$ valores debido a que $\sum_{i=0}^{|\mathcal{Y}|-1}\hat{\mathbb{P}}(Y = i \mid \mathbf{X}) = 1$. Adicionalmente, para la regla plug-in se tiene que $\hat{\mathbb{P}}(Y \mid \mathbf{X}) \rightarrow \mathbb{P}(Y|\mathbf{X})$ cuando $freq(\mathbf{X},Y) \rightarrow \infty\ \forall\mathbf{X}\in\boldsymbol{\mathcal{X}}$ [Devroye *et al.*, 1996].

Sin embargo, en la práctica el número de imágenes de entrenamiento es limitado, lo que implica que N es una cantidad pequeña con respecto al total de probabilidades a estimar. Nótese que esto no necesariamente significa que N es computacionalmente pequeño. Por lo tanto, conseguir una buena estimación de las probabilidades condicionales $\mathbb{P}(Y|\mathbf{X})$ es una tarea difícil a nivel estadístico y computacional tal como se evidencia a través del ejemplo 2.2. Lo usual es que, en las imágenes de entrenamiento, sólo una pequeña porción del número total de configuraciones se observe unas pocas veces, y en casos más extremos incluso sólo una vez. Esto implica a su vez que las frecuencias $freq(\mathbf{X},Y = i)$ tienden a cero. Más crítico aún es el caso de las configuraciones de ventana que no fueron observadas, pues en este caso no se dispone de ninguna información para estimar $\mathbb{P}(Y|\mathbf{X})$. Ante esta

situación, una heurística consiste en asignar a la configuración **X** no observada en el entrenamiento el nivel de gris con mayor probabilidad marginal $\mathbb{P}(Y)$.

Ejemplo 2.2. Para analizar la complejidad estadística y computacional del uso de la regla plug-in sea un problema de segmentación binaria de imágenes de 1000×1000 píxeles con 256 niveles de gris. Para esto se asume que una ventana cuadrada W de $n = 7\times7$ píxeles permite capturar suficiente información para segmentar los objetos de interés. En este caso, el número total de configuraciones de ventana (tamaño del espacio $\mathcal{X}$) para las cuales se debe estimar el valor de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ es $256^{7\times7} \approx 10^{118}$. En perspectiva, la complejidad estadística de este problema implica manejar una cantidad total de configuraciones que es aproximadamente 10^{36} veces más grande que el número de átomos en todo el universo observable. A nivel computacional, asumiendo que cada píxel de las imágenes a procesar resulta en una configuración **X** distinta, el número de imágenes necesario para observar todas las 10^{118} posibles configuraciones que componen $\mathcal{X}$ es 10^{112}. Peor aún, para tener una buena estimación de $\mathbb{P}(Y = 1|\mathbf{X})$ se requiere que $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$ sean cantidades relativamente grandes, con lo cual el número de imágenes necesarias para esta tarea es mucho mayor que 10^{112}.

Para una ventana W de tamaño $n = |W|$ e imágenes observadas binarias, el espacio $\mathcal{X}$ está formado por un total de 2^n posibles configuraciones. Si se considera el caso particular donde el error del operador óptimo ψ_{opt} es cero, $\varepsilon(\psi_{opt}) = 0$, y las configuraciones **X** se observan con probabilidad uniforme $\mathbb{P}(\mathbf{X}) = 1/2^n$, se puede mostrar que el error promedio de la regla plug-in para diseñar W-operadores a partir de conjuntos de entrenamiento con N tuplas está acotado inferiormente por

$$\mathbb{E}[\varepsilon(\psi_N)] \geq (1 - (1/2^n))^N. \tag{2.9}$$

donde la esperanza $\mathbb{E}$ se calcula respecto a la distribución subyacente a la generación de conjuntos de entrenamiento de N tuplas [Devroye *et al.*, 1996]. Para este caso, si $N \leq 2^{n-1}$ entonces la desigualdad 2.9 resulta en $\mathbb{E}[\varepsilon(\psi_N)] \geq 0.5$, que implica un comportamiento peor que el del algoritmo aleatorio. Esto evidencia que, para ventanas grandes y conjuntos de imágenes de entrenamiento pequeños, la calidad de la estimación de las probabilidades condicionales usando la regla plug-in es mala si no se utiliza una adecuada estrategia de generalización.

Para el caso donde las imágenes ideales son binarias, se puede demostrar que el costo de diseño de la regla plug-in está acotado superiormente para la distribución $\Pr(\mathbf{X},Y)$ por:

$$\Delta(\psi_N,\psi_{opt}) \leq 2\mathbb{E}[|\,\mathbb{P}(Y=1|\,\mathbf{X}) - \hat{\mathbb{P}}(Y=1|\,\mathbf{X})\,|], \tag{2.10}$$

donde la esperanza $\mathbb{E}$ se calcula con respecto a la distribución marginal de **X**. Este resultado indica que, en teoría, se podría hacer que el costo de diseño de la regla plug-in sea arbitrariamente pequeño en la medida en que se disponga de un conjunto de ejemplos de entrenamiento suficientemente grande. La regla plug-in es *consistente* para la distribución $\Pr(\mathbf{X},Y)$, lo que implica que $\Delta(\psi_N,\psi_{opt}) \rightarrow 0$ cuando $N \rightarrow \infty$ [Devroye *et al.*, 1996].

Además, para la regla plug-in el valor esperado del costo de diseño o estimación $\mathbb{E}[\Delta(\psi_N,\psi_{opt})]$ es no incremental: $\mathbb{E}[\Delta(\psi_N,\psi_{opt})] \leq \mathbb{E}[\Delta(\psi_{N+1},\psi_{opt})]$. Esto significa que el costo de diseño de la regla plug-in nunca empeora con el incremento de la cantidad de ejemplos de entrenamiento para cualquier distribución conjunta Pr(**X**,*Y*) [Braga-Neto y Dougherty, 2005].

A nivel práctico, la regla plug-in es útil para diseñar W-operadores para procesamiento de imágenes binarias siempre y cuando el problema en cuestión permita el uso de *ventanas pequeñas* que implica un tamaño menor que 3×3 píxeles. Para el caso de imágenes en escala de grises, la regla plug-in es aplicable siempre que las imágenes a procesar contengan pocos niveles de gris, por ejemplo 5 o 10 niveles de gris, y las ventanas requeridas para el procesamiento sean pequeñas. Para problemas que demanden el uso de *ventanas de mediano tamaño* que contienen un número entre 5×5 y 7×7 píxeles, y *ventanas de gran tamaño* que contienen un número mayor que 9×9 píxeles, el uso de esta regla resulta completamente impráctico. Esto debido a que la cantidad de probabilidades condicionales a estimar es extremadamente grande y la cantidad de datos de entrenamiento es fija y limitada para obtener buenas estimaciones.

2.3.2. Análisis Empírico usando Imágenes con Ruido Sintético

En esta sección se analiza la variación de la performance de W-operadores diseñados en base a la regla plug-in en función de la cantidad de ejemplos de entrenamiento y la complejidad del problema en cuestión. La tarea de los W-operadores aquí diseñados es filtrar ruido puntual sintético aditivo y sustractivo uniforme añadido a imágenes oculares binarias segmentadas manualmente.

Base de Datos DRIVE: Para este experimento se utilizaron las imágenes de la base de datos pública DRIVE. Esta base de datos contiene 40 imágenes color de angiografías retinales, o imágenes oculares, con sus respectivas imágenes de gold estándar. Estas últimas son imágenes binarias conteniendo únicamente los vasos sanguíneos de las imágenes color. Para las imágenes oculares de esta base de datos, la segmentación de los vasos sanguíneos fue realizada manualmente con la ayuda de oftalmólogos [Staal *et al.*, 2004]. Los 40 pares de imágenes que forman esta base de datos están divididos en dos subconjuntos: uno para entrenamiento y el otro para testeo. Cada uno de estos subconjuntos contiene 20 pares de imágenes, donde las imágenes color están en el formato TIFF y las imágenes binarias están en el formato GIF. Tanto las imágenes color como las imágenes binarias están compuestas por 584×565 píxeles. Cada canal RGB de las imágenes color contiene 256 niveles de intensidad. La segmentación de los vasos sanguíneos en imágenes de angiografías retinales, o imágenes oculares, provee información importante tanto para aplicaciones médicas como para la implementación de sistemas biométricos de control de acceso [Fraz *et al.*, 2012].

A nivel médico, una angiografía retinal es un examen diagnóstico que usa cámaras de fondo para evaluar la estructura anatómica de la retina. Previo a este examen, se administran gotas oculares que hacen dilatar la pupila para luego tomar fotografías del interior del ojo. Después de tomar el primer grupo de imágenes se inyecta un tinte especial, llamado fluoresceína, dentro de una vena en la región del codo. Posteriormente, la cámara de fondo toma fotografías del interior del ojo a medida que el tinte va pasando a través de sus vasos sanguíneos. Este examen se utiliza para detectar el crecimiento anómalo de nuevos vasos o una inadecuada circulación sanguínea en la retina. También se utiliza para diagnosticar y evaluar la efectividad de un tratamiento para problemas oculares como

hemorragias, exudados, y esclerosis. Estos problemas aparecen a su vez como consecuencia de enfermedades como la diabetes, hipertensión arterial, y alteraciones cardiacas [Staal *et al.*, 2004], [Fraz *et al.*, 2012].

Algunos sistemas biométricos de control de acceso basan su funcionamiento en el hecho de que la morfología del árbol arterial retiniano es única para cada persona [Fraz *et al.*, 2012]. Un componente importante de estos sistemas constituye la segmentación de los vasos sanguíneos de las imágenes oculares de los usuarios de dicho sistema. Una diferencia importante de esta aplicación con respecto a las aplicaciones médicas es que en este caso el procedimiento de escaneo ocular no es invasivo; es decir, no se requiere la inyección de ningún tinte. Tampoco se requiere el uso de fluidos que dilaten la retina previa a la adquisición de la imagen ocular. Más detalles del uso y características de las imágenes de angiografías retinales, o imágenes oculares, se presentarán en los siguientes capítulos, donde se abordará el problema de segmentación de los vasos sanguíneos oculares.

Para la etapa entrenamiento de los W-operadores se trabajó directamente con 7 imágenes gold estándar (21, 26, 27, 29, 35, 36 y 40) seleccionadas aleatoriamente a partir del subconjunto de entrenamiento de la base de datos DRIVE. Para testeo se utilizaron 5 imágenes gold estándar (10, 11, 13, 19 y 20) seleccionadas también aleatoriamente a partir del subconjunto de testeo de esta base de datos. Este número de imágenes fue seleccionado con el objetivo de obtener buenas estimaciones de las probabilidades condicionales y los errores de los operadores diseñados con una moderada complejidad computacional para el experimento.

Para generar las imágenes observadas se añadió ruido puntual sintético aditivo y sustractivo de manera uniforme a las imágenes ideales de entrenamiento y testeo. Las densidades del ruido puntual añadido son 5, 10, 15 y 20 por ciento. Nótese que la complejidad del problema aumenta a medida que se incrementa el nivel de densidad del ruido añadido a cada imagen. En base a los 7 pares de imágenes de entrenamiento se diseñaron W-operadores para eliminar, o filtrar, el ruido sintético agregado. Para esta tarea se utilizaron ventanas de 3×3 y 5×5 píxeles que son tamaños adecuados para el filtrado de ruido puntual. La complejidad estadística y computacional de la regla plug-in depende, en este caso, del número de píxeles que forman la ventana utilizada para estimar el valor de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$. Para el problema en cuestión, $\mathbb{P}(Y = 1|\mathbf{X})$ cuando se ha observado la configuración $\mathbf{X}$ en la ventana centrada en ese píxel.

La etapa de entrenamiento, o aprendizaje, de este experimento consta de dos partes. Para cada densidad de ruido, en la primera parte se escanea cada píxel de las imágenes observadas e ideales de entrenamiento con cada ventana fijada para este experimento. En base a este escaneo se obtiene una tabla de frecuencias en la que se registran las diferentes configuraciones observadas $\mathbf{X}$ con sus correspondientes valores de frecuencia $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$. En la segunda parte se define el valor de la función característica ψ_N para cada configuración observada. Se tiene que $\psi_N(\mathbf{X}) = 1$ si $freq(\mathbf{X},Y = 1) \geq freq(\mathbf{X},Y = 0)$ y $\psi_N(\mathbf{X}) = 0$ en el caso contrario. Para las configuraciones de ventana no observadas se define la heurística $\psi_N(\mathbf{X}) = 1$. Este procedimiento es completamente equivalente a estimar el valor de $\mathbb{P}(Y = 1|\mathbf{X})$ (ecuación 2.5) y luego usar la ecuación 2.8, con $\tau = 0.5$, para obtener el

valor de $\psi_N(\mathbf{X})$. El valor de N se controla incrementando el tamaño de un rectángulo que define la región de escaneo en las imágenes de entrenamiento. Este rectángulo empieza en la esquina superior izquierda de las imágenes de entrenamiento. Para conseguir los valores de N aquí testeados se usaron una y dos imágenes de entrenamiento.

En la etapa de testeo se aplicó cada operador diseñado a las 5 imágenes observadas de testeo. El nivel de ruido usado para definir las imágenes observadas de testeo es igual al nivel usado para definir las imágenes observadas de entrenamiento. La performance de cada operador diseñado se obtuvo comparando las imágenes resultantes con sus respectivas imágenes ideales. El error de cada operador corresponde al porcentaje de píxeles de cada imagen donde difieren los valores de $\psi_N(\mathbf{X})$ e Y. Los errores que se reportan corresponden a los promedios de todos los errores calculados para cada una de las 5 imágenes de testeo.

Con respecto al costo de diseño $\Delta(\psi_N,\psi_{opt})$, en el gráfico de la Figura 2.2 se puede observar que, para ventanas de 3×3 píxeles y diferentes porcentajes de ruido, el error se estabiliza más rápidamente que para el operador diseñado en base a la ventana de 5×5 píxeles. Esto se debe a que la cantidad máxima de posibles configuraciones para los operadores diseñados es $2^9 = 512$ y de $2^{25} = 33.554$M, respectivamente. Por lo tanto, en este último caso hay mucha más diversidad de configuraciones para las cuales se debe estimar la probabilidad condicional (mayor costo de diseño). Por esta razón se necesita una mayor cantidad de puntos para completar dicho aprendizaje con respecto a la cantidad de puntos necesarios para el primer caso. Para ventanas de 3×3 píxeles, el aprendizaje se completa con aproximadamente 3×10^5 píxeles de entrenamiento. Es decir, un sólo par de imágenes (observada e ideal) es suficiente para completar el entrenamiento. Para ventanas de 5×5 píxeles es evidente que se requieren más de dos imágenes para completar el entrenamiento. Consecuentemente, también se incrementa el costo computacional de la regla plug-in usando ventanas de 5×5 píxeles en comparación con ventanas de 3×3 píxeles.

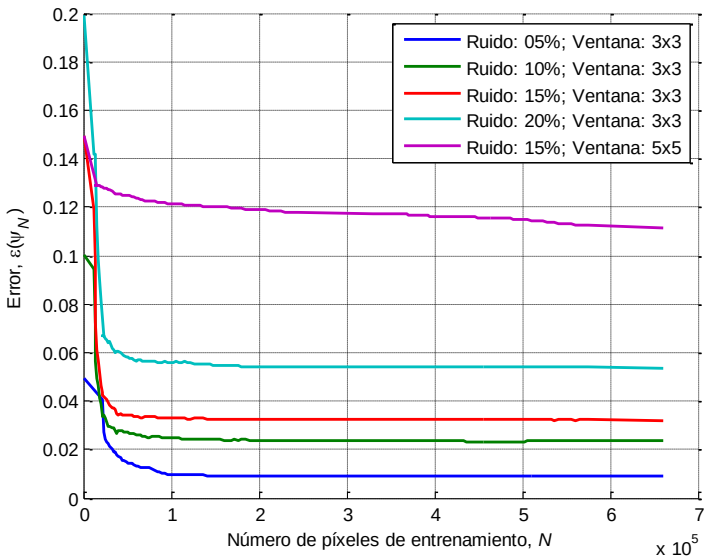


Figura 2.2. Variación del error de W-operadores diseñados en base a la regla plug-in para filtrado de ruido puntual sintético aditivo y sustractivo de imágenes oculares binarias en función del número de píxeles de entrenamiento.

En esta parte se analiza la influencia de la complejidad del problema en cuestión, medido por el error de Bayes $\varepsilon(\psi_{opt})$, sobre el error del operador diseñado. Cabe aclarar que el valor error de Bayes es desconocido en este experimento, pero se sabe que aumenta a medida que se incrementa el porcentaje de error en las imágenes observadas. En la Figura 2.2 también se puede observar que, para ventanas de 3×3 píxeles, los errores medidos a partir de 3×10^5 puntos, y aproximadamente constantes, son más grandes para cuando se incrementa el nivel de ruido en las imágenes observadas. Esto se debe a que un aumento en el nivel de ruido produce también un incremento en el valor de $\varepsilon(\psi_{opt})$. Siguiendo con el análisis para las ventanas de 3×3 píxeles, a partir de valores de $N \geq 10^5$, se puede asumir que el costo de diseño $\Delta(\psi_N,\psi_{opt})$ es constante. En estas condiciones, se comprueba empíricamente lo predicho por la ecuaciones 2.6 y 2.7: un incremento de $\varepsilon(\psi_{opt})$ produce también un incremento de $\varepsilon(\psi_N)$ para un costo de diseño $\Delta(\psi_N,\psi_{opt})$ constante.

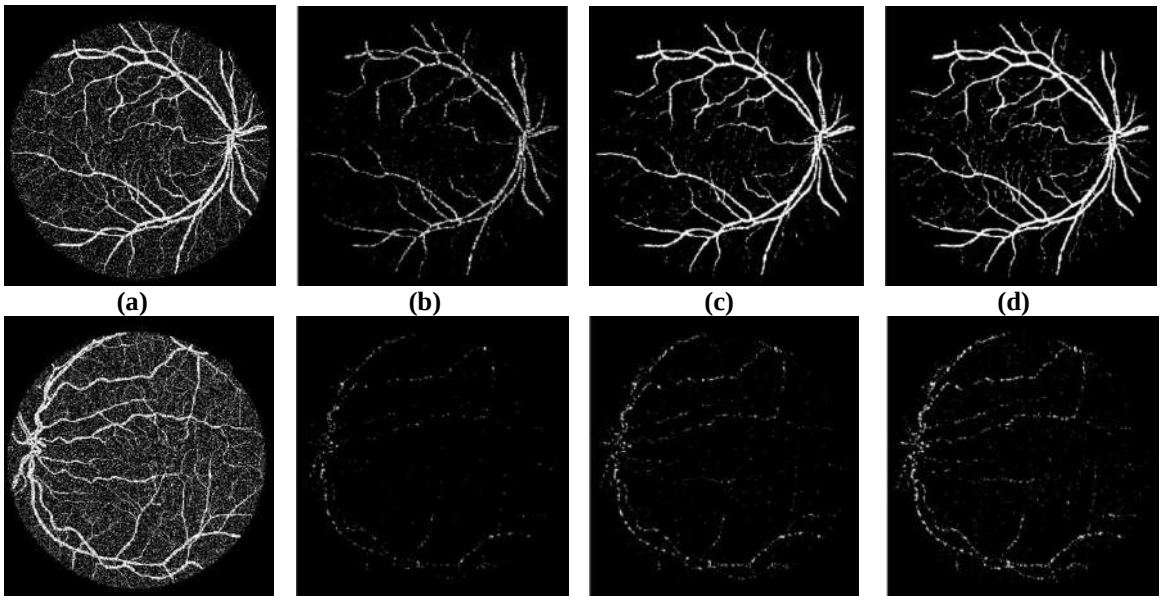


Figura 2.3. Resultados de filtrar ruido puntual sintético aditivo y sustractivo en una imagen binaria ocular con W-operadores diseñados en base a la regla plug-in. **(a)** Imagen observada con una densidad de ruido sintético de 10%. Resultados usando una ventana de 3×3 píxeles y (**b)** 1.75×10^4 (error = 0.0240), **(c)** 3.17×10^4 (error = 0.0236), y **(d)** 6.60×10^5 (error = 0.0234) píxeles de entrenamiento. (**e**) Imagen observada con una densidad de ruido sintético de 15%. Resultados usando una ventana de 5×5 píxeles y (**f)** 8.25×10^4 (error = 0.1223), **(g)** 3.30×10^5 (error = 0.1174), y **(h)** 6.60×10^5 (error = 0.1117) píxeles de entrenamiento.

A manera de ejemplo, en la Figura 2.3 se puede apreciar visualmente el efecto de aplicar W-operadores obtenidos a partir de diferentes cantidades de píxeles de entrenamiento para ventanas de 3×3 y 5×5 píxeles. En estas imágenes se puede notar que los operadores diseñados a partir de ventanas 3×3 píxeles son capaces de eliminar la mayor parte del ruido contenido en la imagen observada. Este efecto aumenta a medida que se incrementa la cantidad de píxeles de entrenamiento. Sin embargo, incluso cuando el diseño se realiza usando 2 imágenes de entrenamiento (Figura 2.3-d), el operador resultante, aparte de eliminar el ruido, también elimina partes de los vasos sanguíneos delgados. Esto causa a su vez que para los vasos sanguíneos delgados se pierda la conectividad entre sus píxeles. Para disminuir este efecto indeseado, se podrían usar ventanas más grandes para el diseño de los W-operadores. Sin embargo, las imágenes de las Figuras 2.3-f a h muestran que el uso de ventanas más grandes, como por ejemplo 5×5 píxeles, requiere de una gran cantidad de

ejemplos de entrenamiento para conseguir buenos resultados. En este caso, ante una cantidad limitada de ejemplos de entrenamiento, los operadores diseñados con ventanas de 5×5 píxeles eliminan no sólo los vasos sanguíneos delgados, sino también partes de los vasos sanguíneos gruesos, produciendo un efecto totalmente contrario al deseado.

2.4. Regla *k*NN

En esta parte se realiza un análisis teórico y experimental de la regla de diseño no paramétrica *k*NN. Es importante mencionar que esta regla no es exclusiva para el diseño de W-operadores, al igual que tampoco lo es la regla plug-in.

2.4.1. Análisis Teórico

El nombre de esta regla proviene del inglés *k-nearest neighbors* o en español *k*-vecinos más cercanos. Aquí se define el valor de la función característica para una configuración de ventana **X** en base a una votación entre las frecuencias de sus *k*-vecinos más cercanos pertenecientes al conjunto de ejemplos de entrenamiento. Formalmente, dado el conjunto de entrenamiento $\mathcal{D} = \{(\mathbf{X}_j, freq(\mathbf{X}_j, Y = 0), ..., freq(\mathbf{X}_j, Y = |\mathcal{Y}| - 1))\}$, con $j = 1,\ldots,N$, el valor de ψ_N para una configuración $\mathbf{X} \in \mathcal{X}$ se obtiene estimando el valor de $\mathbb{P}(Y = i|\mathbf{X})$ mediante la siguiente relación:

$$\hat{\mathbb{P}}(Y = i \mid \mathbf{X}) = \sum_{j=1}^{N} \alpha_j \left(freq(\mathbf{X}_j, Y = i) / \sum_{u=0}^{|\mathcal{Y}|-1} freq(\mathbf{X}_j, Y = u) \right), \tag{2.11}$$

donde $i = 1,\ldots,|\mathcal{Y}| - 1$. Para el caso de los pesos α_j, se tiene que $\alpha_j = 1/k$ si $\mathbf{X}_j$ es uno de los k vecinos más cercanos a la configuración **X** o $\alpha_j = 0$ para el caso contrario, con $j = 1,\ldots,N$. Luego se reemplazan los valores estimados de las probabilidades condicionales en las ecuaciones 2.1 o 2.3 para definir el valor de ψ_N para **X**. Para el caso particular donde $k = 1$, la regla *k*NN toma el nombre de *regla* 1NN.

En lo relacionado al costo de diseño de la regla 1NN se puede demostrar que

$$lim_{N \to \infty} \mathbb{E}[\Delta(\psi_N, \psi_{opt})] \leq \varepsilon(\psi_{opt}), \tag{2.12}$$

para cualquier distribución conjunta Pr(**X**,*Y*) [Cover y Hart, 1967]. Por lo tanto, el costo de diseño asintótico esperado para la regla 1NN es pequeño si el error $\varepsilon(\psi_{opt})$ del operador óptimo ψ_{opt} para Pr(**X**,*Y*) es también pequeño. Para el caso donde $k > 1$, con k impar, el valor esperado del costo de diseño de la regla *k*NN está acotado superiormente por

$$lim_{N \to \infty} \mathbb{E}[\Delta(\psi_N, \psi_{opt})] \leq 1/(ke)^{1/2}, \tag{2.13}$$

para cualquier distribución conjunta Pr(**X**,*Y*), donde *e* es la constante neperiana [Devroye *et al.*, 1996]. Esto implica que, para la regla *k*NN el costo de diseño puede ser reducido arbitrariamente para un valor de k suficientemente grande. Adicionalmente, cuando $k \to \infty$ y $k/N \to 0$ con $N \to \infty$, la regla *k*NN es *universalmente consistente*. Esto implica que $\mathbb{E}[\Delta(\psi_N, \psi_{opt})] \to 0$ para cualquier distribución conjunta Pr(**X**,*Y*).

Para encontrar los k vecinos más cercanos a una configuración $\mathbf{X}$ es necesario definir una métrica de distancia d. En base a esta métrica se ordenan los elementos del conjunto $\mathcal{D}$ en orden decreciente de d. Posteriormente, y previo a la votación, se asigna un peso $\alpha = 1/k$ a las k primeras tuplas de este conjunto ordenado; mientras que las $N - k$ tuplas restantes toman un peso $\alpha = 0$. Es usual para kNN el uso de la distancia euclidiana definida como $d(\mathbf{X},\mathbf{X}') = \sqrt{\sum_{i=1}^{n}(X_i - X'_i)^2}$ para el caso donde las imágenes observadas son imágenes en escala de grises o color. Para el caso de imágenes binarias se puede utilizar la distancia de Hamming definida como $d(\mathbf{X},\mathbf{X}') = \sum_{i=1}^{n}\mathbb{I}(X_i \neq X'_i)$.

Para aplicaciones prácticas de diseño de W-operadores usando la regla kNN otro aspecto clave, aparte de la selección del tamaño de la ventana W, es la selección del valor de k. Cuando $N \to \infty$ se espera que $freq(\mathbf{X},Y) \to \infty \ \forall Y \in \mathcal{Y}$. Por lo tanto, en este caso, la regla 1NN sería suficiente para obtener una muy buena estimación de las probabilidades condicionales para definir la función característica. Desafortunadamente, ante un escenario con una cantidad finita y limitada de imágenes de entrenamiento, la selección de k se debe realizar de manera heurística. Lo usual es asignar a k un valor que sea función del tamaño N del conjunto de entrenamiento $\mathcal{D}$: $k = k(N)$. Las heurísticas más usadas son $k = \sqrt{N}$ para valores de N pequeños y moderados o $k = log_{10}(N)$ si N es grande [Duda *et al.*, 2001].

***Maldición de la Dimensión*:** Para el caso donde se diseñan W-operadores binarios (imágenes de entrada y salida binarias) con valores de k y tamaños de ventana pequeños y conjuntos de entrenamiento suficientemente grandes, la regla kNN puede producir buenos resultados tal como se evidencia más adelante a través de un ejemplo práctico. Sin embargo, para el caso donde se requieren ventanas medianas o grandes y/o las imágenes a procesar son imágenes en escala de grises, la regla kNN puede enfrentar los efectos negativos de la denominada *maldición de la dimensión* (o en inglés *curse of dimensionality*).

Para explicar esta maldición, se considera un ejemplo proporcionado en [Murphy, 2012] (página 18). Sea $\mathcal{X}$ un hypercubo h_1 de volumen 1, dentro del cual se asume que las configuraciones de ventana, de dimensión n, están uniformemente distribuidas. Dada una nueva configuración $\mathbf{X}$, sus probabilidades condicionales $\mathbb{P}(Y|\mathbf{X})$ se estiman haciendo crecer otro hypercubo h_2, cuyo centro se encuentra en $\mathbf{X}$, hasta contener una fracción k/N de configuraciones de entrenamiento. La longitud esperada de un lado de h_2 está dada por $a_n(k/N) = (k/N)^{1/n}$. Si la ventana W tiene un tamaño de $n = 7\times7$ píxeles, y la estimación de $\mathbb{P}(Y|\mathbf{X})$ para $\mathbf{X}$ se basa en un 10% de las configuraciones del conjunto de entrenamiento ($k/N = 0.1$), entonces $a_n(k/N) = 0.9541$. Esto implica que la longitud del hypercubo h_2 se debe extender un 95.41% a lo largo de cada dimensión alrededor de $\mathbf{X}$ para definir una vecindad que contenga el 10% del conjunto de ejemplos de entrenamiento. Si $k/N = 0.001$ (0.1% del total de datos del conjunto de entrenamiento), entonces $a_n(k/N) = 0.8685$ (la longitud del hypercubo h_2 se debe extender un 86.85% en cada dimensión alrededor de $\mathbf{X}$).

El anterior ejemplo evidencia que, para ventanas de moderado y gran tamaño, la estimación de $\mathbb{P}(Y|\mathbf{X})$ implica usar configuraciones que están muy alejadas de $\mathbf{X}$. Esto a su vez podría

causar que la estimación de $\mathbb{P}(Y|\mathbf{X})$ sea imprecisa, con lo cual el resultado del operador diseñado a partir de estas estimaciones, especialmente cuando Pr($\mathbf{X}$,Y) tiene variación no uniforme, será malo. Aplicaciones prácticas evidencian que la regla *k*NN puede ser útil para resolver problemas que involucren el uso de ventanas pequeñas con imágenes observadas e ideales binarias [Kim, 2000], [Kim y Barreto, 2000].

2.4.2. Análisis Empírico usando Imágenes con Ruido Sintético

El objetivo de esta sección es analizar la influencia de la variación de *k* en el desempeño de los W-operadores diseñados con la regla *k*NN para un conjunto de ejemplos de entrenamiento fijo (*N* fijo). Para esto se consideran nuevamente las imágenes binarias oculares de los subconjuntos de entrenamiento y testeo de la base de datos DRIVE. A diferencia del experimento para el análisis de la regla plug-in, en este caso se utiliza un único nivel de ruido puntual uniforme equivalente al 10%. Se diseñan W-operadores usando la regla *k*NN para filtrar el ruido puntual artificial introducido en las imágenes binarias oculares. Las imágenes observadas e ideales para entrenamiento y testeo aquí usadas son las mismas utilizadas en el caso anterior. La métrica de distancia utilizada para encontrar los *k* vecinos más cercanos a una configuración de ventana es la distancia de Hamming. Adicionalmente, también se muestran datos referentes al costo computacional: tiempo de entrenamiento de W-operadores usando la regla *k*NN. El procedimiento aquí usado para el entrenamiento y testeo es similar al procedimiento usado para el experimento de la regla plug-in. La única diferencia consiste en que, para el entrenamiento de cada W-operador, se utilizaron todos los píxeles (329960 píxeles) de cada imagen observada de entrenamiento.

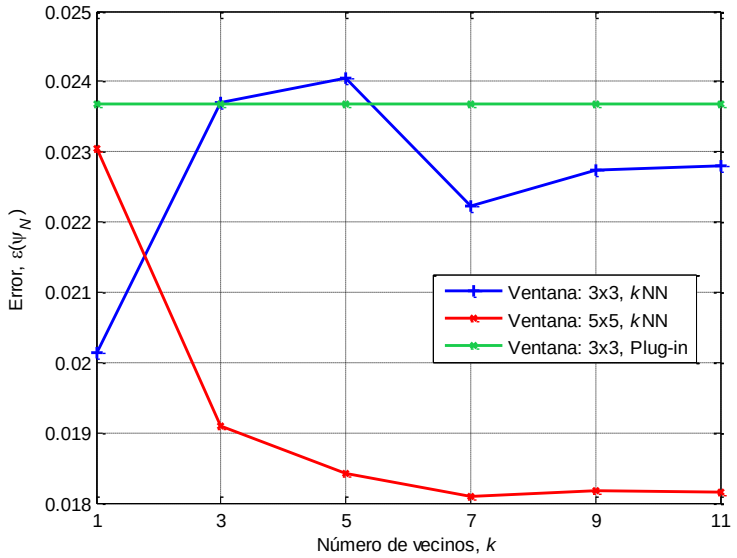


Figura 2.4. Variación del error de W-operadores diseñados usando la regla *k*NN para filtrado de ruido puntual sintético aditivo y sustractivo de imágenes oculares binarias en función del valor de *k* y comparación con los resultados de la regla plug-in.

En la Figura 2.4 se presentan los resultados obtenidos sobre la influencia del valor de *k* en el desempeño de un operador diseñado con la regla *k*NN. Para la ventana de 3×3 píxeles, el mejor resultado se da cuando $k = 1$ (regla 1NN). Para valores de *k* mayores a 1, el error de los operadores aumenta de manera variable, siendo para el caso de $k = 3$ muy similar al error promedio de los W-operadores diseñados en base a la regla plug-in. Para la ventana de

5×5 píxeles, el desempeño de los W-operadores diseñados usando la regla 1NN es bastante similar al resultado de los operadores diseñados usando la regla plug-in con una ventana de 3×3 píxeles. A medida que se incrementa el valor de k, el error de los operadores para la ventana de 5×5 píxeles decrece de manera aproximadamente exponencial y luego empieza a crecer ligeramente a partir de $k = 7$. Aquí el desempeño de los operadores diseñados usando ventanas de 5×5 píxeles y la regla kNN, con $k > 1$, es mejor que el desempeño de los operadores diseñados usando ventanas de 3×3 píxeles en base las reglas kNN y plug-in.

Estos resultados evidencian que, para espacios de configuraciones pequeños como el que se genera cuando se usan ventanas de 3×3 píxeles (512 posibles configuraciones) y un conjunto grande de ejemplos de entrenamiento relativo al tamaño de $\mathcal{X}$, el aumento de k deteriora el desempeño de los operadores diseñados en base a kNN. Esto se debe a que, para la estimación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$, se utilizan configuraciones que están muy alejadas de la configuración de testeo $\mathbf{X}$. Dicho de otro modo, la estimación de $\mathbb{P}(Y = 1|\mathbf{X})$ deja de ser local a medida que k aumenta, situación que va en contra del principio de localidad de kNN.

Por otro lado, el buen desempeño de 1NN para un espacio de configuraciones $\mathcal{X}$ pequeño (ventana de 3×3 píxeles) y un conjunto grande de ejemplos de entrenamiento, relativo al tamaño de $\mathcal{X}$, se debe a que generalmente el vecino más cercano a una configuración de testeo $\mathbf{X}$ es la misma configuración $\mathbf{X}$ o una configuración muy cercana en términos de la distancia de Hamming. Dado que $\mathcal{X}$ es pequeño, para cada configuración del conjunto de entrenamiento se dispone de valores de frecuencias $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$ que son suficientemente grandes para estimar el valor de $\mathbb{P}(Y = 1|\mathbf{X})$ con una adecuada precisión. Por lo tanto, este resultado evidencia que, para espacios de configuraciones pequeños, la regla 1NN es un buen candidato para el diseño de W-operadores.

Para el caso de espacios de configuraciones grandes, relativo al tamaño del conjunto de ejemplos de entrenamiento, la situación cambia de manera drástica en comparación con el caso anteriormente analizado. Para ventanas de 5×5 píxeles, el espacio $\mathcal{X}$ está compuesto por alrededor de 33.55 millones de posibles configuraciones. Hay que tener presente que la cantidad de ejemplos de entrenamiento usada para este experimento es de 0.33 millones de píxeles. En estas condiciones, el aumento de k (hasta un valor de 7) se manifiesta en una mejora notable en la precisión de la estimación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$. En otras palabras, se reduce el costo de diseño a medida que k aumenta según indica la ecuación 2.13. Consecuentemente y dado que la complejidad del problema no cambia, es decir $\varepsilon(\psi_{opt})$ se mantiene constante, el desempeño de los operadores diseñados también mejora. Para $k > 7$, se empieza a evidenciar levemente el efecto de la pérdida de localidad de la regla kNN.

En esta parte se analiza, de manera empírica, el costo computacional de diseño de W-operadores usando la regla kNN. En la Figura 2.5 se presenta un gráfico de la variación del tiempo de procesamiento, o testeo, promedio por imagen en función del valor de k. Para referencia, se anota que el experimento se realizó usando un computador con procesador

Intel core i-5, con velocidad de CPU de 2.30 GHz, y 8GB de memoria RAM. Los tiempos de entrenamiento de cada operador, no incluidos en el gráfico, son aproximadamente 10 y 30 minutos para las ventanas de 3×3 y 5×5 píxeles, respectivamente. Para el caso de la ventana de 3×3 píxeles se puede observar que el tiempo de procesamiento se mantiene aproximadamente constante a medida que se incrementa *k*. Por otro lado, para la ventana de 5×5 píxeles el tiempo de procesamiento por imagen se incrementa con el aumento de *k*. En general, el tiempo de procesamiento con ventanas de 5×5 píxeles es más grande que el tiempo de procesamiento usando ventanas de 3×3 píxeles. Esto se debe a que en el primer caso la cantidad de distancias a calcular y ordenar para encontrar los *k* vecinos más cercanos de una configuración es más grande que para el caso de 3×3 píxeles.

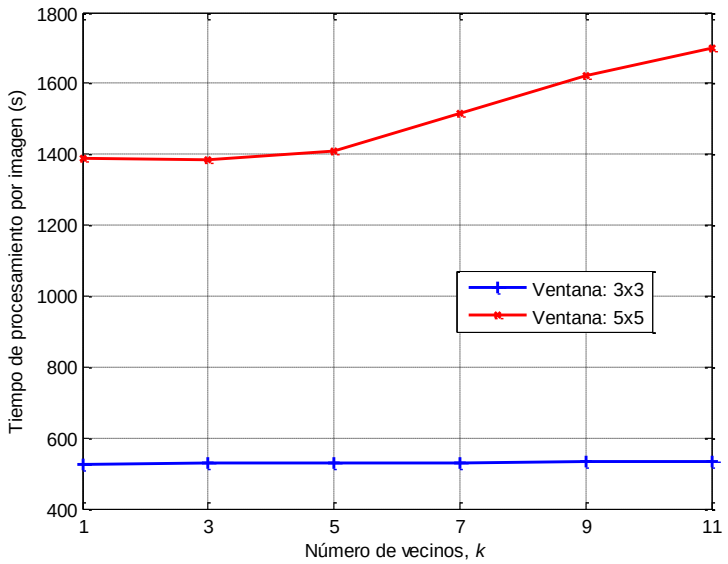


Figura 2.5. Variación del tiempo de procesamiento de W-operadores diseñados con la regla *k*NN para filtrado de ruido puntual sintético aditivo y sustractivo de imágenes oculares binarias en función del valor de *k*.

En esta parte se reportan los tiempos totales de procesamiento para los mejores resultados, en términos de error, anteriormente analizados. El tiempo total para entrenar y aplicar un W-operador usando la regla 1NN, con una ventana de 3×3 píxeles, para una única imagen de entrenamiento y una única imagen de testeo es de aproximadamente 20 minutos. Para la regla 7NN, con una ventana de 5×5 píxeles, el tiempo total es de aproximadamente 1 hora. Estos valores deben ser tomados únicamente como referencia, pues el tiempo de procesamiento depende de varios factores: recursos computacionales disponibles (cantidad de memoria y velocidad del procesador), eficiencia en la implementación del código, lenguaje de programación, número de tareas simultáneas que ejecuta el procesador, entre otros. Estos valores muestran la razón práctica por la que no se realizaron experimentos usando ventanas más grandes que las testeadas o imágenes oculares en escala de grises o color.

En la Figura 2.6 se muestran, a modo de ejemplo, las imágenes resultantes de la aplicación de un W-operador diseñado usando la regla *k*NN con una ventana de 3×3 píxeles y valores de $k = 1$ (Figura 2.6-c), $k = 3$ (Figura 2.6-d), $k = 5$ (Figura 2.6-e), $k = 7$ (Figura 2.6-f), $k = 9$ (Figura 2.6-g) y $k = 11$ (Figura 2.6-h). Las imagen observada y la imagen gold estándar se muestran en la Figura 2.6-a y -b, respectivamente. Al igual que para el caso de la regla plug-in con ventanas de 3×3 píxeles (Figura 2.3, imágenes b hasta la d), los W-operadores

basados en *k*NN son capaces filtrar la mayoría del ruido presente en el árbol arterial principal (vasos sanguíneos gruesos). Sin embargo, para el caso de los vasos sanguíneos delgados, existe una pérdida de conectividad entre sus píxeles, pues algunos de ellos son reconocidos como ruido, y por consiguiente, eliminados de las imágenes resultantes. Adicionalmente, también se puede observar que queda un remanente de ruido en el fondo de las imágenes procesadas, el cual es mayor para los valores de k = 7, 9, y 11 en comparación con los valores de k = 1, 3, y 5.

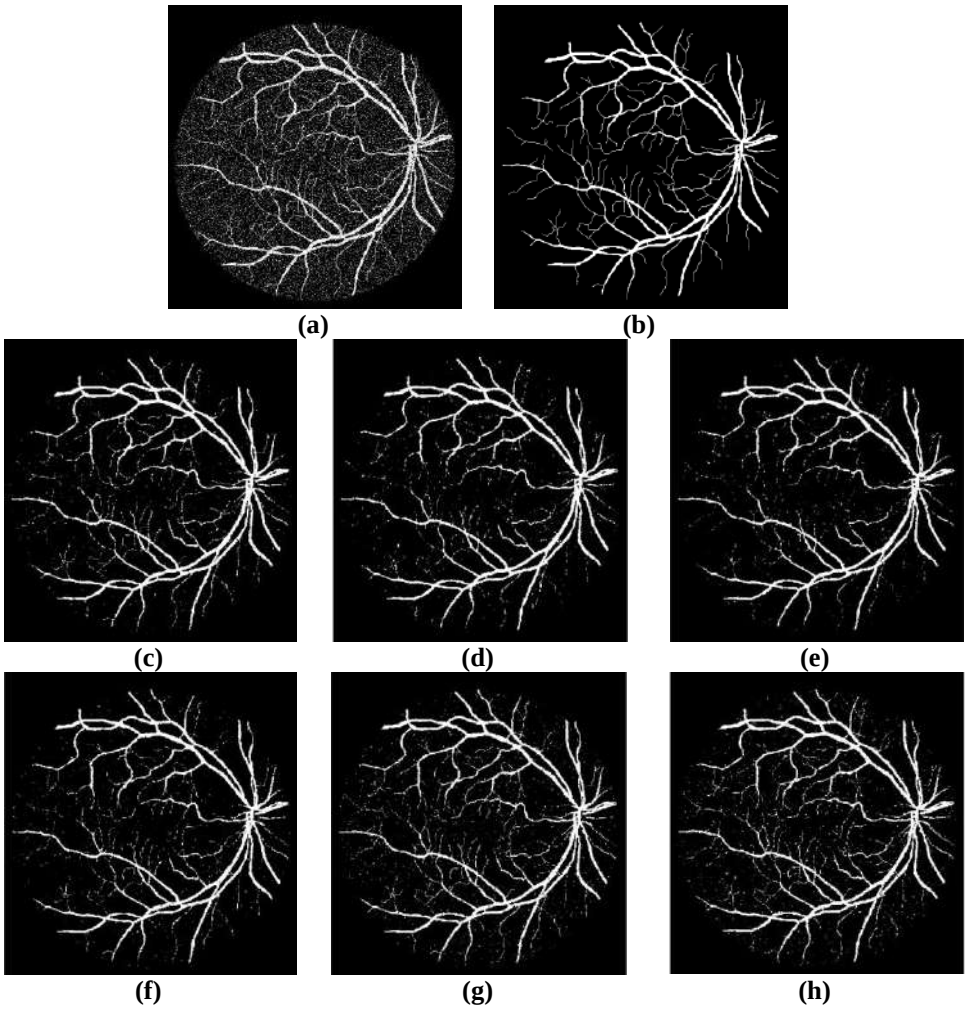


Figura 2.6. Resultados de filtrar ruido puntual sintético aditivo y sustractivo en una imagen binaria ocular con W-operadores diseñados en base a la regla *k*NN. **(a)** Imagen observada con una densidad de ruido de 10% y (**b**) imagen ideal. Resultados usando una ventana de 3×3 píxeles y valores de *k*: (**c)** 1 (error = 0.0214), **(d)** 3 (error = 0.0242), **(e)** 5 (error = 0.0262), (**f**) 7 (error = 0.0235), **(g)** 9 (error = 0.0238), y **(h)** 11 (error = 0.0248).

En la Figura 2.7 se presentan los resultados obtenidos usando una ventana de 5×5 píxeles y las mismas imágenes observada e ideal de la Figura 2.6. Comparando las imágenes de la Figura 2.7 (imágenes b hasta la f) con las imágenes de Figura 2.6 (imágenes c hasta la h), se puede observar que los W-operadores diseñados en base a ventanas de 5×5 píxeles retornan imágenes con menor proporción de ruido en el fondo. Sin embargo, en este caso los operadores tampoco son capaces de mantener intactos los vasos sanguíneos delgados durante el filtrado del ruido.

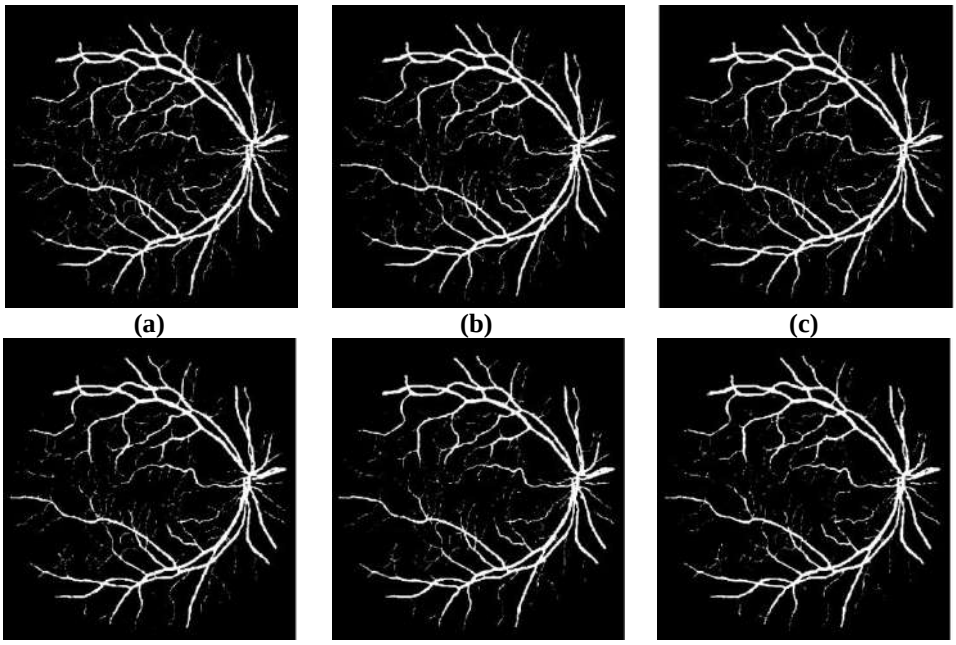


Figura 2.7. Resultados de filtrar ruido puntual sintético aditivo y sustractivo en una imagen binaria ocular con W-operadores diseñados en base a la regla *k*NN usando una ventana de 5×5 píxeles y valores de *k*: (**a)** 1 (error = 0.0245), **(b)** 3 (error = 0.0214), **(c)** 5 (error = 0.0210), (**d**) 7 (error = 0.0210), **(e)** 9 (error = 0.0210), y **(f)** 11 (error = 0.0215).

En esta parte se comparan visualmente los resultados de W-operadores diseñados usando la regla plug-in (Figura 2.8, imágenes i a k) con los resultados en base a la regla *k*NN para ventanas de 5×5 píxeles y un nivel ruido del 15%. Las imágenes observada e ideal se muestran en las Figuras 2.8-a y -b, respectivamente. Las imágenes c a h muestran los resultados de usar valores de k = 1 a 11, respectivamente. Estos resultados muestran claramente que para el problema de filtrado de ruido en cuestión, la regla *k*NN proporciona mejores resultados con una menor cantidad de datos de entrenamiento. Una vez más se anota que el árbol arterial principal se preserva durante el filtrado de ruido, pero los vasos sanguíneos delgados son eliminados de las imágenes resultantes.

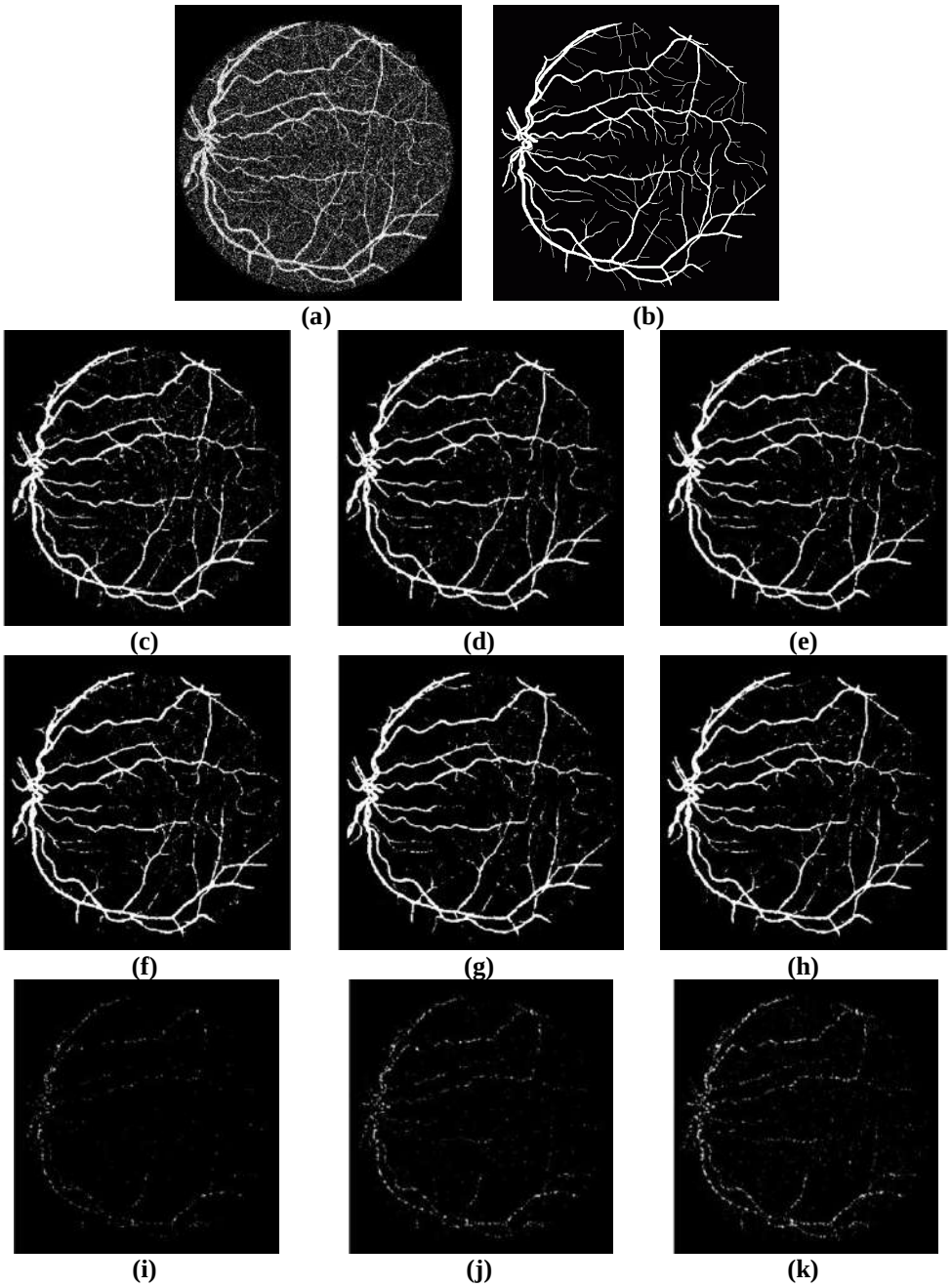


Figura 2.8. Resultados de filtrar ruido puntual sintético aditivo y sustractivo en una imagen binaria ocular con W-operadores diseñados en base a la regla *k*NN y plug-in. **(a)** Imagen observada con una densidad de ruido sintético de 15% y (**b**) imagen ideal. Resultados usando la regla *k*NN, una ventana de 5×5 píxeles, 3.30×10^5 píxeles de entrenamiento, y valores de *k*: (**c**) 1 (error = 0.0431), **(d)** 3 (error = 0.0386), **(e)** 5 (error = 0.0364), (**f**) 7 (error = 0.0368), **(g)** 9 (error = 0.0371), y **(h)** 11 (error = 0.0365). Resultados usando la regla plug-in, una ventana de 5×5 píxeles, y (**i**) 8.25×10^4 (error = 0.1223), **(j)** 3.30×10^5 (error = 0.1174), y **(k)** 6.60×10^5 (error = 0.1117) píxeles de entrenamiento.

2.5. ¿Existe un Método de Diseño Intrínsecamente Mejor que Otro?

Antes de proseguir con la revisión de métodos de diseño automático de W-operadores, se plantea la pregunta de si es posible que alguno de ellos sea mejor que los demás en términos de performance o error y dejando de lado, por un momento, su costo computacional. La respuesta a esta pregunta es *no* tanto para W-operadores como en general para problemas de aprendizaje computacional supervisado y problemas de

búsqueda u optimización. El sustento matemático para estas afirmaciones está dado por los denominados *teoremas de no free lunch* presentados en [Wolpert, 1996] y [Wolpert y Macready, 1997], respectivamente.

Para problemas de aprendizaje computacional supervisado estos teoremas asumen como criterio de evaluación del desempeño de un algoritmo el costo 0-1. Para problemas de búsqueda u optimización se consideran espacios de búsqueda finitos o discretos y no se admiten revisitas de puntos ya explorados. Según los teoremas de no free lunch, para todos los problemas de aprendizaje supervisado/optimización que cumplan con las condiciones establecidas *no existe el algoritmo óptimo*. Esto implica que si $\mathcal{F}$ denota el conjunto de todos los posibles problemas de aprendizaje supervisado/optimización, el hecho de que un algoritmo funcione bien para un conjunto de problemas $\mathcal{F}_1 \subset \mathcal{F}$ necesariamente se compensa con una mala performance sobre el conjunto de problemas restantes $\mathcal{F}$ - $\mathcal{F}_1$.

Para aprendizaje computacional supervisado, el foco de esta tesis, lo anterior involucra que *el desempeño promedio de todos los algoritmos, evaluado mediante el costo 0-1, sobre todos los posibles problemas es el mismo*. En otras palabras, dados dos algoritmos $\mathcal{A}_1$ y $\mathcal{A}_2$, el hecho de que $\mathcal{A}_1$ muestre buena performance sobre un conjunto de problemas, en los que $\mathcal{A}_2$ falla, no implica una inherente superioridad de $\mathcal{A}_1$ sobre $\mathcal{A}_2$. Los teoremas de no free lunch garantizan que en este caso habrá otro conjunto de problemas para los que se cumpla el caso contrario, donde $\mathcal{A}_2$ es mejor que $\mathcal{A}_1$. Por lo tanto, el desempeño promedio de $\mathcal{A}_1$ y $\mathcal{A}_2$ sobre todo el conjunto de problemas posibles es el mismo. Sorprendentemente, esto también aplica para el caso donde uno de los algoritmos, ya sea $\mathcal{A}_1$ o $\mathcal{A}_2$, es el algoritmo aleatorio. En cierto modo, los teoremas de no free lunch para aprendizaje computacional supervisado u optimización se asemejan, por ejemplo, a las leyes de conservación de la energía, momento, entre otras, de la Física. Sin embargo, hay casos y métricas de evaluación de error para los cuales estos teoremas no aplican. Estos casos son denominados *casos de free lunch* [Wolpert y Macready, 2005], [Auger y Teytaud, 2010], [Yang, 2012]**.**

De manera más formal, sea el espacio discreto de configuraciones de ventana $\boldsymbol{\mathcal{X}}$, el conjunto finito $\mathcal{Y}$, y una función determinística f: $\boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$ tal que $Y = f(\mathbf{X})$ o una distribución probabilística f: $\boldsymbol{\mathcal{X}} \rightarrow [0,1]$ tal que $Y = argmax_{Y \in \mathcal{Y}} f(Y|\mathbf{X})$. En estas condiciones, los teoremas de no free lunch establecen que, independientemente de la distribución de $\mathbf{X}$ y el número de puntos de entrenamiento N, todos los algoritmos de aprendizaje supervisado que estiman Y, tienen el mismo desempeño promedio evaluado mediante cualquiera de los siguientes criterios: $\mathbb{E}[\varepsilon(\mathcal{A})|\boldsymbol{\mathcal{D}}]$, $\mathbb{E}[\varepsilon(\mathcal{A})|f,\boldsymbol{\mathcal{D}}]$, $\mathbb{E}[\varepsilon(\mathcal{A})|f,N]$ y $\mathbb{E}[\varepsilon(\mathcal{A})|N]$. Aquí $\mathbb{E}[\varepsilon(\mathcal{A})|\boldsymbol{\mathcal{D}}]$ es el error promedio de $\mathcal{A}$ calculado de manera uniforme sobre todas las distribuciones *a priori* $\Pr(f)$ y cualquier conjunto de entrenamiento $\boldsymbol{\mathcal{D}}$. $\mathbb{E}[\varepsilon(\mathcal{A})|f,\boldsymbol{\mathcal{D}}]$ es el error promedio de $\mathcal{A}$ calculado de manera uniforme sobre todas las funciones f y cualquier conjunto de entrenamiento $\boldsymbol{\mathcal{D}}$. $\mathbb{E}[\varepsilon(\mathcal{A})|f,N]$ es el error promedio de $\mathcal{A}$ calculado de manera uniforme sobre todas las

funciones f. $\mathbb{E}[\varepsilon(\mathcal{A})|N]$ es el error promedio de $\mathcal{A}$ calculado de manera uniforme sobre todas las distribuciones *a priori* Pr(f) [Wolpert, 1996], [Duda *et al.*, 2001].

En el párrafo anterior, el error $\varepsilon(\mathcal{A})$ del algoritmo de aprendizaje supervisado $\mathcal{A}$ se calcula usando el costo 0-1: $r(\psi_{\mathcal{A}}(\mathbf{X})) = \mathbb{I}(\psi_{\mathcal{A}}(\mathbf{X}) \neq Y)\ \forall(\mathbf{X},Y)\notin\boldsymbol{\mathcal{D}}$ con $\psi_{\mathcal{A}}\in\mathcal{A}$. En palabras, esto significa que el error $\varepsilon(\mathcal{A})$ se calcula únicamente a partir de los pares $(\mathbf{X},Y)$ no contenidos en el conjunto de entrenamiento $\boldsymbol{\mathcal{D}}$. Esto difiere con respecto al criterio asumido en esta tesis para el cálculo del error: independencia en el muestreo e idéntica distribución. Según este criterio es posible que exista un solapamiento entre el conjunto de entrenamiento y el conjunto de testeo. Dicho de otro modo, es posible que una configuración pueda estar contenida tanto el conjunto de entrenamiento como en el conjunto de testeo. Sin embargo, en la práctica para ventanas W de mediano y gran tamaño, el número de configuraciones que forman el espacio $\boldsymbol{\mathcal{X}}$ es muy grande relativo al tamaño N de $\boldsymbol{\mathcal{D}}$. Por lo tanto, es muy probable que el solapamiento entre el conjunto de entrenamiento y el conjunto de testeo sea mínimo o incluso nulo. En estas condiciones, *los teoremas de no free lunch aplican para el diseño automático de W-operadores*.

Para un determinado problema de aprendizaje supervisado sea la distribución condicional $\Pr(\psi|\boldsymbol{\mathcal{D}})$ de que $\mathcal{A}$ retorne la función ψ dado el conjunto de entrenamiento $\boldsymbol{\mathcal{D}}$. Para algoritmos determininísticos como kNN y plug-in, la distribución condicional $\Pr(\psi|\boldsymbol{\mathcal{D}})$ está concentrada en una única función ψ; es decir, $\mathbb{P}(\psi|\boldsymbol{\mathcal{D}}) = 1$ para ψ y 0 en los demás casos. Para algoritmos estocásticos, usualmente, la distribución $\Pr(\psi|\boldsymbol{\mathcal{D}})$ es distinta de cero para cada función ψ. $\Pr(f|\boldsymbol{\mathcal{D}})$ es la distribución condicional *a posteriori* de que f sea la función que genera los valores de Y a ser estimados usando $\mathcal{A}$ y dado el conjunto $\boldsymbol{\mathcal{D}}$. El buen desempeño de un algoritmo de aprendizaje computacional supervisado para un problema determinado está condicionado a que tan similares, o alineadas, estén las distribuciones condicionales $\Pr(\psi|\boldsymbol{\mathcal{D}})$ y $\Pr(f|\boldsymbol{\mathcal{D}})$ [Duda *et al.*, 2001], [Wolpert, 2002].

Los teoremas de no free lunch no involucran únicamente a los algoritmos de aprendizaje supervisado, sino también a las heurísticas o reglas de oro desarrolladas para mejorar la performance de dichos algoritmos [Wolpert, 1996], [Wolpert, 2002]. Por ejemplo, una regla de oro comúnmente utilizada en aprendizaje computacional es la denominada *navaja de Ocamm* (en inglés *Ocamm's razor*) [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012]. Según esta regla, si el desempeño de dos funciones características diseñadas en base a algoritmos diferentes es el mismo dado un conjunto de entrenamiento, entonces se debe elegir la función diseñada en base al algoritmo más simple. Para estos casos, y otros similares, los teoremas de no free lunch garantizan que éstas técnicas no presentan un beneficio efectivo cuando se aplican sobre todo el conjunto de problemas posibles. Sin embargo, esto no implica, en ningún modo, que dichas técnicas o heurísticas no sean útiles para problemas puntuales. Por el contrario, para un problema determinado estas técnicas pueden ayudar a alinear la estructura del algoritmo $\Pr(\psi|\boldsymbol{\mathcal{D}})$ con la estructura del problema $\Pr(f|\boldsymbol{\mathcal{D}})$.

En base a todo lo anteriormente expuesto, no hay razón teórica para *a priori* tener predilección por uno u otro método de diseño. Sin embargo, la complejidad computacional también juega un papel importante, junto al error, a la hora de decidir qué métodos de diseño utilizar para resolver un problema determinado. Después de todo, nadie está dispuesto a esperar una semana o más para procesar una única imagen de un conjunto de, por ejemplo, de 10 o más imágenes. Adicionalmente, los teoremas de no free lunch son una de las razones que justifican el desarrollo e investigación de nuevos métodos de diseño que, en general, tendrán un desempeño promedio igual al de los métodos existentes, pero que para un problema determinado o conjunto de problemas particular pueden resultar muy útiles (a costa de un mal desempeño sobre otro conjunto de problemas).

Ejemplo 2.3. En este ejemplo se analiza uno de los casos de los teoremas de no free lunch. Sea $\mathcal{X}$ el espacio de todas las posibles configuraciones $\mathbf{X} = (X_1,X_2)$, donde cada X_i puede tomar uno de los valores del conjunto $\{0,1,2\}$ con $i = 1,2$. Adicionalmente, sea $\mathcal{Y} = \{0,1,2\}$. En la Tabla 2.1 se muestran los datos para un problema de aprendizaje supervisado, donde el conjunto $\mathcal{D}$ está formado por $N = 5$ pares de entrenamiento $(\mathbf{X},Y)$. Sean dos algoritmos de aprendizaje supervisado: $\mathcal{A}_1$ y $\mathcal{A}_2$. Las reglas de generalización de estos algoritmos son $\psi_1 = max(X_1,X_2)$ y $\psi_2 = min(X_1,X_2)$, respectivamente, para todas aquellas configuraciones no registradas en el entrenamiento.

Tabla 2.1. Ejemplo de los teoremas de no free lunch.

	(X_1,X_2)	$Y = f(\mathbf{X})$	ψ_1	ψ_2
entrenamiento $\mathcal{D}$	0 0	0	0	0
	0 1	1	1	1
	0 2	2	2	2
	1 0	1	1	1
	1 1	0	0	0
	1 2	1	1	1
testeo	2 0	2	2	0
	2 1	0	2	1
	2 2	2	2	2

Calculando el error, en base al costo 0-1, para los dos algoritmos sobre el conjunto de testeo se tiene que $\varepsilon(\mathcal{A}_1|f,\mathcal{D}) = 1/3$ y $\varepsilon(\mathcal{A}_2|\, f,\mathcal{D}) = 2/3$. Si se usa el error cuadrático medio como criterio de evaluación se tiene que $\varepsilon_{ECM}(\mathcal{A}_1|f,\mathcal{D}) = 4/3$ y $\varepsilon_{ECM}(\mathcal{A}_2|f,\mathcal{D}) = 5/3$. Por lo tanto, según estos dos criterios de error $\mathcal{A}_1$ es superior a $\mathcal{A}_2$ para el problema considerado. Sin embargo, en la práctica no se conoce la función f. Por lo tanto, una forma de comparar el desempeño de $\mathcal{A}_1$ y $\mathcal{A}_2$ es calculando su error promedio de manera uniforme sobre todas las funciones f cuyos valores de Y sean consistentes con los valores de Y que forman el conjunto de entrenamiento. Nótese que existe un total de $3^3 = 27$ casos consistentes con el conjunto de entrenamiento de la Tabla 2.1. Si se promedian los valores de $\varepsilon(\mathcal{A}_1|f,\mathcal{D})$ y $\varepsilon(\mathcal{A}_2|f,\mathcal{D})$ de los 27 casos consistentes con $\mathcal{D}$, entonces $\mathbb{E}[\varepsilon(\mathcal{A}_1)|f,\mathcal{D}] = \mathbb{E}[\varepsilon(\mathcal{A}_2)|f,\mathcal{D}] = 2/3$. Esto evidencia que los dos algoritmos tienen el mismo desempeño promedio en base al costo 0-1. Por otro lado, se tiene que $\mathbb{E}[\varepsilon_{ECM}(\mathcal{A}_1)|f,\mathcal{D}] = 5/3$ y $\mathbb{E}[\varepsilon_{ECM}(\mathcal{A}_2)|f,\mathcal{D}] = 4/3$. Esto muestra una superioridad de $\mathcal{A}_2$ sobre $\mathcal{A}_1$. Con este contraejemplo se muestra que los

teoremas de no free lunch no aplican para cuando el criterio de evaluación es el error cuadrático medio.

2.6. Representación Computacional y Morfológica de W-operadores: Núcleo y Base

La *representación computacional* de un W-operador tiene que ver con la implementación en hardware o software de su función característica. Para el caso de la regla plug-in, la función característica ψ de un W-operador $\mathbf{\Psi}$ se puede representar computacionalmente mediante el uso de una *tabla de búsqueda* $\mathcal{T} = \{(\mathbf{X}_1,\psi(\mathbf{X}_1)),...,(\mathbf{X}_N,\psi(\mathbf{X}_N))\}$. En la tabla de búsqueda se almacenan todas las distintas configuraciones $\mathbf{X}$ observadas en la etapa de entrenamiento con sus respectivos valores estimados de $\psi(\mathbf{X})$. Es conveniente recordar que $\psi(\mathbf{X})$ se define en base a las estimaciones de las probabilidades condicionales $\mathbb{P}(Y|\mathbf{X})$ y la regla de decisión presentada en las ecuaciones 2.1 o 2.3. Para facilitar, durante la etapa de testeo, la localización de las configuraciones en tabla de búsqueda se puede definir un determinado orden de indexamiento para los pares $(\mathbf{X},\psi_N(\mathbf{X}))$ dentro de $\mathcal{T}$. Para la regla *k*NN, la representación computacional de un W-operador se puede realizar a través de la tabla de frecuencias o conjunto $\mathcal{D}$ obtenido en la etapa de entrenamiento.

Para ventanas pequeñas es probable que una configuración se observe más de una vez en las imágenes de entrenamiento. En este caso el número de filas de la tabla de búsqueda y la tabla de frecuencias será pequeño relativo al total de píxeles escaneados. Por el contrario, para ventanas de mediano y gran tamaño es poco probable que una configuración se observe más de una vez. Consecuentemente, en este caso el número de filas de la tabla de búsqueda y la tabla de frecuencias será similar al total de píxeles contenidos en las imágenes de entrenamiento. Para este último caso, y problemas donde el tamaño de las imágenes a procesar y/o el número de imágenes de entrenamiento es grande, el tamaño de la tabla de búsqueda y la tabla de frecuencias será también grande. En estas condiciones, el costo computacional referido al tiempo de búsqueda de una configuración es estas tablas durante la etapa de testeo será muy alto. En algunos casos prácticos esto puede volver incluso intratable el uso de los métodos de diseño de W-operadores antes descritos.

Una forma de compactar la representación computacional de un W-operador es a través de su *representación morfológica* usando el núcleo, o *kernel*, de su función característica [Maragos y Schafer, 1987], [Banon y Barrera, 1991], [Banon y Barrera, 1993], [Heijmans, 1994], [Barrera y Brun, 1998], [Barrera *et al.*, 2005]. Para facilidad del análisis se considera una función característica binaria ψ: $\mathcal{X} \rightarrow \mathcal{Y}$, con dominio $\mathcal{X} = \{0,1\}^n$ y rango $\mathcal{Y} = \{0,1\}$. El *núcleo* $\mathcal{K}[\psi](u)$ de ψ, con $u \in \{0,1\}$, se define como el conjunto de todas las configuraciones pertenecientes a $\mathcal{X}$ cuyo valor de la función característica ψ sea u (ecuación 2.14).

$$\mathcal{K}[\psi](u) = \{\mathbf{X} \in \mathcal{X} \mid \psi(\mathbf{X}) = u\}. \tag{2.14}$$

Nótese que para $u = 1$ el núcleo $\mathcal{K}[\psi](1)$ de ψ engloba tanto a las configuraciones observadas y no observadas durante la etapa de entrenamiento con $\psi(\mathbf{X}) = 1$. Una situación similar se tiene para el caso donde $u = 0$. Por lo tanto, la definición de núcleo establece también una *generalización* del valor de la función característica. Esto es útil para aquellas configuraciones no observadas durante la etapa de entrenamiento, o lo que es lo mismo,

para aquellas configuraciones que no están registradas en la tabla de búsqueda $\mathcal{T}$ o la tabla de frecuencias $\mathcal{D}$ [Hirata *et al.*, 2006].

El núcleo $\mathcal{K}[\psi](1)$ de la función característica ψ puede ser también reducido, o compactado, a un colección de todos sus intervalos maximales $\mathcal{M}[\psi]$ [Maragos y Schafer, 1987], [Banon y Barrera, 1991], [Banon y Barrera, 1993], [Heijmans, 1994], [Barrera y Brun, 1998], [Dougherty y Barrera, 2002], [Barrera *et al.*, 2005]. Sean dos configuraciones de ventana $\mathbf{A},\mathbf{B}\in\mathcal{X}$ tales que $\mathbf{A} \leq \mathbf{B}$, donde la relación de orden está dada por la inclusión. Un *intervalo* [**A**,**B**] es un conjunto de configuraciones definido como

$$[\mathbf{A},\mathbf{B}] = \{\mathbf{X}\in\mathcal{X} \mid \mathbf{A} \leq \mathbf{X} \leq \mathbf{B}\}. \tag{2.15}$$

Dado un intervalo [**A**,**B**], un *operador sup-generador* $\lambda_{\mathbf{A},\mathbf{B}}$: $\mathcal{X} \rightarrow \mathcal{Y}$ es una función tal que

$$\lambda_{\mathrm{A,B}}(\mathbf{X}) = \begin{cases} 1 & \text{si } \mathbf{A} \leq \mathbf{X} \leq \mathbf{B} \\ 0 & \text{en el caso contrario} \end{cases}. \tag{2.16}$$

El intervalo [**A**,**B**] es un *intervalo maximal* dentro de una colección dada de intervalos si ningún otro intervalo dentro de dicha colección lo contiene. $\mathcal{M}[\psi]$ denota la colección de todos los intervalos maximales dentro de la colección de todos los intervalos de $\mathcal{K}[\psi](1)$.

A partir de lo anterior, una función característica ψ puede ser representada en función de un operador sup-generador $\lambda_{\mathbf{A},\mathbf{B}}$ que opera sobre la colección de intervalos maximales $\mathcal{M}[\psi]$ de su núcleo $\mathcal{K}[\psi](1)$:

$$\psi(\mathbf{X}) = sup\{\lambda_{\mathbf{A},\mathbf{B}}(\mathbf{X}) \mid [\mathbf{A},\mathbf{B}]\in\mathcal{M}[\psi]\}. \tag{2.17}$$

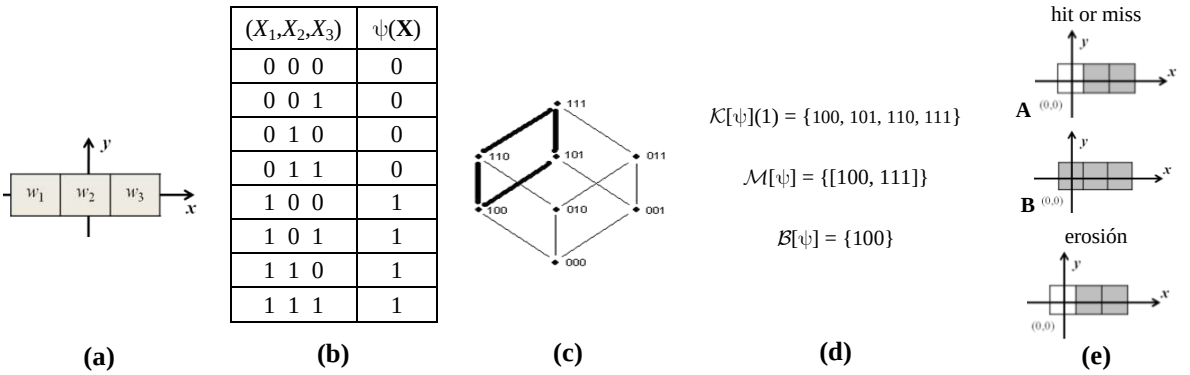
El supremo de la ecuación 2.17 es 1 si existe un intervalo $[\mathbf{A},\mathbf{B}]\in\mathcal{M}[\psi]$ que contiene a la configuración **X** y 0 en el caso contrario. De manera alternativa, un W-operador puede ser también representado como un supremo de operaciones hit or miss:

$$\psi(\mathbf{X}) = sup\{\mathbf{X}\otimes\mathbf{C} \mid \mathbf{C} = \{\mathbf{A},\sim\mathbf{B}\} \text{ y } [\mathbf{A},\mathbf{B}]\in\mathcal{M}[\psi]\}, \tag{2.18}$$

donde $\mathbf{X}\otimes\mathbf{C}$ denota la operación de hit or miss de **X** por el elemento estructurante compuesto $\mathbf{C} = \{\mathbf{A},\sim\mathbf{B}\}$ tal que $\mathbf{X}\otimes\mathbf{C} = (\mathbf{X}\ominus\mathbf{A})\cap(\sim\mathbf{X}\ominus\sim\mathbf{B})$. $\mathrm{X}\ominus\mathbf{A}$ denota la erosión de **X** por **A** que es igual a 1 si $\mathbf{A} \leq \mathbf{X}$ y 0 en el caso contrario [Matheron, 1975], [Serra, 1982]. De forma equivalente se define la erosión de $\sim\mathbf{X}$ por el elemento estructurante $\sim\mathbf{B}$, donde $\sim\mathbf{X}$ y $\sim\mathbf{B}$ denotan los complementos de **X** y **B**, respectivamente. La ecuación 2.18 es posible gracias a la relación que existe entre el operador sup-generador y la operación morfológica de hit or miss: $\lambda_{\mathbf{A},\mathbf{B}}(\mathbf{X}) = \mathbf{X}\otimes\mathbf{C}$, donde $\mathbf{C} = \{\mathbf{A},\sim\mathbf{B}\}$ [Hirata, 2011]. Adicionalmente, si se cumple que $\mathbf{A} \leq \mathbf{B} \Leftrightarrow \psi(\mathbf{A}) \leq \psi(\mathbf{B})\ \forall\mathbf{A},\mathbf{B}\in\mathcal{X}$, entonces el W-operador $\mathbf{\Psi}$ con función característica ψ es un *W-operador incremental*, o *creciente*; en el caso contrario el W-operador se denomina *W-operador no incremental*, o *no creciente*. Una configuración **X** es una *configuración minimal* si no contiene a otra configuración. La *base* $\mathcal{B}[\psi]$ de un

operador incremental ψ se define como el conjunto de todas las configuraciones minimales pertenecientes a su núcleo $\mathcal{K}[\psi](1)$. Por lo tanto, para el caso de operadores incrementales, la función característica se puede representar mediante el supremo de un conjunto de erosiones [Matheron, 1975]:

$$\psi(\mathbf{X}) = sup\{X \ominus \mathbf{B} \mid \mathbf{B} \in \mathcal{B}[\psi]\}. \qquad (2.19)$$



(X_1,X_2,X_3)	$\psi(\mathbf{X})$
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

(a) **(b)** **(c)** **(d)** **(e)**

Figura 2.9. Representación computacional y morfológica de W-operadores. (**a**) Ventana *W* de 1×3 píxeles. (**b**) Representación computacional de ψ a través de una tabla de búsqueda. (**c**) Diagrama de Hasse o reticulado del espacio de 8 posibles configuraciones que se pueden observar a través de *W*. (**d**) Representación morfológica de ψ mediante su núcleo $\mathcal{K}[\psi](1)$, colección de intervalos maximales $\mathcal{M}[\psi]$ y base $\mathcal{B}[\psi]$. En este caso es posible el uso de la base debido a que ψ es incremental. (**e**) Elementos estructurante de 1×3 píxeles para las operaciones de hit or miss (**A** y **B**) y erosión.

Ejemplo 2.4. Aquí se ilustra el concepto de núcleo, colección de intervalos maximales, y base de un W-operador. En la Figura 2.9-a se muestra una ventana *W* de 1×3 píxeles usada para diseñar un W-operador cuya función característica ψ se representa través de una tabla de búsqueda (Figura 2.9-b). En la Figura 2.9-d se presenta el núcleo $\mathcal{K}[\psi](1)$, colección de intervalos maximales $\mathcal{M}[\psi]$, y la base $\mathcal{B}[\psi]$ de la función ψ. Para poder definir la base se asume que ψ es un operador incremental. El diagrama de Hasse (Figura 2.9-c) muestra de manera ordenada, usando la relación de inclusión, el conjunto de todas las 8 posibles configuraciones que se pueden observar a través de *W*. A esta forma de representar el conjunto de configuraciones de ventana se denomina *reticulado*.

En el reticulado de la Figura 2.9-c, las esquinas del lado resaltado con negrilla corresponden a las 4 configuraciones pertenecientes al núcleo de ψ. Nótese como el reticulado permite visualizar fácilmente las configuraciones que actúan como cota inferior, (1,0,0), y cota superior, (1,1,1), del intervalo maximal contenido en $\mathcal{M}[\psi]$. Para este ejemplo, el operador ψ se puede definir en función de un operador sup-generador como $\psi(\mathbf{X}) = \lambda_{(100),(111)}(\mathbf{X})$. ψ puede ser también definido mediante la operación morfológica de hit or miss usando los dos primeros elementos estructurantes de la Figura 2.9-e. Alternativamente, el operador ψ también puede ser representado a través del operador morfológico de erosión usando el tercer elemento estructurante mostrado en la Figura 2.9-e. Esto es posible porque ψ es un operador incremental. Para los elementos estructurantes mostrados en la Figura 2.9-e, blanco representa 1 y negro representa 0.

2.7. Diseño de W-operadores usando una Representación Morfológica

Aparte de compactar la representación computacional de un W-operador, la representación morfológica junto con la regla plug-in pueden también ser usados para el diseño automático, especialmente para la generalización, de W-operadores. En este caso, el diseño de un W-operador consiste de dos etapas: en la primera etapa se diseña la función característica de un W-operador usando la regla plug-in y luego, en la segunda etapa, se obtiene el conjunto de intervalos maximales o la base de dicha función característica. Es conveniente recordar que el cálculo de la base es posible únicamente si el W-operador a diseñar es incremental [Dougherty, 1992a]. Por ejemplo, en [Mathew *et al.*, 1993], los autores proponen un método de diseño de W-operadores binarios incrementales. En la primera etapa, usando la regla plug-in, se estiman las probabilidades condicionales y el valor de la función característica de un operador no incremental. Esto se realiza para todas las configuraciones registradas en el conjunto de ejemplos de entrenamiento. Para aquellas configuraciones no observadas se define un valor $\psi(\mathbf{X}) = 1$.

En la segunda etapa de este método, se determina la base de un W-operador incremental. Para esto se parte del núcleo del W-operador no incremental diseñado en la primera etapa. Usando un algoritmo de intercambio de configuraciones se procede a sacar e incorporar configuraciones desde y hacia el núcleo del operador diseñado en la primera etapa. El intercambio de configuraciones se realiza de tal modo que, el incremento de error del operador diseñado calculado en base a las probabilidades condicionales estimadas en la primera etapa no sea muy elevado. En la etapa de testeo, los operadores diseñados se aplican mediante una serie de erosiones utilizando cada configuración de la base como un elemento estructurante. Los experimentos realizados con este método consisten en la restauración de imágenes binarias contaminadas con ruido tipo sal y pimienta y la reconstrucción de imágenes binarias de huellas digitales. Para estos dos experimentos se utilizan ventanas rectangulares de 2×3 píxeles. Los resultados obtenidos tanto para el filtrado de ruido como para la restauración de imágenes muestran que los W-operadores diseñados mejoran la calidad de los resultados en comparación con el filtro mediana.

Para el caso de ventanas de mediano y gran tamaño el tiempo de búsqueda de los elementos de la base usando el algoritmo de intercambio de configuraciones antes descrito es muy elevado. Debido a este inconveniente, en [Hirata *et al.*, 2000b] se propone uno nuevo algoritmo de intercambio de configuraciones que opera sobre un conjunto de inversión definido para el operador diseñado en la primera etapa. Este conjunto de inversión está formado por todas aquellas configuraciones del núcleo que en el reticulado tienen configuraciones con $\psi(\mathbf{X}) = 0$ por encima de ellas, y todas aquellas configuraciones fuera del núcleo que en el reticulado tienen configuraciones con $\psi(\mathbf{X}) = 1$ por debajo de ellas. En cada iteración de este nuevo algoritmo de intercambio se reduce el tamaño del conjunto de inversión hasta obtener como resultado final un W-operador incremental en base al operador no incremental de la primera etapa.

Los experimentos realizados con este método de diseño consisten en el filtrado de ruido sintético de imágenes binarias usando ventanas de 3×3 hasta 7×7 píxeles. Los casos considerados para los experimentos son imágenes con figuras geométricas contaminadas con ruido tipo sal-pimienta; imágenes de texto donde los bordes de las letras fueron deteriorados introduciendo ruido puntual artificial; imágenes de granos cuadrados con artefactos de ruido artificial que consisten en cuadrados de 3×3 píxeles; e imágenes de

líneas de diferente grosor con los bordes contaminados con ruido puntual artificial uniforme. Los resultados obtenidos muestran una reducción notable del ruido artificial introducido en cada caso. Sin embargo, los tiempos de procesamiento reportados indican que el costo computacional del algoritmo de intercambio es alto, especialmente para ventanas de 5×5 y 7×7 píxeles. Una alternativa para disminuir el costo computacional de este método consiste en reducir el tamaño inicial del conjunto de inversión a costa de un potencial deterioro en la performance de los operadores diseñados.

Otra alternativa para el diseño de W-operadores binarios consiste en representar la función característica obtenida en base a la regla plug-in mediante una suma de productos de funciones booleanas [Barrera *et al.*, 1995], [Dougherty, 1999]. Este método es similar al método usado para el diseño de circuitos electrónicos digitales. Aquí a las configuraciones **X** no observadas en la etapa de entrenamiento se les asigna la condición de *no importa*; es decir, $\psi(\mathbf{X}) = 0$ o $\psi(\mathbf{X}) = 1$. Debido a que el número de productos de las funciones booleanas puede ser muy grande, en este método se requiere de una posterior simplificación de la expresión booleana resultante. Para esto se pueden emplear algoritmos de reducción lógica como el de Quine-McCluskey [Quine, 1952], [McCluskey, 1956] o los mapas de Karnaugh [Karnaugh, 1953]. El principal problema de este método es su elevado costo computacional para diseñar W-operadores, incluso con ventanas de pequeño tamaño. Para el caso de ventanas de mediano y gran tamaño el uso de estos métodos es completamente impráctico.

También se ha propuesto el uso de un algoritmo de división incremental de intervalos del núcleo de un operador diseñado en base a la regla plug-in (*algoritmo ISI*, del inglés *incremental splitting intervals*) [Barrera *et al.*, 1996]. El objetivo en este caso es diseñar y representar un W-operador en base a la colección de intervalos maximales de su núcleo. En [Barrera *et al.*, 1997b] los autores aplican este método a problemas de detección de bordes en imágenes binarias con y sin ruido puntual aditivo y sustractivo uniforme. También se experimenta con la detección de puntos de entrada y salida de circuitos electrónicos digitales en imágenes binarias con ruido puntual uniforme. Otros casos de experimentación incluyen el reconocimiento de caracteres, reconocimiento de texturas, restauración de imágenes de texto contaminadas con ruido puntual uniforme, detección de líneas defectuosas en imágenes de aleaciones eutécticas, y reconocimiento de códigos de barras. Para estos experimentos se usaron ventanas de 3×3, 5×5, y 11×11 píxeles. La etapa de entrenamiento de los W-operadores diseñados involucra el uso de uno y dos pares para la detección de entradas y salidas en circuitos digitales, y cuatro pares para el reconocimiento de código de barras. Mientras tanto, la etapa de testeo se basa en el uso de una sola imagen observada con su correspondiente imagen ideal para la estimación del error de los operadores diseñados.

Los resultados obtenidos en el trabajo descrito en el párrafo anterior muestran una ventaja notable del algoritmo ISI sobre el algoritmo de Quine-McCluskey a nivel de tiempo de entrenamiento (costo computacional). En términos del error empírico, los resultados obtenidos son comparables con otros métodos propuestos para cada caso experimentado. Sin embargo, es importante anotar que estos experimentos involucran el uso de imágenes binarias pequeñas. Adicionalmente, los resultados pueden ser sobreoptimistas debido a que son calculados únicamente a partir de una sola imagen de testeo, descartando de la evaluación la variabilidad que puede existir entre varias imágenes de un mismo problema. En [Hirata *et al.*, 2002] los autores introducen algunas heurísticas para mejorar la performance del algoritmo ISI. Los casos considerados incluyen problemas de extracción

de objetos en imágenes binarias de diagramas de flujo y reconocimiento de texturas en base a ventanas de 9×17 y 7×5 píxeles, respectivamente. En base a los resultados obtenidos, los autores reconocen que se requieren de nuevas mejoras para que los W-operadores diseñados en base al algoritmo ISI retornen buenos resultados en escenarios que involucren el uso de grandes ventanas.

Así mismo, en [Kim, 1997] se propone una nueva forma de diseñar W-operadores en dos etapas usando la representación morfológica de un W-operador en base a una colección de intervalos maximales. En la primera etapa se diseña un W-operador usando la regla plug-in. En la segunda etapa se propone el uso de un nuevo algoritmo de división rápida del espacio de configuraciones creando una estructura de árbol que representa computacionalmente al W-operador diseñado. Los resultados obtenidos aplicando este método a la detección de bordes en imágenes binarias, en escala de grises, e imágenes color muestran una superioridad del método propuesto en términos de costo computacional (memoria y tiempo de entrenamiento) con respecto al algoritmo ISI. Para el procesamiento de imágenes binarias se emplean ventanas con 12, 25, y 49 píxeles. Para el caso de imágenes en escala de gris y color se usan únicamente ventanas de 4 píxeles. La evaluación del desempeño de los operadores diseñados se realiza usando una sola imagen de testeo. Por lo tanto, a pesar de los avances realizados, todavía no se resuelve el problema de diseño de W-operadores usando ventanas de mediano y gran tamaño para imágenes binarias, en escala de grises, y color.

En los siguientes trabajos: [Dougherty, 1992a], [Dougherty y Loce, 1993], [Dougherty y Loce, 1994] se presenta un análisis teórico de los costos de estimación o diseño de W-operadores binarios usando métodos de representación basados en secuencias de erosiones y secuencias de hit or miss. La principal conclusión de estos trabajos es que el costo de diseño aumenta para operadores basados en secuencias de erosiones o secuencias de hit or miss a medida que se aumenta el tamaño de la ventana tal como es de esperarse. Sin embargo, el número de ejemplos de entrenamiento requeridos para diseñar W-operadores basados en secuencias de erosiones puede ser mucho menor que el número de ejemplos necesarios para el caso donde los operadores se diseñan en base a secuencias de hit or miss. Esto se debe que sólo los W-operadores binarios incrementales, una subclase de los W-operadores binarios, se pueden representar mediante secuencias de erosiones. Por el contrario, cualquier W-operador binario puede ser representado mediante una secuencia de operaciones hit or miss u operadores sup-generadores. En [Dougherty, 1992b] y [Dougherty, 1994] se extiende este análisis para el caso de imágenes en escala de grises.

En [Barrera *et al.*, 2005], [Hirata *et al.*, 2007] se propone una extensión de los métodos de diseño de W-operadores usando la colección de intervalos maximales o la base del núcleo de la función característica para el caso de imágenes en escala de grises. Sin embargo, no se presenta ningún caso de aplicación debido al alto costo computacional que demanda la implementación práctica de esta extensión. A nivel general, el diseño y representación de un W-operador usando la colección de intervalos maximales o la base del núcleo de la función característica es viable desde el punto de vista práctico para problemas de PDI que involucren ventanas de pequeño tamaño e imágenes binarias. A pesar de su sólido sustento matemático, estas técnicas demandan un alto costo computacional para ser aplicadas a problemas donde se requiera el uso de ventanas de mediano y gran tamaño, o peor aún, cuando las imágenes a procesar sean imágenes en escala de grises o color.

Un punto importante a tener en cuenta es que, el uso de algunas de las técnicas de diseño antes descritas involucra imponer restricciones. Estas restricciones actúan sobre el espacio de las funciones características que se consideran como candidatas para el diseño. Por ejemplo, las técnicas que retornan W-operadores incrementales descartan de su espacio de búsqueda a aquellos W-operadores no incrementales. Si para un problema dado, el operador óptimo es no incremental, incluso el mejor operador incremental podría ser una mala solución para el problema en cuestión. Por lo tanto, aparte del costo diseño antes considerado, es importante también analizar el impacto de la restricción impuesta por un determinado método sobre el desempeño final, o error, de los W-operadores diseñados. En la siguiente sección se realiza este análisis.

2.8. Diseño de W-operadores usando Restricciones

Dado que PDI es una disciplina aplicada, no es suficiente con sólo proponer formas de representación y/o diseño de W-operadores que cumplan con interesantes propiedades matemáticas. También se requiere que dichas representaciones o métodos de diseño se puedan implementar en un contexto práctico. Por esta razón, es usual que en el diseño automático de W-operadores se termine imponiendo restricciones sobre la clase o familia de operadores a ser usadas en la etapa de aprendizaje o entrenamiento. A nivel teórico, en la primera sección de este capítulo se mostró que para el caso de la regla plug-in se puede reducir arbitrariamente el costo de diseño en un escenario con un número infinito de muestras de entrenamiento. Del mismo modo, para el caso de *k*NN se puede reducir el costo de diseño, hasta llegar a un valor de cero, si el valor de *k* crece mucho más lentamente con relación al crecimiento de la cantidad de ejemplos de entrenamiento disponibles.

El problema en estos dos casos es que, en la práctica, la cantidad de ejemplos de entrenamiento es finita y limitada. Esto causa a su vez que, a medida que se incrementa el tamaño de las ventanas para el procesamiento de imágenes binarias o cuando se trabajan con imágenes en escala de grises y color, la performance de los W-operadores diseñados en base a estas reglas empeora a pesar de sus interesantes propiedades matemáticas. Esto último se debe a un incremento del costo de diseño con un consecuente aumento del error de los operadores diseñados (ecuación 2.6). Aparte de estos problemas, otro aspecto a tener en cuenta es el costo computacional que implicaría el diseñar un W-operador usando *k*NN o plug-in asumiendo que se contaran con suficientes datos de entrenamiento para explotar sus propiedades de convergencia.

Para los métodos de diseño basados en la representación morfológica, uno de los objetivos es reducir la representación computacional de un W-operador. Sin embargo, en un contexto práctico, sobre todo en aplicaciones reales, es posible que un W-operador esté representado por una base compuesta por un número grande de elementos estructurantes. Situación similar podría ocurrir para la representación morfológica usando una colección de intervalos maximales. Adicionalmente, los casos de aplicación reportados en la sección anterior evidencian que estos métodos involucran también un alto costo computacional cuando se usan con ventanas de mediano y gran tamaño para procesar imágenes binarias. Más crítica aún es la situación cuando se desea aplicar dichos métodos a problemas donde las imágenes observadas son imágenes en escala de grises y color, incluso usando ventanas de pequeño tamaño. Por este motivo, algunos de los métodos reportados en la sección anterior consideran únicamente el diseño de W-operadores incrementales, que son una subclase de todos los posibles W-operadores.

Cuando se dispone de una cantidad finita y limitada de ejemplos de entrenamiento, una forma de reducir el costo de diseño o estimación de un método de diseño consiste en limitar, o restringir, la cantidad de funciones características candidatas para la búsqueda del mejor operador. Otra alternativa consiste en la reducción del número de variables que forman las configuraciones de ventana. En cualquiera de estos casos, y otros que se mencionarán más adelante, se debe procurar que estas restricciones no deterioren significativamente la performance del operador diseñado. En lo sucesivo de esta tesis se considera el diseño de W-operadores usando restricciones. Se analiza el costo de usar una restricción y su influencia sobre el desempeño de los W-operadores diseñados.

Para facilidad del siguiente análisis, y sin pérdida de generalidad, se considera que las imágenes observadas son imágenes en escala de grises y que las imágenes ideales son imágenes binarias. El espacio de todas las posibles imágenes observadas es $L^{\boldsymbol{E}}$, recordando que $\boldsymbol{E}\subset\mathbb{Z}^2$ y el conjunto de niveles de gris es $L = \{0,1,\ldots,l - 1\}$ con $l\in\mathbb{Z}^+$. El espacio de todas las posibles imágenes binarias es $\{0,1\}^{\boldsymbol{E}}$. Para una ventana W de tamaño $n = |W|$, el espacio de todas las posibles configuraciones n-dimensionales $\mathbf{X}$ que se pueden observar a través de W es $\boldsymbol{\mathcal{X}}$. Por lo tanto, el tamaño de $\boldsymbol{\mathcal{X}}$ es $|\boldsymbol{\mathcal{X}}| = l^n$. Los posibles valores de Y que se pueden observar en las imágenes ideales están dados por el conjunto $\mathcal{Y} = \{0,1\}$. La función característica de un W-operador $\boldsymbol{\Psi}$ es una función ψ: $\boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$. La familia o conjunto de todas las posibles funciones características, también denominado espacio de búsqueda en el contexto de optimización, con dominio en $\boldsymbol{\mathcal{X}}$ y rango en $\mathcal{Y}$ se denota mediante C_0. En este caso, C_0 está formado por un total de $|C_0| = 2^{l^n}$ posibles funciones características.

El uso de una restricción involucra utilizar un espacio de búsqueda C_1 que es un subconjunto del espacio de búsqueda inicial C_0: $C_1\subset C_0$. Al reducir el espacio de búsqueda, y dado un número fijo N de ejemplos de entrenamiento, el costo de estimación de un W-operador $\Delta(\psi_N,\psi_{\text{opt}})$ se reduce. Esto se debe a que ahora hay un número menor de funciones candidatas entre las cuales se debe seleccionar a la mejor. Sin embargo, esta reducción del espacio de búsqueda trae consigo un *costo de restricción* dado por

$$\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}) = \varepsilon(\psi_{C,\text{opt}}) - \varepsilon(\psi_{\text{opt}}), \tag{2.20}$$

donde $\varepsilon(\psi_{C,\text{opt}})$ es el error de la función característica óptima en el nuevo espacio de búsqueda C_1, $\varepsilon(\psi_{C,\text{opt}}) < \varepsilon(\psi)\ \forall\psi\in C_1$. Debido a la optimalidad de la función ψ_{opt} siempre se cumple que $\varepsilon(\psi_{\text{opt}}) < \varepsilon(\psi_{C,\text{opt}})$ para cualquier $C_1\subset C_0$, con lo cual el costo de restricción es siempre positivo o a lo sumo 0: $\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}) \geq 0$. El error $\varepsilon(\psi_{C,N})$ de un W-operador $\psi_{C,N}$ diseñado en base a un algoritmo de aprendizaje supervisado $\mathcal{A}$ utilizando el nuevo espacio de búsqueda C_1 y un conjunto $\boldsymbol{\mathcal{D}}$ de N ejemplos de entrenamiento se obtiene reemplazando en la ecuación 2.6 los valores de $\Delta(\psi_N,\psi_{\text{opt}})$ y $\varepsilon(\psi_{\text{opt}})$ con $\Delta(\psi_{C,N},\psi_{C,\text{opt}})$ y $\varepsilon(\psi_{C,\text{opt}})$, respectivamente:

$$\varepsilon(\psi_{C,N}) = \Delta(\psi_{C,N},\psi_{C,\text{opt}}) + \varepsilon(\psi_{C,\text{opt}}), \tag{2.21}$$

donde $\Delta(\psi_{C,N},\psi_{C,\text{opt}})$ denota el costo de diseño, o estimación, del operador $\varepsilon(\psi_{C,N})$ dentro del espacio de búsqueda reducido C_1.

Reemplazando la expresión para $\varepsilon(\psi_{C,\text{opt}})$ de la ecuación 2.20 en la ecuación 2.21 se obtiene

$$\varepsilon(\psi_{C,N}) = \Delta(\psi_{C,N},\psi_{C,\text{opt}}) + \Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}) + \varepsilon(\psi_{\text{opt}}). \quad (2.22)$$

Para un problema determinado, el valor de $\varepsilon(\psi_{\text{opt}})$ es constante. De la misma forma, una vez fijada una restricción, el valor de $\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}})$ es también constante y la única alternativa para reducir el error del operador a diseñar es reduciendo el costo de estimación $\Delta(\psi_{C,N},\psi_{C,\text{opt}})$. Usualmente, diferentes conjuntos de entrenamiento resultan en diferentes W-operadores. Por lo tanto, para propósitos de comparación de métodos de diseño automático de W-operadores se considera su error esperado calculado en base a conjuntos $\mathcal{D}$ formados por N ejemplos de entrenamiento:

$$\mathbb{E}[\varepsilon(\psi_{C,N})] = \mathbb{E}[\Delta(\psi_{C,N},\psi_{C,\text{opt}})] + \Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}) + \varepsilon(\psi_{\text{opt}}), \quad (2.23)$$

donde la esperanza $\mathbb{E}$ se calcula con respecto a la distribución subyacente a la generación de los conjuntos de N ejemplos de entrenamiento.

En base a lo anterior, la restricción cuyo espacio de búsqueda es C_1 será buena sí y solamente si el valor de $\mathbb{E}[\varepsilon(\psi_{C,N})]$ es menor que el valor de $\mathbb{E}[\varepsilon(\psi_N)]$ cuando se diseñan W-operadores usando el espacio de búsqueda sin restricciones C_0: $\mathbb{E}[\varepsilon(\psi_{C,N})] < \mathbb{E}[\varepsilon(\psi_N)]$, donde $\mathbb{E}[\varepsilon(\psi_N)]$ se calcula usando la ecuación 2.7. Esto a su vez implica que

$$\mathbb{E}[\Delta(\psi_N,\psi_{\text{opt}})] - \mathbb{E}[\Delta(\psi_{C,N},\psi_{C,\text{opt}})] > \Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}). \quad (2.24)$$

En palabras, el resultado de la desigualdad 2.24 indica que, para que una restricción sea buena, la reducción del costo de diseño, o mejora en la estimación de las probabilidades condicionales, $\mathbb{E}[\Delta(\psi_N,\psi_{\text{opt}})] - \mathbb{E}[\Delta(\psi_{C,N},\psi_{C,\text{opt}})]$, debe superar al costo de restricción $\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}})$. En la Figura 2.10 se muestra una ilustración gráfica del análisis antes presentado. En este caso se asume que el operador óptimo ψ_{opt} del espacio sin restricción C_0 no está incluido dentro del espacio restringido C_1.

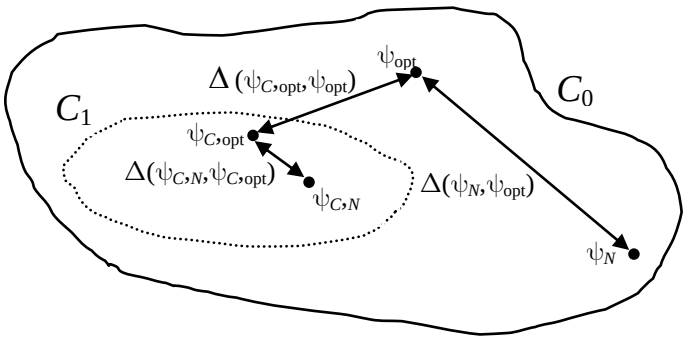


Figura 2.10. Ilustración de los espacios de búsqueda para el diseño de W-operadores usando restricciones.

Para la reducción del costo de diseño se tiene que $(\mathbb{E}[\Delta(\psi_N,\psi_{\text{opt}})] - \mathbb{E}[\Delta(\psi_{C,N},\psi_{C,\text{opt}})]) \to 0$ cuando $N \to \infty$. Esto evidencia que el uso de restricciones es útil solamente para el caso donde la cantidad de ejemplos de entrenamiento es finita, situación que es el caso de toda

aplicación práctica. Adicionalmente, para el caso de una regla de estimación consistente y una distribución conjunta Pr($\mathbf{X}$,Y) para la cual $\mathbb{E}[\Delta(\psi_{C,N+1},\psi_{C,\text{opt}})] \leq \mathbb{E}[\Delta(\psi_{C,N},\psi_{C,\text{opt}})]$, una restricción es útil siempre y cuando la cantidad de ejemplos de entrenamiento disponibles sea menor a un cierto tamaño. Por encima de dicho tamaño, la restricción es detrimental [Dougherty, 1999], [Braga-Neto y Dougherty, 2005]. Por lo tanto, en el contexto práctico, el diseño de W-operadores consiste en la búsqueda de restricciones que cumplan con la condición establecida en la desigualdad 2.24 teniendo en cuenta la cantidad de ejemplos disponibles para el diseño. El dilema en este caso es que *fuertes restricciones reducen significativamente el costo de diseño a costa de un aumento en el costo de restricción*.

En la práctica, lo deseable es que C_1 contenga al operador óptimo, ψ_{opt}, en cuyo caso el costo de restricción es nulo: $\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}) = 0$. Si este es el caso, la desigualdad 2.24 se reduce a demostrar que el costo de estimación en el nuevo espacio de búsqueda C_1 es menor que el costo de estimación en el espacio original C_0: $\mathbb{E}[\Delta(\psi_N,\psi_{\text{opt}})] > \mathbb{E}[\Delta(\psi_{C,N},\psi_{C,\text{opt}})]$. A modo de ejemplo, sea C_0 el conjunto de todos los W-operadores para una ventana W de tamaño n. C_1 puede ser la familia de todos los W-operadores que pueden ser representados por un conjunto predefinido de elementos estructurantes. Esta será una buena restricción solamente si ψ_{opt} puede ser representando morfológicamente a través de una secuencia de erosiones con los elementos estructurantes predefinidos. Caso contrario, inclusive el mejor operador dentro de la restricción C_1 podría ser una mala solución para el problema en cuestión.

Desafortunadamente, en un contexto práctico encontrar "buenas" restricciones para el diseño de W-operadores no es una tarea trivial. Para una aplicación práctica lo usual es seleccionar un espacio de búsqueda C_1 esperando que su costo de diseño y restricción sean bajos, de tal modo de satisfacer la desigualdad 2.24, [Barrera *et al.*, 2005], [Dougherty y Barrera, 2002]. Esto es equivalente a esperar que la distribución del algoritmo de aprendizaje utilizado Pr($\psi|\boldsymbol{\mathcal{D}}$) y la distribución a posteriori del problema en cuestión Pr($f|\boldsymbol{\mathcal{D}}$) estén alineadas, situación descrita en el análisis de los teoremas de no free lunch (Sección 2.5). Dado que en la práctica se desconoce la distribución conjunta Pr($\mathbf{X}$,Y), la única forma de saber si una restricción C_1 es buena es observando el desempeño, o calculando el error empírico, del W-operador diseñado a partir de la misma.

En un contexto práctico, una restricción puede ser dividida en dos partes: (**a**) *pre-procesamiento* de las configuraciones de ventana, donde se reduce el tamaño del espacio $\boldsymbol{\mathcal{X}}$ obteniendo un nuevo espacio de configuraciones $\boldsymbol{\mathcal{Z}}$, donde $|\boldsymbol{\mathcal{Z}}| < |\boldsymbol{\mathcal{X}}|$; y (**b**) *selección de una clase de funciones características* $C \subset C_0$, definidas sobre el espacio $\boldsymbol{\mathcal{Z}}$, para la representación de los W-operadores. Adicionalmente, en el diseño también se puede incluir información *a priori* sobre el problema en cuestión, sin que ésta represente una restricción. En función de lo anterior, la ecuación 2.21 puede ser expresada como

$$\varepsilon(\psi_{C,N}) = \Delta(\psi_{\xi,C,N},\psi_{\xi,C,\text{opt}}) + \Delta(\psi_{\xi,C,\text{opt}},\psi_{C,\text{opt}}) + \Delta(\psi_{\xi,\text{opt}},\psi_{\text{opt}}) + \quad \varepsilon(\psi_{\text{opt}}) - \Delta_{\text{priori}}. \quad (2.25)$$

En la ecuación 2.25, el término $\Delta(\psi_{\xi,C,N},\psi_{\xi,C,\text{opt}})$ denota el costo de diseño o estimación de un W-operador usando la clase de funciones características C definidas sobre $\boldsymbol{\mathcal{Z}}$;

$\Delta(\psi_{\xi,C,\text{opt}},\psi_{C,\text{opt}})$ denota el costo de restringir al conjunto C la clase de funciones características definidas sobre el nuevo espacio de configuraciones $\mathcal{Z}$; $\Delta(\psi_{\xi,\text{opt}},\psi_{\text{opt}})$ denota el costo de reducir el espacio de configuraciones $\mathcal{X}$ para obtener el nuevo espacio $\mathcal{Z}$; $\varepsilon(\psi_{\text{opt}})$ es el error del operador óptimo sin restricciones; y Δ_{priori} denota la reducción del costo de diseño usando información *a priori*. Este último término muestra el rol que cumple el uso de información *a priori* para el diseño automático de W-operadores. Sin embargo, Δ_{priori} puede ser negativo si la información *a priori* no es correcta y cero si no se dispone de la misma. Nótese que para el caso donde $\Delta_{\text{priori}} < 0$, el error $\varepsilon(\psi_{C,N})$ del operador diseñado $\psi_{C,N}$ aumenta. Por lo tanto, aunque la información *a priori* no constituye una forma de restricción, ésta debe ser correcta para reducir el error de un W-operador.

A continuación se clasifican y analizan las restricciones propuestas para el diseño automático de W-operadores. La clasificación utilizada en esta tesis es en función del tipo de imágenes para el cual fueron propuestas originalmente. Este tipo de clasificación se adopta porque facilita ubicar en el contexto de PDI el grado de avance teórico y práctico que se ha logrado en lo referente al diseño automático de W-operadores. De modo más general, esto también permite comparar el diseño automático de operadores morfológicos con otros enfoques existentes en PDI, situación que está fuera del alcance de la presente revisión bibliográfica. Es importante también resaltar que la clasificación aquí usada no es exclusiva, pues en varios casos una misma restricción puede servir para trabajar con diferentes tipos de imágenes, situaciones en las cuales se anota este hecho.

La clasificación aquí usada es diferente a la clasificación originalmente propuesta en esta área y descrita en [Dougherty y Barrera, 1999]. Aquí los autores clasifican las restricciones propuestas en dos categorías: restricciones independientes y dependientes. Se consideran como *restricciones independientes* a aquellas donde la decisión de ubicar a una configuración dentro del núcleo de la función característica depende solamente de la configuración en cuestión. Aquellas restricciones donde el criterio de ubicación de una configuración dentro del núcleo de un operador depende de la configuración en sí misma y de otras configuraciones se denominan *restricciones dependientes* [Barrera *et al.*, 2005]. En este último caso no siempre es posible establecer cuál es la relación de dependencia. Adicionalmente, no siempre es posible en la práctica utilizar la representación morfológica que permite el cálculo del núcleo de un operador debido a su elevado costo computacional.

2.8.1. Restricciones para Diseño de W-operadores Binarios

Uso de Operadores Morfológicos: En [Schmitt, 1989], uno de los primeros trabajos desarrollados dentro del diseño automático de operadores morfológicos, se propone combinar de manera automática y usando sistemas de inteligencia artificial las operaciones morfológicas de un conjunto de operadores predefinido para un problema determinado. Las operaciones consideradas son erosión, dilatación, y hit or miss. En este caso, la restricción consiste en la definición del tipo y número de operaciones morfológicas a combinar. El sistema propuesto permite especificar el problema a resolver mediante el uso de un único par de imágenes de entrenamiento (imágenes observada e ideal). Los experimentos realizados consisten en detectar discontinuidades en las líneas presentes en dos imágenes de aleaciones eutécticas laminares. Los resultados obtenidos ponen de manifiesto por primera vez la potencial ventaja de combinar operadores morfológicos de manera automática versus

una estrategia de combinación heurística en términos de costo computacional y performance de los operadores diseñados.

En [Joo *et al.*, 1990] los autores proponen el diseño automático de operadores morfológicos transformando reglas lingüísticas en inglés en una *lógica de predicados* de primer orden. Las reglas lingüísticas describen en este caso a las operaciones morfológicas a ser usadas durante el diseño. Las operaciones morfológicas consideradas son erosión, dilatación, hit or miss, apertura, y cerradura. Es importante remarcar que, en este caso, se propone también una nueva forma de diseño de W-operadores que consiste en el uso de la lógica de predicados. Aquí la descripción del problema a resolver está dada por las imágenes observadas; mientras tanto que la salida deseada consiste en la descripción lingüística, en inglés, del algoritmo morfológico a ser utilizado, incluyendo la forma y tamaño de los elementos estructurantes a ser empleados. Se proponen algoritmos para la remoción de objetos pequeños con formas específicas, por ejemplo cuadrados de 5×5 píxeles, detección de objetos, por ejemplo células, y detección de fallas. A pesar de la extensa formulación teórica presentada, no se incluyen casos prácticos de aplicación.

W-operadores Incrementales: Para algunos casos analizados en la Sección 2.7, y tal como se mencionó en su debido momento, una de las restricciones utilizadas consiste en asumir que el operador a diseñar es incremental. Otra forma de restricción, aparte de la de operador incremental, consiste en limitar la cantidad y forma de los elementos estructurantes con los que se representará el W-operador diseñado [Salembier, 1992]. Para este caso se diseñan y representan W-operadores incrementales mediante una secuencia de erosiones con elementos estructurantes seleccionados a partir de una librería predefinida de elementos estructurantes. Esta librería se compone de elementos estructurantes obtenidos a partir de la experiencia de diseñadores. De modo similar se procede cuando se diseñan W-operadores no incrementales mediante secuencias de hit or miss [Dougherty, 1994]. Otras restricciones incluyen limitar el tamaño de la base o el número de intervalos maximales [Dougherty y Barrera, 1999].

Uso de Información a Priori: Se han propuesto también métodos de diseño automático de W-operadores basados en el uso de información *a priori*. En [Dougherty y Loce, 1996] y [Barrera *et al.*, 1997a] se propone el diseño de un W-operador con función característica ψ_N partiendo de una función característica inicial ψ_0 definida por el diseñador. En función de los datos de entrenamiento, y un factor inercial no negativo $\omega(\mathbf{X})$, se ajusta la función ψ_0 para obtener ψ_N. Para esto, si $\psi_0(\mathbf{X}) = 0$, entonces $\psi_N(\mathbf{X}) = 1$ si la estimación de $\mathbb{P}(Y = 1|\mathbf{X})$ es mayor que $0.5 + \omega(\mathbf{X})/(freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1))$. Por el contrario, si $\psi_0(\mathbf{X}) = 1$, entonces $\psi_N(\mathbf{X}) = 0$ si la estimación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ es menor o igual que $0.5 - \omega(\mathbf{X})/(freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1))$. Si $(freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1)) = 0$, lo cual implica que $\mathbf{X}$ no fue observada en el entrenamiento, entonces $\psi_N(\mathbf{X}) = \psi_0(\mathbf{X})$. El término $\omega(\mathbf{X})$ representa el grado de confianza del diseñador en la función inicial ψ_0.

Otra forma de uso de información *a priori* es el caso donde tanto las imágenes observadas como las imágenes ideales son simuladas mediante el uso de modelos de degradación. En base a estas simulaciones y uno de los métodos descritos al inicio de este capítulo, plug-in

por ejemplo, se pueden diseñar W-operadores para la restauración de imágenes [Dougherty, 1999]. En este caso, para lograr un desempeño satisfactorio de los W-operadores diseñados, las realizaciones de las imágenes observadas e ideales utilizadas para el entrenamiento deben estar acorde con las imágenes reales a procesar. En otras palabras, la distribución probabilística subyacente al modelo de generación de las imágenes debe ser similar a la distribución probabilística del problema en cuestión, la cual es desconocida. Justamente es éste uno de los principales inconvenientes de este método de diseño. Adicionalmente, para cada problema a resolver se requiere encontrar un nuevo modelo de generación de las imágenes de entrenamiento.

En [Loce y Dougherty, 1997] se propone el diseño de W-operadores partiendo de la función *a priori* $\psi_0(\mathbf{X}) = 1\ \forall \mathbf{X} \in \mathcal{X}$ y la definición de un umbral $\tau > 0$. Si para una configuración $\mathbf{X}$ se cumple que $(freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1)) \geq \tau$, entonces se usa este valor para estimar la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ usando la regla plug-in. A partir de esta estimación se define el valor de la función característica $\psi_N(\mathbf{X})$ del operador diseñado (ecuación 2.3). Caso contrario se define $\psi_N(\mathbf{X}) = 1$.

Otra alternativa consiste en modelar a cada probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ como una variable aleatoria gobernada por una determinada distribución *a priori* y luego, en un contexto Bayesiano, estimar su valor utilizando el conjunto de ejemplos de entrenamiento [Dougherty y Barrera, 1997], [Kamat y Dougherty, 2000]. Un inconveniente de este método de diseño es que tiene un alto costo computacional debido a que el tamaño del espacio de configuraciones crece exponencialmente con el aumento del tamaño de la ventana. A nivel general, el principal desafío del uso de información *a priori* es el tratar de lograr que ésta sea correcta de tal modo de no incrementar el error del operador diseñado (ecuación 2.25).

Otro tipo de restricción consiste en limitar el número de erosiones y dilataciones con las que se representa el W-operador diseñado, así como el tamaño mínimo y máximo de los elementos estructurantes. En base a esta restricción en [Harvey y Marshall, 1996] se diseñan W-operadores usando algoritmos genéticos con ventanas de pequeño y de mediano tamaño. También se procesan imágenes en escala de grises usando ventanas pequeñas. En [Quintana *et al.*, 2006] también se emplean algoritmos genéticos para el diseño de W-operadores. En este caso los algoritmos genéticos se usan para dos tareas. La primera consiste en ajustar la forma de los elementos estructurantes iniciales definidos por el usuario. La segunda tarea consiste en encontrar secuencias de erosiones y dilataciones cuyos resultados son combinados mediante operaciones booleanas.

Los experimentos realizados en los trabajos antes descritos consisten en el reconocimiento de objetos contenidos en imágenes binarias sintéticas y el reconocimiento de un cierto tipo de símbolos en imágenes de pentagramas musicales. Para el primer experimento se usan elementos estructurantes iniciales de 3×3 píxeles, un par de imágenes de entrenamiento y 10 imágenes de testeo de 640×480 píxeles. Para el segundo experimento se usan elementos estructurantes iniciales de 3×3, 5×5, y 7×7 píxeles e imágenes de 16×16, 32×32, y 64×64 píxeles. Los resultados obtenidos muestran un mejor desempeño del algoritmo propuesto para el segundo experimento en comparación con el primero.

Como se vio anteriormente, la teoría garantiza que *no se puede diseñar un operador que supere el desempeño del operador óptimo* para un problema determinado. Por lo tanto, se pensaría que aumentando el tamaño de la ventana también aumenta la posibilidad de encontrar al operador óptimo, o uno muy cercano al mismo. Esto debido a que se incrementa el tamaño del espacio de búsqueda (comportamiento de $\varepsilon(\psi_{C,\text{opt}})$ en la Figura 2.11). Si bien esta idea es correcta, la misma sólo es válida para escenarios donde la cantidad de ejemplos de entrenamiento es infinita, situación contraria a un escenario práctico. Para una cantidad fija y finita de ejemplos de entrenamiento, una forma relativamente fácil de reducir el costo de estimación es reduciendo el tamaño, o resolución, de la ventana usada para la recolección de observaciones. Esto provoca a su vez una reducción del tamaño del espacio de configuraciones y consecuentemente una reducción del espacio de búsqueda.

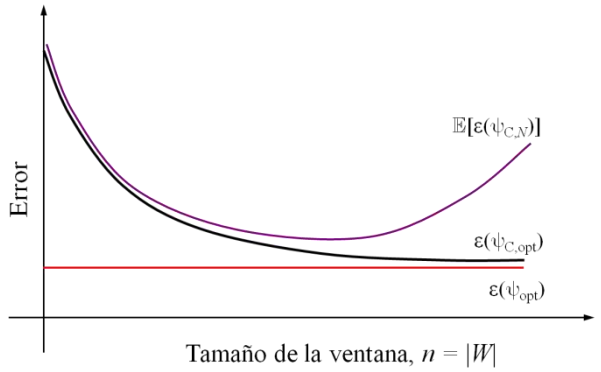


Figura 2.11. Ilustración de la curva de error "U" para el diseño de W-operadores. En este caso la restricción *C* consiste en la variación del tamaño *n* de la ventana *W*. La cantidad de ejemplos de entrenamiento *N* es fija.

Curva de error "U": En [Hirata, 2011] se presenta un estudio empírico donde se analiza la influencia del tamaño de la ventana sobre el desempeño del W-operador diseñado en un escenario con una cantidad fija de datos o ejemplos de entrenamiento. Al graficar la curva de error de testeo del W-operador diseñado (eje *y*) en función del número de píxeles que conforman la ventana (eje *x*), se obtiene una curva con forma de "U" (comportamiento de $\mathbb{E}[\varepsilon(\psi_{C,N})]$ en la Figura 2.11). Para ventanas con pocos píxeles (lado izquierdo del eje *x*), el error del W-operador diseñado es grande. Esto se debe a que el costo de diseño es bajo, pero el costo de restricción es demasiado alto. A medida que se mueve hacia el lado derecho del eje *x*, el error del W-operador va decreciendo hasta llegar a estabilizarse. En el punto donde el error se estabiliza es donde se consigue un equilibrio entre el costo de diseño y el costo de restricción. Posteriormente, y a medida que el número de píxeles que forman la ventana aumenta, también empieza a crecer el valor de error. Esto se debe a que el costo de estimación aumenta debido al crecimiento exponencial del número de funciones características que conforman el espacio de búsqueda, teniendo en este caso un bajo costo de restricción.

Restricción Secundaria: Al reducir el tamaño de la ventana, inmediatamente surge la pregunta sobre qué píxeles eliminar. Una alternativa consiste en eliminar o restringir la contribución de información de los píxeles que se encuentran en los bordes de la ventana asumiendo que el píxel a procesar es el píxel central. La suposición en este caso es que los píxeles más cercanos al píxel a procesar son los que mayor influencia ejercen sobre el

diseño del W-operador. Esto debido a que en esta región la correlación entre los valores de los píxeles es alta. Mientras tanto, los píxeles que se encuentran en los bordes de la ventana tienen menor contribución debido a que están menos correlacionados con el píxel a procesar, pero aumentan exponencialmente la demanda de datos de entrenamiento.

Un ejemplo de esta idea es el uso de las denominadas restricciones secundarias. Aquí no existe eliminación de píxeles de la ventana, pero se restringe la contribución de aquellos píxeles que se encuentran en el borde de la ventana durante el diseño del W-operador [Sarca *et al.*, 1998]. En este trabajo se propone dividir la ventana de observación inicial W_0 de n píxeles en dos subventanas disjuntas: una ventana primaria W_1 con p píxeles y la otra denominada ventana secundaria W_2 con q píxeles tal que $n = p + q$. La ventana primaria W_1 contiene los píxeles de W_0 más cercanos al píxel a procesar, usualmente el píxel central. La ventana secundaria W_2 contiene los píxeles que se encuentran en los bordes de W_0. Se diseñan y representan W-operadores por medio de funciones booleanas definidas sobre las n variables que se pueden observar a través de W_0. Las funciones Booleanas consideradas como candidatas para la optimización incluyen a todas las funciones características que se pueden definir sobre las p variables de W_1. Sin embargo, el espacio de búsqueda no incluye a todas las funciones características que se pueden definir sobre las n variables de W_0. Esto implica que se restringe el espacio de búsqueda para el caso de las q variables que se pueden ver mediante W_2.

El método de diseño antes descrito se evalúa en un problema de restauración de imágenes binarias de texto escaneado. El modelo de degradación usado consiste en añadir ruido puntual artificial a cada letra de modo que la cantidad de ruido en los bordes internos y externos de cada letra es mayor en relación al resto de la imagen. Las ventanas W_0 utilizadas son de 3×3 hasta 7×7 píxeles con valores de q iguales a 5 y 9. Los resultados obtenidos muestran que los errores de los W-operadores diseñados usando la restricción son ligeramente mayores a los W-operadores sin restricción. Sin embargo, el costo computacional (cantidad de memoria) del diseño restringido se reduce notablemente comparado con el diseño sin restricciones. Adicionalmente, para una cantidad fija de datos de entrenamiento y los tamaños de ventana considerados, el error de los W-operadores disminuye a medida que se incrementa el tamaño de la ventana.

Uso de Ventanas Dispersas: Otra forma de restricción para reducir el tamaño del espacio de configuraciones consiste en el uso de ventanas de mediano o gran tamaño dispersas o ralas. Por ejemplo, si se realiza un submuestreo de una ventana "compacta" de 6×6 píxeles de tal modo que se consideran únicamente las observaciones de los píxeles de la primera, tercera y quinta filas y si se procede de igual manera para el caso de las columnas, entonces la cantidad efectiva de píxeles que se observan con esta ventana son 3×3 píxeles. Esto es equivalente a reducir el tamaño de las imágenes a procesar usando este mismo método de submuestreo y realizar el escaneo con una ventana compacta de 3×3 píxeles. Dicho de otro modo, el diseño de W-operadores usando la ventana dispersa de 6×6 píxeles y la imagen original es equivalente a diseñar W-operadores usando una ventana compacta de 3×3 píxeles pero con la imagen original submuestreada. Esta idea se utiliza en [Hirata y Dornelles, 2009] para resolver problemas de reconocimiento de caracteres en imágenes de texto, identificación de texturas en imágenes de mapas, y la segmentación de objetos circulares y cuadrados en imágenes de diagramas de flujo. Las ventanas ralas utilizadas tienen un tamaño total de 18×14 píxeles, con una cantidad efectiva de 9×7 píxeles.

Diseño Iterativo: El diseño iterativo de W-operadores es otra alternativa de restricción que ha sido investigada. Aquí una ventana W de mediano o largo tamaño puede ser representada mediante una serie de subventanas de pequeño tamaño $W_0,W_1,\ldots,W_q$ de tal manera que se satisfaga la relación $W = W_0 \oplus W_1 \oplus,\ldots,\oplus W_q$, donde $\oplus$ denota el operador morfológico de dilatación. La restricción en este caso consiste en el uso de una clase de funciones características que se pueden representar como $\psi = \psi_q\psi_{q-1},\ldots,\psi_1$, donde ψ_i se define sobre el conjunto de variables que se pueden observar con W_i, siendo $i = 1,\ldots q$. Nótese que la función característica ψ se define sobre la ventana W.

Por ejemplo, dado una par de imágenes observada e ideal, en una primera etapa se puede diseñar un W-operador usando una ventana pequeña W y la imagen observada de tal manera que se minimice el error entre la imagen filtrada y la imagen ideal. En una segunda etapa se puede diseñar un segundo W-operador con la misma ventana de la primera etapa W, donde la imagen observada es la imagen filtrada de la primera etapa y la minimización del error se da entre la imagen filtrada de esta segunda etapa y la imagen ideal. El W-operador resultante se obtiene poniendo en cascada los dos W-operadores diseñados en la primera y segunda etapas y su ventana se obtiene dilatando W consigo misma. La ventaja en este caso es que en cada etapa del diseño se usan ventanas pequeñas que involucran tanto un bajo costo de diseño o estimación como un bajo o moderado costo computacional [Sarca *et al.*, 1999]. En [Hirata *et al.*, 2000a] se extiende esta idea de diseño iterativo para casos donde el diseño involucra a más de dos etapas, llamado diseño multietapa.

En [Hirata, 2009] se diseñan W-operadores en dos etapas usando la regla plug-in. Para esto se definen *a priori* una colección de ventanas $\mathcal{V} = \{W_1,W_2,\ldots,W_q\}$, donde el solapamiento entre cada par de ventanas Wi y W_j es mínimo para todo $i,j = 1,\ldots,q$ e $i \neq j$. En base a cada ventana de $\mathcal{V}$, en una primera etapa se diseñan los W-operadores $\mathbf{\Psi}_1,\mathbf{\Psi}_2,\ldots,\mathbf{\Psi}_q$ usando un único par de imágenes de entrenamiento (O,I). Para una segunda etapa, y dado el par de imágenes (O',I'), se diseña un solo W-operador $\mathbf{\Psi}$ en base a las imágenes resultantes de la primera etapa $\mathbf{\Psi}_1(O'),\mathbf{\Psi}_2(O'),\ldots,\mathbf{\Psi}_q(O')$. Para esto se utiliza una ventana de 1 píxel y las configuraciones de la forma $\mathbf{Z} = (\psi_1(\mathbf{X}_1),\psi_2(\mathbf{X}_2),\ldots,\psi_q(\mathbf{X}_q))$, donde $\psi_1,\psi_2,\ldots,\psi_q$ son las funciones características de los W-operadores $\mathbf{\Psi}_1,\mathbf{\Psi}_2,\ldots,\mathbf{\Psi}_q$, respectivamente. $\mathbf{X}_1,\mathbf{X}_2,\ldots,\mathbf{X}_q$ son las configuraciones observadas en un píxel arbitrario t de la imagen O' a través de las ventanas $W_1,W_2,\ldots,W_q$, respectivamente. El valor de Y se obtienen de la manera usual a partir de la imagen I': $Y = I'(t)$. El operador resultante $\mathbf{\Psi}$ se define sobre la vecindad formada por todas las ventanas del conjunto $\mathcal{V}$. Se propone también diseñar W-operadores en tres y cuatro etapas.

Los experimentos realizados con el método de diseño antes descrito consisten en la segmentación de texto en imágenes de revistas, reconocimiento de caracteres, identificación de texturas en imágenes de mapas, y segmentación de objetos circulares y cuadrados en imágenes de diagramas de flujo. El número de ventanas usadas para $\mathcal{V}$ varía entre 5 y 8. El tamaño de las vecindades que cubren las ventanas utilizadas varía entre 9×7 y 25×25 píxeles. Los resultados obtenidos en términos de error son bastante buenos. En cuanto a la complejidad computacional, el tiempo de entrenamiento aumenta de manera abrupta a medida que se incrementa el tamaño de la vecindad que cubren las ventanas usadas. Aunque este método permite diseñar W-operadores cuya vecindad de observación es

grande, el problema es que cuando se cuentan con imágenes de entrenamiento de gran tamaño o más de un par de imágenes de entrenamiento para cada etapa, el método presenta un muy elevado costo computacional.

Siguiendo esta misma línea de diseño, en [Santos *et al.*, 2010] los autores proponen definir de manera automática el número y forma de las ventanas que conforman el conjunto $\mathcal{V}$ para diseñar W-operadores en dos etapas. Una vez definido $\mathcal{V}$ de manera automática, sus ventanas se usan para diseñar los W-operadores de la primera etapa. En la segunda etapa se procede de la misma manera que en el trabajo descrito en el párrafo anterior. La regla de diseño de W-operadores usada en cada etapa es plug-in. En esta propuesta se requiere que el diseñador defina una ventana inicial de gran tamaño a partir de la cual se obtienen los elementos de $\mathcal{V}$ usando teoría de información. En un trabajo más reciente [Dornelles y Hirata, 2012] se propone un diseño iterativo de W-operadores en dos etapas, donde se define *a priori* un conjunto de ventanas $\mathcal{V}$. Usando algoritmos genéticos se selecciona una subcolección de ventanas a partir del conjunto $\mathcal{V}$ para diseñar los W-operadores de la primera etapa.

Multi-resolución Piramidal: Una restricción que busca conseguir un equilibrio entre el costo de diseño y el costo de restricción es el uso de multi-resolución piramidal [Dougherty *et al.*, 2001]. En este caso se define un conjunto anidado de ventanas $\mathcal{V} = \{W_1, W_2, \ldots, W_q\}$, tal que W_1 es la ventana de mayor tamaño, o resolución, y W_q es la ventana de menor tamaño, usualmente conteniendo sólo un píxel. Por lo tanto, se tiene que $W_1 \supset W_2 \supset \ldots \supset W_q$ y $|W_1| > |W_2| > \ldots > |W_q|$. Al apilar las ventanas de $\mathcal{V}$ empezando en la base por la ventana más grande W_1 y terminando en la cima con la ventana más pequeña W_q se forma una pirámide (Figura 2.12). De aquí el nombre de multi-resolución piramidal. Se denota mediante $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_q$ a las configuraciones que se pueden observar a través de las ventanas $W_1, W_2, \ldots, W_q$, respectivamente, donde $\mathbf{X} = \mathbf{Z}_1$. Observar una imagen mediante una pirámide de ventanas es equivalente a aplicar una secuencia de transformaciones $\xi_1, \xi_2, \ldots, \xi_q$ que eliminan 0, $|W_2| - |W_1|, \ldots, |W_q| - |W_1|$ componentes, respectivamente, del vector $\mathbf{X}$, donde ξ_1 es la transformación identidad tal que $\mathbf{Z}_1 = \xi_1(\mathbf{X})$ con $\mathbf{Z}_1 = \mathbf{X}$.

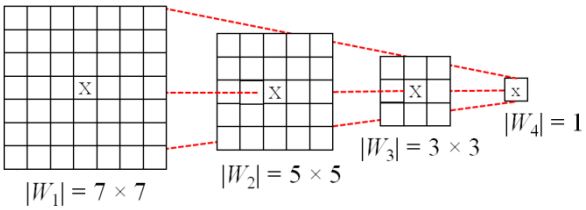


Figura 2.12. Ilustración de una pirámide compuesta por 4 ventanas. El píxel a procesar se marca con una x.

En la etapa de entrenamiento de multi-resolución piramidal se utiliza la regla plug-in para estimar los valores de las probabilidades condicionales $\mathbb{P}(Y = 1|\mathbf{Z})$ para las configuraciones observadas con cada ventana de $\mathcal{V}$. En base a estas estimaciones se construyen q tablas de búsqueda de la forma $(\mathbf{Z}, \psi(\mathbf{Z}))$, con una tabla de búsqueda por cada ventana. La restricción

de multi-resolución piramidal consiste en la estimación de las probabilidades condicionales con ventanas de menor tamaño con respecto a W_1. Para la etapa de testeo, y cada píxel a procesar, primero se obtiene la configuración $\mathbf{X}$ con la ventana W_1 y luego la configuración $\mathbf{Z}_1 = \xi_1(\mathbf{X})$. Si se cuenta con una estimación de la probabilidad $\mathbb{P}(Y = 1|\mathbf{Z}_1)$ en la tabla de búsqueda correspondiente a W_1, entonces se define $\psi_N(\mathbf{X}) = \psi(\mathbf{Z}_1)$. En el caso contrario, se aplica este mismo procedimiento para la segunda ventana W_1, y así sucesivamente hasta poder definir un valor para $\psi_N(\mathbf{X})$. Si esto no es posible incluso con la ventana W_q, entonces se asigna a $\psi_N(\mathbf{X})$ el nivel de gris más observado en las imágenes ideales correspondiente a la ventana conjunto vacío: $W_{q+1} = \varnothing$.

En la propuesta original de multi-resolución piramidal es el diseñador el que debe definir las ventanas que conforman la pirámide a utilizar durante el diseño. Por otro lado, en [Vaquero *et al.*, 2005] se propone un criterio para seleccionar automáticamente una pirámide dado un conjunto de pirámides definido *a priori* por el diseñador. Este criterio está basado en seleccionar la pirámide con menor entropía condicional media de Shannon. Esta entropía se calcula a partir de todas las estimaciones de las probabilidades condicionales para todas las ventanas de cada pirámide a testear. Para casos ambiguos donde la entropía condicional media de dos o más pirámides es muy baja, se propone usar el error empírico sobre el conjunto de entrenamiento para la selección de la pirámide a ser usada para el diseño. La idea detrás de esta propuesta es seleccionar la pirámide, de un conjunto de pirámides, cuya distribución conjunta inducida sea la más informativa. Es decir, se trata de seleccionar la pirámide con una cantidad mínima de casos en los que las probabilidades condicionales estimadas para cada ventana sean cercanas a 0.5. A nivel general, para este método los resultados serán buenos si la distribución conjunta inducida por la pirámide seleccionada es similar a la distribución conjunta real.

A pesar de las interesantes propiedades teóricas que posee multi-resolución piramidal, el principal inconveniente que se enfrenta a nivel práctico es su alto costo computacional tanto a nivel de uso de memoria como a nivel de tiempo de entrenamiento y testeo. Este problema se vuelve muy crítico para ventanas de mediano y gran tamaño e imágenes en escala de grises o color. Se debe tener presente también que en este caso se almacena una tabla de búsqueda por cada ventana que conforma la pirámide en cuestión. Por esta razón, esta técnica de diseño ha sido utilizada en los trabajos antes mencionados únicamente considerando ventanas pequeñas. Los casos experimentados incluyen la segmentación de objetos en imágenes binarias sintéticas contaminadas con ruido y el reconocimiento de dígitos en imágenes binarias de códigos postales.

Envelope: En el diseño por envelope, o envolvente, la restricción consiste en asumir que el operador a diseñar se encuentra contenido entre dos W-operadores diseñados heurísticamente por el diseñador [Barrera *et al.*, 2000]. Sea ψ el operador diseñado de manera automática, y α y β los operadores diseñados de manera heurística, llamado envelope. En este método de diseño se tiene que $\alpha \leq \psi \leq \beta$. El costo de diseño de W-operadores usando envelopes es bajo siempre y cuando α y β no estén muy alejados el uno del otro [Brun *et al.*, 2004]. El costo de restricción es nulo en el caso donde el operador óptimo ψ_{opt} está contenido entre α y β. En el caso extremo, si $\alpha(\mathbf{X}) = 0$ y $\beta(\mathbf{X}) = 1$ $\forall \mathbf{X} \in \mathcal{X}$,

entonces no existe restricción. El punto clave de este método consiste en definir los operadores α y β de tal modo que el costo de la restricción no sea demasiado alto. Desafortunadamente, esta tarea no es para nada trivial y además demanda de experiencia del diseñador.

A nivel práctico, el método por envelope involucra tanto el diseño heurístico del envelope α y β como de un W-operador no restringido ψ_0 usando, por ejemplo, la regla plug-in. Luego, para una configuración arbitraria **X**, el valor de la función característica $\psi_N(\mathbf{X})$ del W-operador envelope se define como: $\psi_N(\mathbf{X}) = \alpha(\mathbf{X})$ si $\psi_0(\mathbf{X}) < \alpha(\mathbf{X})$, $\psi_N(\mathbf{X}) = \psi_0(\mathbf{X})$ si $\alpha(\mathbf{X}) \leq \psi_0(\mathbf{X}) \leq \beta(\mathbf{X})$, y $\psi_N(\mathbf{X}) = \beta(\mathbf{X})$ si $\psi_0(\mathbf{X}) > \beta(\mathbf{X})$. Por lo tanto, el diseño envelope puede ser considerado como un híbrido entre un diseño heurístico en el que se define α y β y un diseño automático en el que se define ψ_0. En [Brun *et al.*, 2003] se propone el diseño automático de ψ_0 usando multi-resolución piramidal y en [Brun *et al.*, 2004] se presenta una extensión de este método de diseño para el caso de imágenes en escala de grises.

Aplicaciones de la restricción envelope a imágenes binarias incluyen la detección de bordes en imágenes sintéticas contaminadas con artefactos de ruido, restauración de objetos a partir de sus bordes deteriorados con ruido, detección de texturas y reconocimiento de caracteres [Barrera *et al.*, 2000]. El método usado para diseñar el W-operador ψ_0 previo a la aplicación de la restricción envelope es el algoritmo ISI. Al comparar los resultados del envelope combinado con el algoritmo ISI (método híbrido) con respecto a los resultados del algoritmo ISI (método automático), se muestra que el diseño híbrido presenta un mejor desempeño que el método automático para conjuntos de imágenes entrenamiento menores a un cierto tamaño. Posteriormente, los dos algoritmos tienen un desempeño similar. Este resultado evidencia que, siempre y cuando la definición del envelope sea correcta, ésta ayuda a reducir de manera más rápida el costo de diseño en comparación con el método automático. Como ya se anotó anteriormente el problema es que no siempre resulta fácil definir un envelope, en cuyo caso resulta preferible usar un método puramente automático que con una cantidad suficiente de datos resulta en un desempeño similar al método híbrido. En [Brun *et al.*, 2004] se considera el filtrado de ruido tipo sal y pimienta en imágenes con 256 niveles de gris. En este caso se usan ventanas espaciales de 3×3 y 5×5 píxeles.

2.8.2. Restricciones para Diseño de W-operadores en Escala de Grises

Filtros Stack: Una de las primeras técnicas propuestas para el diseño automático de W-operadores para imágenes en escala de grises consiste en el uso de la técnica de stacking o apilamiento de operadores binarios [Coyle y Lin, 1988]. La clase de W-operadores en escala de grises que pueden ser descompuestos en un conjunto de W-operadores binarios se denomina *filtros stack*. Un ejemplo de esta clase de operadores es el filtro mediana. Para este caso, el resultado del filtro mediana de una imagen en escala de grises es igual a la suma de los filtros mediana aplicados sobre las imágenes binarias resultantes de umbralar la imagen original en cada nivel gris perteneciente a su rango [Hirata, 2008].

Sean las imágenes observada *O* e ideal *I*, donde *O* e *I* pueden ser imágenes en escala de grises u *O* una imagen en escala de grises e *I* una imagen binaria. Sea la transformación de

umbralamiento ξ_k: $L \rightarrow \{0,1\}$ definida como $\xi_k(a) = 1$ si $a \geq k$ o caso contrario $\xi_k(a) = 0$, donde $0 \leq k \leq |L|$ y L $\{0,1,\ldots,l - 1\}$. Aplicando la transformación ξ_k a cada píxel de las imágenes del par de entrenamiento (O,I) se obtiene el par de imágenes binarias (O_k,I_k) para un umbral k. Si I es una imagen binaria, entonces ξ_k se aplica únicamente a la imagen observada O. Usando varios valores para el umbral k se puede obtener el conjunto de imágenes binarias $\{(O_1,I_1),..., (O_m,I_m)\}$. En base a este conjunto y una ventana W, se puede diseñar un W-operador binario ζ_N. Este operador puede o no estar sujeto a alguna restricción. El operador morfológico ψ_N sobre las imágenes en escala de grises se define mediante $\psi_N(\mathbf{X}) = \sum_{k=1}^{m}(\zeta_N(\mathbf{X}^{(k)}))$, donde $\mathbf{X}^{(k)} = (\xi_k(X_1),\ldots,\xi_k(X_n))$. Para el caso donde la imagen ideal I es una imagen binaria, ψ_N se define mediante una votación de los m valores de ζ_N.

Dado que ψ_N está definido mediante un único operador binario, ζ_N, entonces ψ_N es también un operador binario actuando sobre imágenes en escala de grises. Adicionalmente, dado un determinado tamaño de ventana, el espacio de configuraciones para operadores binarios es mucho menor que el espacio de configuraciones para operadores en escala de grises. Por lo tanto, el costo de diseño de un W-operador diseñado usando la técnica de stacking es, generalmente, menor que el costo de diseño de W-operadores en escala de grises. Sin embargo, existe también un incremento en el costo de restricción [Kuosmanen y Astola, 1995], [Astola y Kuosmanen, 1999]. W-operadores diseñados mediante esta técnica han sido testeados en aplicaciones de filtrado de ruido impulsivo y ruido speckle usando ventanas formadas por un total de 17 píxeles [Hirata *et al.*, 1999]. El principal inconveniente de esta técnica es el alto costo computacional que implica el diseño de los W-operadores binarios que forman el stack de operadores, especialmente para ventanas de mediano y largo tamaño e imágenes con una cantidad significativa de niveles de gris.

Filtros Aperture: Otra manera de reducir el tamaño del espacio $\mathcal{X}$ consiste en limitar o disminuir el número de niveles de gris de cada configuración de ventana a un rango de valores predefinido, restricción que toma el nombre de aperture. Los W-operadores diseñados en base a esta restricción se denominan filtros aperture [Hirata *et al.*, 2000c]. Sea una configuración $\mathbf{X} = (X_1,\ldots,X_n)$ y un nuevo conjunto de niveles de gris $K = \{-k,-k+1,...,k\}$ conteniendo un número de $2k + 1$ niveles de gris con $k \in \mathbb{Z}^+$ y $k << |L|$. Una configuración aperture $\mathbf{Z} = (Z_1,\ldots,Z_n)$ se obtiene aplicando la transformación de reducción de niveles de gris ξ: $L^n \rightarrow K^n$ al vector $\mathbf{X}$ para obtener $\mathbf{Z}$: $\mathbf{Z} = \xi(\mathbf{X})$. Cada componente Z_i, con $i = 1,\ldots,n$, de $\mathbf{Z}$ se obtiene mediante

$$Z_i = min(max(-k, X_i - z), k), \tag{2.26}$$

donde z es un escalar que depende de las componentes de la configuración $\mathbf{X}$: $z = z(\mathbf{X})$. Por ejemplo, se puede definir a z como la mediana entre las componentes de la configuración $\mathbf{X}$: $z = median(X_1,\ldots,X_n)$. También se puede asignar a z el valor correspondiente al píxel para el cual se obtuvo la configuración $\mathbf{X}$. Nótese que en este método de diseño la restricción aumenta a medida que se disminuye el valor de k. Es importante también mencionar que en [Hirata *et al.*, 2000c] se postula la definición de

filtros aperture considerando que tanto las imágenes ideales como las imágenes observadas son imágenes en escala de grises.

Para una ventana espacial *W*, una ventana de rango $K = \{-k,-k+1,\ldots,k\}$ y una imagen observada *O*, a nivel práctico la definición anterior es equivalente primero a trasladar espacialmente *W* por *t* (Figura 2.13-a). Segundo, obtener la configuración de ventana **X**. Tercero, trasladar verticalmente por *z* la configuración **X**, lo que implica restar de cada componente de **X** el valor *z* (Figura 2.13-b y -c). Cuarto, proyectar hacia la parte superior o inferior del producto Cartesiano *W*×*K*, llamado aperture, los valores que caen por encima o por debajo de *K*, respectivamente (Figura 2.13-d). Finalmente, se registra en el vector **Z** los valores observados a través del aperture *W*×*K*. En base a las configuraciones registradas de esta manera, se puede diseñar un filtro aperture usando por ejemplo la regla plug-in. También se ha combinado la restricción aperture con la restricción envelope para el diseño de W-operadores [Brun *et al.*, 2004]. Otra idea consiste en el uso de multi-resolución tanto a nivel de espacio como de rango [Hirata *et al.*, 2006]. También se ha propuesto combinar la información proporcionada por un conjunto predefinido de apertures para diseñar W-operadores en base a la regla plug-in para el procesamiento digital de señales [Green *et al.*, 2003]. La idea en este caso es que cada aperture, en función de su forma, busque un cierto tipo de patrón en la señal a procesar.

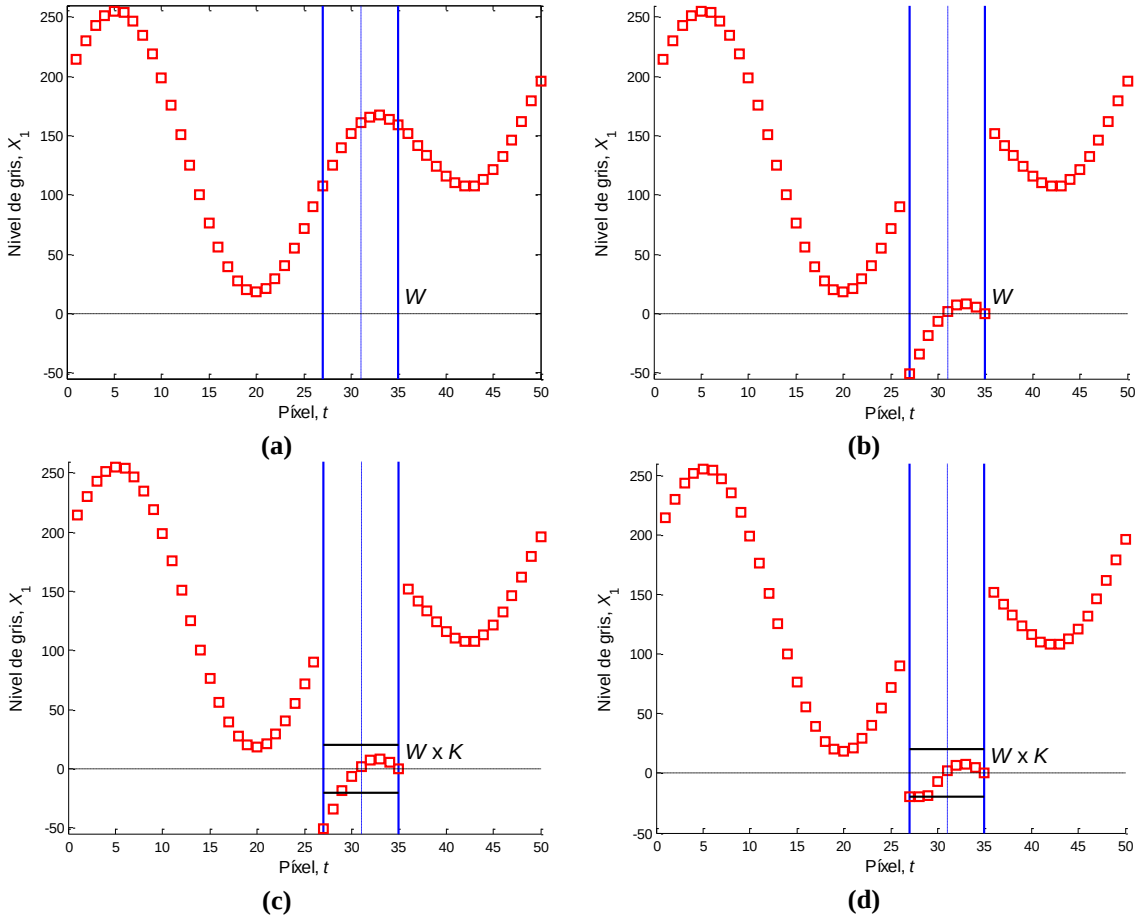


Figura 2.13. Ilustración del proceso para obtener configuraciones aperture para una señal unidimensional. (**a**) Traslación de la ventana espacial W por $t = 32$. (**b**) Traslación vertical de la configuración de ventana **X** = (108,125,140,152,161,166,167,164,159) por su mediana $z = 159$. (**c**) Proyección de los niveles de gris que caen fuera de la ventana de rango $K = \{-20,-19,\ldots,20\}$ a los valores de -20 y 20. (**d**) Configuración aperture **Z** = (-20,-20,-19,-7,2,7,8,5,0).

A pesar del bien fundamentado sustento matemático bajo el cual se postula la restricción aperture son muy pocas sus aplicaciones prácticas existentes. Los experimentos realizados incluyen tareas de deblurring de imágenes sintéticas conteniendo sólo 16 niveles de gris y usando ventanas espaciales de entre 3×3 y 5×5 píxeles y valores de *k* para las ventanas de rango entre 3 y 5 [Hirata *et al.*, 2000c]. A nivel general, la restricción aperture permite simplificar notablemente a nivel computacional y estadístico el diseño de W-operadores; sin embargo, sigue siendo un problema la técnica empleada para diseñar un W-operador en base a configuraciones aperture.

El diseño de filtros aperture aumenta su complejidad a medida que se aumenta el tamaño de la ventana espacial y/o el tamaño de la ventana de rango. En estas circunstancias, el uso de la regla plug-in para el diseño de filtros aperture resulta inapropiado, pues los valores de frecuencia para una configuración aperture observada son usualmente muy pequeños, e incluso iguales a 1, con lo cual se vuelve imposible obtener una buena estimación de las probabilidades condicionales. Peor es el caso en el que no se observan algunas configuraciones aperture en el escaneo de las imágenes de entrenamiento. Una situación similar se tiene para cuando se emplea multi-resolución piramidal para el diseño de filtros aperture. Aquí la función característica se termina definiendo en su gran mayoría en base a las configuraciones cercanas a la cima de la pirámide (ventanas de baja resolución). En este escenario, el costo de estimación es muy bajo, pero el costo de restricción es demasiado elevado, de tal manera que la performance del filtro diseñado resulta siendo mala.

Uso de Subventanas de Pequeño Tamaño: Como se analizó para el caso de imágenes binarias, una forma de reducir el espacio de configuraciones es disminuyendo el tamaño de la ventana inicial usada para el diseño de W-operadores. En [Martins-Jr *et al.*, 2005], los autores proponen encontrar una subventana óptima a partir de los potenciales subconjuntos de ventanas que se pueden formar con los puntos que componen una ventana *W* definida *a priori* por el diseñador. Aquí la idea es encontrar la subventana cuya entropía condicional media de Shannon sea la mínima. La entropía condicional media de Shannon para cada subventana candidata se calcula a partir de las probabilidades condicionales de las configuraciones observadas con dicha subventana y usando la regla plug-in. Dado que aquí se resuelve un problema que involucra un número extremadamente grande de combinaciones, o potenciales subventanas candidatas, el uso de una búsqueda exhaustiva; es decir, testear todas las posibles subventanas resulta impráctico. En su lugar se emplea un algoritmo de búsqueda automático que, en cada iteración, aumenta el número de puntos de la subventana a ser obtenida. Adicionalmente, para disminuir la cantidad de probabilidades condicionales a estimar se propone aplicar una cuantización de los niveles de gris de las configuraciones obtenidas a partir de cada subventana.

Este método se aplica para la clasificación de 9 tipos diferentes de texturas en imágenes con 256 niveles de gris. La ventana inicial contiene 7×7 píxeles y la cantidad de niveles de gris usados por cada componente de una configuración de ventana varían entre 2, 4 y 8. Las subventanas resultantes contienen entre 6 y 15 píxeles de un total inicial de 49 píxeles. La característica principal de las ventanas resultantes es que son dispersas. Como es de esperarse, los resultados mejoran a medida que se utiliza una mayor cantidad de niveles de gris para cada configuración. Sin embargo, esto también provoca un aumento en la complejidad del proceso de búsqueda de la ventana óptima, con lo cual el costo computacional se eleva notablemente. Como nota adicional, en este caso un aumento del número de niveles de gris por cada configuración no siempre mejora la calidad de los

resultados, pues la cantidad de ejemplos de entrenamiento es fija (escenario de la curva de error "U").

2.8.3. Restricciones para Diseño de W-operadores Color

En [Martins-Jr *et al.*, 2006] los autores extienden la idea descrita en la parte final de la sección anterior para el caso de imágenes color RGB. En este caso, para una ventana W de n puntos, los vectores $\mathbf{X}$ están formados por un número total de $3n$ componentes. Cada vector $\mathbf{X}$ contiene los valores de intensidad de los tres canales que componen las imágenes color RGB. Aquí se define *a priori* la estructura del vector $\mathbf{X}$, la cual no influye en el diseño del W-operador siempre y cuando ésta se mantenga invariable durante las etapas de entrenamiento y testeo.

Este método se testea en el mismo problema descrito anteriormente, pero ahora usando imágenes color RGB. Los resultados obtenidos muestran una reducción del error de los W-operadores diseñados en comparación con aquellos diseñados a partir de las imágenes en escala de grises. Esta reducción se atribuye al aumento del poder discriminativo de los W-operadores diseñados gracias al uso de la información de color. Un segundo caso de aplicación consiste en la segmentación de los anillos que forman el tronco de un árbol en imágenes de cortes transversales. Una vez más, los W-operadores diseñados en base a la información de color presentan un mejor desempeño comparado con los W-operadores en niveles de gris, W-operadores diseñados con ventanas de 1 píxel y clasificadores diseñados en base a perceptrones multicapa [Bishop, 2005].

2.9. Diagnóstico del Estado del Arte

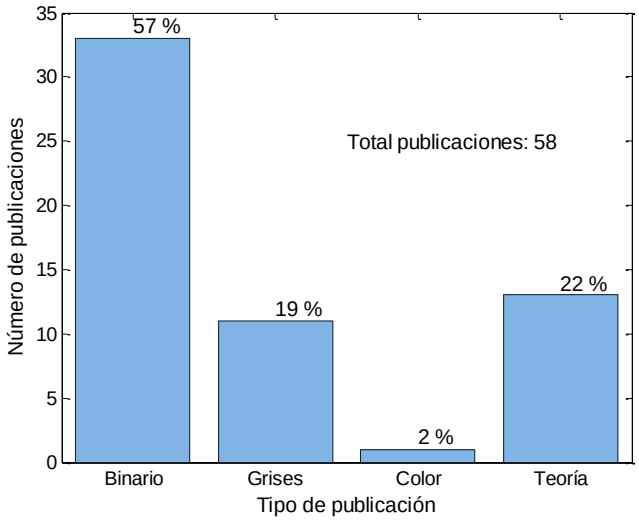


Figura 2.14. Estadística de publicaciones científicas relacionadas con el diseño automático de W-operadores: cantidad de publicaciones en función del tipo de contribución.

En la Figura 2.14 se presentan las estadísticas obtenidas a partir de la descripción y análisis de los métodos de diseño automático de W-operadores presentados en la sección anterior. Para el resumen estadístico se consideran 4 categorías de publicaciones que son: desarrollos teóricos y métodos de diseño de W-operadores para imágenes binarias, imágenes en escala de grises, e imágenes color. Es importante mencionar que, la categorización de cada publicación se realizó en función de los objetivos propuestos en su introducción, lo cual no

significa para nada que las publicaciones categorizadas como aplicaciones no contengan desarrollos teóricos o viceversa. De hecho, la mayoría de las publicaciones que involucran un avance práctico tienen un sólido fundamento teórico sobre el cual se basa el método de diseño o aplicación presentado en cada paper. De un total de 58 publicaciones revisadas, un 57% está dedicado al desarrollo de métodos de diseño de W-operadores para imágenes binarias, un 22% corresponde a desarrollos teóricos, seguido por un 19% de publicaciones donde se proponen nuevos métodos de diseño de operadores para imágenes en escala de grises, y apenas un 2% corresponde a métodos de diseño para el caso de imágenes color.

A nivel general, a pesar de la gran cantidad de trabajos desarrollados para el caso de imágenes binarias, uno de los problemas no resueltos consiste en el diseño automático de W-operadores para ventanas de gran tamaño. Para el caso de imágenes en escala de grises y color se requiere el desarrollo de nuevos métodos o la extensión de los existentes para ventanas de mediano y gran tamaño. A nivel práctico, el principal problema evidenciado es la carencia de aplicaciones que involucren imágenes reales. Si bien las imágenes o problemas sintéticos permiten evaluar ciertas propiedades teóricas de los operadores diseñados en un ambiente controlado, en problemas reales el diseñador no puede manipular el proceso o la distribución probabilística subyacente a un problema dado. En particular, en los trabajos revisados no se registra ninguna aplicación de W-operadores al problema de procesamiento digital de imágenes médicas. A nivel teórico, existe un importante marco de referencia sobre el cual se puede fundamentar el desarrollo de nuevos avances teóricos y/o nuevos métodos o algoritmos de diseño.

Centrando el análisis sobre la capacidad de estimación y generalización de la función característica, se puede notar el papel preponderante que ha cumplido la regla plug-in, especialmente para el caso de imágenes binarias. Sin embargo, el problema principal de esta regla es que por sí sola no tiene capacidad de generalización. Esto significa que, en base a esta regla, no se puede predecir el valor de la función característica para configuraciones no registradas en la etapa de testeo. Para esto es necesario definir algún tipo de heurística o generalización. Otra alternativa es el uso de la regla *k*NN que sí tiene capacidad de generalización. Sin embargo, esta capacidad de generalización se consigue a costa de un aumento en el costo computacional que involucra la búsqueda de los *k* vecinos más cercanos a una determinada configuración. Esta situación hace que, desde el punto de vista computacional, su uso se vuelva prohibitivo para problemas donde se requieren de ventanas de mediano y gran tamaño.

Con relación a la representación matemática y computacional de la función característica, la mayoría de métodos propuestos involucran el almacenamiento de grandes tablas de búsqueda o frecuencias que representan al operador diseñado. Esta situación se vuelve muy problemática para el caso de ventanas de mediano y gran tamaño, donde la cantidad de posibles configuraciones aumenta de manera exponencial. Otra alternativa propuesta consiste en usar una representación morfológica para el operador diseñado. Esto implica la búsqueda ya sea de los elementos estructurantes que forman la base o la colección de intervalos maximales para una secuencia de erosiones o para el operador sup-generador (u operador hit or miss), respectivamente. El problema aquí es doble, pues el tiempo de búsqueda de los elementos estructurantes o de la colección de intervalos maximales aumenta drásticamente cuando crece el espacio de configuraciones o cuando la complejidad de un problema es alta. Adicionalmente, para problemas complejos el número de elementos estructurantes o intervalos maximales puede ser muy grande.

En lo referente al diseño bajo restricciones, los teoremas de no free lunch afirman que no existen buenas o malas restricciones a nivel general, al igual que tampoco existen métodos de diseño que sean superiores a otros en términos de error. Para un problema determinado, el hecho de que una restricción funcione bien o mal depende de los costos de diseño y restricción. En lo relacionado al primero (costo de diseño), en la teoría formulada para W-operadores no existe un análisis formal de la relación entre el costo de diseño y la cantidad de ejemplos de entrenamiento disponibles. En lo relacionado al segundo (costo de restricción), la teoría garantiza que, para un problema determinado, si el operador óptimo está dentro de la restricción, entonces el costo de restricción es nulo. Sin embargo, dado que en la práctica, y sobre todo en problemas reales, no se conoce el proceso aleatorio subyacente a la generación de las imágenes, no existe forma de calcular este costo y por ende tampoco hay forma de saber *a priori* si una restricción será buena o mala para un problema determinado. Después de todo, si se conociera el proceso aleatorio que genera las imágenes, mucho de lo hasta ahora presentado no tendría sentido. Esto debido a que a partir de este proceso se podría encontrar tanto al operador óptimo como calcular su error exacto sin tener que definir restricciones ni algoritmos de diseño, ni tampoco métodos de estimación de error.

En función de este análisis se puede notar que *la aplicación directa de los métodos de diseño propuestos al campo del procesamiento digital de imágenes médicas no es posible*. Esto se debe a que la mayoría de equipos de imagenología médica entregan imágenes en escala de grises o color [Dougherty, 2009], para los cuales los métodos de diseño propuestos de W-operadores son escasos. Si bien la extensión de las definiciones de los métodos binarios para el caso de imágenes en escala de grises no representa, en la mayoría de casos, un problema a nivel teórico, sí lo es a nivel práctico. Esto último causado por el incremento dramático del costo computacional. Para el caso de imágenes multicanal, y en particular de imágenes color, el desafío se presenta tanto a nivel teórico como a nivel práctico, pues en este caso el valor de un píxel ya no es un escalar, sino un vector.

2.10. Anotaciones Finales

En este capítulo se han presentado y analizado exhaustivamente los métodos propuestos en la literatura científica para el diseño automático de W-operadores. Esta revisión y análisis inicia presentando a dos de los métodos más utilizados como son la regla plug-in y la regla *k*NN. Luego se presentó un análisis de las implicaciones de los teoremas de no free lunch que, en el contexto de esta tesis, afirman que todos los métodos de diseño de W-operadores tienen el mismo desempeño promedio evaluado mediante el costo 0-1. En base a este análisis, se concluyó que no hay razón para, *a priori*, seleccionar un método de diseño. Lo bien o mal que funciona un método de diseño para un problema particular depende de la similitud que existe entre su estructura probabilística y la estructura probabilística del problema en cuestión.

Posteriormente se analizó la representación computacional y morfológica de la función característica de un W-operador. En esta parte se presentaron y analizaron los métodos de diseño de W-operadores basados en una representación morfológica mediante operadores sup-generadores, erosiones, y operaciones hit or miss. Posteriormente, se abordó el diseño de W-operadores tratado como un problema de definición de restricciones. Se analizó el costo de diseño y costo de restricción del espacio de funciones características. Se determinó que el diseño de W-operadores usando restricciones consiste en conseguir un balance entre

el costo de restricción y el costo de diseño, sin que exista un método para, *a priori*, definir buenas restricciones.

Se propuso también una clasificación de las restricciones existentes en función de los tipos de imágenes para los que se utilizan. El orden seleccionado para su descripción y análisis tiene que ver con el orden histórico en el que fueron propuestos. En base a este análisis se concluye que no existen métodos de diseño de W-operadores aplicados al procesamiento de imágenes médicas. La mayoría de métodos existentes han sido aplicados a problemas que involucran imágenes sintéticas binarias con ventanas de pequeño y mediano tamaño. Para imágenes en niveles de gris y color, los métodos propuestos son escasos e inadecuados, desde el punto de vista computacional, para problemas que involucren ventanas medianas y grandes.

En función de la revisión bibliográfica y análisis presentado a lo largo del presente capítulo, de aquí en adelante en esta tesis se persiguen dos objetivos generales:

a) Desarrollar nuevos métodos de diseño automático de W-operadores considerando los problemas de definición de nuevas restricciones o extensión de alguna de las existentes, generalización, y representación matemática y computacional de la función característica. Aquí la definición del problema a resolver se realiza a través mediante una colección o conjunto de pares de imágenes observada e ideal (conjunto de imágenes de entrenamiento).

b) Testear los métodos propuestos en problemas de PDI que involucren el uso de imágenes reales, y en particular, en problemas que involucren el uso de imágenes médicas.

En los capítulos siguientes se presentan el paradigma propuesto y los principales avances teóricos y prácticos que se han alcanzado en esta tesis.

2.11. Referencias

[1] Y.S. Abu-Mostafa, M. Magdon-Ismail and H.T. Lin, *Learning from data*, AMLBook, Pasadena CA, 2012.

[2] J.T. Astola and P. Kuosmanen, "Representation and Optimization of Stack Filters," in *Nonlinear Filters for Image Processing*, E.R. Dougherty and J.T. Astola, Eds. Belingham: SPIE and IEEE Presses, 1999.

[3] A. Auger and O. Teytaud, "Continuous Lunches Are Free Plus the Design of Optimal Optimization Algorithms," *Algorithmica*, vol. 57, pp. 121-46, 2010.

[4] G. Banon and J. Barrera, "Minimal Representations for Translation-Invariant Set Mappings by Mathematical Morphology," *SIAM Journal on Applied Mathematics*, vol. 51, pp. 1782-98, 1991.

[5] G.J.F. Banon and J. Barrera, "Decomposition of mappings between complete lattices by mathematical morphology, Part I. General lattices," *Signal Processing*, vol. 30, pp. 299-327, 1993.

[6] J. Barrera, G.J.F. Banon and E.R. Dougherty, "Automatic design of morphological operators," in *Space, Structure and Randomness*, M. Bilodeau, F. Meyer, and M. Schmitt, Eds.: Springer, 2005, pp. 257-78.

[7] J. Barrera and M. Brun, "Translation invariant transformations of discrete random sets," *Proc. Computer Graphics, Image Processing, and Vision, 1998. Proceedings. SIBGRAPI '98. International Symposium on*, Rio de Janeiro, 450-5 (1998).

[8] J. Barrera, E.R. Dougherty and M. Brun, "Hybrid human–machine binary morphological operator design. An independent constraint approach," *Signal Processing*, vol. 80, pp. 1469-87, 2000.

[9] J. Barrera, E.R. Dougherty and N.S.T. Hirata, "Design of Optimal Morphological Operators Using Prior Filters," *Acta Stereologica*, vol. 16, pp. 193-200, 1997a.

[10] J. Barrera, E.R. Dougherty and N.S. Tomita, "Automatic Programming of Binary Morphological Machines by Design of Statistically Optimal Operators in the Context of Computational Learning Theory," *Electronic Imaging*, vol. 6, pp. 54–67, 1997b.

[11] J. Barrera, R. Terada, F. Da Silva and N. Tomita, "Automatic Programming of MMach's for OCR," in *Mathematical Morphology and its Applications to Image and Signal Processing*. vol. 5, P. Maragos, R. Schafer, and M. Butt, Eds.: Springer US, 1996, pp. 385-92.
[12] J. Barrera, N.S. Tomita, F.S. Correa da Silva and R. Terada, "Automatic programming of binary morphological machines by PAC learning," *Proc.* 233-44 (1995).
[13] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Cambridge, 2005.
[14] U.M. Braga-Neto and E.R. Dougherty, "Classification," in *Genomic Signal Processing and Statistics*. vol. 2, E.R. Dougherty, I. Shmulevich, J. Chen, and Z.J. Wang, Eds.: Hindawi Publishing Corporation, 2005, pp. 93-128.
[15] M. Brun, E.R. Dougherty, R. Hirata and J. Barrera, "Design of optimal binary filters under joint multiresolution–envelope constraint," *Pattern Recognition Letters*, vol. 24, pp. 937-45, 2003.
[16] M. Brun, R. Hirata, J. Barrera and E.R. Dougherty, "Nonlinear Filter Design Using Envelopes," *Journal of Mathematical Imaging and Vision*, vol. 21, pp. 81-97, 2004.
[17] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, pp. 21-7, 1967.
[18] E.J. Coyle and J.H. Lin, "Stack filters and the mean absolute error criterion," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, pp. 1244-54, 1988.
[19] L. Devroye, L. Györfi and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer, New York, 1996.
[20] M.M. Dornelles and N.S.T. Hirata, "A genetic algorithm based approach for combining binary image operators," *Proc. Pattern Recognition (ICPR), 2012 21st International Conference on*, 3184-7 (2012).
[21] E. Dougherty, "Optimal mean-absolute-error filtering of gray-scale signals by the morphological hit-or-miss transform," *Journal of Mathematical Imaging and Vision*, vol. 4, pp. 255-71, 1994.
[22] E. Dougherty and J. Barrera, "Pattern Recognition Theory in Nonlinear Signal Processing," *Journal of Mathematical Imaging and Vision*, vol. 16, pp. 181-97, 2002.
[23] E.R. Dougherty, "Optimal mean-square N-observation digital morphological filters: I. Optimal binary filters," *CVGIP: Image Understanding*, vol. 55, pp. 36-54, 1992a.
[24] E.R. Dougherty, "Optimal mean-square N-observation digital morphological filters: II. Optimal gray-scale filters," *CVGIP: Image Understanding*, vol. 55, pp. 55-72, 1992b.
[25] E.R. Dougherty, *Random Processes for Image and Video Processing*, IEEE Press, New York, 1999.
[26] E.R. Dougherty and J. Barrera, "Bayesian design of optimal morphological operators based on prior distributions for conditional probabilities," *Acta Steriologica*, vol. 16, pp. 167-74, 1997.
[27] E.R. Dougherty and J. Barrera, "Logical Image Operators," in *Nonlinear Filters for Image Processing*, E.R. Dougherty and J.T. Astola, Eds. Bellingham: SPIE and IEEE Press, 1999, pp. 1-60.
[28] E.R. Dougherty, J. Barrera, G. Mozelle, S. Kim and M. Brun, "Multiresolution Analysis for Optimal Binary Filters," *Journal of Mathematical Imaging and Vision*, vol. 14, pp. 53-72, 2001.
[29] E.R. Dougherty and R.P. Loce, "Optimal mean-absolute-error hit-or-miss filters: morphological representation and estimation of the binary conditional expectation," *Optical Engineering*, vol. 32, pp. 815-27, 1993.
[30] E.R. Dougherty and R.P. Loce, "Precision of morphological-representation estimators for translation-invariant binary filters: Increasing and nonincreasing," *Signal Processing*, vol. 40, pp. 129-54, 1994.
[31] E.R. Dougherty and R.P. Loce, "Optimal binary differencing filters: design, logic complexity, precision analysis, and application to digital document processing," *Journal of Electronic Imaging*, vol. 5, pp. 66-86, 1996.
[32] G. Dougherty, *Digital image processing for medical applications*, Cambridge Press, New York, 2009.
[33] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification*, 2001.
[34] M.M. Fraz, P. Remagnino, A. Hoppe, B. Uyyanonvara, A.R. Rudnicka, C.G. Owen and S.A. Barman, "Blood vessel segmentation methodologies in retinal images – A survey," *Computer Methods and Programs in Biomedicine*, vol. 108, pp. 407-33, 2012.
[35] A.C. Green, S. Marshall, D. Greenhalgh and E.R. Dougherty, "Design of multi-mask aperture filters," *Signal Processing*, vol. 83, pp. 1961-71, 2003.
[36] N.R. Harvey and S. Marshall, "The use of genetic algorithms in morphological filter design," *Signal Processing: Image Communication*, vol. 8, pp. 55-71, 1996.
[37] H.J.A.M. Heijmans, *Morphological Image Operators*, Academic Press, Boston, 1994.
[38] N.S.T. Hirata, "Stack Filters: From Definition to Design Algorithms," in *Advances in Imaging and Electron Physics*. vol. 152, P.W. Hawkes, Ed.: Elsevier, 2008, pp. 1-47.
[39] N.S.T. Hirata, "Multilevel Training of Binary Morphological Operators," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, pp. 707-20, 2009.
[40] N.S.T. Hirata, "Morphological Operator Design from Training Data," in *Innovations in Intelligent Image Analysis*. vol. 339, H. Kwaśnicka and L. Jain, Eds.: Springer Berlin Heidelberg, 2011, pp. 31-58.

[41] N.S.T. Hirata, J. Barrera and E.R. Dougherty, "Design of statistically optimal stack filters," *Proc. XII Brazilian Symposium on Computer Graphics and Image Processing*, Campinas, (1999).
[42] N.S.T. Hirata, J. Barrera, R. Terada and E.R. Dougherty, "The Incremental Splitting of Intervals Algorithm for the Design of Binary Image Operators," *Proc. Procedings of the Sixth International Symposium on Mathematical Morphology*, 219-28 (2002).
[43] N.S.T. Hirata and M.M. Dornelles, "The Use of High Resolution Images in Morphological Operator Learning," *Proc. Computer Graphics and Image Processing (SIBGRAPI), 2009 XXII Brazilian Symposium on*, 141-8 (2009).
[44] N.S.T. Hirata, E.R. Dougherty and J. Barrera, "Iterative design of morphological binary image operators," *Optical Engineering*, vol. 39, pp. 3106-23, 2000a.
[45] N.S.T. Hirata, E.R. Dougherty and J. Barrera, "A switching algorithm for design of optimal increasing binary filters over large windows," *Pattern Recognition*, vol. 33, pp. 1059-81, 2000b.
[46] N.S.T. Hirata, R. Hirata and J. Barrera, "Basis Computation Algorithms," *Proc. Eight International Symposium on Mathematical Morphology*, Rio de Janeiro, 15-26 (2007).
[47] R. Hirata, M. Brun, J. Barrera and E.R. Dougherty, "Aperture filters: theory, application, and multiresolution analysis," in *Advances in Nonlinear Signal and Image Processing*: Hindawi Publishing, 2006, pp. 15-48.
[48] R. Hirata, E.R. Dougherty and J. Barrera, "Aperture filters," *Signal Processing*, vol. 80, pp. 697-721, 2000c.
[49] H. Joo, R.M. Haralick and L.G. Shapiro, "Toward the automatic generation of mathematical morphology procedures using predicate logic," *Proc. Computer Vision, 1990. Proceedings, Third International Conference on*, 156-65 (1990).
[50] V.G. Kamat and E.R. Dougherty, "Multiresolution Bayesian design of binary filters," *Electronic Imaging*, vol. 9, pp. 283-95, 2000.
[51] M. Karnaugh, "The map method for synthesis of combinational logic circuits," *American Institute of Electrical Engineers, Part I: Communication and Electronics, Transactions of the*, vol. 72, pp. 593-9, 1953.
[52] H.Y. Kim, "Quick construction of efficient morphological operators by computational learning," *Electronics Letters*, vol. 33, pp. 286-7, 1997.
[53] H.Y. Kim, "Binary Operator Design by k-Nearest Neighbor Learning with Application to Image Resolution Increasing," *International Journal on Imaging Systems and Technology*, vol. 11, pp. 331-9, 2000.
[54] H.Y. Kim and P.S.L.M. Barreto, "Fast binary image resolution increasing by k-nearest neighbor learning," *Proc. Image Processing, 2000. Proceedings. 2000 International Conference on*, 327-30 vol.2 (2000).
[55] P. Kuosmanen and J.T. Astola, "Optimal Stack Filters Under Rank Selection and Structural Constraints," *Signal Processing*, vol. 41, pp. 309-38, 1995.
[56] R.P. Loce and E.R. Dougherty, *Enhancement and restoration of digital documents: Statistical design of nonlinear algorithms*, Bellingham, 1997.
[57] P. Maragos and R.W. Schafer, "Morphological filters: part I: their set-theoretic analysis and relations to linear shift invariant filters," *IEEE Transactions in Acoustics, Speech and Signal Processing*, vol. 35, pp. 1153- 69 1987.
[58] D.C. Martins-Jr, R.M. Cesar-Jr and J. Barrera, "Automatic Window Design for Gray-Scale Image Processing Based on Entropy Minimization," in *Progress in Pattern Recognition, Image Analysis and Applications*. vol. 3773, A. Sanfeliu and M. Cortés, Eds.: Springer Berlin Heidelberg, 2005, pp. 813-24.
[59] D.C. Martins-Jr, R.M. Cesar-Jr, J. Barrera, M.A.S. Liziér and L.G. Nonato, "W-operator window design for color texture classification," *Proceedings of XIX Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), Manaus/AM-Brazil.(abstract for poster presentation)*, vol. pp. 2006.
[60] G. Matheron, *Random Sets and Integral Geometry*, New York, 1975.
[61] A.V. Mathew, E.R. Dougherty and V. Swarnakar, "Efficient derivation of the optimal mean-square binary morphological filter from the conditional expectation via a switching algorithm for discrete power-set lattice," *Circuits, Systems and Signal Processing*, vol. 12, pp. 409-30, 1993.
[62] E.J. McCluskey, "Minimization of Boolean function," *The Bell System Technical Journal* vol. 35, pp. 1417–44, 1956.
[63] K.P. Murphy, *Machine Learning: A probabilistic perspective*, MIT Press, Cambridge MA, 2012.
[64] W.V. Quine, "The problem of simplifying truth functions," *The American Mathematical Monthly*, vol. 59, pp. 521–31, 1952.
[65] M.I. Quintana, R. Poli and E. Claridge, "Morphological algorithm design for binary images using genetic programming," *Genetic Programming and Evolvable Machines*, vol. 7, pp. 81-102, 2006.

[66] P. Salembier, “Structuring element adaptation for morphological filters,” *Journal of Visual Communication and Image Representation*, vol. 3, pp. 115-36, 1992.
[67] C.S. Santos, N.S.T. Hirata and R. Hirata, “An Information Theory framework for two-stage binary image operator design,” *Pattern Recognition Letters*, vol. 31, pp. 297-306, 2010.
[68] O.V. Sarca, E.R. Dougherty and J. Astola, “Secondarily constrained Boolean filters,” *Signal Processing*, vol. 71, pp. 247-63, 1998.
[69] O.V. Sarca, E.R. Dougherty and J. Astola, “Two-stage binary filters,” *Journal of Electronic Imaging*, vol. 8, pp. 219-32, 1999.
[70] M. Schmitt, “Mathematical morphology and artificial intelligence: An automatic programming system,” *Signal Processing*, vol. 16, pp. 389-401, 1989.
[71] J. Serra, “Image Analysis and Mathematical Morphology,” vol. I, pp. 1982.
[72] J. Staal, M.D. Abràmoff, M. Niemeijer, M.A. Viergever and B. Ginneken, “Ridge-Based Vessel Segmentation in Color Images of the Retina,” *IEEE Transactions On Medical Imaging*, vol. 23, pp. 501-9, 2004.
[73] D.A. Vaquero, J. Barrera and R. Hirata, “A Maximum-Likelihood Approach for Multiresolution W-Operator Design,” *Proc. Computer Graphics and Image Processing, 2005. SIBGRAPI 2005. 18th Brazilian Symposium on*, 71-8 (2005).
[74] D.H. Wolpert, “The lack of a priori distinctions between learning algorithms,” *Neural Computation*, vol. 8, pp. 1341-90, 1996.
[75] D.H. Wolpert, “The Supervised Learning No-Free-Lunch Theorems,” in *Soft Computing and Industry*, R. Roy, M. Köppen, S. Ovaska, T. Furuhashi, and F. Hoffmann, Eds.: Springer London, 2002, pp. 25-42.
[76] D.H. Wolpert and W.G. Macready, “No free lunch theorems for optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67-82, 1997.
[77] D.H. Wolpert and W.G. Macready, “Coevolutionary free lunches,” *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 721-35, 2005.
[78] X.S. Yang, “International Journal of Artificial Intelligence Tools,” *Free Lunch or No Free Lunch: That is not Just a Question?*, vol. 21, pp. 1-13, 2012.

CAPÍTULO III

Reconocimiento de Patrones y Paradigma Propuesto

Resumen: En este capítulo se propone un nuevo paradigma para el diseño automático de W-operadores basado en reconocimiento de patrones. El capítulo empieza con una breve descripción de las principales características y aplicaciones del reconocimiento de patrones y su relación con el aprendizaje computacional. Posteriormente se definen los conceptos de clasificador y error de clasificación, y se plantea el problema de diseño estadístico de clasificadores. En base a estas definiciones se propone definir y representar a un operador de ventana o W-operador a través de un sistema de reconocimiento de patrones. Luego se realiza una breve comparación histórica del desarrollo teórico de W-operadores y reconocimiento de patrones. A continuación se presenta la teoría de aproximación y generalización de Vapnik y Chervonenkis para diseño estadístico de clasificadores usando ejemplos de entrenamiento. En base a esta teoría se realiza una comparación entre los modelos de clasificación paramétricos y los modelos no paramétricos. Finalmente se describe a las redes neuronales artificiales tipo feed-forward que son el modelo paramétrico propuesto en esta tesis para el diseño automático de W-operadores.

3.1. Introducción

El *reconocimiento de patrones*, al igual que el aprendizaje computacional, es una rama de la inteligencia artificial dedicada a la búsqueda automática de *estructuras*, *patrones*, o *regularidades*, en conjuntos de datos u observaciones [Bishop, 2006], [Duda *et al.*, 2001]. A partir de las estructuras detectadas, o regularidades encontradas, el objetivo es realizar predicciones con la mayor exactitud posible sobre el comportamiento futuro de nuevos conjuntos de datos u observaciones (objetivo principal del método científico). Nótese que aquí se asume implícitamente que los datos tienen una estructura, patrón, o regularidad que se puede encontrar.

El reconocimiento de patrones involucra a problemas de clasificación, regresión, agrupamiento o clustering [Dalton *et al.*, 2013, 2014], estimación de densidad, y compresión o reducción de la dimensión de los datos [Bishop, 2006]. Es muy difícil establecer una frontera clara entre el reconocimiento de patrones y el aprendizaje computacional, pues las dos tienen el mismo dominio de problemas y cuando se habla de la una es muy común también referirse a la otra, incluso aunque no sea manera explícita. Tal es así que en muchos casos se termina considerando al reconocimiento de patrones y al aprendizaje computacional como términos sinónimos.

No es un objetivo de esta tesis el proponer una frontera de división entre estas dos áreas de la inteligencia artificial, sino solamente establecer un criterio que permita diferenciarlas. Aprendizaje computacional involucra una colección de algoritmos o métodos de diseño de sistemas para encontrar patrones en conjuntos de datos. Mientras tanto, el reconocimiento de patrones se refiere al análisis teórico del escenario donde operan y el comportamiento de dichos algoritmos para tales escenarios. De modo más simple, el aprendizaje computacional engloba una variedad de modelos para hacer predicciones basadas en la extracción automática de información a partir de las observaciones; mientras que el reconocimiento de patrones permite estudiar el funcionamiento y las características del problema donde se puede usar cada modelo [Devroye *et al.*, 1996], [Bishop, 2006].

3.2. Reconocimiento de Patrones y W-operadores

En esta sección se presenta la teoría que soporta el paradigma propuesto en esta tesis para el diseño automático de W-operadores.

3.2.1. Definiciones de Reconocimiento de Patrones

Son claves para el reconocimiento de patrones los conceptos de observación y clasificador. Una observación es un vector formado por un conjunto de medidas numéricas y un clasificador es una función que asigna una etiqueta a cada observación. Formalmente, y con un abuso de notación, se define a una *observación* como un vector $\mathbf{X} = (X_1,...,X_n)$ compuesto por n variables aleatorias llamadas *características*, *descriptores*, o en inglés *features*. El conjunto, o espacio, de todas las posibles observaciones se denota mediante $\mathcal{X}$. Usualmente, $\mathcal{X}$ es un subconjunto del espacio euclidiano $\mathbb{R}^n$. Sea Y una variable aleatoria categórica con dominio en $\mathcal{Y} = \{0,1,\ldots,c - 1\}$, con $c \in \mathbb{Z}^+$, donde cada elemento representa una *etiqueta*, *clase*, o *categoría*. En este caso, Y denota la clase o categoría a la que pertenece una observación $\mathbf{X}$. Si $\mathcal{Y}$ es un conjunto infinito de números reales o un intervalo, el problema en cuestión se denomina *regresión*. En esta tesis se considera el problema de clasificación. El espacio de todas las posibles funciones con dominio en $\mathcal{X}$ y rango en $\mathcal{Y}$ se denota mediante $\mathcal{Y}^{\mathcal{X}}$. Adicionalmente, sea $\Pr(\mathbf{X},Y)$ la distribución que gobierna el espacio de observaciones y etiquetas $\{\mathcal{X}\times\mathcal{Y}\}$.

Un *clasificador* es una función $\varphi \in \mathcal{Y}^{\mathcal{X}}$ que asigna una etiqueta $\varphi(\mathbf{X}) \in \mathcal{Y}$ al vector de características u observación $\mathbf{X} \in \mathcal{X}$. Por lo tanto, φ es una función que estima el valor de Y a partir de la observación de $\mathbf{X}$. La obtención de vectores de características y la reducción de su dimensión se denominan *extracción de características* y *selección de características*, respectivamente. En inglés estas tareas se denominan *feature extraction* y *feature selection*, respectivamente. Es importante remarcar que estas tareas son parte de la estructura del clasificador [Duda *et al.*, 2001], [Murphy, 2012].

Dado un *clasificador* $\varphi \in \mathcal{Y}^{\mathcal{X}}$, su *error de clasificación* $\varepsilon(\varphi)$ está dado por la probabilidad de que $\varphi(\mathbf{X})$ e Y sean diferentes para cualquier par $(\mathbf{X},Y)$:

$$\varepsilon(\varphi) = \mathbb{P}(\varphi(\mathbf{X}) \neq Y), \tag{3.1}$$

donde la probabilidad $\mathbb{P}(\varphi(\mathbf{X}) \neq Y)$ se calcula a partir de la distribución conjunta $\Pr(\mathbf{X},Y)$. En palabras, el error de un clasificador φ está dado por la probabilidad de observar un par $(\mathbf{X},Y)$, donde la etiqueta predicha $\varphi(\mathbf{X})$ para $\mathbf{X}$ sea distinta de su etiqueta asociada Y. El clasificador $\varphi_{\text{opt}} \in \mathcal{Y}^{\mathcal{X}}$ con el menor valor de error entre todos los clasificadores que forman el espacio $\mathcal{Y}^{\mathcal{X}}$ se llama *clasificador óptimo*, o *clasificador de Bayes*, y su error $\varepsilon(\varphi_{\text{opt}})$ se denomina *error del clasificador óptimo*, o *error de Bayes*: $\varepsilon(\varphi_{\text{opt}}) \leq \varepsilon(\varphi)\ \forall \varphi \in \mathcal{Y}^{\mathcal{X}}$.

Para facilitar el análisis, y sin pérdida de generalidad, de aquí en adelante se asume que el conjunto $\mathcal{Y}$ está formado únicamente por dos clases o elementos, $c = 2$: $\mathcal{Y} = \{0,1\}$ y $\boldsymbol{\mathcal{X}}$ es un espacio discreto n-dimensional. Para este caso se tiene un problema de clasificación binaria, donde el espacio de todos los posibles clasificadores binarios con dominio en $\boldsymbol{\mathcal{X}}$ y rango en el conjunto $\{0,1\}$ se denota mediante $\{0,1\}^{\boldsymbol{\mathcal{X}}}$. Para una observación arbitraria $\mathbf{X} \in \boldsymbol{\mathcal{X}}$, la respuesta o salida del clasificador óptimo binario φ_{opt} está dada por

$$\varphi_{\text{opt}}(\mathbf{X}) = \begin{cases} 1 & \text{si } \mathbb{P}(\mathbf{X}, Y = 1) \geq \mathbb{P}(\mathbf{X}, Y = 0) \\ 0 & \text{si } \mathbb{P}(\mathbf{X}, Y = 1) < \mathbb{P}(\mathbf{X}, Y = 0) \end{cases}, \tag{3.2}$$

donde $\mathbb{P}(\mathbf{X}, Y = 0)$ y $\mathbb{P}(\mathbf{X}, Y = 1)$ son las probabilidades conjuntas de que el vector $\mathbf{X}$ tenga asociada la etiqueta $Y = 0$ o $Y = 1$, respectivamente, con $\mathbb{P}(\mathbf{X}, Y = 0) + \mathbb{P}(\mathbf{X}, Y = 1) = \mathbb{P}(\mathbf{X})$.

Si en la ecuación 3.2 se reemplaza $\mathbb{P}(\mathbf{X}, Y = 1)$ con $\mathbb{P}(Y = 1|\mathbf{X})\mathbb{P}(\mathbf{X})$, donde $\mathbb{P}(Y = 1|\mathbf{X})$ es la probabilidad condicional *a posteriori* de que la etiqueta sea $Y = 1$ dado $\mathbf{X}$, y $\mathbb{P}(\mathbf{X})$ es la probabilidad marginal con la que aparece $\mathbf{X}$, entonces se obtiene

$$\varphi_{\text{opt}}(\mathbf{X}) = \begin{cases} 1 & \text{si } \mathbb{P}(Y = 1 \mid \mathbf{X}) \geq 0.5 \text{ o } \mathbb{P}(Y = 1 \mid \mathbf{X}) \geq \mathbb{P}(Y = 0 \mid \mathbf{X}) \\ 0 & \text{caso contrario} \end{cases}. \tag{3.3}$$

En función de lo anterior, el error del clasificador óptimo binario está dado por

$$\varepsilon(\varphi_{\text{opt}}) = \sum_{\{\mathbf{X}:\, \varphi_{\text{opt}}(\mathbf{X}) = 1\}} \mathbb{P}(Y = 0 \mid \mathbf{X})\mathbb{P}(\mathbf{X}) + \sum_{\{\mathbf{X}:\, \varphi_{\text{opt}}(\mathbf{X}) = 0\}} \mathbb{P}(Y = 1 \mid \mathbf{X})\mathbb{P}(\mathbf{X}). \tag{3.4}$$

Para aplicaciones prácticas, usualmente se desconoce la distribución conjunta $\Pr(\mathbf{X}, Y)$, pero en su lugar se tiene acceso a un conjunto de datos de entrenamiento y un conjunto de datos de testeo $\boldsymbol{\mathcal{D}}_{\text{train}} = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_N, Y_N)\}$ y $\boldsymbol{\mathcal{D}}_{\text{test}} = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_M, Y_M)\}$, respectivamente, los cuales son generados independientemente. En realidad, en la práctica sólo tiene acceso a un único conjunto de datos $\boldsymbol{\mathcal{D}}$ con P pares $(\mathbf{X}, Y)$, el cual se divide en los subconjuntos: $\boldsymbol{\mathcal{D}}_{\text{train}}$ y $\boldsymbol{\mathcal{D}}_{\text{test}}$, donde $P = M + N$. Más adelante se volverá evidente el por qué de esta división. Para los conjuntos de entrenamiento y testeo se asume que cada par $(\mathbf{X}, Y)$ fue generado de manera independiente y a partir de la misma distribución conjunta $\Pr(\mathbf{X}, Y)$. En base a $\boldsymbol{\mathcal{D}}_{\text{train}}$ y un algoritmo de aprendizaje supervisado se diseña un clasificador φ_N minimizando la cantidad de errores de φ_N sobre las observaciones de $\boldsymbol{\mathcal{D}}_{\text{train}}$. El valor del error verdadero $\varepsilon(\varphi_N)$ de φ_N se puede estimar usando los elementos del conjunto $\boldsymbol{\mathcal{D}}_{\text{test}}$ y se denota mediante $\varepsilon_M(\varphi_N)$. El valor de $\varepsilon_M(\varphi_N)$ es el *error empírico* calculado como el número de veces que el clasificador diseñado predice erróneamente las etiquetas para los vectores $\mathbf{X}$ del conjunto $\boldsymbol{\mathcal{D}}_{\text{test}}$ dividido para el total sus elementos, M [Duda *et al.*, 2001], [Murphy, 2012].

La forma usual de diseñar clasificadores consiste en particionar el espacio de características $\boldsymbol{\mathcal{X}}$ en un número de regiones igual al número de clases del problema en cuestión. Para clasificación binaria, el espacio de características se particiona en dos regiones $\mathcal{R}_0$ y $\mathcal{R}_1$, donde la frontera que separa estas dos regiones, llamada *frontera de decisión*, se define

través de una función perteneciente a una clase o conjunto de funciones C (por ejemplo las funciones lineales o los polinomios de un determinado orden). Esto se puede conseguir modelando, o aproximando, la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ mediante una función paramétrica $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$: $\mathbb{R}^n \rightarrow [0,1]$, donde $\boldsymbol{\beta}$ denota el conjunto de parámetros que definen a un clasificador. La frontera de decisión es la función que se obtiene para cuando $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta}) = 0.5$. Si $\boldsymbol{\beta}$ es un conjunto fijo, pero desconocido, el paradigma de diseño de clasificadores es *frecuentista* [Devroye *et al.*, 1996], [Vapnik, 2000], [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012], [Mohri *et al.*, 2012]. Si $\boldsymbol{\beta}$ se modela como un conjunto aleatorio con distribución *a priori* $\Pr(\boldsymbol{\beta})$, entonces el paradigma de diseño es *Bayesiano* [Bishop, 2006], [Murphy, 2012]. Esta tesis se enmarca dentro del paradigma frecuentista. Una familia o conjunto de funciones $\{\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})\}$ se denomina *modelo de clasificación paramétrico*. En el diseño de clasificadores también se pueden emplear *modelos de clasificación no paramétrico*s como *k*NN o plug-in (Capítulo II, Secciones 2.3 y 2.4) [Devroye *et al.*, 1996], [Duda *et al.*, 2001].

Dado un modelo de clasificación paramétrico $\{\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})\}$, o modelo de aprendizaje supervisado, el diseño de un clasificador se reduce a ajustar los valores del conjunto de parámetros, $\boldsymbol{\beta}$, usando un algoritmo de optimización y los elementos $(\mathbf{X},Y)$ del conjunto $\boldsymbol{\mathcal{D}}_{\text{train}}$. A este procedimiento se denomina *entrenamiento* del clasificador. Por ejemplo, para el entrenamiento de un clasificador se puede maximizar la función de verosimilitud del conjunto de entrenamiento $\boldsymbol{\mathcal{D}}_{\text{train}}$ dado el conjunto de parámetros del modelo $\boldsymbol{\beta}$, $\mathbb{P}(\boldsymbol{\mathcal{D}}_{\text{train}}|\boldsymbol{\beta})$. Otra alternativa es minimizar el número de errores que comete el clasificador sobre los elementos del conjunto $\boldsymbol{\mathcal{D}}_{\text{train}}$, llamado minimización del error empírico [Vapnik, 2000]. Una vez entrenado el clasificador, la clase o etiqueta $\varphi_N(\mathbf{X})$ para un vector $\mathbf{X}$ se determina calculando el valor de $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ para $\mathbf{X}$ y luego reemplazando este valor en la ecuación 3.3. En este caso la frontera de decisión que separa las regiones $\mathcal{R}_0$ y $\mathcal{R}_1$ se obtiene cuando $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta}) = 0.5$. Es posible mover ligeramente esta frontera de decisión modificando el valor de umbral de 0.5 a otro valor. Durante el *testeo* se aplica el clasificador diseñado a los elementos del conjunto $\boldsymbol{\mathcal{D}}_{\text{test}}$ y se estima el valor de su error verdadero.

3.2.2. Relación entre Clasificación y Diseño de W-operadores

Es interesante observar la gran cantidad de similitudes que existen entre las definiciones presentadas en esta sección con las definiciones presentadas para el caso de W-operadores. En particular, nótese que la función característica ψ de un W-operador $\boldsymbol{\Psi}$ y un clasificador φ cumplen una tarea similar que es la de particionar el espacio de configuraciones de ventana y el espacio de vectores de características, respectivamente. Una diferencia es que el espacio de configuraciones para el caso de W-operadores es discreto, mientras que el espacio de vectores de características para clasificación puede ser discreto (como se asume en esta tesis) o también continuo.

Si se considera la representación morfológica de los W-operadores, se puede establecer una analogía entre el conjunto de elementos de la base o el conjunto de intervalos maximales que definen a un W-operador y el conjunto de parámetros β que definen a un clasificador paramétrico. Para el caso de modelos de clasificación no paramétricos, el equivalente a la regla plug-in es regla de clasificación por histogramas. Aquí se particiona el espacio $\mathbb{R}^n$ en hipercubos de igual lado. Para clasificar un vector **X** se realiza una votación entre las etiquetas de los vectores del hipercubo que contiene a **X**, asignándole la etiqueta mayoritaria [Devroye *et al.*, 1996]. Adicionalmente, la definición de restricciones para un problema de diseño de W-operadores es equivalente a la selección de un modelo de clasificación, incluyendo extracción y selección de características.

Por lo tanto, en base a lo anterior en esta tesis se considera *al diseño automático de la función característica de un W-operador como un problema de reconocimiento de patrones que consiste en el diseño de un clasificador*. Aquí el clasificador asigna una etiqueta a un vector de características **X** observado a través de una ventana *W*. En este caso los valores de cada descriptor o *feature* son los niveles de intensidad que se observan en cada punto o píxel de la ventana *W*. En este contexto, la ventana se considera como el elemento a través del cual se realiza la tarea de extracción de características o *feature extraction*. La reducción de la dimensión del espacio de configuraciones puede ser tratada como un problema de selección de características o *feature selection*.

Nótese que lo aquí asumido no altera para nada la función que cumple un W-operador cuya entrada y salida son imágenes. La ventaja en este caso es que el reconocimiento de patrones ofrece un nuevo paradigma para el diseño de W-operadores con un sólido sustento teórico. Una de las primeras ventajas que se puede advertir es el menor costo computacional de representación de una función característica al ser considerada como un clasificador paramétrico. Adicionalmente, de aquí en adelante no se hará más una distinción entre clasificador y función característica. Por el contrario, se considerarán como términos sinónimos que se denotan mediante ψ.

3.2.3. Breve Comparación Histórica entre W-operadores y Reconocimiento de Patrones

El reconocimiento de patrones despunta a finales de los años 50 y toda la década de los 60 del siglo pasado. En este período de tiempo se analizan ciertas propiedades de los métodos de estimación de densidades no paramétricos [Rosenblatt, 1956], [Cover y Hart, 1967] y también se propone y analiza el primer algoritmo de aprendizaje computacional como es el caso del *perceptrón* [Rosenblatt, 1962]. Por su parte, la teoría de W-operadores aparece con el nacimiento de la morfología matemática en 1964 a partir del trabajo desarrollado por Matheron y presentado formalmente en [Matheron, 1975]. Desde su nacimiento cada una de estas disciplinas se desarrolló de manera independiente. Es recién en el trabajo de Dougherty y Barrera [Dougherty y Barrera, 2002], donde se establece por primera vez un vínculo formal entre las mismas. Posteriormente, en [Martins-Jr *et al.*, 2006] se presenta una de las primeras propuestas donde se trata formalmente al diseño de W-operadores como un problema de reconocimiento de patrones. En ese trabajo se hace uso de técnicas de *feature selection* para el diseño de la ventana de W-operadores binarios y en escala de grises. Sin embargo, a pesar de estos trabajos, hasta el inicio de esta tesis no se registra un avance significativo dentro del diseño y aplicaciones prácticas de W-operadores bajo el contexto de reconocimiento de patrones.

3.3. Diseño de Clasificadores: Teoría de Aproximación y Generalización

Como se anotó anteriormente, el diseño de un clasificador se inicia con la selección de un modelo de clasificación. En el caso de modelos paramétricos, esto involucra la selección de una clase de funciones paramétricas. En base a esta clase de funciones, en la etapa de entrenamiento, se ajusta el valor de los parámetros del modelo, lo cual equivale a definir una frontera de decisión usando el conjunto de ejemplos de entrenamiento $\mathcal{D}_{\text{train}}$.

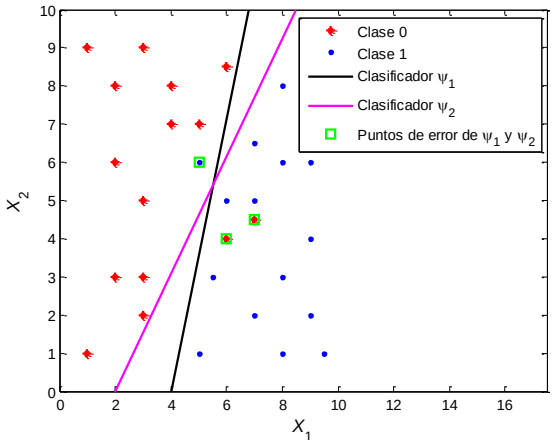


Figura 3.1. Clasificación binaria usando fronteras de decisión lineales. El espacio de características está compuesto por 30 puntos, 15 puntos por cada clase. Las dos fronteras de decisión particionan el espacio de características en dos regiones. Sin embargo, los dos clasificadores producen el mismo etiquetado o clasificación efectiva de los puntos.

Por ejemplo, para el problema de clasificación binaria mostrado en la Figura 3.1 el espacio de observaciones está formado por un total de 30 puntos en $\mathbb{R}^2$, con 15 puntos por cada clase. Para la implementación de la frontera de decisión se usa la clase de funciones lineales paramétricas $C = \{h(\mathbf{X};\boldsymbol{\beta}) \mid h = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \text{ y } \boldsymbol{\beta} = (\beta_0,\beta_1,\beta_2) \in \mathbb{R}^3\}$, donde la etiqueta para cada vector $\mathbf{X} = (X_1,X_2)$ es $\psi(\mathbf{X}) = 0$ si $h(\mathbf{X}) < 0$, o $\psi(\mathbf{X}) = 1$ si $h(\mathbf{X}) \geq 0$. Alternativamente, se puede transformar el valor de una función $h(\mathbf{X};\boldsymbol{\beta}) \in C$ en un valor de probabilidad usando $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta}) = 1/(1 + exp(-h(\mathbf{X};\boldsymbol{\beta})))$, donde $\boldsymbol{\beta} = \{\beta_0,\beta_1,\beta_2\}$, y luego se puede comparar este valor con un umbral de 0.5 para definir la etiqueta de un vector $\mathbf{X}$.

Para el ejemplo en cuestión, se consideran dos fronteras de decisión que, a su vez, definen dos clasificadores: ψ_1 y ψ_2. Nótese que a pesar de que las fronteras de decisión son diferentes, y por ende las regiones definidas son distintas, la clasificación efectiva de los puntos es la misma. De hecho, si se considera al error empírico como una medida de la precisión de la clasificación, entonces los dos clasificadores mostrados en la Figura 3.1 son equivalentes, pues se tiene que $\varepsilon_N(\psi_1) = \varepsilon_N(\psi_2) = 3/30$. En realidad, en este caso no sólo existen dos fronteras de decisión que generan la misma clasificación, sino un número infinito de funciones que pueden generar la misma clasificación que ψ_1 y ψ_2.

Dado un conjunto cualquiera de N puntos $\boldsymbol{\mathcal{D}}_N = \{\mathbf{X}_1,\ldots,\mathbf{X}_N\}$, con $\mathbf{X}_i \in \boldsymbol{\mathcal{X}} \subset \mathbb{R}^n$ e $i = 1,\ldots,N$, una *dicotomía* es una partición del conjunto $\boldsymbol{\mathcal{D}}_N$ en dos subconjuntos disjuntos $\boldsymbol{\mathcal{D}}_0$ y $\boldsymbol{\mathcal{D}}_1$ tales que $\boldsymbol{\mathcal{D}}_N = \boldsymbol{\mathcal{D}}_0 \cup \boldsymbol{\mathcal{D}}_1$, donde $\boldsymbol{\mathcal{D}}_0$ y $\boldsymbol{\mathcal{D}}_1$ contienen los puntos pertenecientes a la clase 0 y la clase 1, respectivamente [Abu-Mostafa *et al.*, 2012]. El número máximo posible de dicotomías para todos los conjuntos formados por N puntos es 2^N. La *función de crecimiento* para una clase de funciones C y todos los conjuntos con un número fijo de N puntos $\{\boldsymbol{\mathcal{D}}_N\}$ se denota mediante $m_C(N)$. El valor de $m_C(N)$ corresponde al número máximo de dicotomías que se pueden generar usando las funciones de C aplicadas a cada $\boldsymbol{\mathcal{D}}_N$ del conjunto $\{\boldsymbol{\mathcal{D}}_N\}$, donde N es un número entero cualquiera. Dado que 2^N es la cantidad máxima de posibles dicotomías para todos los conjuntos formados por N puntos, entonces $m_C(N) \leq 2^N$ [Vapnik y Chervonenkis, 1971], [Vapnik, 2000].

Es conveniente remarcar que la función de crecimiento es de naturaleza puramente combinatoria y su valor depende únicamente de la clase de funciones C utilizada y de la cantidad de puntos en cuestión N, más no de la forma en la que éstos están distribuidos en el espacio. Por ejemplo, para un espacio de observaciones bidimensional y la clase de funciones lineales, el valor de la función de crecimiento para los conjuntos formados por 2 puntos es $m_C(2) = 4$ de un total de 4 posibles dicotomías. Para conjuntos de 3 puntos se tiene $m_C(3) = 8$ de un total de 8 posibles dicotomías. Nótese que, en este caso, si un conjunto está formado por 3 puntos colineales, entonces no se pueden generar todas las 8 posibles dicotomías; sin embargo, la función de crecimiento tiene un valor de 8 porque hace referencia al máximo número de dicotomías sobre todos los posibles conjuntos formados por 3 puntos. Para un conjunto de 4 puntos se tiene $m_C(4) = 14$ de un total de 16 posibles dicotomías [Abu-Mostafa *et al.*, 2012].

La *dimensión de Vapnik-Chervonenkis*, o *dimensión VC*, de una clase de funciones C es el valor máximo de N para el cual la función de crecimiento es $m_C(N) = 2^N$ [Vapnik, 2000]. Si para una clase de funciones C se tiene que $m_C(N) = 2^N$ para todo valor de N, entonces la dimensión VC de C es infinita, $VC = \infty$. La dimensión VC indica el tamaño máximo de los conjuntos de puntos para los cuales se pueden generar todas las posibles dicotomías usando una clase de funciones C. En otras palabras, la dimensión VC permite conocer la "cantidad efectiva" de clasificadores que se pueden implementar con las funciones de C.

Es oportuno remarcar que la dimensión VC es independiente de la distribución marginal $\Pr(\mathbf{X})$. Dada un clase de funciones C con dimensión VC finita, si $VC > 2$, entonces se tiene que $m_C(N) \leq N^{VC}$, lo cual implica un crecimiento polinómico de $m_C(N)$ con el aumento de N [Devroye *et al.*, 1996] [Abu-Mostafa *et al.*, 2012]. Por ejemplo, para la clase de funciones lineales paramétricas $C = \{f(\mathbf{X};\boldsymbol{\beta}) \mid \boldsymbol{\beta} \in \mathbb{R}^{n+1}\}$ la dimensión VC es $n + 1$ que, en este caso, corresponde al número total de parámetros que se pueden ajustar durante el entrenamiento del clasificador. Nótese también que, aunque C está formado por una cantidad infinita de funciones, la cantidad efectiva de dicotomías que se pueden generar usando funciones lineales es 2^{n+1}.

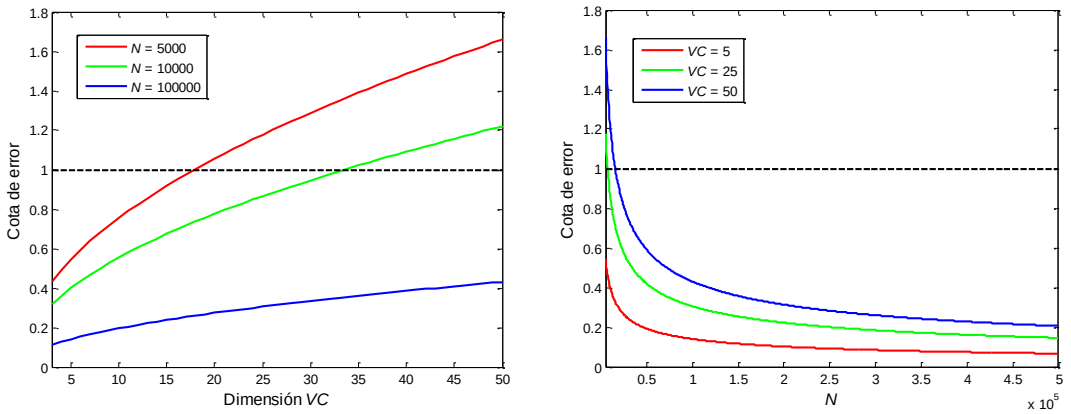
En base a lo anterior se analiza el costo de diseño de clasificadores usando restricciones. Sea $\psi_{C,\text{opt}}$ el clasificador óptimo para una clase de funciones C con dimensión VC finita y

acotada inferiormente por $VC > 2$: $\varepsilon(\psi_{C,\text{opt}}) < \varepsilon(\psi_C)\ \forall \psi_C \in C$. Los errores $\varepsilon(\psi_{C,\text{opt}})$ y $\varepsilon(\psi_C)$ se calculan usando la ecuación 3.4. Dado un conjunto de entrenamiento $\mathcal{D}$ formado por N pares $(\mathbf{X},Y)$ generados independientemente a partir de una distribución conjunta y fija $\Pr(\mathbf{X},Y)$, el clasificador que minimiza el error empírico sobre $\mathcal{D}$, llamado *clasificador óptimo empírico*, se denota mediante $\psi_{C,N,*}$. El error verdadero del clasificador óptimo empírico $\psi_{C,N,*}$ es $\varepsilon(\psi_{C,N,*}) = \mathbb{P}(\psi_{C,N,*}(\mathbf{X}) \neq Y)$. En estas condiciones, el valor esperado del costo de diseño del clasificador óptimo empírico usando la clase de funciones C está acotado superiormente por la *desigualdad de Vapnik-Chervonenkis*:

$$\mathbb{E}[\Delta(\psi_{C,N,*},\psi_{C,\text{opt}})] \leq 8\sqrt{(VC\ln(N)+4)/(2N)}\,, \tag{3.5}$$

donde la esperanza $\mathbb{E}$ se calcula con respecto a la distribución subyacente a la generación de conjuntos $\mathcal{D}$ formados por N puntos [Devroye *et al.*, 1996]. Esta desigualdad es válida para cualquier distribución conjunta $\Pr(\mathbf{X},Y)$.

La desigualdad 3.5 indica que, para una clase de funciones C con dimensión VC finita y mayor que 2, la cota del costo de diseño del clasificador óptimo empírico decrece en un orden de $\sqrt{\ln(N)/N}$ a medida que se aumenta la cantidad de datos de entrenamiento, N, manteniendo fija la dimensión VC (Figura 3.2-a). En el otro caso, para una cantidad de datos de entrenamiento fija, el aumento de la complejidad de la clase de funciones C, que implica un aumento de su dimensión VC, hace que también se incremente el tamaño de la cota de error (Figura 3.2-b). En los dos gráficos de la Figura 3.2 se puede notar que para determinados valores de VC y N, la cota de error se vuelve trivial (mayor que 1). En este caso no se puede afirmar nada respecto al comportamiento del costo de diseño del clasificador óptimo empírico.



(a) **(b)**

Figura 3.2. Cotas de error para el costo de diseño del clasificador óptimo empírico. (**a**) Variación de la cota de error en función del aumento de la complejidad de la clase de funciones (dimensión VC) y un número fijo N de ejemplos de entrenamiento. (**b**) Variación de la cota de error en función del aumento de la cantidad de ejemplos de entrenamiento N y dimensión VC fija.

La desigualdad 3.5 hace referencia al costo de diseño del clasificador óptimo empírico dados una clase de funciones C y conjuntos de ejemplos de entrenamiento con N puntos. Sin embargo, en la práctica la búsqueda del clasificador óptimo empírico, con frecuencia, trae consigo un muy elevado costo computacional [Duda *et al.*, 2001]. En su lugar, lo usual

es diseñar un clasificador subóptimo ψ_N con un error empírico de entrenamiento $\varepsilon_N(\psi_N)$ pequeño. En función de esto se espera que el bajo error de entrenamiento de ψ_N signifique un buen desempeño de dicho clasificador en el testeo [Abu-Mostafa *et al.*, 2012]. Esto es equivalente a esperar que los valores de $\varepsilon_N(\psi)$ y $\varepsilon(\psi)$ sean muy similares, en un sentido probabilístico, para cualquier clasificador $\psi \in C$, a lo cual se denomina *generalización*. Aquí los términos $\varepsilon_N(\psi)$ y $\varepsilon(\psi)$ denotan el error empírico de entrenamiento y el error verdadero del clasificador $\psi \in C$, respectivamente. El problema de generalización fue investigado por Vapnik y Chervonenkis en el contexto de convergencia uniforme de las frecuencias relativas a los valores de probabilidad, obteniendo la siguiente desigualdad [Vapnik y Chervonenkis, 1971], [Vapnik, 2000]:

$$\mathbb{P}(sup_{\psi \in \mathcal{C}} |\varepsilon_N(\psi) - \varepsilon(\psi)| > \tau) \leq 4m_C(2N)exp(-\tau^2 N/8), \tag{3.6}$$

donde la probabilidad se calcula con respecto a la distribución subyacente a la generación de conjuntos de entrenamiento con N pares $(\mathbf{X},Y)$ y $\tau > 0$ es un umbral arbitrario. Esta desigualdad es válida para cualquier distribución conjunta $\Pr(\mathbf{X},Y)$.

La desigualdad 3.6 implica que, siempre que la dimensión *VC* de una clase de funciones sea finita, y la cantidad de ejemplos de entrenamiento sea suficientemente grande, es posible obtener una buena *aproximación* de $\varepsilon(\psi)$ mediante el error empírico $\varepsilon_N(\psi)$. Esto es posible debido a que el valor de la función de crecimiento $m_C(2N)$ tiene un crecimiento polinómico con el aumento de N si la dimensión VC de una clase de funciones C es finita. En otras palabras, y bajo las condiciones anotadas, la desigualdad de Vapnik-Chervonenkis afirma que es posible que buen desempeño del clasificador durante la etapa de entrenamiento signifique también un buen desempeño durante la etapa de testeo.

El problema de diseño de clasificadores ahora se reduce a, dado un modelo de clasificación, encontrar, en la etapa de entrenamiento, un clasificador con un bajo nivel de error empírico $\varepsilon_N(\psi)$. Sin embargo, se puede advertir fácilmente que existen casos donde el lado derecho de la desigualdad 3.6 puede ser mayor que 1: $4m_C(2N)exp(-\tau^2 N/8) > 1$. En estos casos, la desigualdad 3.6 se vuelve trivial y no se puede afirmar nada para la etapa de testeo a partir de la información de entrenamiento. Es decir, un bajo valor del error de entrenamiento $\varepsilon_N(\psi)$ no necesariamente implica que $\varepsilon(\psi)$ sea también bajo.

Peor aún, para los casos donde $4m_C(2N)exp(-\tau^2 N/8) > 1$, un bajo error de entrenamiento podría significar un pobre desempeño, o una mala generalización, del clasificador diseñado al ser utilizado durante la etapa de testeo: $\varepsilon_N(\psi) << \varepsilon(\psi)$. Esto sucede cuando se diseña un clasificador ψ a partir de una clase de funciones C muy compleja tal que su dimensión VC sea mayor que o igual al número de ejemplos de entrenamiento disponibles: $VC > N$. Este problema es conocido como *sobre-entrenamiento* del clasificador, o en inglés *overfitting* [Duda *et al.*, 2001], [Mohri *et al.*, 2012], [Murphy, 2012]. En una situación de overfitting, la mala performance de un clasificador durante la etapa de testeo se debe a que, por su elevada complejidad, predice con alta, o completa, exactitud los datos de entrenamiento

incluyendo el ruido: $\varepsilon_N(\psi) \approx 0$. En este escenario, el clasificador no es capaz de descubrir ninguna estructura o patrón en los datos de entrenamiento; por el contrario, simplemente los "memoriza". Esto evidencia que *no siempre un bajo error de entrenamiento implica una buena generalización*. Usando un ejemplo de la Literatura Argentina, el overfitting genera en un clasificador una situación similar a la que vivía *Funes el Memorioso* de Jorge Luis Borges. A este personaje, por ejemplo, "…no sólo le costaba comprender que el símbolo genérico perro abarcara tantos individuos dispares de diversos tamaños y diversa forma; le molestaba que el perro de las tres y catorce (visto de perfil) tuviera el mismo nombre que el perro de las tres y cuarto (visto de frente)…"

Es conveniente remarcar que en ninguna parte del análisis anterior se toma en cuenta al clasificador óptimo ψ_{opt} para la distribución conjunta $Pr(\mathbf{X},Y)$, ni tampoco a su error $\varepsilon(\psi_{opt})$. La razón es simple, en la práctica no se conoce el proceso aleatorio que genera las imágenes del problema en cuestión, ni tampoco la distribución conjunta $Pr(\mathbf{X},Y)$ de las configuraciones de ventana y sus etiquetas. Solamente se tiene acceso a un conjunto de imágenes de entrenamiento para diseñar un operador y otro conjunto de imágenes de testeo para evaluar su desempeño. En estas condiciones, lo que se puede aspirar es diseñar un operador que pueda generalizar bien el concepto aprendido durante la etapa de entrenamiento.

Para un problema determinado, lo antes mencionado involucra seleccionar una clase de funciones *C*, o modelo de clasificación, con una complejidad (dimensión VC) acorde a la cantidad de datos de entrenamiento disponibles, *N*: $VC << N$. En base a la clase de funciones seleccionada *C*, un conjunto de *N* ejemplos de entrenamiento, y un algoritmo de optimización, se diseña un clasificador ψ_N procurando obtener un bajo error empírico $\varepsilon_N(\psi_N)$ de entrenamiento. Para evaluar la capacidad predictiva de ψ_N usualmente se utiliza al error empírico $\varepsilon_M(\psi_N)$ calculado al aplicar ψ_N sobre el conjunto de testeo. Esto porque el error de entrenamiento $\varepsilon_N(\psi_N)$ de ψ_N es una estimación, usualmente, sobreoptimista del valor de $\varepsilon(\psi_N)$. El valor de $\varepsilon_M(\psi_N)$ será una buena aproximación del error verdadero $\varepsilon(\psi_N)$ si el número de datos de testeo, *M*, es suficientemente grande tal como se evidencia a través de la *desigualdad de Hoeffding* [Hoeffding, 1963], [Abu-Mostafa *et al.*, 2012]

$$\mathbb{P}(|\varepsilon_M(\psi_N) - \varepsilon(\psi_N)| > \tau) \leq 2exp(-2\tau^2 M), \tag{3.7}$$

donde $\tau > 0$ es un umbral arbitrario.

Como nota final de este análisis es importante mencionar que, debido al carácter general de las desigualdades 3.6 y 3.7, sus predicciones son usualmente muy pesimistas debido a que representan el peor escenario posible. Por este motivo, dentro de reconocimiento de patrones es común utilizar técnicas como *validación cruzada* para comparar diferentes modelos de clasificación y seleccionar a aquel que presente mejores resultados para el diseño de un clasificador [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012]. En esta tesis, se aplica este protocolo para la selección de modelos de clasificación. Adicionalmente, por el defecto antes anotado, tampoco es común reportar las cotas de error para un problema determinado. En su lugar se reporta únicamente el error de testeo como medida de la capacidad predictiva, o poder de generalización, del operador diseñado.

3.4. Modelos de Clasificación Paramétricos versus Modelos No Paramétricos

El diseño de un clasificador se inicia con la selección de un modelo de clasificación. Ante este hecho surge la pregunta sobre ¿*qué modelo de clasificación utilizar*? Una vez más lo teoremas de no free lunch indican que no hay un modelo de clasificación que funcione mejor que otro sin conocer el problema en el que se aplicarán. En promedio, todos los modelos de clasificación se comportan igual. Sin embargo, es conveniente realizar las siguientes acotaciones. En los modelos paramétricos, la cantidad de parámetros o complejidad del modelo (dimensión VC) es fija; mientras que en los modelos no paramétricos, la complejidad del modelo varía en función de la cantidad de ejemplos de entrenamiento. Así mismo, los modelos paramétricos usualmente convergen más rápido a la solución óptima dentro de la clase de funciones seleccionada [Devroye *et al.*, 1996], [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012], [Murphy, 2012].

En contraste con la ventaja descrita anteriormente para el caso de los modelos paramétricos está el hecho de que, en este tipo de modelos, se realizan fuertes suposiciones que pueden implicar restricciones acerca de la forma y tipo de la distribución subyacente a la generación de los datos. Si la distribución del modelo de clasificación dista mucho de la distribución real de los datos, entonces los resultados podrían ser malos. En esta parte, los modelos no paramétricos son más flexibles, pues no se asume una distribución específica. Sin embargo, cuando se trabaja con ventanas de mediano y gran tamaño e imágenes en escala de grises o color, los modelos no paramétricos presentan un elevado costo computacional y adicionalmente demandan de una gran cantidad de datos de entrenamiento para obtener una buena generalización (Capítulo II, Secciones 2.3 y 2.4).

3.5. Redes Neuronales Artificiales tipo Feed-Forward

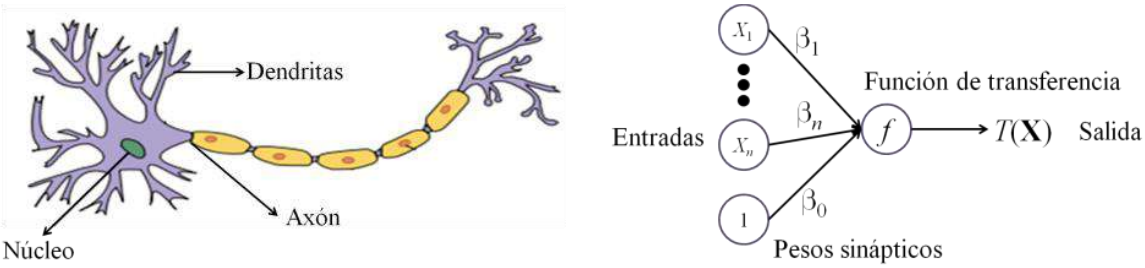


Figura 3.3. Neurona humana y neurona artificial**.** Estructura de: (**a**) una neurona humana y (**b**) una neurona artificial. En la neurona artificial, las líneas que conectan cada variable de entrada con el nodo de la función de transferencia son equivalentes a las dendritas de la neurona humana. La salida de la neurona artificial es equivalente al axón de la neurona humana.

Las redes neuronales artificiales son un modelo paramétrico de aprendizaje computacional que se pueden utilizar para resolver problemas de clasificación y regresión [Hagan *et al.*, 1996], [Hastie *et al.*, 2001], [Duda *et al.*, 2001], [Egmont-Petersen *et al.*, 2002], [Bishop, 2005]. Estos modelos deben su nombre al hecho de que emulan la estructura cerebral humana formada por la interconexión de billones de neuronas (Figura 3.3-a). Para las redes neuronales artificiales, al igual que en el cerebro humano, el elemento fundamental se llama neurona (Figura 3.3-b). Las conexiones entre las entradas $X_1,\ldots,X_n$ de una neurona artificial y el nodo donde se localiza la función de transferencia f son equivalentes a las dendritas de una neurona humana. Los parámetros $\beta_1,\ldots,\beta_n$ se denominan *pesos sinápticos*. Estos pesos controlan la intensidad de la conexión entre cada variable de entrada y el nodo donde se

localiza la función de transferencia. El parámetro β_0 se denomina *umbral de activación* de la neurona artificial.

La respuesta $T(\mathbf{X})$ de una neurona artificial, para un vector de entrada $\mathbf{X} = (X_1,\ldots,X_n)$, se obtiene calculando el valor de la función de transferencia f para la combinación lineal, o producto escalar, entre el vector de entrada $\mathbf{X}$ y el vector de pesos sinápticos $\boldsymbol{\beta} = (\beta_1,\ldots,\beta_n)$ más el valor de umbral activación β_0:

$$T(\mathbf{X}) = f(\beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n). \tag{3.8}$$

Si se define la función f: $\mathbb{R} \rightarrow \{0,1\}$ tal que $f(a) = 1$ para $a \geq 0$ ó $f(a) = 0$ para el caso contrario, entonces $T(\mathbf{X})$ es la respuesta de un *perceptrón* [Rosenblatt, 1962]. También se pueden utilizar *funciones sigmoideas*, que son funciones monótonamente crecientes, tales que $f(-\infty) = -1$ y $f(\infty) = 1$, o $f(-\infty) = 0$ y $f(\infty) = 1$, o a su vez *funciones lineales* $f(a) = a$.

Cuando se interconectan varias neuronas artificiales, la estructura resultante se denomina red neuronal artificial. En particular, si se conectan perceptrones agrupados por capas, el modelo resultante se denomina perceptrón multicapa [Duda *et al.*, 2001]. Si las neuronas se agrupan por capas y se interconectan de modo tal que sólo se permiten las conexiones hacia adelante entre las neuronas de dos capas consecutivas, entonces el modelo resultante se denomina *red neuronal artificial tipo feed-forward* (Figura 3.4). La complejidad de una red neuronal artificial tipo feed-forward está determinada por el número de entradas, número de capas, y el número de neuronas presentes en cada capa. Para simplificar la terminología a ser empleada en lo posterior a las redes neuronales artificiales, las neuronas artificiales, los pesos sinápticos, y los umbrales de activación se los llama redes neuronales, neuronas, pesos, y umbrales, respectivamente.

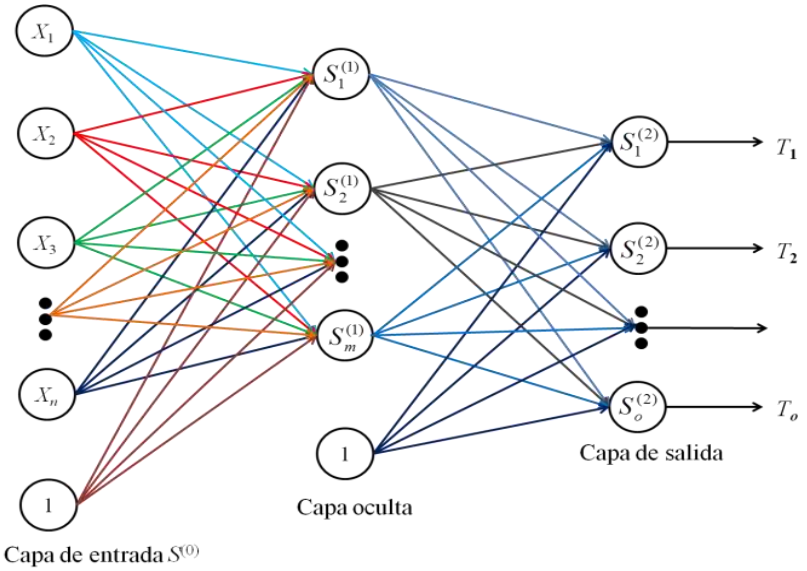


Figura 3.4. Red neuronal tipo feed-forward compuesta por tres capas: entrada $S^{(0)}$, oculta $S^{(1)}$, y salida $S^{(2)}$.

Sea una arquitectura de red neuronal tipo feed-forward de tres capas: entrada $S^{(0)}$, oculta $S^{(1)}$ y salida $S^{(2)}$ con m neuronas, $S_1^{(1)} \ldots, S_m^{(1)}$, en la capa oculta y una única neurona $S_1^{(2)}$ en la capa de salida (Figura 3.5). Las funciones de transferencia $f^{(1)}$ de las neuronas de la capa

oculta son funciones sigmoideas y la función de transferencia $f^{(2)}$ de la capa de salida es una función lineal $f^{(2)}(a) = a$. En la Figura 3.5, los términos $\mathbf{P}^{(1)} \in \mathbb{R}^{m \times n}$ y $\mathbf{P}^{(2)} \in \mathbb{R}^{m}$ denotan las matrices de pesos de la capa oculta $S^{(1)}$ y de la capa de salida $S^{(2)}$, respectivamente. Los términos $\mathbf{b}^{(1)} \in \mathbb{R}^{m}$ y $b^{(2)} \in \mathbb{R}$ denotan el vector de umbrales de la capa oculta $S^{(1)}$ y el umbral de la capa de salida $S^{(2)}$, respectivamente. El conjunto de todos los p parámetros de la red neuronal se denota mediante β, $\beta = \{\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \mathbf{b}^{(1)}, b^{(2)}\} \in \mathbb{R}^{p}$.

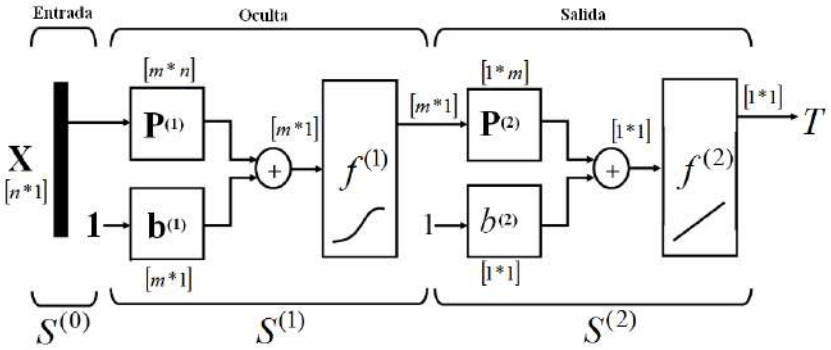


Figura 3.5. Arquitectura de una red neuronal feed-forward de tres capas con una única neurona de salida. Las neuronas de la capa oculta tienen funciones de transferencia sigmoideas y la neurona de la capa de salida tiene una función de transferencia lineal.

Para la red neuronal de la Figura 3.5, la respuesta $T_i^{(1)}(\mathbf{X})$ de la i-ésima neurona $S_i^{(1)}$ de la capa oculta para el vector de entrada $\mathbf{X} = (X_1,\ldots,X_n)^{\mathrm{T}}$ está dada por

$$T_i^{(1)}(\mathbf{X}) = f^{(1)}\left(\left(\sum_{j=1}^{n} P_{i,j}^{(1)} X_j\right) + b_i^{(1)}\right), \tag{3.9}$$

donde $P_{i,j}^{(1)}$ y $b_i^{(1)}$ denotan el peso de la conexión entre la variable de entrada $X_j \in \mathbf{X}$ y la neurona $S_i^{(1)}$, y el umbral de activación de dicha neurona, respectivamente. La respuesta total $T(\mathbf{X})$ de la red neuronal para el vector de entrada $\mathbf{X}$ se calcula mediante

$$T(\mathbf{X}) = f^{(2)}\left(\left(\sum_{i=1}^{m} P_i^{(2)} T_i^{(1)}(\mathbf{X})\right) + b^{(2)}\right), \tag{3.10}$$

donde $P_i^{(2)}$ y $b^{(2)}$ denotan el peso de la conexión entre la neurona $S_i^{(1)}$ de la capa oculta y la neurona $S_1^{(2)}$ de la capa de salida, y el umbral de activación de la neurona $S_1^{(2)}$ de la capa de salida, respectivamente. Dado que $f^{(2)}(a) = a$, entonces la ecuación 3.10 resulta en

$$T(\mathbf{X}) = \left(\sum_{i=1}^{m} P_i^{(2)} T_i^{(1)}(\mathbf{X})\right) + b^{(2)}. \tag{3.11}$$

El conjunto de todas las infinitas funciones que se pueden implementar usando la arquitectura de red neuronal tipo feed-forward antes descrita es $C = \{T(\mathbf{X};\beta) \mid \beta \in \mathbb{R}^{p}\}$. La dimensión VC de la clase de funciones C está acotada inferior y superiormente por

$$2\lfloor (m+1)/2 \rfloor n \ \leq VC \leq \ 2p(log_2(e(m+1))), \tag{3.12}$$

donde $\lfloor (m+1)/2 \rfloor$ denota el entero más grande que es igual a o menor que $(m + 1)/2$ y e es la constante neperiana. El valor de la suma $m + 1$ es el número total de neuronas de la red, n

es la dimensión del vector de entrada $\mathbf{X}$ y $p = m \times n + 2m + 1$ es número total de parámetros (pesos y umbrales) que se pueden ajustar durante el entrenamiento de la red neuronal [Baum y Haussler, 1989], [Bishop, 2005].

En la Figura 3.6 se puede observar la variación del valor de las cotas inferior y superior de la dimensión VC de una red neuronal tipo feed-forward en función del número de neuronas m de la capa oculta $S^{(1)}$ y para diferentes tamaños n del vector de entrada $\mathbf{X}$. En esta figura se puede notar que el tamaño mínimo de los conjuntos de vectores $\mathbf{X}$ para los que se pueden implementar todas las posibles dicotomías, usando una red neuronal tipo feed-forward, no aumenta de manera significativa con el incremento de los valores de m o n (cotas inferiores). Por el contrario, el tamaño máximo de los conjuntos de vectores $\mathbf{X}$ para los que se pueden implementar todas las posibles dicotomías aumenta de manera muy significativa con el incremento de m o n (cotas superiores). Vale la pena recordar que m es el número de neuronas de la capa oculta y n es la dimensión de los vectores de características $\mathbf{X}$.

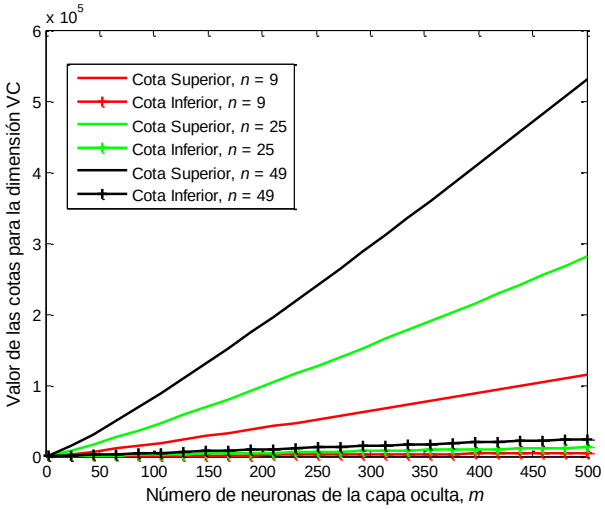


Figura 3.6. Variación del valor de las cotas inferior y superior de la dimensión VC en función del número de neuronas de la capa oculta para la clase de funciones implementadas por una red neuronal tipo feed-forward.

Al procedimiento de ajuste de los pesos y umbrales de la red neuronal que resulta en un conjunto de parámetros fijo $\boldsymbol{\beta} \in \mathbb{R}^p$ se denomina *entrenamiento de la red neuronal*. La función resultante $T(\mathbf{X};\boldsymbol{\beta}) \in C$ al fijar el conjunto de parámetros $\boldsymbol{\beta}$ se denomina *red neuronal entrenada*. Una red neuronal entrenada $T(\mathbf{X};\boldsymbol{\beta})$ para clasificación binaria define una frontera de decisión no lineal que particiona el espacio de configuraciones o características $\boldsymbol{\mathcal{X}}$ en dos regiones: una para el conjunto de vectores $\mathbf{X}$ pertenecientes a la clase 0 y la otra para el conjunto de vectores $\mathbf{X}$ pertenecientes a la clase 1. Para una red neuronal entrenada, la etiqueta para un vector $\mathbf{X}$ se define como $\psi(\mathbf{X}) = 1$ si $T(\mathbf{X};\boldsymbol{\beta}) \geq \tau$ y $\psi(\mathbf{X}) = 0$ en el caso contrario, donde $\tau > 0$ es el umbral de clasificación.

Para una dimensión fija de $\mathbf{X}$ (n constante), si se selecciona un determinado número de neuronas, m, y un determinado tipo de función de transferencia sigmoidea, $f^{(1)}$, para la capa

oculta, entonces se define una clase de funciones $C_p = \{T(\mathbf{X};\boldsymbol{\beta}) \mid \boldsymbol{\beta}\in\mathbb{R}^p$, p y $f^{(2)}$ son fijos$\}$. Sea $\psi_{C,\text{opt}}$ el clasificador óptimo dentro de la clase de funciones C_p: $\varepsilon(\psi_{C,\text{opt}}) < \varepsilon(\psi)$, con $\varepsilon(\psi) = \mathbb{P}(\psi(\mathbf{X}) \neq Y)\ \forall\psi\in C_p$. El costo de restricción al utilizar la clase de funciones C_p se denota mediante $\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}})$, donde ψ_{opt} es el clasificador óptimo para la distribución conjunta Pr($\mathbf{X}$,Y). Las redes neuronales tipo feed-forward son *aproximadores universales*: $\Delta(\psi_{C,\text{opt}},\psi_{\text{opt}}) \rightarrow 0$ cuando $m \rightarrow \infty$ para cualquier distribución conjunta Pr($\mathbf{X}$,Y) [Hornik *et al.*, 1989], [Devroye *et al.*, 1996]. Esto implica que las redes neuronales tipo feed forward tienen la capacidad de aproximar al clasificador óptimo de cualquier distribución conjunta Pr($\mathbf{X}$,Y), siempre y cuando, el número de neuronas m de la capa oculta sea suficientemente grande.

Sea $\psi_{C,N,*}$ un clasificador óptimo empírico diseñado a partir de una clase de funciones C_p y un conjunto de entrenamiento $\mathcal{D}$ formado por N tuplas ($\mathbf{X}$,*freq*($\mathbf{X}$,Y = 0),*freq*($\mathbf{X}$,Y = 1)). El costo promedio de diseño de clasificadores óptimos empíricos usando la clase de funciones C_p se denota mediante $\mathbb{E}[\Delta(\psi_{C,N,*},\psi_{C,\text{opt}})]$. Este costo promedio de diseño está acotado superiormente por

$$\mathbb{E}[\Delta(\psi_{C,N,*},\psi_{C,\text{opt}})] \leq 4\sqrt{\left((m+3)ln(2)+n(m+1)ln(2N+1)\right)/N}\ . \qquad (3.13)$$

A partir de lo anterior se ha mostrado que si $m \rightarrow \infty$ y $m\times ln(N)/N \rightarrow 0$ con $N \rightarrow \infty$, entonces $\mathbb{E}[\Delta(\psi_{C,N,*},\psi_{C,\text{opt}})] \rightarrow 0$ para cualquier distribución conjunta Pr($\mathbf{X}$,Y), lo cual implica *consistencia universal de las redes neuronales tipo feed-forward* [Faragó y Lugosi, 1993]. Cabe recordar que *k*NN también es una regla de clasificación universalmente consistente (Capítulo II, Sección 2.4.1). En este contexto, la diferencia entre *k*NN y las redes neuronales tipo feed-forward es que *k*NN está afectado por la maldición de la dimensión. Esto implica que para lograr un determinado nivel de aproximación, o costo de diseño, la regla *k*NN requiere usualmente de una cantidad mucho más grande de datos entrenamiento que las redes neuronales tipo feed-forward [Faragó y Lugosi, 1993].

Desafortunadamente, las propiedades de aproximación y consistencia universal aplican para los casos donde las redes tipo feed-forward tienen una cantidad extremadamente grande de neuronas en la capa oculta. Además también se requiere de un conjunto muy grande de ejemplos de entrenamiento para reducir en gran medida el costo de diseño. En la práctica sucede todo lo contrario, la capacidad computacional es limitada al igual que los datos de entrenamiento y no siempre es posible encontrar el clasificador óptimo empírico. A pesar de esto, las redes neuronales tipo feed-forward convergen, usualmente, a buena solución de manera más rápida que los métodos no paramétricos como *k*NN o plug-in [Duda *et al.*, 2001], [Bishop, 2005], [Murphy, 2012]. Esto a su vez permite implementar fronteras de decisión complejas en escenarios con cantidades finitas y limitadas de ejemplos de entrenamiento. Una ventaja adicional de las redes neuronales tipo feed-forward, en el contexto de W-operadores, es que la representación computacional del operador diseñado tiene un bajo costo computacional. Esto debido a que sólo se almacena en la memoria RAM del computador el conjunto de parámetros ajustados $\boldsymbol{\beta}$, cuyo tamaño es muy pequeño en comparación con las tablas de búsqueda o colecciones de elementos estructurantes.

3.6. Anotaciones Finales

En este capítulo se ha propuesto un nuevo paradigma para el diseño automático de operadores de ventana o W-operadores. Bajo este paradigma, el problema de diseño de un W-operador se define como un problema de reconocimiento de patrones que consiste en el diseño estadístico de un clasificador. En el contexto de los W-operadores, el dominio de un clasificador es el espacio de las configuraciones de ventana y el rango es un conjunto de etiquetas. La entrada de un clasificador es un vector de variables aleatorias formado por los valores de los píxeles que se observan a través de una ventana. Para este vector es frecuente que los valores de sus variables estén correlacionados. Esto debido a que, en las imágenes, los niveles de gris o vectores de los píxeles más cercanos al píxel a procesar, usualmente, son similares o tienen poca variación.

Bajo el dominio de reconocimiento de patrones se analizó la teoría de aproximación y generalización de Vapnik y Chervonenkis para el diseño estadístico de clasificadores paramétricos. Esta teoría garantiza, en un sentido probabilístico que, si la cantidad de ejemplos de entrenamiento es mucho mayor que la complejidad de un modelo de clasificación, dada por su dimensión VC, entonces un bajo error de entrenamiento significa una buena capacidad de generalización del clasificador diseñado. Esto implica que, en las condiciones establecidas, un clasificador puede descubrir una estructura subyacente a los datos de entrenamiento (asumiendo que existe tal estructura), lo cual le permite tener un buen desempeño en la etapa de testeo. Por otro lado, cuando la complejidad del modelo de clasificación es mayor que el número de ejemplos de entrenamiento disponibles, el riesgo de sobreajuste u overfitting de sus parámetros es alto. El overfitting significa que un clasificador puede modelar los ejemplos de entrenamiento sin descubrir estructura alguna, con lo cual su performance en la etapa de testeo será mala. Para la teoría de Vapnik-Chervonenkis se asume independencia e idéntica distribución probabilística en los ejemplos de entrenamiento y testeo.

La sección final del presente capítulo se ha dedicado a la descripción del modelo de clasificación seleccionado para el diseño automático de W-operadores: las redes neuronales tipo feed-forward. En función de este modelo, el diseño de un W-operador consiste en el ajuste de los parámetros de una red neuronal tipo feed-forward usando un conjunto de configuraciones extraídas a partir de las imágenes de entrenamiento por medio de una ventana. Para un determinado problema, la arquitectura de la red que define su complejidad (dimensión VC) se selecciona teniendo en cuenta el número de ejemplos de entrenamiento disponibles. Se ha seleccionado a las redes neuronales tipo feed-forward debido a que son aproximadores universales y porque permiten implementar fronteras de decisión complejas

En los capítulos siguientes se hace uso de la teoría presentada para diseñar W-operadores que se aplican a problemas que involucran el uso de imágenes reales, y en particular, a problemas que involucran imágenes médicas.

3.7. Referencias

[1] Y.S. Abu-Mostafa, M. Magdon-Ismail and H.T. Lin, *Learning from data*, AMLBook, Pasadena CA, 2012.

[2] E.B. Baum and D. Haussler, "What Size Net Gives Valid Generalization?," *Neural Computation*, vol. 1, pp. 151-60, 1989.

[3] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Cambridge, 2005.

[4] C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

[5] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, pp. 21-7, 1967.
[6] L.A. Dalton, M.E. Benalcázar, M. Brun and E.R. Dougherty, "Bayes clustering operators for known random labeled point processes," *Proc. Signals, Systems and Computers, 2013 Asilomar Conference on*, 893-7 (2013).
[7] L.A. Dalton, M.E. Benalcázar, M. Brun and E.R. Dougherty, "Bayes Labeling and Bayes Clustering Operators for Random Labeled Point Processes," *IEEE Transaction on Signal Processing*, vol. Submitted paper, pp. 2014.
[8] L. Devroye, L. Györfi and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer, New York, 1996.
[9] E. Dougherty and J. Barrera, "Pattern Recognition Theory in Nonlinear Signal Processing," *Journal of Mathematical Imaging and Vision*, vol. 16, pp. 181-97, 2002.
[10] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification*, 2001.
[11] M. Egmont-Petersen, D.d. Ridder and H. Handels, "Image processing with neural networks—a review," *Pattern Recognition*, vol. 35, pp. 2279-301, 2002.
[12] A. Faragó and G. Lugosi, "Strong universal consistency of neural network classifiers," *Information Theory, IEEE Transactions on*, vol. 39, pp. 1146-51, 1993.
[13] M. Hagan, H. Demuth and M. Beale, *Neural Network Design*, PWS Publishing, Boston MA, 1996.
[14] T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer, New York, 2001.
[15] W. Hoeffding, "Probability Inequalities for Sums of Bounded Ra," *Journal of the American Statistical Association*, vol. 58, pp. 13-30, 1963.
[16] K. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-66, 1989.
[17] D.C. Martins-Jr, R.M. Cesar-Jr and J. Barrera, "W-operator window design by minimization of mean conditional entropy," *Pattern Analysis and Applications*, vol. 9, pp. 139-153, 2006.
[18] G. Matheron, *Random Sets and Integral Geometry*, New York, 1975.
[19] M. Mohri, A. Rostamizadeh and A. Talwalkar, *Foundations of Machine Learning*, MIT Press, London, 2012.
[20] K.P. Murphy, *Machine Learning: A probabilistic perspective*, MIT Press, Cambridge MA, 2012.
[21] D.E. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *Annals of Mathematical Statistics*, vol. 27, pp. 642-69, 1956.
[22] F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, Spartan Books, Washington DC, 1962.
[23] V. Vapnik and A. Chervonenkis, "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities," *Theory of Probability & Its Applications*, vol. 16, pp. 264-80, 1971.
[24] V.N. Vapnik, *The Nature of Statistical Learning*, Springer, New York, 2000.

CAPÍTULO IV

Método Propuesto para Diseño Automático de W-operadores Binarios

Resumen: Uno de los problemas relacionados con el diseño automático de W-operadores usando la regla plug-in es el de generalización. Como una posible solución para este problema, bajo el dominio de reconocimiento de patrones, en el presente capítulo se propone un nuevo método de generalización, restringido al diseño automático de W-operadores binarios. Este método consiste en el diseño de un clasificador binario que etiqueta las configuraciones observadas a través de una ventana. El diseño del clasificador se realiza mediante la regla plug-in y una regresión no lineal entre el cociente de las probabilidades de clase de las configuraciones de ventana y la respuesta de una red neuronal tipo feed-forward de tres capas. Para el entrenamiento de la red neuronal se utiliza como función de costo al error cuadrático medio ponderado. En la primera parte de este capítulo se propone el diseño de W-operadores binarios como un problema de diseño de clasificadores binarios. Luego se presenta un breve resumen del diseño de W-operadores usando la regla plug-in considerando el problema de generalización. En base a este análisis y teoría se presenta de manera formal el método propuesto para el diseño de W-operadores binarios. Luego se abordan aspectos concernientes a su implementación práctica. Finalmente, se presentan cuatro experimentos donde uno de ellos involucra el procesamiento de imágenes médicas y los tres restantes son problemas generales de PDI. El problema de imágenes médicas consiste en el filtrado de ruido en la segmentación de vasos sanguíneos en imágenes oculares. Los problemas generales de PDI aquí considerados son la detección de bordes en imágenes contaminadas con ruido artificial tipo sal-pimienta, identificación de texturas en imágenes de mapas, y reconocimiento de caracteres en imágenes de texto. En base a estos experimentos se analizan y comparan los resultados obtenidos con los resultados provenientes de la aplicación de operadores de imágenes diseñados usando la regla *k*NN, multi-resolución piramidal, máquinas de soporte vectorial, gradiente morfológico, y redes neuronales convolucionales.

4.1. Introducción

En este capítulo se propone un nuevo método para el diseño automático de W-operadores binarios donde la imagen de entrada (imagen observada) y la imagen de salida (imagen ideal) son imágenes binarias denotadas mediante O e I, respectivamente. Se asume que O e I son realizaciones de un par de procesos conjuntamente estacionarios $(\mathbf{O},\mathbf{I})$. Formalmente, un W-operador binario es una función $\boldsymbol{\Psi}: \{0,1\}^{\boldsymbol{E}} \rightarrow \{0,1\}^{\boldsymbol{E}}$ tal que la imagen resultante $\boldsymbol{\Psi}(O)$ de aplicar el operador $\boldsymbol{\Psi}$ a la imagen binaria observada O es una estimación de la imagen binaria ideal I, donde $\boldsymbol{E}\subset\mathbb{Z}^2$ y $\{0,1\}$ son el dominio y el rango de las imágenes binarias, respectivamente.

Para un W-operador binario $\boldsymbol{\Psi}$, su función característica es un clasificador $\psi: \boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$ que asigna una etiqueta $\psi(\mathbf{X})$ del conjunto de etiquetas $\mathcal{Y} = \{0,1\}$ a un vector formado por n características o features $\mathbf{X} = (X_1,\ldots,X_n)\in\boldsymbol{\mathcal{X}}$, con $X_i\in\{0,1\}$ $\forall X_i\in\mathbf{X}$. Aquí $\boldsymbol{\mathcal{X}}$ denota el espacio de todos los posibles $|\boldsymbol{\mathcal{X}}| = 2^n$ vectores binarios de n características o features. El vector de características o features $\mathbf{X}$ contiene una configuración observada en la imagen binaria O a través de una ventana $W = \{w_1,\ldots,(0,0),\ldots,w_n\}\subset\mathbb{Z}^2$ para un píxel arbitrario $t\in\boldsymbol{E}$ de dicha imagen. La etiqueta $\psi(\mathbf{X})$ para el vector $\mathbf{X}$ constituye una estimación de la variable aleatoria $Y = I(t)$. La distribución probabilística inducida por el par de procesos estocásticos conjuntamente estacionarios $(\mathbf{O},\mathbf{I})$ sobre la generación de los pares de vectores de características y etiquetas, $(\mathbf{X},Y)$, es $\Pr(\mathbf{X},Y)$. Por lo tanto, en el contexto de reconocimiento

de patrones, el problema de diseño automático de un W-operador binario consiste en el diseño de un clasificador binario.

4.2. Diseño de W-operadores Binarios usando la Regla Plug-in

La etiqueta que un clasificador binario ψ asigna a una configuración $\mathbf{X}$ depende de las probabilidades condicionales $\mathbb{P}(Y = 1|\mathbf{X})$ y $\mathbb{P}(Y = 0|\mathbf{X})$, con $\mathbb{P}(Y = 0|\mathbf{X}) = 1 - \mathbb{P}(Y = 1|\mathbf{X})$ (Capítulo III, Sección 3.2.1). Dado que la probabilidad conjunta $\mathbb{P}(\mathbf{X},Y = 1)$ de $(\mathbf{X},Y = 1)$ y la probabilidad marginal $\mathbb{P}(\mathbf{X})$ de $\mathbf{X}$ son desconocidas en una aplicación práctica, donde $\mathbb{P}(\mathbf{X},Y = 1) = \mathbb{P}(Y = 1|\mathbf{X})\mathbb{P}(\mathbf{X})$, entonces resulta imposible el cálculo del valor de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$. Ante esto, la regla plug-in permite estimar el valor de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ usando un conjunto de ejemplos de entrenamiento, o tabla de frecuencias, $\boldsymbol{\mathcal{D}} = \{(\mathbf{X}_i,freq(\mathbf{X}_i,Y = 0),freq(\mathbf{X}_i,Y = 1))\}$, con $i = 1,\ldots,N$ (Capítulo II, Sección 2.3). Aquí se asume que cada par vector-etiqueta $(\mathbf{X},Y)$, observado en las imágenes de entrenamiento, es una realización independiente de una distribución conjunta, fija, y desconocida $\Pr(\mathbf{X},Y)$. La estimación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ para una configuración $\mathbf{X}$ se realiza calculando el cociente entre la frecuencia del par $(\mathbf{X},Y = 1)$, $freq(\mathbf{X},Y = 1)$, y la frecuencia total del vector $\mathbf{X}$, $(freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1))$, mediante la siguiente expresión:

$$\hat{\mathbb{P}}(Y = 1|\mathbf{X}) = freq(\mathbf{X},Y = 1)/ (freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1)). \quad (4.1)$$

Se puede también estimar el valor de la probabilidad marginal $\mathbb{P}(\mathbf{X})$ del vector de características $\mathbf{X}$ calculando el cociente entre la frecuencia total de $\mathbf{X}$ y la suma de todas las frecuencias registradas en el conjunto de ejemplos de entrenamiento $\boldsymbol{\mathcal{D}}$:

$$\hat{\mathbb{P}}(\mathbf{X}) = (freq(\mathbf{X},Y = 0) + freq(\mathbf{X},Y = 1))/\sum_{(\mathbf{X},Y)\in\boldsymbol{\mathcal{D}}} freq(\mathbf{X},Y). \quad (4.2)$$

La etiqueta $\psi_N(\mathbf{X})$ estimada para un vector $\mathbf{X}$ depende del valor estimado $\hat{\mathbb{P}}(Y = 1|\mathbf{X})$ de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$, donde $\psi_N(\mathbf{X}) = 1$ si $\hat{\mathbb{P}}(Y = 1|\mathbf{X}) \geq \tau$ y $\psi_N(\mathbf{X}) = 0$ para el caso contrario. Si se selecciona un valor de umbral $\tau = 0.5$, entonces el clasificador ψ_N se denomina *clasificador óptimo empírico* debido a que es el clasificador con el mínimo error empírico $\varepsilon_N(\psi_N)$ calculado sobre los N elementos del conjunto de entrenamiento $\boldsymbol{\mathcal{D}}$. Usando este procedimiento, es posible definir las etiquetas únicamente para las configuraciones pertenecientes al conjunto $\boldsymbol{\mathcal{D}}$. Dicho de otro modo, con la regla plug-in sólo se dispone de información para etiquetar las configuraciones observadas durante el escaneo de las imágenes de entrenamiento. Sin embargo, el objetivo del diseño de W-operadores es encontrar un clasificador que permita predecir la etiqueta para toda configuración $\mathbf{X}\in\boldsymbol{\mathcal{X}}$ que se pueda registrar en las imágenes observadas dada una ventana W. A esta falta de capacidad predictiva de la regla plug-in para las configuraciones no registradas en la etapa de entrenamiento se denomina *problema de generalización*.

En este capítulo se propone un nuevo método para el diseño automático de W-operadores que resuelve el problema de generalización de la regla plug-in usando redes neuronales artificiales tipo feed-forward, o con conexión hacia adelante (Capítulo III, Sección 3.5). El método propuesto consiste en la estimación de las probabilidades condicionales $\mathbb{P}(Y = 1|\mathbf{X})$ para todas aquellas configuraciones de ventana $\mathbf{X}$ pertenecientes al conjunto $\mathcal{D}$ usando la regla plug-in. En base a estas estimaciones se diseña un clasificador binario ψ que particiona el espacio de configuraciones o características $\mathcal{X}$ en dos regiones disjuntas mediante funciones no lineales implementadas a través del uso de redes neuronales artificiales tipo feed-forward. El método propuesto y los resultados obtenidos fueron publicados en [Benalcázar *et al.*, 2012] y [Benalcázar *et al.*, 2013].

4.3. Método Propuesto: Regla Plug-in y Redes Neuronales Tipo Feed-Forward

En esta sección se presentan las definiciones y el procedimiento de implementación práctica del método propuesto.

4.3.1. Definiciones

Para una configuración de ventana $\mathbf{X}\in\mathcal{X}$ cualquiera, la suma de las probabilidades condicionales de clase debe ser igual a 1: $\mathbb{P}(Y = 0|\mathbf{X}) + \mathbb{P}(Y = 1|\mathbf{X}) = 1$. Si a la probabilidad condicional de cada clase se le suma una cantidad infinitesimal $\delta\in\mathbb{R}^+$, cuyo uso se volverá evidente más adelante, entonces se obtiene la expresión

$$(\mathbb{P}(Y = 0|\mathbf{X}) + \delta) + (\mathbb{P}(Y = 1|\mathbf{X}) + \delta) = 1 + 2\delta. \tag{4.3}$$

Si se define la función ϕ: $\mathcal{X} \rightarrow \mathbb{R}$ como el logaritmo natural del cociente entre los dos términos del lado izquierdo de la ecuación anterior, se obtiene la relación

$$\phi(\mathbf{X}) = ln\left(\frac{\mathbb{P}(Y=1|\,\mathbf{X})+\delta}{\mathbb{P}(Y=0|\,\mathbf{X})+\delta}\right) = ln\left(\frac{\mathbb{P}(Y=1|\,\mathbf{X})+\delta}{1-\mathbb{P}(Y=1|\,\mathbf{X})+\delta}\right). \tag{4.4}$$

Si se aproxima la función ϕ mediante una red neuronal tipo feed-forward de tres capas, $T(\bullet;\beta)\in C$ (Capítulo III, Sección 3.5), entonces se define la regresión no lineal dada por la ecuación 4.5.

$$\phi(\mathbf{X}) = ln\left(\frac{\mathbb{P}(Y=1|\,\mathbf{X})+\delta}{1-\mathbb{P}(Y=1|\,\mathbf{X})+\delta}\right) = T(\mathbf{X};\beta). \tag{4.5}$$

Resolviendo la ecuación 4.5 para $\mathbb{P}(Y = 1|\mathbf{X})$ en función de la respuesta de la red neuronal, se obtiene

$$\mathbb{P}(Y = 1|\mathbf{X};\beta) = \frac{exp(T(\mathbf{X};\beta))(1+\delta)-\delta}{1+exp(T(\mathbf{X};\beta))}, \tag{4.6}$$

donde β en $\mathbb{P}(Y = 1|\mathbf{X};\beta)$ enfatiza la aproximación realizada de la probabilidad condicional de la clase 1, $\mathbb{P}(Y = 1|\mathbf{X})$, dada la configuración de ventana **X** mediante una red neuronal tipo feed-forward de tres capas.

En este caso, la etiqueta $\psi(\mathbf{X})$ que la red neuronal asigna a una configuración **X** es

$$\psi(\mathbf{X}) = \begin{cases} 1 & \text{si } \mathbb{P}(Y = 1|\mathbf{X};\beta) \geq 0.5 \\ 0 & \text{en el caso contrario} \end{cases}. \tag{4.7}$$

Lo anterior es completamente equivalente a umbralar el valor de la respuesta $T(\mathbf{X};\beta)$ de la red neuronal para el vector de entrada **X** en 0:

$$\psi(\mathbf{X}) = \begin{cases} 1 & \text{si } T(\mathbf{X};\beta) \geq 0 \\ 0 & \text{en el caso contrario} \end{cases}. \tag{4.8}$$

4.3.2. Implementación Práctica

Para una configuración de ventana **X**, sean $\hat{\mathbb{P}}(Y = 1|\mathbf{X})$ y $\hat{\mathbb{P}}(\mathbf{X})$ los valores estimados de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ y la probabilidad marginal $\mathbb{P}(\mathbf{X})$, respectivamente. Estos valores estimados se obtienen usando la regla plug-in (ecuaciones 4.1 y 4.2) y un conjunto de N ejemplos de entrenamiento $\mathcal{D} = \{(\mathbf{X}_i, freq(\mathbf{X}_i, Y = 0), freq(\mathbf{X}_i, Y = 1))\}$. A partir de estos valores estimados de probabilidad se puede calcular el valor de $\phi(\mathbf{X})$ usando la ecuación 4.5 y fijando un valor infinitesimal para el parámetro δ. Este parámetro permite representar de manera computacional, con cantidades finitas, los valores de la función $\phi(\mathbf{X})$ para cuando el término $\hat{\mathbb{P}}(Y = 1|\mathbf{X})$ toma los valores de 0 ó 1.

Para una red neuronal tipo feed-forward con un número m fijo de neuronas y funciones de transferencia $f^{(1)}$ sigmoideas en la capa oculta, y una función de transferencia lineal $f^{(2)}$ para la única neurona de la capa de salida, la clase de funciones que se pueden implementar es $C_p = \{T(\mathbf{X};\beta) \mid \beta \in \mathbb{R}^p\}$. Dada la red neuronal C_p, la calidad de la aproximación de los valores de $\phi(\mathbf{X})$ calculados mediante la ecuación 4.5 con una función $T(\mathbf{X};\beta) \in C_p$ se mide mediante el error cuadrático medio (ECM) empírico ponderado dado por la ecuación 4.9.

$$\varepsilon_{ECMP,N}(\beta) = (1/2N) \sum_{i=1}^{N} (T(\mathbf{X}_i;\beta) - \phi(\mathbf{X}_i))^2 Pe(\mathbf{X}_i). \tag{4.9}$$

En la ecuación 4.9, el valor del peso $Pe(\mathbf{X}_i) \in [0,1]$ con el que la configuración $\mathbf{X}_i$ contribuye al valor de error se calcula mediante la siguiente expresión:

$$Pe(\mathbf{X}_i) = \frac{\hat{\mathbb{P}}(\mathbf{X}_i)}{max(\hat{\mathbb{P}}(\mathbf{X}_1),...,\hat{\mathbb{P}}(\mathbf{X}_N))}. \tag{4.10}$$

4.3.3. Entrenamiento de las Redes Neuronales usando el ECM Ponderado

El entrenamiento de una red neuronal consiste en encontrar un conjunto de parámetros que minimicen el ECM ponderado empírico dado por la ecuación 4.9. Debido a que el ECM

ponderado es una función no convexa en el espacio de los parámetros de la red neuronal, entonces existen varios mínimos locales (Figura 4.1). Por lo tanto, el entrenamiento o aprendizaje de la red neuronal consiste en resolver un problema de optimización en el que se encuentra, usualmente, un mínimo local para la función objetivo o función de costo dada por la ecuación 4.9. Si se representan los p elementos del conjunto $\boldsymbol{\beta}$ mediante los parámetros $\beta_1,\ldots,\beta_p$, entonces la minimización de la ecuación 4.9, usando el procedimiento usual del cálculo diferencial, equivale a resolver el sistema de ecuaciones 4.11.

$$\frac{\partial \varepsilon_{ECMP,N}(\boldsymbol{\beta})}{\partial \beta_i} = 0, \qquad \forall i = 1,\ldots,p. \tag{4.11}$$

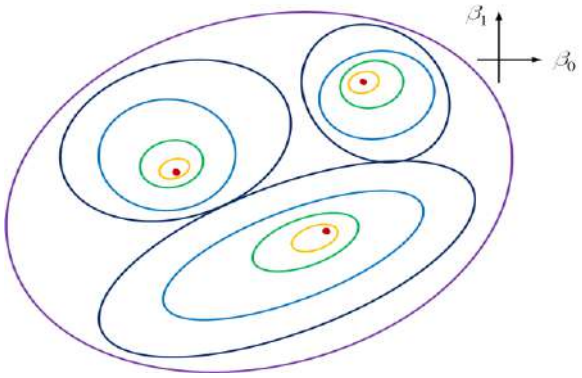


Figura 4.1. Ilustración de la superficie de la función de error $\varepsilon_{ECMP,N}(\boldsymbol{\beta})$ en el espacio de pesos $\boldsymbol{\beta}$ de una red neuronal mediante el uso de isolíneas. Los 3 mínimos locales están marcados con puntos rojos.

Lamentablemente, el sistema de ecuaciones 4.11 contiene ecuaciones trascendentes; es decir, ecuaciones que no pueden ser expresadas mediante una secuencia de operaciones algebraicas (suma, multiplicación, y raíz cuadrada). Por lo tanto, su minimización se debe realizar utilizando un método de optimización numérica. Es muy frecuente en este caso el uso del *método de descenso de gradiente* [Bishop, 2005], donde el ajuste de pesos de la red neuronal se realiza usando la ecuación 4.12.

$$\beta_i(k+1) \leftarrow \beta_i(k) - \alpha \frac{\partial \varepsilon_{ECMP,N}(\boldsymbol{\beta}(k))}{\partial \beta_i(k)}, \qquad \forall i = 1,\ldots,p. \tag{4.12}$$

donde k denota el número de *época de entrenamiento*, y α es un escalar que controla la *velocidad de aprendizaje* de la red neuronal, o el *tamaño del descenso de gradiente*. Las derivadas parciales de la función de error $\varepsilon_{ECMP,N}(\boldsymbol{\beta})$ con respecto a todos los parámetros $\beta_i \in \boldsymbol{\beta}$ de la red neuronal, o gradiente de la función de error, se calculan mediante el *algoritmo de retro-propagación del error*, o en inglés *error backpropagation algorithm* [Rumelhart *et al.*, 1986], [Duda *et al.*, 2001], [Bishop, 2005], [LeCun, 1988]. Este algoritmo es una implementación eficiente de la regla de la cadena del cálculo diferencial para el cálculo del valor de las derivadas parciales de la ecuación 4.12. En función de $\varepsilon_{ECMP,N}(\boldsymbol{\beta})$, las derivadas parciales de la ecuación 4.12 se pueden expresar mediante la ecuación 4.13.

$$\frac{\partial \varepsilon_{ECMP,N}(\boldsymbol{\beta})}{\partial \beta_i} = \frac{1}{N}\sum_{i=1}^{N}(T(\mathbf{X}_i;\beta) - \phi(\mathbf{X}_i))\,Pe(\mathbf{X}_i)\frac{\partial T(\mathbf{X}_i;\beta)}{\partial \beta_i}. \quad (4.13)$$

El uso del ECM ponderado, definido mediante la ecuación 4.9, como función de costo para el entrenamiento de una red neuronal, presenta dos ventajas importantes frente al ECM: (1) Las configuraciones de ventana contaminadas con ruido tienen menor influencia sobre el valor total de la función de costo y, (2) si las configuraciones están libres de ruido, entonces se puede prevenir el sobre-entrenamiento de la red [Sai *et al.*, 2009]. Para el primer caso se asume que las configuraciones con ruido se observan con menor frecuencia con respecto a las configuraciones libres de ruido. En función de esto, el peso del error cometido por la red neuronal para las configuraciones con ruido (ecuaciones 4.9 y 4.13) es menor que el peso de las configuraciones libres de ruido. Para la ecuación 4.9, una configuración $\mathbf{X}_i$ observada con probabilidad marginal estimada baja $\hat{\mathbb{P}}(\mathbf{X}_i)$ hace que el valor de $(T(\mathbf{X}_i;\beta) - \phi(\mathbf{X}_i))^2$ tenga una baja contribución sobre el valor total de $\varepsilon_{ECMP,N}(\boldsymbol{\beta})$. Del mismo modo, para la ecuación 4.13, un valor bajo para el peso $Pe(\mathbf{X}_i)$ de una configuración $\mathbf{X}_i$ hace que su contribución sobre el valor total de la derivada parcial también sea bajo.

Para el segundo caso descrito anteriormente, lo usual es que los valores de las probabilidades marginales de configuraciones muy cercanas a una configuración observada frecuentemente sean grandes. Del mismo, es usual también que las configuraciones muy próximas a una configuración observada con baja frecuencia tengan valores bajos de probabilidad marginal. En estas condiciones, el clasificar erróneamente una configuración observada con alta frecuencia, o sus vecinos más cercanos, contribuye de manera más significativa al aumento del error total de clasificación. Por el contrario, clasificar erróneamente una configuración con baja frecuencia, o sus vecinos más cercanos, no afecta de manera significativa al error total de clasificación.

Se debe tener en cuenta que el error de clasificación es función tanto de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ como de la probabilidad marginal $\mathbb{P}(\mathbf{X})$ (Capítulo III, Sección 3.2). Para este caso, el uso del ECM ponderado hace que la frontera de decisión sea ajustada durante el entrenamiento de la red con mayor precisión en las regiones del espacio $\mathcal{X}$ donde se encuentran las configuraciones observadas con alta frecuencia. Para el caso de las configuraciones observadas con baja frecuencia, si bien el valor de $(T(\mathbf{X}_i;\beta) - \phi(\mathbf{X}_i))^2$ puede ser alto, éste disminuye al ser multiplicado por un valor de peso $Pe(\mathbf{X}_i)$ bajo. Para el caso donde las configuraciones se observan con una probabilidad marginal estimada aproximadamente uniforme, el valor de los pesos es similar para todas las configuraciones observadas. En este escenario, es la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$, o su estimado, la que determina el ajuste de la frontera de decisión que implementa la red neuronal.

En base a este análisis, el entrenamiento de una red neuronal que define a un W-operador binario se resume en los siguientes pasos:

1) Inicialización del conjunto de pesos de la red neuronal. Debido a la forma en la que opera el algoritmo de retro-propagación del error es contraproducente inicializar todos los pesos en un mismo valor, por ejemplo 0. Para este caso, el ajuste de pesos utilizando el método de descenso de gradiente y el algoritmo de retro-propagación ocasionaría que todas

las neuronas de la red terminen con los mismos pesos al final del entrenamiento. En su lugar, lo usual es realizar una inicialización aleatoria de todos los pesos mediante, por ejemplo, una distribución normal multivariable estándar [Bishop, 2005]. Es este paso el que hace que las redes neuronales sean un método de aprendizaje computacional supervisado estocástico.

2) Cálculo de la respuesta de la red neuronal para todas las configuraciones del conjunto de entrenamiento, o propagación hacia adelante (ecuaciones 3.9 y 3.10).

3) Cálculo del error de entrenamiento (ecuaciones 4.9 y 4.10).

4) Cómputo del valor de las derivadas parciales (ecuación 4.13) de la función de costo (ecuación 4.9) con respecto a todos los parámetros de la red neuronal utilizando el algoritmo de retro-propagación del error. Este paso también se denomina cálculo del valor del gradiente de la función de costo.

5) Ajuste del valor de los pesos (ecuación 4.12).

El número de épocas de entrenamiento (pasos del 2 al 5) se puede fijar previo al ajuste de los parámetros de la red. También se puede finalizar el entrenamiento en base al valor del error de entrenamiento calculado para cada época en el paso 3. Cuando este error es menor que un umbral determinado o cuando su variación entre épocas consecutivas es menor que un umbral determinado se detiene el entrenamiento. Se ha demostrado también que, si la cantidad de datos de entrenamiento es suficientemente grande para evitar el sobre-entrenamiento de la red u overfitting, se debe minimizar el error de entrenamiento tanto como sea posible [Cataltepe *et al.*, 1999]. Sin embargo, para los casos donde el número de ejemplos de entrenamiento es grande, esto puede acarrear un elevado costo computacional de entrenamiento.

Otra alternativa consiste en, previo al entrenamiento de la red, dividir el conjunto de datos $\mathcal{D}$ en un subconjunto para el ajuste de parámetros y otro subconjunto para validar el poder predictivo de la red en cada época de aprendizaje. En este caso como resultado del entrenamiento de la red se retorna el conjunto de parámetros con el mejor desempeño sobre el conjunto de datos de validación (Figura 4.2). La proporción en la que se divide $\mathcal{D}$ es usualmente 70% para entrenamiento y el 30% restante para validación [Bishop, 2005]. Es usual que esta última alternativa resulte en mejores resultados que las opciones antes mencionadas debido a que el error de validación es menos optimista que el error de entrenamiento [Bishop, 2005], [Abu-Mostafa *et al.*, 2012]. Además en base a esta técnica de validación se puede también seleccionar el mejor de todos los mínimos locales encontrados al entrenar la red varias veces utilizando diferentes puntos de inicio. El costo computacional (tiempo y memoria) de cada época de entrenamiento de una red neuronal depende de la cantidad de ejemplos de entrenamiento y el método utilizado para la optimización.

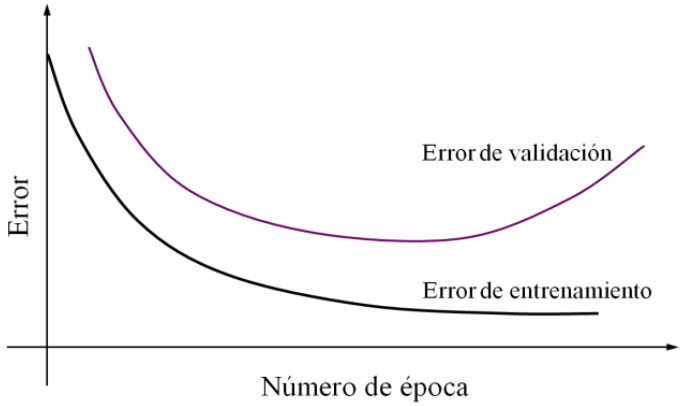


Figura 4.2. Ilustración de las curvas de error de entrenamiento y error de validación para el entrenamiento por épocas de una red neuronal.

Para el método de descenso de gradiente, la velocidad de convergencia hacia un mínimo local depende del valor inicial de los pesos, del valor de la velocidad de aprendizaje, y de la forma de la superficie de error en el espacio de parámetros de la red neuronal. En la Figura 4.3 se muestra un caso donde la velocidad de convergencia será lenta debido a que el gradiente calculado en el punto correspondiente a los pesos iniciales no apunta en la dirección del mínimo local. Nótese también la forma elipsoidal de las isolíneas de la función de costo de la red neuronal en el espacio de parámetros. En este caso, en cada época, el descenso de gradiente oscilará perpendicularmente con relación a la dirección en la que se encuentra el mínimo local.

Para el caso descrito anteriormente, el acercamiento efectivo del descenso de gradiente hacia el mínimo local es demasiado lento, requiriendo de un número grande de épocas para su convergencia hacia el mínimo local. Este problema se agrava cuando se consideran valores muy altos para la velocidad de aprendizaje de la red neuronal. A nivel práctico, esto incrementa notablemente el tiempo de entrenamiento y la demanda de memoria RAM. Una alternativa para disminuir el nivel de oscilación es variar la velocidad de aprendizaje durante el entrenamiento y/o modificar el punto de inicio del descenso de gradiente, sin que esto garantice evitar el problema de oscilación, [Vogl *et al.*, 1988], [LeCun *et al.*, 1998].

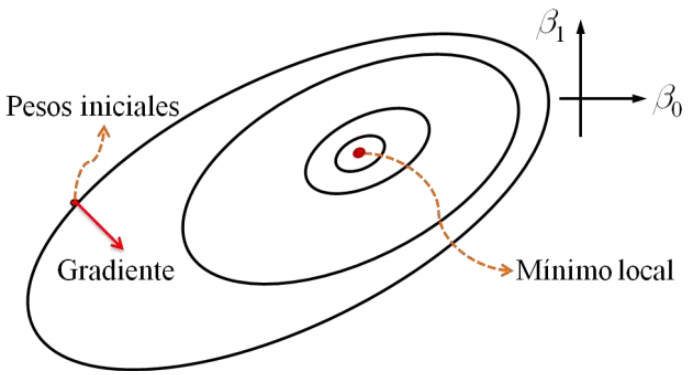


Figura 4.3. Ilustración de la influencia del valor inicial de los pesos sobre la velocidad de convergencia del método de descenso de gradiente para el entrenamiento de una red neuronal.

A pesar de que el algoritmo de descenso de gradiente, con velocidad de aprendizaje fija, permite explicar muy fácilmente el proceso de entrenamiento de una red neuronal, su velocidad de convergencia hacia un mínimo local es usualmente lenta. Por esta razón, en

este trabajo se utiliza el *método de optimización de Levenberg-Marquardt* para el entrenamiento de las redes neuronales [Levenberg, 1944], [Marquardt, 1963]. Este método sirve para la minimización únicamente de funciones de costo cuadráticas como la función de error presentada en la ecuación 4.9. Para ello este método monitorea la función de error y modifica continuamente el tamaño del descenso de gradiente o el inverso de la velocidad de aprendizaje. De esta manera se asegura una progresiva disminución del valor de la función de error por cada época de entrenamiento [Hagan y Menhaj, 1994], [Bishop, 2005].

La actualización de los pesos con el método de Levenberg-Marquardt se realiza en base a

$$\mathbf{B}(k+1) \leftarrow \mathbf{B}(k) - [\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}]^{-1}\mathbf{J}^T\mathbf{E}, \tag{4.14}$$

donde **B** es una matriz conteniendo todos los parámetros de la red neuronal, **J** es la matriz jacobiana de la función de costo, **E** es un vector conteniendo todos los N valores de error calculados mediante la ecuación 4.9, λ controla el tamaño del descenso de gradiente, e **I** es la matriz identidad. La matriz jacobiana **J** contiene el valor de todas las derivadas parciales de la función de costo con respecto a todos los parámetros de la red neuronal (ecuación 4.13). Los valores de esta matriz se obtienen usando el algoritmo de retro-propagación del error.

Durante el entrenamiento de la red, si el valor de la función de error disminuye, entonces se reduce también el valor del parámetro λ aumentando de esta manera la velocidad de aprendizaje. Luego se actualizan los valores de los pesos y se pasa a la siguiente época. Por el contrario, si el valor de la función de error aumenta, entonces se incrementa el valor de λ, disminuyendo así la velocidad de aprendizaje. A continuación se calcula nuevamente el valor de la función de costo en base a los pesos obtenidos con esta nueva velocidad de aprendizaje. Si esta actualización de los pesos hace que el valor de la función de costo (ecuación 4.9) disminuya, entonces se pasa a la siguiente época; en el caso contrario, se repite este procedimiento aumentando nuevamente el valor de λ.

Según lo anterior, el número de épocas requerido para alcanzar un mínimo local usando el método de Levenberg-Marquardt es usualmente menor que el número de épocas utilizando el método de descenso de gradiente. Sin embargo, en algunos casos es posible que este método tarde un largo período de tiempo por cada época monitoreando la función de error y modificando la velocidad de aprendizaje. Además el uso de memoria en este caso es mayor que en el descenso de gradiente. Esto se debe a que en este método se realiza el cálculo de la inversa de la matriz $[\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}]$ varias veces por cada época de entrenamiento. A pesar de estos inconvenientes, en la práctica este método es uno de los más rápidos para el entrenamiento de redes neuronales conteniendo un número de parámetros del orden de las centenas [Hagan *et al.*, 1996].

4.4. Entrenamiento y Aplicación de W-operadores en base al Método Propuesto

A continuación se presenta el procedimiento tanto de entrenamiento como de aplicación para el diseño automático de W-operadores binarios en base al método propuesto.

4.4.1. Procedimiento de Entrenamiento

Sea el par de imágenes de entrenamiento binarias (O,I), que son realizaciones de un par de procesos estocásticos conjuntamente estacionarios $(\mathbf{O},\mathbf{I})$ (el cual es desconocido en la práctica), y la ventana W conteniendo n píxeles. Los pasos que integran el procedimiento de entrenamiento son:

***1. Recolección de observaciones*:** Aquí se escanea cada píxel t de las imágenes O e I usando la ventana W para obtener los pares $(\mathbf{X},Y)$. En base a estos pares se construye una tabla de frecuencias, o conjunto de ejemplos de entrenamiento, $\mathcal{D}$ donde se guardan cada una de las diferentes configuraciones $\mathbf{X}$ observadas con sus respectivas frecuencias $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$: $\mathcal{D} = \{(\mathbf{X}_i,freq(\mathbf{X}_i,Y = 0),freq(\mathbf{X}_i,Y = 1))\}$, con $i = 1,\ldots,N$ (Capítulo I, Sección 1.5.3).

2. Estimación de probabilidades: Usando el conjunto de ejemplos de entrenamiento $\mathcal{D}$, y en base a la regla plug-in, se estima el valor de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ y el valor de la probabilidad marginal $\mathbb{P}(\mathbf{X})$ para cada configuración $\mathbf{X}$ registrada en el paso anterior.

3. Entrenamiento de la red neuronal: Primero se define una arquitectura de red neuronal tipo feed-forward con m neuronas para la capa oculta. Las funciones de transferencia para la capa oculta y la capa de salida son $f^{(1)}: \mathbb{R} \rightarrow [-1,1]$ y $f^{(2)}: \mathbb{R} \rightarrow \mathbb{R}$, respectivamente. Estas funciones se definen como $f^{(1)}(z) = tansig(z) = (2/(1 + exp(-2z))) - 1$ y $f^{(2)}(z) = z$. Se selecciona la función tangente hiperbólica sigmoidea *tansig* (Figura 4.4) para la función de transferencia $f^{(1)}$ debido a que, en conjunto con la función lineal de la capa de salida, permite la aproximación de funciones cuyo rango está formado por números positivos y negativos, incluyendo el cero [Hagan *et al.*, 1996].

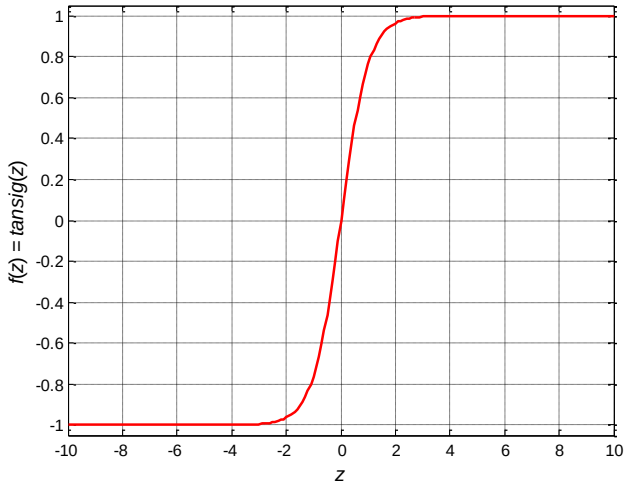


Figura 4.4. Forma de la función tangente hiperbólica sigmoidea *tansig* para el entrenamiento de redes neuronales.

El entrenamiento de la red neuronal se realiza utilizando el método de optimización de Levenberg-Marquardt, donde la función a optimizar es la función de error dada por la

ecuación 4.9. Para esto se usan las tripletas $(\mathbf{X},\phi(\mathbf{X}),Pe(\mathbf{X}))$, donde $\phi(\mathbf{X})$ se calcula mediante la ecuación 4.5 en base a los estimados de las probabilidades condicionales obtenidos en el paso anterior. Se selecciona un valor $\delta = 2^{-52}$ que corresponde a la distancia entre el número 1 y el número de doble precisión más cercano que se puede representar en un computador. El peso $Pe(\mathbf{X})$ se calcula en base al estimado de la probabilidad marginal usando la ecuación 4.10.

Ejemplo 4.1. En la Figura 4.5 se ilustra gráficamente el procedimiento antes descrito aplicado al diseño automático de W-operadores para el filtrado de ruido puntual aditivo y sustractivo en imágenes binarias. La imagen observada *O* es una versión deteriorada de la imagen ideal *I*. En este caso se utiliza una ventana cuadrada *W* de $n = 3\times3$ píxeles para el escaneo del par de imágenes de entrenamiento *O* e *I*. Como resultado del diseño se obtiene una red neuronal entrenada que representa la función característica del W-operador diseñado.

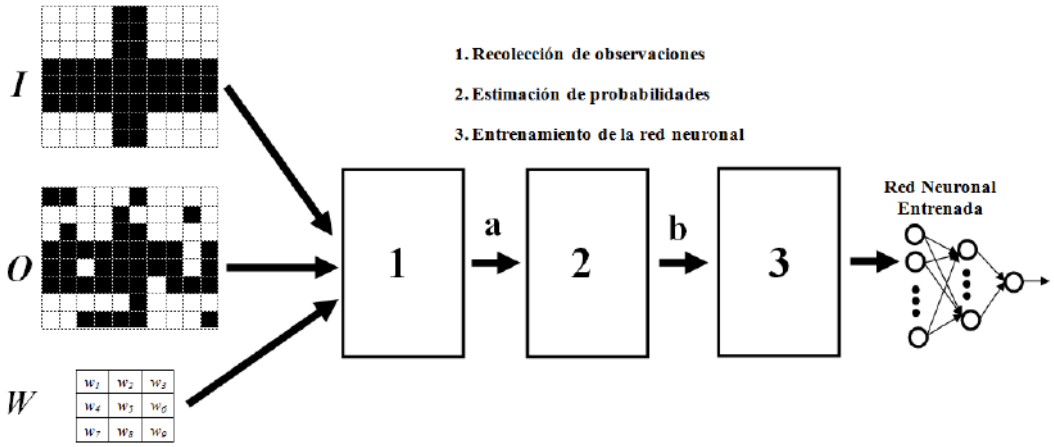


Figura 4.5. Esquema de entrenamiento de W-operadores diseñados en base a redes neuronales tipo feed-forward para el filtrado de ruido puntual aditivo y sustractivo en imágenes binarias.

4.4.2. Procedimiento de Aplicación

Una vez entrenada la red neuronal, a continuación se aplica el operador diseñado a una imagen *O*', que es una realización del proceso estocástico estacionario **O**. Para esto se utiliza el siguiente procedimiento.

***1. Recolección de observaciones*:** Consiste en obtener la observación **X**, usando la ventana *W*, para el punto *t* de la imagen a procesar *O*'.

2. Consulta a la red neuronal: Para la observación **X** se calcula la respuesta $T(\mathbf{X};\beta)$ de la red neuronal entrenada. A partir de este valor se obtiene la cantidad $\mathbb{P}(Y = 1|\mathbf{X};\beta)$, usando la ecuación 4.6, que es un estimado de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ dada el vector de configuración **X**. Es importante mencionar que este último cálculo no es necesario si el umbral para $T(\mathbf{X};\beta)$ es 0, lo que es completamente equivalente a seleccionar un umbral de 0.5 para $\mathbb{P}(Y = 1|\mathbf{X};\beta)$. Para el caso contrario, resulta más intuitivo umbralar un valor que representa una medida de probabilidad que una función con rango infinito.

***3. Estimación de la función característica*:** Se define la etiqueta $\psi(\mathbf{X})\in\{0,1\}$ para la configuración $\mathbf{X}$ utilizando el valor de $T(\mathbf{X};\beta)$ o el estimado $\mathbb{P}(Y = 1|\mathbf{X};\beta)$ en base a la regla de decisión presentada en las ecuaciones 4.13 y 4.14, respectivamente.

***4. Asignación de Valores*:** Se asigna al punto t de la imagen resultante del procesamiento $\mathbf{\Psi}(O')$ el valor de $\psi(\mathbf{X})$: $\mathbf{\Psi}(O')(t) = \psi(\mathbf{X})$.

Este procedimiento se repite para todo punto t de la imagen observada O', siempre que la ventana W esté totalmente contenida en su dominio. Caso contrario, el valor del punto t en la imagen de salida es igual al valor de dicho punto en la imagen de entrada. Esto implica que los píxeles de borde de la imagen O' se mantienen intactos, lo cual es posible debido a que las imágenes de entrada y salida del procesamiento son imágenes binarias. Este criterio no constituye un problema para el procesamiento debido a que, usualmente, los objetos de interés se encuentran cercanos al centro de las imágenes.

Ejemplo 4.2. En la Figura 4.6 se ilustra gráficamente el procedimiento de aplicación de W-operadores para el filtrado de ruido puntual aditivo y sustractivo en imágenes binarias. En este caso la imagen a ser procesada es una imagen binaria O' compuesta por tres líneas horizontales deterioradas por ruido puntual aditivo y sustractivo. Para esto se debe utilizar la misma ventana W empleada para la etapa de entrenamiento. Como resultado de la aplicación del W-operador, representado mediante una red neuronal entrenada, se obtiene la imagen binaria procesada $\mathbf{\Psi}(O')$. Nótese como los valores de los píxeles de borde no se modifican, lo cual es posible gracias a que tanto la imagen de entrada como la imagen de salida del procesamiento son imágenes binarias. Se puede también observar que la restauración de la línea central no es perfecta.

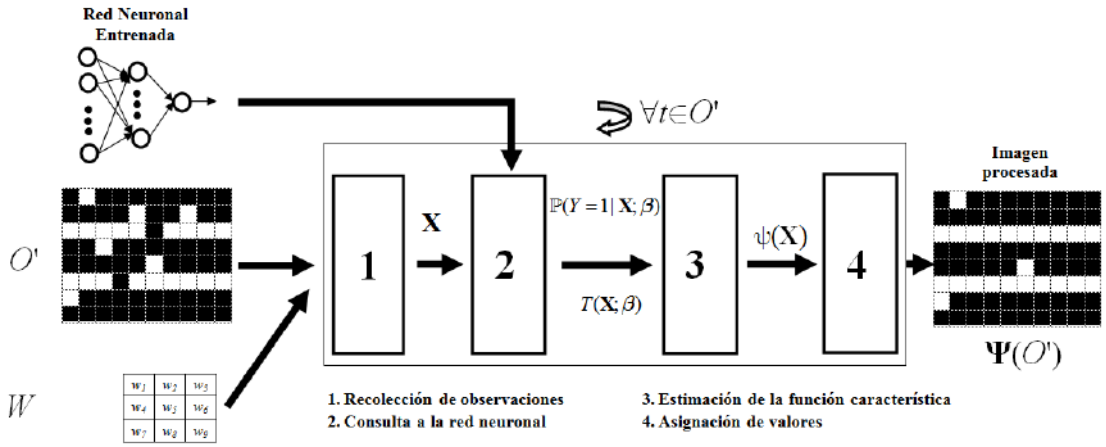


Figura 4.6. Esquema de aplicación de W-operadores diseñados en base a redes neuronales tipo feed-forward para el filtrado de ruido puntual aditivo y sustractivo en imágenes binarias.

4.5. Experimentos, Resultados, y Discusión

En esta sección se presentan cuatro aplicaciones del método propuesto que consisten en: (1) filtrado de ruido en la segmentación de los vasos sanguíneos de imágenes oculares, (2) detección de bordes en imágenes contaminadas con ruido puntual aditivo y sustractivo, (3) identificación de texturas en imágenes con escaneados de mapas, y (4) reconocimiento de caracteres en imágenes con escaneados de texto.

4.5.1. Métricas de Evaluación y Comparación de Resultados

Para la evaluación de la calidad de los resultados, en todos los casos se realizó una comparación entre las imágenes procesadas con sus respectivas imágenes de gold estándar. En base a esta comparación se realizó el cálculo de la tasa de error (ER del inglés *error rate*), la tasa de falsos positivos (FPR del inglés *false positive rate*), y la tasa de falsos negativos (FNR del inglés *false negative rate*). En cada caso, a menos que se aclare algo diferente, ER representa la proporción de píxeles que difieren entre la imagen resultante del procesamiento y su correspondiente imagen ideal (o gold estándar). FPR representa la proporción de píxeles del fondo de la imagen ideal que aparecen como parte del objeto de interés en la imagen resultante. FNR representa la proporción de píxeles del objeto de interés en la imagen ideal que aparecen como fondo en la imagen resultante.

En función de lo anterior se puede notar que ER es una estimación de la probabilidad de clasificar erróneamente un píxel, $\mathbb{P}(\psi(\mathbf{X}) \neq Y)$. FPR es una estimación de la probabilidad de que un píxel sea etiquetado con 1 dado que la etiqueta verdadera es 0, $\mathbb{P}(\psi(\mathbf{X}) = 1|Y = 0)$. FNR es una estimación de la probabilidad de que un píxel sea etiquetado con 0 dado que la etiqueta verdadera es 1, $\mathbb{P}(\psi(\mathbf{X}) = 0|Y = 1)$.

4.5.2. Descripción de los Métodos de Diseño Utilizados

Los resultados obtenidos con el método propuesto se comparan con aquellos obtenidos utilizando métodos de diseño de W-operadores como *k*NN y multi-resolución piramidal. También se realizan comparaciones, para todos los experimentos, con uno de los algoritmos clásicos frecuentemente utilizado dentro de reconocimiento de patrones como es la máquina de soporte vectorial (SVM del inglés *support vector machine*) [Vapnik, 2000]. Para el problema de detección de bordes también se realizan comparaciones con el gradiente morfológico que es el método tradicional de la morfología matemática para esta tarea. Para el reconocimiento de caracteres se realizan también comparaciones con los resultados obtenidos al utilizar redes neuronales convolucionales (CNNs del inglés *convolutional neural networks*). Este método de aprendizaje computacional supervisado fue propuesto para el reconocimiento de números en imágenes de códigos postales [LeCun y Bengio, 1995].

Redes Neuronales tipo Feed-Forward: El número *m* de neuronas de la capa oculta para las redes neuronales tipo feed-forward utilizadas en cada experimento se obtuvo utilizando la regla $m = 4\sqrt{n}$, donde *n* es el tamaño de la ventana utilizada en cada experimento. Esta heurística fue seleccionada teniendo en cuenta el tamaño promedio de las imágenes utilizadas en cada experimento. Este tamaño promedio de las imágenes influye directamente sobre el número *N* de distintas configuraciones para el entrenamiento de la red neuronal. Al definir $m = 4\sqrt{n}$ se procura que la cota superior del costo de diseño de un clasificador utilizando redes neuronales tipo feed-forward (Capítulo III, ecuación 3.13) sea mucho menor que 1 para todos los experimentos. En estas condiciones se previene el sobre-entrenamiento de las redes neuronales (overfitting) que conduciría a una mala performance de los operadores diseñados durante el testeo.

Regla kNN: Para el caso de la regla *k*NN se utiliza la distancia de Hamming para medir la similitud entre dos configuraciones de ventana. Para el caso del número de vecinos más cercanos, *k*, se utiliza la regla heurística $k = \lceil log_{10}(N) \rceil$, donde $\lceil log_{10}(N) \rceil$ es el entero impar más cercano que es mayor o igual que $log_{10}(N)$.

Multi-resolución: Para el diseño de W-operadores con multi-resolución (Capítulo II, Sección 2.8.1), las pirámides utilizadas en cada experimento se representan gráficamente mediante un esquema de ventana. La Figura 4.7 muestra un ejemplo de este esquema de representación para una pirámide formada por 5 ventanas: $\mathcal{V} = \{W_1,\ldots,W_5 \mid W_1 \supset \ldots \supset W_5\}$. En este caso, la ventana W_i de la pirámide $\mathcal{V}$ se obtiene seleccionando únicamente los puntos cuyos valores son iguales a o mayores que *i*, con $i = 1,\ldots,5$. Para el diseño de W-operadores en base al método propuesto siempre se utiliza la ventana más grande de las pirámides indicadas en cada experimento

2	**1**	**2**	**1**	**2**
0	**4**	**3**	**4**	**0**
2	**3**	**5**	**3**	**2**
0	**4**	**3**	**4**	**0**
2	**1**	**2**	**1**	**2**

Figura 4.7. Representación de una pirámide formada por 5 ventanas.

Máquina de Soporte Vectorial: El *SVM* es un clasificador que asigna etiquetas en función de la siguiente regla de decisión:

$$\varphi(\mathbf{X}) = \begin{cases} 1 & \text{si } \sum_{i=1}^{q} Y_i \beta_i K(\mathbf{X}, \mathbf{X}_i) - \beta_0 \geq 0 \\ -1 & \text{en el caso contrario} \end{cases}, \tag{4.15}$$

donde $K(\mathbf{X},\mathbf{X}_i)$ denota el valor del kernel entre el vector $\mathbf{X}$ a ser clasificado y el vector de soporte $\mathbf{X}_i$. La variable $Y_i \in \{-1,1\}$ denota la etiqueta del vector de soporte $\mathbf{X}_i$. Los términos β_0 y β_i denotan un valor de umbral y el peso que tiene el valor del kernel $K(\mathbf{X},\mathbf{X}_i)$ dentro de la clasificación, respectivamente. Una determinada función puede ser considerada como un kernel para SVM si es continua, positiva, y simétrica. Los tipos de kernel más utilizados son el lineal definido como $K(\mathbf{X},\mathbf{X}') = \mathbf{X} \bullet \mathbf{X}'$, donde $\mathbf{X} \bullet \mathbf{X}'$ denota el producto escalar entre los vectores $\mathbf{X}$ y $\mathbf{X}'$; el polinómico de orden *d* definido como $K(\mathbf{X},\mathbf{X}') = (\mathbf{X} \bullet \mathbf{X}' + 1)^d$; o la función de base radial Gausiana $K(\mathbf{X},\mathbf{X}') = exp(-b\|\mathbf{X} - \mathbf{X}'\|^2)$, donde $b > 0$ es un factor de escala. El papel de la función de kernel dentro de SVM es medir el grado de similitud que existe entre el vector a clasificar $\mathbf{X}$ y el vector de soporte $\mathbf{X}'$ [Burges, 1998], [Cristianini y Shawe-Taylor, 2000], [Vapnik, 2000], [Scholkopf y Smola, 2002]. Para todos los experimentos realizados se utilizó el kernel de base radial Gausiano con $b = 1$.

Nótese que SVM es un clasificador lineal, lo cual es evidente para el caso del kernel lineal, donde la frontera de decisión en el espacio $\mathcal{X}$ es el hiperplano óptimo o hiperplano de máxima margen [Vapnik, 2000]. Para los tipos de kernel no lineales, SVM sigue siendo un clasificador lineal (en función del vector de parámetros β). Aquí SVM separa las configuraciones de las clases a clasificar mediante la implementación del hiperplano

óptimo, o hiperplano de máximo margen, en un espacio de características $\mathcal{Z}$ de mayor dimensión que el espacio $\mathcal{X}$. El entrenamiento de los SVMs se realiza utilizando los pares $(\mathbf{X},Y)$, con $Y \in \{-1,1\}$. Para todos los experimentos, la etiqueta Y para $\mathbf{X}$ se obtiene umbralando el valor estimado de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ en base a la regla plug-in. Dado el conjunto $\mathcal{D} = \{(\mathbf{X}_i,Y)\}$, con $i = 1,\ldots,N$, el entrenamiento de un SVM consiste en encontrar los vectores de soporte entre las configuraciones del conjunto $\mathcal{D}$, y los pesos β_i, con $i = 0,\ldots,n$, mediante la resolución de un problema de programación cuadrática [Vapnik, 2000]. Esto hace que los SVMs sean factibles, desde el punto de vista computacional, para problemas donde $\mathcal{D}$ contiene una cantidad de ejemplos N de un orden menor que o igual a las decenas de miles.

Redes Neuronales Convolucionales: CNN es un tipo de red neuronal que permite clasificar o etiquetar imágenes. Este modelo está basado en dos operaciones que son la convolución y el pooling u operación de asociación. En base a estas dos operaciones se extrae un conjunto de características que permiten clasificar a una imagen. En la Figura 4.8 se presenta una ilustración de las operaciones de convolución y pooling de CNN. En esta figura, la imagen a ser clasificada, mostrada en verde, está formada por 5x5 píxeles. La máscara de convolución, mostrada en amarillo, tiene 3x3 píxeles. La imagen resultante de la convolución está formada por 3x3 píxeles. En la Figura 4.8 la operación de pooling corresponde al cálculo del valor máximo de los 9 píxeles de la imagen resultante de la convolución, resultando en el valor de 4. Dependiendo de la aplicación, es posible el uso de otras estadísticas como el mínimo, el promedio, o la mediana. También se puede realizar un submuestreo de la imagen resultante de la convolución para obtener una nueva imagen de menor tamaño.

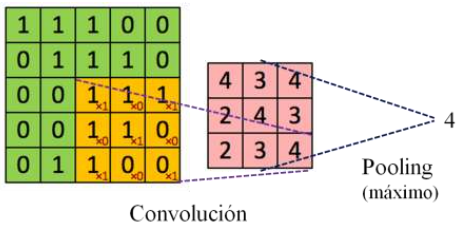


Figura 4.8. Ilustración de las operaciones de convolución y pooling de CNN.

En la Figura 4.9 se muestra una imagen a clasificar compuesta por 20×20 píxeles. Al convolucionar esta imagen con 2 máscaras diferentes se obtienen 2 imágenes, en cada una de las cuales se marcan 4 regiones disjuntas. Cada región se compone de 9×9 píxeles. Por cada región de cada imagen resultante de la convolución se obtiene una característica a través de la operación de pooling. En total, en este ejemplo se obtienen 8 características, las cuales son alimentadas a un clasificador para estimar la etiqueta correspondiente a la imagen original. En este caso la red neuronal convolucional está formada por cuatro capas. La capa 0 es la capa de entrada, donde usualmente no se realiza ningún cálculo. En la capa 1 se realiza la operación de convolución con dos máscaras diferentes. En la capa 2 se realiza la operación de pooling. En la capa 3 se realiza la clasificación.

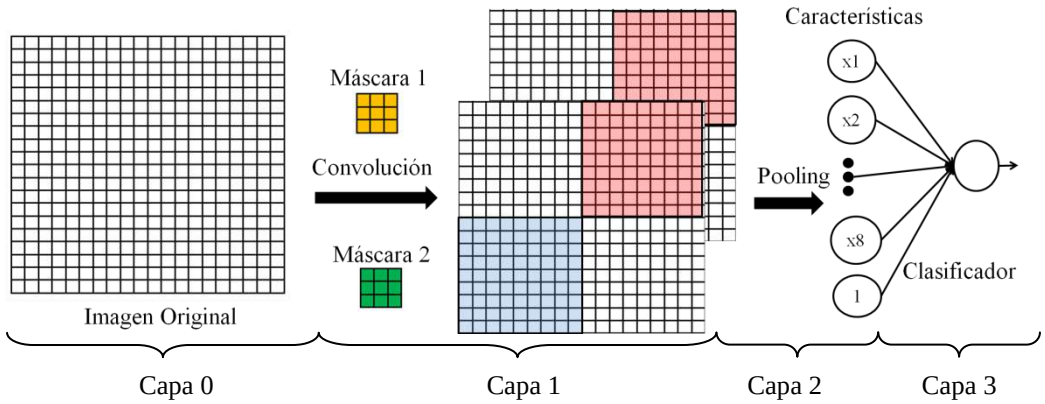


Figura 4.9. Ilustración de una red neuronal convolucional de cuatro capas. La capa 0 es la capa de entrada, donde no se realiza ningún cálculo. La capa 1 está compuesta por dos máscaras de convolución. La capa 2 realiza la operación de pooling. La capa 3 es un clasificador que permite etiquetar a la imagen original en base al vector de características obtenido a partir de las capas 0, 1, y 2.

En la práctica es usual añadir varias capas de convolución y pooling previo a la etapa de clasificación. Es común también el uso de clasificadores lineales que asignan una etiqueta del conjunto {-1,1} al umbralamiento del producto punto entre el vector de características y el vector de pesos de la capa de salida. También se pueden utilizar clasificadores no lineales como redes neuronales tipo feed-forward. En general, el entrenamiento de una red neuronal convolucional consiste en ajustar los coeficientes de las máscaras de convolución de cada capa y los pesos del clasificador de la capa de salida. Además se debe definir la operación de pooling de cada capa. Este entrenamiento se realiza utilizando el método de descenso de gradiente combinado con retro-propagación del error [LeCun y Bengio, 1995].

4.5.3. Protocolo para el Diseño y Testeo

Para cada experimento se utilizó un conjunto de 4 pares de imágenes de entrenamiento (observada e ideal). De este conjunto se usó un único par de imágenes para entrenamiento, mientras que los 3 pares restantes se utilizaron para el testeo. Este procedimiento se ejecutó 4 veces, de tal manera que en cada iteración se utilizó un par de imágenes diferente para entrenamiento y los pares restantes para el testeo. Este procedimiento se denomina validación cruzada [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012]. Los resultados numéricos que se reportan en la Tabla 4.1, al final de esta sección, representan el promedio de los resultados de todas las 4 iteraciones ejecutadas en cada experimento.

4.5.4. Filtrado de Ruido en Imágenes Oculares

El objetivo de este experimento es diseñar automáticamente W-operadores para filtrar el ruido en la segmentación de los vasos sanguíneos en imágenes oculares. Los vasos sanguíneos fueron segmentados automáticamente utilizando un algoritmo diseñado heurísticamente en base a morfología matemática difusa y presentado en [Bouchet *et al.*, 2010]. Este algoritmo aplica el top-hat difuso por apertura al canal correspondiente al color verde de las imágenes oculares color RGB. Se utiliza este canal debido a que es el canal que presenta el nivel más bajo de ruido y el nivel más alto de contraste entre los vasos sanguíneos y el fondo de la imagen en comparación con los dos canales restantes (rojo y azul) [Staal *et al.*, 2004], [Fraz *et al.*, 2012]. Posterior al top-hat se aplica un procedimiento de umbralamiento para obtener una imagen binaria con los vasos sanguíneos segmentados. La principal característica de estas imágenes es que, aparte de los vasos sanguíneos,

contienen un tipo de ruido similar a un ruido puntual aditivo y sustractivo cuya densidad es mayor en las regiones cercanas a los vasos sanguíneos (Figura 4.10, imágenes b y e).

Las imágenes observadas para este experimento son las imágenes binarias segmentadas en base al algoritmo de morfología matemática difusa antes descrito. Es conveniente mencionar también que, para este experimento, las imágenes binarias con los vasos sanguíneos segmentados son dadas, lo que implica que no se puede modificar el algoritmo que las generó. Por lo tanto, en el contexto general de segmentación de imágenes oculares, el diseño de W-operadores aquí realizado actúa únicamente como un método para el filtrado de ruido o pos-procesamiento de las imágenes segmentadas. Las imágenes ideales son las imágenes binarias con los vasos sanguíneos segmentados manualmente obtenidas de la base de datos pública DRIVE descrita en el Capítulo II, Sección 2.3.2 [Staal *et al.*, 2004]. Para este experimento cada imagen utilizada tiene un tamaño de 565×584 píxeles. Las ventanas utilizadas en este caso tienen un tamaño de 5×5 píxeles. Esto implica que las redes neuronales tipo feed-forward tienen un total de 20 neuronas en la capa oculta y los vectores de entrada tienen una dimensión de 25.

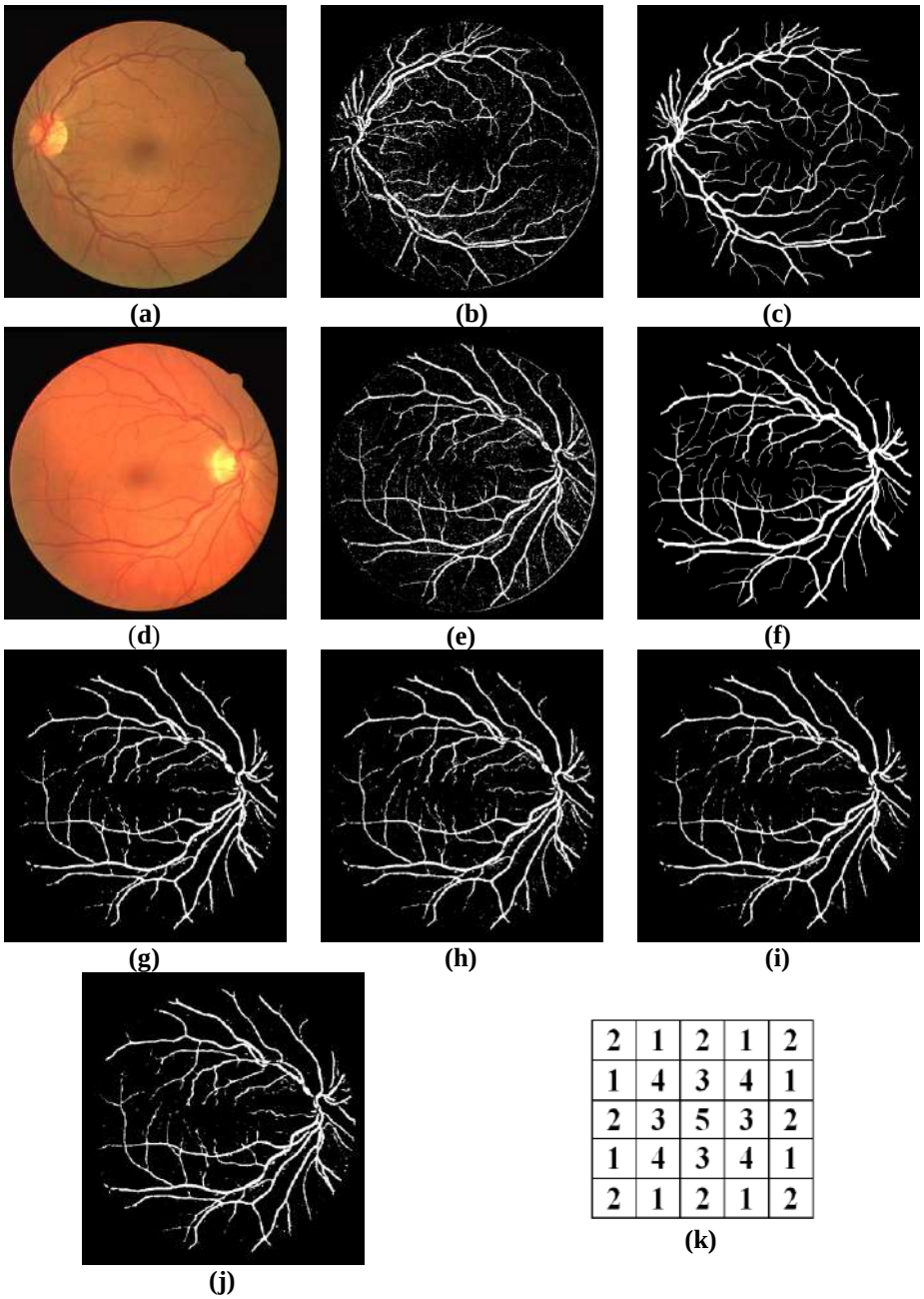


Figura 4.10. Resultados del filtrado de ruido en imágenes oculares. (**a**) Imagen original color de entrenamiento. Par de imágenes de entrenamiento: (**b**) imagen observada, e (**c**) imagen ideal. (**d**) Imagen original color de testeo. (**e**) Imágenes observada e (**f**) ideal de testeo y los resultados en base a: (**g**) red neural tipo feed-forward, (**h**) multi-resolución piramidal, (**i**) *k*NN, y (**j**) clasificador SVM. (**k**) Pirámide de ventanas para multi-resolución.

En la primera parte de la Tabla 4.1 se resumen los resultados obtenidos, tanto antes como después de la aplicación de los W-operadores. De acuerdo con los valores de ER, la reducción del nivel de ruido del método propuesto es del 13%. Para el caso de los W-operadores diseñados con multi-resolución piramidal y *k*NN esta reducción es del 8% y 12%, respectivamente. Los clasificadores diseñados usando SVM producen una reducción del ruido del 12.5%. Según estos resultados, el enfoque propuesto para el diseño automático de operadores morfológicos produce la mayor reducción del nivel ruido en la segmentación de los vasos sanguíneos en comparación con los demás algoritmos testeados. Sin embargo, el nivel de reducción de ruido del enfoque propuesto es comparable con aquellos obtenidos

en base a W-operadores diseñados usando la regla *k*NN y los clasificadores SVM. Esta similitud en los resultados se debe a que estos tres métodos (redes neuronales tipo feed-forward, *k*NN, y SVM) eliminan aproximadamente la misma cantidad de falsos positivos, lo cual se evidencia través de los valores de FPR. La leve superioridad del método propuesto se debe principalmente a una mayor reducción del nivel de falsos negativos (valores de FNR) en comparación con *k*NN y SVM. Multi-resolución piramidal es el método de diseño de W-operadores que disminuye en mayor medida el número de falsos negativos. Sin embargo, esta superioridad se ve contrastada con sólo una leve disminución de la cantidad de falsos positivos.

En la Figura 4.10 se muestra, como ejemplo, las imágenes resultantes de la aplicación de los W-operadores diseñados en base a: redes neuronales tipo feed-forward (imagen g), multi-resolución piramidal (imagen h), *k*NN (imagen i), y SVM (imagen j). Estos resultados se obtuvieron utilizando la pirámide de ventanas de la Figura 4.10-k. Para estos ejemplos, los operadores fueron diseñados en base al par de imágenes de entrenamiento presentado en las imágenes b y c como imágenes observada e ideal, respectivamente. La imagen de testeo es la mostrada en la Figura 4.10-e. Las imágenes mostradas en la Figura 4.10-b y -e se obtuvieron mediante el algoritmo de segmentación basado en morfología matemática difusa aplicado a las imágenes color RGB de la Figura 4.10-a y -d, respectivamente. Las imágenes mostradas como ejemplo de resultado corroboran los resultados numéricos obtenidos.

En la Figura 4.10-b y -e se puede observar que las imágenes obtenidas mediante el algoritmo de segmentación basado en morfología matemática difusa contienen sólo una pequeña cantidad de vasos sanguíneos delgados. Esto es evidente si se comparan estas imágenes con sus respectivas imágenes ideales presentadas en la Figura 4.10-c y -f. En su mayoría, los vasos sanguíneos delgados son eliminados por completo de las imágenes segmentadas. Es esta la razón por la que los valores de FNR en la primera parte de la Tabla 4.1 son bastante altos tanto antes como después la aplicación de todos los operadores diseñados para el filtrado de ruido. Por otra parte, el árbol arterial principal compuesto por los vasos sanguíneos gruesos es segmentado casi en su totalidad; sin embargo, se puede observar que éste se encuentra deteriorado por el ruido generado durante la segmentación.

4.5.5. Detección de Bordes en Imágenes con Ruido

Este experimento consiste en diseñar automáticamente un W-operador que detecte los bordes de los objetos presentes en una imagen con ruido sintético. Para este propósito se utilizó un conjunto de trabajo formado por 4 imágenes de 350×156, 300×270, 637×563, y 401×393 píxeles. Para la obtención de las imágenes observadas para el entrenamiento de los W-operadores, a cada imagen del conjunto de trabajo se le añadió ruido artificial tipo sal y pimienta, con un nivel de densidad de 0.1. Por otra parte, las imágenes ideales (imágenes con los bordes) fueron obtenidas aplicando el gradiente morfológico por erosión $H - (H \ominus B)$ a cada imagen del conjunto de trabajo, donde H es la imagen de la que se desea extraer los bordes, y $H \ominus B$ es la erosión de esta imagen por el elemento estructurante B. En este caso el elemento estructurante utilizado es un cuadrado de 3×3 píxeles.

En este experimento se realizan comparaciones de los resultados obtenidos en base al método propuesto con aquellos obtenidos mediante la aplicación de multi-resolución

piramidal, *k*NN, y clasificadores SVM que son métodos de diseño automático, y el gradiente morfológico que es un método de diseño heurístico de operadores morfológicos. Es importante también mencionar que las ventanas utilizadas en este experimento son las mismas ventanas empleadas para el experimento anterior (5×5 píxeles).

La segunda parte de la Tabla 4.1 resume los resultados obtenidos en este experimento. De la comparación de los valores de ER, se infiere que los W-operadores diseñados con redes neuronales tipo feed-forward mejoran la calidad de los resultados de la multi-resolución piramidal, el gradiente morfológico, *k*NN, y SVM en 28.6%, 93%, 5.5%, y 1.27%, respectivamente. Esta superioridad de las redes neuronales tipo feed-forward se da también al comparar los valores de FNR. Estos valores evidencian que los W-operadores diseñados usando redes neuronales son más robustos a la presencia de ruido tipo sal y pimienta que los operadores diseñados con multi-resolución piramidal y gradiente morfológico. Al comparar los resultados del método propuesto con los W-operadores diseñados usando la regla *k*NN y los clasificadores SVM, se tiene que los resultados obtenidos para todos estos métodos son bastante similares.

La robustez frente al ruido tipo sal y pimienta de las redes neuronales tipo feed-forward se debe, en gran medida, al tipo de función de error utilizada para su entrenamiento. Hay que recordar que el ECM ponderado reduce la influencia que ejercen las configuraciones que contienen ruido sobre el entrenamiento de las redes neuronales (Sección 4.4.2). La robustez de los clasificadores SVM frente al ruido tipo y sal pimienta se debe al uso de las funciones de kernel y a la selección de los vectores de soporte que en su mayoría están libres de ruido. La regla *k*NN disminuye la influencia del ruido en sus predicciones de la probabilidad condicional de la clase 1, $\mathbb{P}(Y = 1|\mathbf{X})$, al calcular un promedio de las probabilidades condicionales de los k vecinos más cercanos a la configuración a clasificar $\mathbf{X}$. Es conveniente también remarcar que, a nivel general, los métodos de diseño automático empleados en este experimento para el problema de detección de bordes son mucho más robustos al ruido tipo sal y pimienta que el método de diseño heurístico (gradiente morfológico).

La Figura 4.11 muestra, a manera de ejemplo, un par de imágenes de entrenamiento, observada (imagen a) e ideal (imagen b), y los resultados obtenidos a partir del procesamiento de la imagen observada de la Figura 4.11-c con W-operadores diseñados usando: redes neuronales (imagen d), multi-resolución piramidal (imagen e), gradiente morfológico (imagen f), *k*NN (imagen g), y SVM (imagen h). La pirámide utilizada en este experimento se muestra en la Figura 4.11-i. En este caso, se puede apreciar visualmente que los resultados de las redes neuronales tipo feed-forward, *k*NN, y SVM contienen mucho menos ruido que los resultados de la multi-resolución piramidal y el gradiente morfológico. En particular, se evidencia visualmente que el gradiente morfológico es un método de detección de bordes muy sensible al ruido tipo sal y pimienta. Se puede observar que la imagen procesada con este método (imagen f) contiene los bordes del objeto de interés así como también los bordes de muchos granos de ruido.

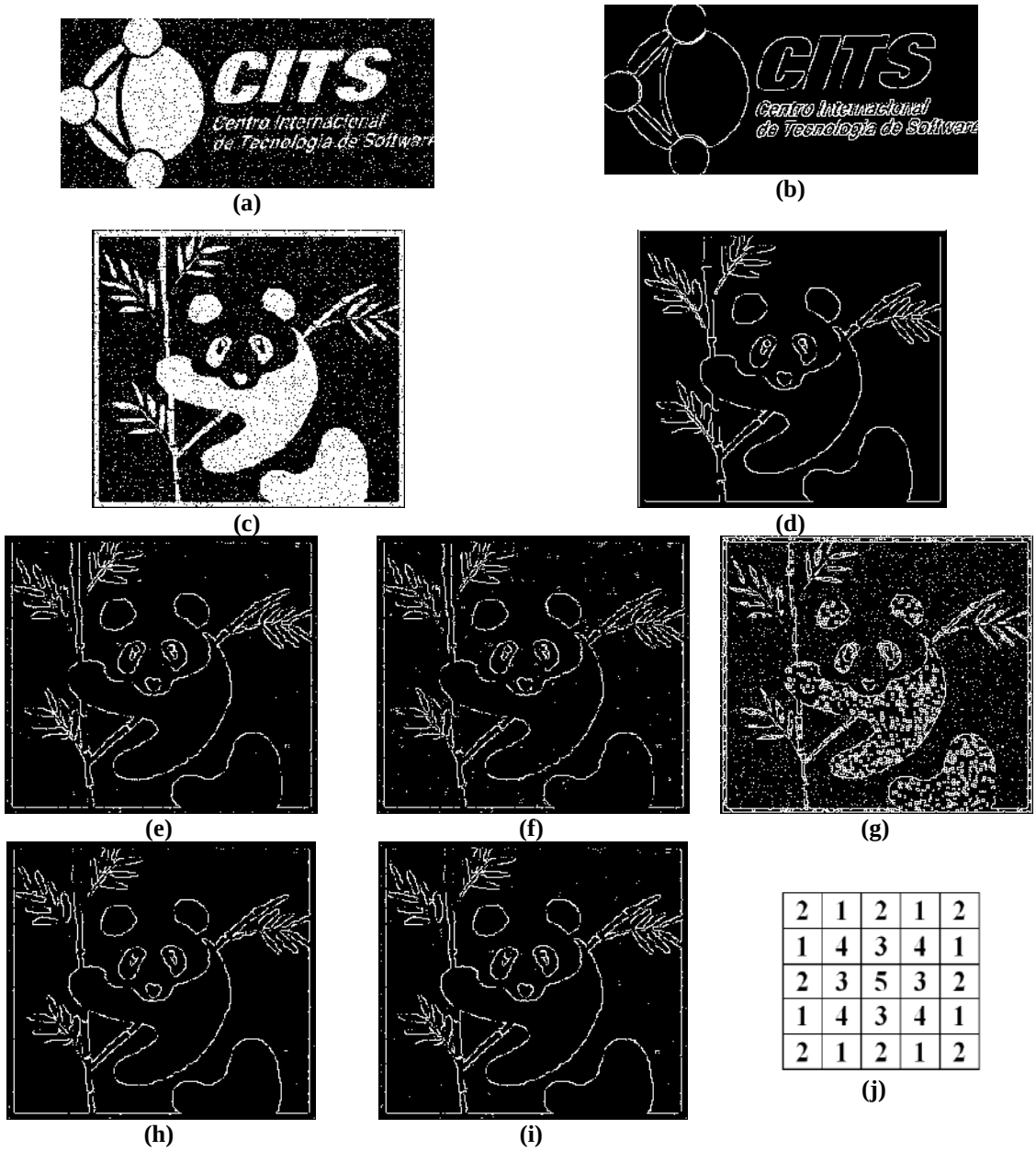
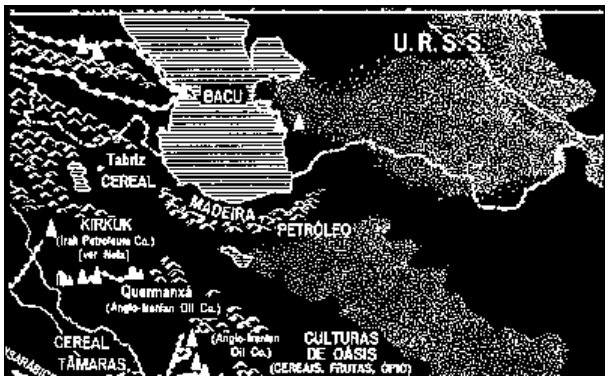


Figura 4.11. Resultados de la detección de bordes en imágenes con ruido. Par de imágenes de entrenamiento: (**a**) imagen observada e (**b**) imagen ideal. (**c**) Imágenes observada e (**d**) ideal de testeo y los resultados obtenidos usando: (**e**) red neuronal tipo feed-forward, (**f**) multi-resolución piramidal, (**g**) gradiente morfológico, (**h**) *k*NN, y (**i**) clasificador SVM. (**j**) Pirámide utiliza para multi-resolución.

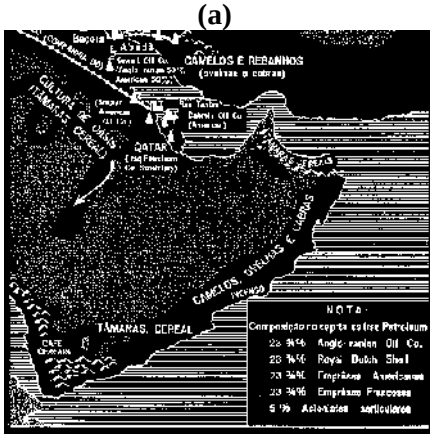
4.5.6. Identificación de Texturas

En este tercer experimento el objetivo es diseñar automáticamente un W-operador que identifique un determinado patrón de textura en imágenes de escaneados de mapas. El conjunto de trabajo está compuesto por 4 pares de imágenes con 250×326, 447×326, 400×250, y 400×395 píxeles. En este caso se utilizan ventanas de mediano tamaño compuestas por 7×7 píxeles. Esto implica que las redes neuronales tipo feed-forward aquí utilizadas están compuestas por 28 neuronas en la capa oculta. La dimensión de los vectores de características o configuraciones de ventana es 49.

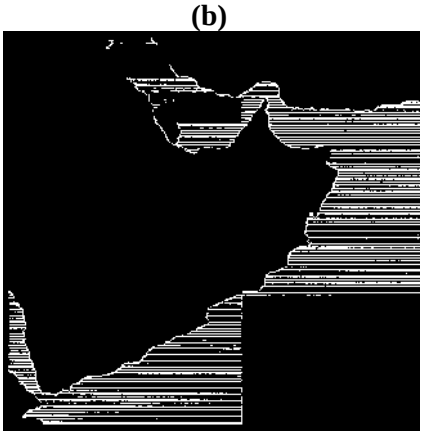


(a)

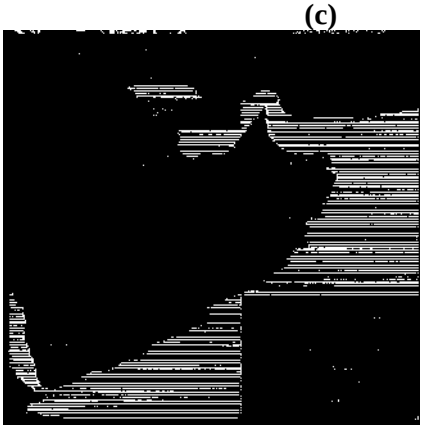
(b)



(c)



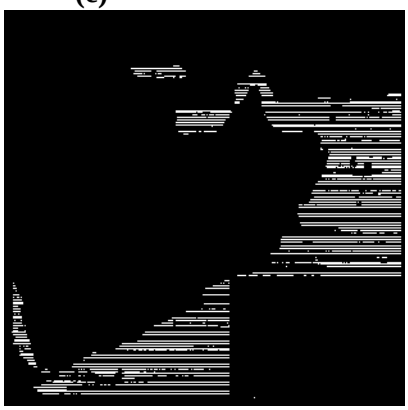
(d)



(e)

(f)

(g)



(h)

2	1	2	1	2	1	2
1	4	3	4	3	4	1
2	3	6	5	6	3	2
1	4	5	7	5	4	1
2	3	6	5	6	3	2
1	4	3	4	3	4	1
2	1	2	1	2	1	2

(i)

Figura 4.12. Resultados obtenidos de la identificación de texturas en imágenes de mapas. Par de imágenes de entrenamiento: (**a**) imagen observada e (**b**) imagen ideal. (**c**) Imágenes observada e (**d**) ideal de testeo y los resultados obtenidos utilizando: (**e**) una red neural tipo feed-forward, (**f**) multi-resolución piramidal, (**g**) *k*NN, y (**h**) un clasificador SVM. (**i**) Pirámide utilizada para la multi-resolución.

La tercera parte de la Tabla 4.1 resume los resultados obtenidos en este experimento. De acuerdo con los valores de ER, los W-operadores diseñados usando las redes neuronales tipo feed-forward mejoran el desempeño de los W-operadores en base a multi-resolución piramidal y *k*NN, y los clasificadores SVM, en 49.5%, 3%, y 45% respectivamente. En este caso los resultados de las redes neuronales son comparables con los resultados de la regla *k*NN tanto a nivel de falsos positivos (valor de FPR) como a nivel de falsos negativos (valor de FNR). Comparando la generalización de las redes neuronales con la de la multi-resolución piramidal y SVM, se puede notar la superioridad de las redes neuronales tipo feed-forward en la detección de texturas.

La Figura 4.12 muestra como ejemplo un par de imágenes de entrenamiento: imagen observada (Figura 4.12-a) e imagen ideal (Figura 4.12-b), y las imágenes resultantes de la aplicación de los operadores morfológicos diseñados automáticamente utilizando: redes neuronales (imagen e), multi-resolución piramidal (imagen f), *k*NN (imagen g), y SVM (imagen h). La imagen observada de testeo se muestra en la Figura 4.12-c y su respectiva imagen ideal se presenta en la Figura 4.12-d. La pirámide de ventanas aquí usada se muestra en la Figura 4.12-i.

El nivel de ruido presente en la imagen resultante de la multi-resolución piramidal y sus altos valores de ER, FPR, y FNR evidencian que el costo de diseño de este método para los tamaños de ventana aquí considerados es alto. Esto significa que el procesamiento de las imágenes se realiza, en su mayoría, en base a las probabilidades condicionales estimadas con las ventanas de bajo tamaño que se encuentran en la cima de la pirámide utilizada. La imagen con el menor nivel de ruido es la obtenida con SVM. Sin embargo, en este caso existen también regiones de textura que no son identificadas, lo cual se manifiesta en su alto valor de FNR. Visualmente, se puede notar que las identificaciones de textura realizadas por *k*NN y las redes neuronales tipo feed-forward son similares, aunque *k*NN introduce un mayor nivel de ruido en la imagen resultante en comparación con las redes neuronales.

4.5.7. Reconocimiento de Caracteres

Este experimento consiste en el diseño automático de W-operadores para el reconocimiento automático de la letra "a" en imágenes con escaneados de texto en el idioma portugués. Aquí los operadores de imágenes a ser diseñados en base a redes neuronales tipo feed-forward, multiresolution piramidal, *k*NN, y SVM deben actuar como un marcador de la letra "a". Esto es, grupos de al menos 25 píxeles espacialmente conectados de cada letra "a" presente en la imagen de entrada O' deben aparecer en la imagen resultante del procesamiento $\mathbf{\Psi}(O')$. A partir de estas marcas o grupos de píxeles, y mediante un proceso de pos-procesamiento, se puede realizar una reconstrucción de todas las letras detectadas.

El pos-procesamiento aplicado en este experimento consiste de tres pasos. Primero, se filtra el ruido de las imágenes con las marcas detectadas. En este paso se utiliza un operador de imagen binario γ que elimina o filtra todos los objetos de la imagen $\mathbf{\Psi}(O')$ que contengan un tamaño menor que 25 píxeles, obteniéndose así la imagen binarias $\gamma(\mathbf{\Psi}(O'))$. Segundo, se eliminan todas las marcas detectadas de la imagen $\gamma(\mathbf{\Psi}(O'))$ que no pertenecen a ninguna letra en O' obteniéndose así la imagen $\Omega(\gamma(\mathbf{\Psi}(O')),O')$, donde $\Omega(O,O')(t) = min(O(t),O'(t))$ es un operador binario píxel a píxel. Tercero, se aplica un procedimiento de reconstrucción

morfológica [Vincent, 1993]. Para esta reconstrucción la imagen binaria $\Omega(\gamma(\mathbf{\Psi}(O')),O')$ que contiene las marcas de las letras "a" detectadas en el primer y segundo pasos actúa como imagen marcador y la imagen original O' actúa como máscara.

En este experimento también se testean las redes neuronales convolucionales. Para el entrenamiento del clasificador CNN se utilizan los pares (H,U), donde H es una imagen binaria de 32×32 píxeles que contiene cada letra a ser clasificada, y U es la etiqueta de H que es igual a +1 si H contiene la letra "a" o -1 en el caso contrario. El tamaño de 32×32 píxeles fue seleccionado de tal modo que H contenga completamente a cada letra a ser clasificada sin solaparse o contener porciones de los caracteres vecinos en la imagen original O'.

Los clasificadores CNN aquí utilizados están compuestos por 8 capas, según lo sugerido por sus inventores en [LeCun y Bengio, 1995]. La capa 0, que es la primera capa de la red, no realiza ningún cálculo. En la segunda capa se realiza la convolución de la imagen de entrada con 6 máscaras diferentes de 5×5 píxeles cada una. En la tercera capa se realiza la operación de pooling que consiste en un submuestreo, con un factor de 2, de las imágenes convolucionadas de la segunda capa. La cuarta capa ejecuta una nueva convolución de las imágenes resultantes de la tercera capa usando 16 máscaras de 5×5 píxeles. La quinta capa realiza un submuestreo de las imágenes de la cuarta capa con un factor de 2. La sexta capa es de convolución usando 120 máscaras de 5×5 píxeles. La séptima y la octava capas forman un clasificador no lineal que es una red neuronal tipo feed-forward. Este clasificador tiene como entrada un vector de 120 características. La séptima capa está formada por 84 neuronas y la octava capa está formada por una única neurona. Todas las neuronas de estas dos últimas capas están formadas por funciones de transferencia *tansig*.

En función de lo anterior se puede notar que las 6 primeras capas del clasificador CNN utilizado en este experimento actúan como extractores de características; mientras que las 2 últimas capas realizan la clasificación de los vectores conteniendo dichas características. El ajuste, tanto de los coeficientes de las máscaras de convolución como de los pesos de la red neuronal tipo feed-forward en el clasificador CNN, se realiza utilizando el método de optimización de Levenberg-Marquardt combinado con el método de retro-propagación del error. En base a estos métodos se minimiza el error cuadrático medio entre la respuesta del clasificador CNN y la etiqueta verdadera de la imagen a clasificar.

El conjunto de datos utilizados para este experimento se compone de 4 pares de imágenes, cuyos tamaños son: 1884×1121 píxeles con 903 caracteres, 1878×722 píxeles con 670 caracteres, 1884×957 píxeles con 952 caracteres, y 1886×722 píxeles con 658 caracteres. Para este experimento se utilizan ventanas ralas que cubren una área de 11×11 píxeles con un total de 49 píxeles observados. Por lo tanto, el tamaño, o dimensión, de los vectores de características para el diseño de W-operadores con redes neuronales tipo-feed forward es 49. Esto a su vez implica que la redes neuronales tipo feed-forward están compuestas por 28 neuronas en la capa oculta según la regla heurística definida en la Sección 4.5.2.

En la última parte de la Tabla 4.1 se resumen los resultados obtenidos en este experimento. En este caso los errores no se calculan píxel a píxel, sino caracter a caracter. Para cada

algoritmo testeado, el ER es el cociente entre el número de caracteres incorrectamente clasificados como “a” sobre el total de caracteres clasificados. El FPR es el cociente entre el número de caracteres que, no siendo “a”, son clasificados como “a” sobre el número total de caracteres que no son “a”. El FNR es el cociente entre el número de caracteres que, siendo “a”, no son clasificados como “a” sobre el total de caracteres que son “a”.

De acuerdo con los valores de error de la Tabla 4.1, el reconocimiento de la letra “a” realizado por los W-operadores diseñados usando las redes neuronales tipo feed-forward es superior al de multi-resolución, *k*NN, SVM, y CNN en 81.3%, 66.7%, 57.1%, y 90%. Es importante también mencionar que el desempeño de todos los métodos testeados es bueno, pues sus errores (valores de ER) son menores a un 0.5%. Por otra parte, a partir de los valores de FPR y FNR, se puede notar que los W-operadores basados en redes neuronales tipo feed-forward fallan tanto a nivel de falsos positivos como a nivel de falsos negativos. Los W-operadores basados en multi-resolución piramidal y *k*NN, y los clasificadores SVM detectan todas las letras “a”, razón por la cual sus valores de FNR son nulos. Adicionalmente, estos métodos también detectan algunos falsos positivos; es decir, caracteres distintos a la letra “a”. El número de falsos positivos de estos métodos es mayor que para el caso de las redes neuronales tipo feed-forward.

Los clasificadores CNN no sólo no reconocen algunas letras “a”, sino que también reconocen erróneamente algunas caracteres que no son “a” en mayor proporción que el enfoque basado en el uso de W-operadores. Es interesante analizar este resultado teniendo en cuenta que los clasificadores CNN utilizan también una red neuronal tipo feed-forward que es más grande que la red neuronal empleada para los W-operadores. Esto implicaría a su vez una mejor capacidad de aproximación, y por ende mejores resultados, considerando que las dos redes fueron entrenadas con el mismo método de optimización. Sin embargo, hay que tener en cuenta el tamaño de los espacios de búsqueda para los dos tipos de clasificadores y la cantidad de datos disponibles para su entrenamiento. Son precisamente estos dos factores los que podrían ser los responsables del mejor desempeño de los W-operadores en relación a los clasificadores CNN.

Para dar soporte a la anterior conjetura hay que considerar el siguiente análisis. La red neuronal del clasificador CNN utiliza vectores de 120 variables obtenidos automáticamente a partir de la convolución y pooling de las imágenes a clasificar formadas por 32×32 píxeles. Esto implica que los clasificadores CNN tienen como dominio un espacio de características formado por 1024 variables. Para los W-operadores, los vectores de características están formados por los valores de los píxeles observados a través de una ventana de 7×7 píxeles, resultando en vectores de 49 variables. En función de esta comparación, los clasificadores CNN tienen mayor capacidad de aproximación al clasificador óptimo que los W-operadores, o lo que es lo mismo, un menor costo de restricción. Sin embargo, la cantidad de ejemplos de entrenamiento para los clasificadores CNN es mucho menor que para los W-operadores. Esto debido a que los primeros operan por caracteres y los W-operadores operan píxel a píxel. Esto implica que los W-operadores disponen de más información para realizar el ajuste de sus pesos sobre un espacio de búsqueda de menor tamaño que el de los clasificadores CNN. Esto significa un menor costo de diseño de los W-operadores frente a los clasificadores CNN. Esto se traduciría finalmente en un mejor desempeño de los W-operadores frente a los clasificadores CNN.

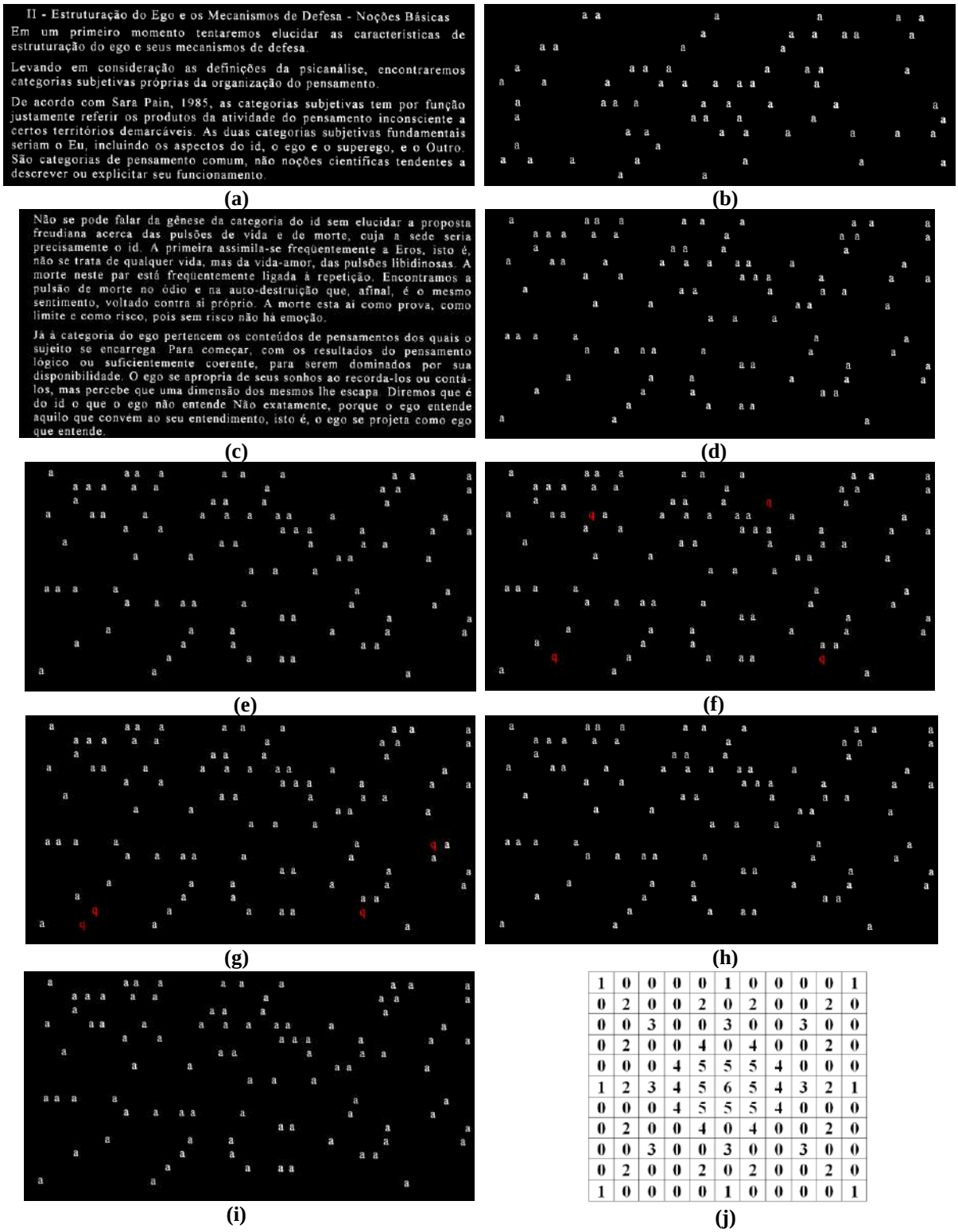


Figura 4.13. Resultados obtenidos en el reconocimiento automático de la letra "a" en imágenes de texto. Par de imágenes de entrenamiento: (**a**) imagen original e (**b**) imagen ideal. (**c**) Imágenes observada e (**d**) ideal de testeo y los resultados obtenidos utilizando: (**e**) una red neuronal tipo feed-forward, (**f**) multi-resolución piramidal, (**g**) regla *k*NN, (**h**) un clasificador SVM, y (**i**) un clasificador CNN. (**j**) Pirámide utiliza para la multi-resolución.

En la Figura 4.13 se muestra, como ejemplo, un par de imágenes de entrenamiento, observada (imagen a) e ideal (imagen b), y las imágenes resultantes de la aplicación de W-operadores diseñados usando redes neuronales (imagen e), multi-resolución piramidal

(imagen f), *k*NN (imagen g), SVM (imagen h), y CNN (imagen i). La imágenes observada e ideal de testeo se muestran en la Figura 4.13-c y -d, respectivamente. En la Figura 4.13-i se muestra la pirámide utilizada para el diseño de los W-operadores.

Tabla 4.1. Resultados de la aplicación W-operadores binarios diseñados automáticamente utilizando redes neuronales tipo feed-forward, multi-resolución piramidal, *k*NN y clasificadores SVM y CNN aplicados al filtrado de ruido en imágenes oculares, detección de bordes en imágenes con ruido, identificación de texturas, y reconocimiento de caracteres (letra "a").

	ER [%]	**FPR [%]**	**FNR [%]**
Filtrado de Ruido en Imágenes Oculares			
Valores Originales (algoritmo de segmentación)	4.53	1.89	32.78
Redes neuronales tipo feed-forward	*3.94*	*1.50*	*28.86*
Multi-resolución piramidal	4.15	1.85	27.66
*k*NN	3.97	1.51	30.07
SVM	3.96	1.50	29.37
Detección de Bordes en Imágenes con Ruido			
Redes neuronales tipo feed-forward	*1.55*	*0.46*	*17.49*
Multi-resolución piramidal	2.17	0.60	24.33
Gradiente morfológico	22.36	18.37	94.83
*k*NN	1.64	0.41	19.11
SVM	1.57	0.43	17.90
Identificación de Texturas			
Redes neuronales tipo feed-forward	*1.59*	*0.56*	*12.66*
Multi-resolución piramidal	3.15	1.74	18.78
*k*NN	1.64	0.86	10.09
SVM	2.89	0.06	35.63
Reconocimiento de Caracteres (letra "a")			
Redes neuronales tipo feed-forward	*0.03*	*0.01*	*0.19*
Multi-resolución piramidal	0.16	0.18	0
*k*NN	0.09	0.11	0
SVM	0.07	0.09	0
CNN	0.31	0.56	0.28

En la Figura 4.14 se muestran algunos ejemplos de las curvas de entrenamiento de las redes neuronales tipo feed-forward utilizadas para el diseño de W-operadores en cada uno de los experimentos antes descritos. Para mejorar la visualización de la variación de los valores de error de entrenamiento y error de validación se utiliza una escala logarítmica para el eje *y*. El eje *x* está en escala lineal. El criterio utilizado para detener el entrenamiento de las redes en cada experimento está basado en un máximo de 50 épocas ó cuando no existe una variación del error de validación durante 5 épocas consecutivas. El error de validación fue calculado sobre un 30% de los datos de entrenamiento. El 70% restante se utilizó para ajustar los pesos de la red neuronal y para calcular el error de entrenamiento.

En todos los gráficos se evidencia que el valor del error de entrenamiento siempre disminuye. En los casos correspondientes a los gráficos de las imágenes a, b, y c se evidencia que el entrenamiento de la red finaliza debido que no existe un cambio en el error de validación durante 5 épocas consecutivas. En el caso del gráfico de la imagen d, el entrenamiento finaliza porque se alcanza el máximo número de épocas permitido (50 épocas). Esto muestra que, cuando se entrena una red neuronal en base al error de validación, es importante la selección de un número máximo de épocas para evitar que este procedimiento se prolongue por un largo período de tiempo en el caso donde la curva U tenga un amplio valle (Figura 4.2).

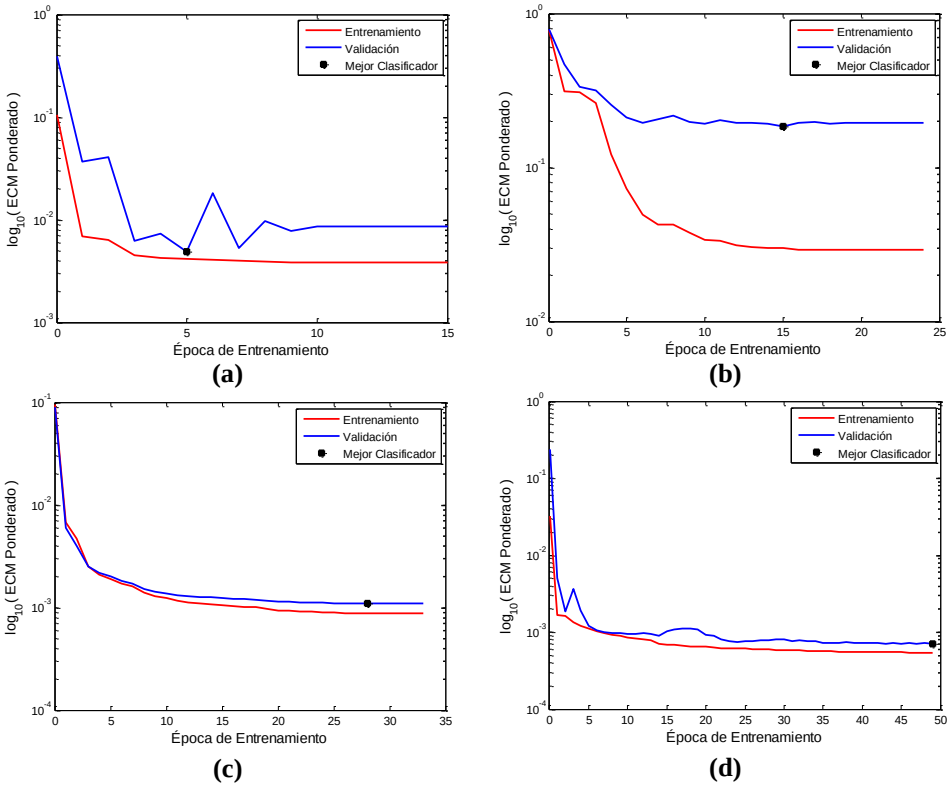


Figura 4.14. Curvas de entrenamiento y validación de las redes neuronales tipo feed-forward. Redes entrenadas para: (**a**) filtrado de ruido en imágenes oculares. (**b**) detección de bordes en imágenes con ruido. (**c**) Identificación de texturas. (**d**) Reconocimiento de caracteres.

En la Tabla 4.2 se muestran los datos referentes al costo computacional de cada uno de los métodos testeados. En la segunda columna de esta tabla se muestran los tiempos promedio de entrenamiento más testeo por cada imagen utilizada en cada experimento. En la tercera columna se presenta el número promedio de tuplas ($\mathbf{X}$,*freq*($\mathbf{X}$,Y = 0),*freq*($\mathbf{X}$,Y = 1)), o número promedio N de ejemplos de entrenamiento utilizado para el diseño automático de los W-operadores por cada etapa de la validación cruzada. En la cuarta columna se presenta el número promedio de píxeles escaneados por cada imagen de entrenamiento. La quinta columna contiene el número de parámetros (pesos y umbrales) ajustados en las redes neuronales tipo feed-forward de los W-operadores diseñados en cada iteración de la validación cruzada. Todos los experimentos fueron realizados usando un computador con procesador Intel core i-5, con velocidad de CPU de 2.30 GHz, y 8 GB de memoria RAM. Los algoritmos para cada experimento fueron implementados en Matlab®.

A nivel de tiempo de entrenamiento más tiempo de procesamiento, tanto el enfoque propuesto en este capítulo como la multi-resolución piramidal presentan mejor desempeño comparado con *k*NN y SVM. Para el caso de los problemas de identificación de texturas y reconocimiento de caracteres, los tiempos empleados por *k*NN y SVM son extremadamente grandes en comparación con los tiempos de los demás métodos testeados. Si bien a nivel de error *k*NN y SVM permiten obtener resultados que son comparables con el enfoque propuesto en esta tesis, a nivel computacional estos métodos implican un alto costo.

El alto costo computacional de *k*NN se debe al cálculo de las distancias de Hamming y el posterior ordenamiento de dichas distancias para encontrar los k vecinos más cercanos a cada configuración a clasificar. Esto sucede a pesar de que se ha utilizado un método de búsqueda optimizado para encontrar los k vecinos más cercanos a cada configuración a

clasificar [Friedman *et al.*, 1977]. Los valores de *k* para los experimentos realizados son 6 para los tres primeros experimentos y 7 para el experimento de reconocimiento de caracteres. Para el caso de SVM, el alto costo computacional se debe a que la búsqueda de los vectores de soporte y el ajuste de parámetros involucra la resolución de un problema de programación cuadrática. El costo computacional de la programación cuadrática es muy alto cuando el problema de clasificación comprende a cantidades de datos entrenamiento de un orden igual a o mayor que las decenas de miles.

Tabla 4.2. Datos del costo computacional de entrenamiento y diseño de W-operadores binarios utilizando redes neuronales tipo feed-forward, multi-resolución piramidal, *k*NN, y clasificadores SVM y CNN aplicados al filtrado de ruido en imágenes oculares, detección de bordes en imágenes con ruido, identificación de texturas, y reconocimiento de caracteres (letra "a").

	Tiempo promedio por imagen [h]	**Número promedio de ejemplos de entrenamiento, *N***	**Número promedio de píxeles escaneados por imagen**	**Número de parámetros de las redes neuronales tipo feed-forward**
Filtrado de Ruido en Imágenes Oculares				
Redes neuronales tipo feed-forward	*0.013*	23.6K	325.4K	541
Multi-resolución piramidal	0.027			----------
*k*NN	1.001			----------
SVM	1.071			----------
Detección de Bordes en Imágenes con Ruido				
Redes neuronales tipo feed-forward	*0.014*	47.6K	159.9K	541
Multi-resolución piramidal	0.011			----------
Gradiente morfológico	0.01			----------
*k*NN	1.168			----------
SVM	0.379			----------
Identificación de Texturas				
Redes neuronales tipo feed-forward	*0.137*	57.6K	117.2K	1.4K
Multi-resolución piramidal	0.048			----------
*k*NN	4.635			----------
SVM	1.863			----------
Reconocimiento de Caracteres (letra "a")				
Redes neuronales tipo feed-forward	*0.684*	85.6K	1641.6K	1.4K
Multi-resolución piramidal	0.829			----------
*k*NN	4.035			----------
SVM	4.143			----------
CNN	0.5			----------

El costo de representación computacional de los W-operadores diseñados en base a las redes neuronales tipo feed-forward es mucho menor que el costo de representación de los W-operadores diseñados usando multi-resolución piramidal y *k*NN. Para estos métodos la representación computacional involucra almacenar tablas de búsqueda con un número de configuraciones igual a los valores de la tercera columna de la Tabla 4.2. Para el caso de los W-operadores diseñados usando las redes neuronales tipo feed-forward sólo es necesario almacenar una cantidad de números reales, o de punto flotante, igual al número de cada fila de la quinta columna de la Tabla 4.2.

4.6. Anotaciones Finales

En este capítulo se ha propuesto un nuevo enfoque para el diseño automático de operadores de ventana, o W-operadores, binarios. La propuesta consiste de dos etapas. Primero se realiza una estimación de las probabilidades condicionales de clase, usando la regla plug-in,

para las configuraciones observadas en el escaneo de las imágenes de entrenamiento. Segundo, en base a estos estimados se diseña un clasificador binario utilizando una red neuronal tipo feed-forward de tres capas: entrada, oculta, y salida. Las funciones de transferencia utilizadas para la red neuronal son la función *tansig* para las neuronas de la capa oculta y la función lineal con pendiente de 45º para la única neurona de la capa salida. Se utilizaron redes neuronales tipo feed-forward porque permiten implementar fronteras de decisión no lineales y porque son aproximadores universales.

Para el entrenamiento de las redes neuronales feed-forward se ha propuesto el uso del error cuadrático medio ponderado. Este error permite que los clasificadores diseñados tengan cierta robustez ante el ruido durante el entrenamiento y el testeo. Adicionalmente, esta función de error ayuda a prevenir el sobre-entrenamiento de las redes neuronales durante el proceso de optimización para el ajuste de sus parámetros. El método de optimización seleccionado para el entrenamiento de las redes neuronales tipo feed-forward es el método de Levenberg-Marquardt combinado con el método de retro-propagación de error para el cálculo de la matriz jacobiana de la función de error. Esta matriz contiene todos los valores de las derivadas parciales de la función de error con respecto a todos los parámetros de la red neuronal.

El método propuesto permite representar matemáticamente la función característica de un W-operador mediante una red neuronal tipo feed-forward. Esta red neuronal realiza la tarea de clasificación de las configuraciones de ventana. La representación computacional de los W-operadores diseñados se realiza a través del almacenamiento de las matrices y vectores que contienen los parámetros ajustados (pesos y umbrales) de la red neuronal, tanto para la capa oculta como para la capa de salida. Esta es una forma de representación computacional compacta de los W-operadores. Gracias a esto se evita el almacenamiento de gran cantidad de datos (configuraciones de ventana y etiquetas de clase) necesarios para la representación computacional de la función característica en el caso de multi-resolución piramidal y *k*NN. Se ha evidenciado también que el entrenamiento de las redes neuronales es un proceso menos complejo, desde el punto de vista computacional, en comparación con el caso de los SVMs, especialmente para conjuntos de entrenamiento que superan las decenas de miles de configuraciones de ventana.

El método propuesto ha sido utilizado como un método de pos-procesamiento en la segmentación de imágenes médicas, el mismo que consiste en el filtrado de ruido en las segmentaciones de los vasos sanguíneos de imágenes oculares. Estas imágenes fueron segmentadas automáticamente en base a un algoritmo diseñado heurísticamente utilizando morfología matemática difusa. También se consideran los casos de detección de bordes en imágenes contaminadas con ruido tipo sal y pimienta con una densidad de 0.1, identificación de texturas en escaneados de mapas, y reconocimiento de la letra “a” en imágenes de texto. Las ventanas utilizadas en todos estos experimentos tienen tamaños que van entre 5×5 a 11×11 píxeles. Las ventanas de 11×11 píxeles son ventanas ralas con una cantidad efectiva de 49 puntos. En todos los casos testeados, el método propuesto permite obtener mejores resultados en términos del error de clasificación. En algunos casos, los resultados de los demás métodos testeados son comparables a los resultados obtenidos con el método propuesto.

En la misma línea del presente capítulo, en el capítulo V se presenta un nuevo método para el diseño automático de W-operadores para procesamiento de imágenes en escala de grises. Bajo el paradigma de reconocimiento de patrones, se considera el problema de diseño de

clasificadores para imágenes en escala de grises. Se presentan también casos de aplicación de la metodología desarrollada para la segmentación de imágenes médicas.

4.7. Referencias

[1] Y.S. Abu-Mostafa, M. Magdon-Ismail and H.T. Lin, *Learning from data*, AMLBook, Pasadena CA, 2012.

[2] M.E. Benalcázar, M. Brun and V.L. Ballarin, "Artificial neural networks applied to statistical design of window operators," *Pattern Recognition Letters*, vol. 34, pp. 970–9, 2013.

[3] M.E. Benalcázar, M. Brun, V.L. Ballarin, I. Passoni, G. Meschino and L. Pra, "Automatic Design of Binary W-Operators Using Artificial Feed-Forward Neural Networks Based on the Weighted Mean Square Error Cost Function," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. vol. 7441, L. Alvarez, M. Mejail, L. Gomez, and J. Jacobo, Eds.: Springer Berlin Heidelberg, 2012, pp. 495-502.

[4] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Cambridge, 2005.

[5] A. Bouchet, M. Brun and V. Ballarin, "Morfología Matemática Difusa aplicada a la segmentación de angioagrafías retinales," *Revista Argentina de Bioingeniería*, vol. 16, pp. 7-10, 2010.

[6] C.J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-67, 1998.

[7] Z. Cataltepe, Y.S. Abu-Mostafa and M. Magdon-Ismail, "No free lunch for early stopping," *Neural Computation*, vol. 11, pp. 995-1009, 1999.

[8] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, Cambridge, 2000.

[9] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification*, 2001.

[10] M.M. Fraz, P. Remagnino, A. Hoppe, B. Uyyanonvara, A.R. Rudnicka, C.G. Owen and S.A. Barman, "Blood vessel segmentation methodologies in retinal images – A survey," *Computer Methods and Programs in Biomedicine*, vol. 108, pp. 407-33, 2012.

[11] J.H. Friedman, J. Bentely and R.A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Transactions on Mathematical Software*, vol. 3, pp. 209-26, 1977.

[12] M. Hagan, H. Demuth and M. Beale, *Neural Network Design*, PWS Publishing, Boston MA, 1996.

[13] M.T. Hagan and M. Menhaj, "Training feed-forward networks with the Marquardt algorithm," *EEE Transactions on Neural Networks*, vol. 5, pp. 989–93, 1994.

[14] Y. LeCun, "A theoretical framework for Back-Propagation," *Proc. 1988 Connectionist Models Summer School*, Pittsburgh PA, 21-8, 1988.

[15] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," in *The Handbook of Brain Theory and Neural Networks*, M.A. Arbib, Ed.: MIT Press, 1995.

[16] Y. LeCun, L. Bottou, G.B. Orr and K.R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade*. vol. 1524, G.B. Orr and K.R. Müller, Eds.: Springer Berlin Heidelberg, 1998, pp. 9-50.

[17] K. Levenberg, "A Method for the Solution of Certain Problems in Least Squares," *The Quarterly of Applied Mathematics*, vol. 2, pp. 164-8, 1944.

[18] D.T. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *SIAM Journal on Applied Mathematics*, vol. 11, pp. 431-41, 1963.

[19] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. vol. 1, D.E. Rumelhart and J.L. McClelland, Eds. Cambridge MA: MIT Press, 1986.

[20] Y. Sai, R. Jinxia and L. Zhongxia, "Learning of Neural Networks Based on Weighted Mean Squares Error Function," *Proc. Computational Intelligence and Design, 2009. ISCID '09. Second International Symposium on*, 241-4 (2009).

[21] B. Scholkopf and A.J. Smola, *Learning with Kernels*, MIT Press, Cambridge MA, 2002.

[22] J. Staal, M.D. Abràmoff, M. Niemeijer, M.A. Viergever and B. Ginneken, "Ridge-Based Vessel Segmentation in Color Images of the Retina," *IEEE Transactions On Medical Imaging*, vol. 23, pp. 501-9, 2004.

[23] V.N. Vapnik, *The Nature of Statistical Learning*, Springer, New York, 2000.

[24] L. Vincent, "Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms," *IEEE Transactions on Image Processing* vol. 2, pp. 176-201, 1993.

[25] T.P. Vogl, J.K. Mangis, A.K. Rigler, W.T. Zink and D.L. Alkon, "Accelerating the convergence of the back-propagation method," *Biological Cybernetics*, vol. 59, pp. 257-63, 1988.

CAPÍTULO V

Método Propuesto para Diseño de W-operadores en Escala de Grises

Resumen: En este capítulo se propone un nuevo método para el diseño automático de W-operadores para procesamiento de imágenes en escala de grises. Bajo el paradigma de reconocimiento de patrones, este problema se formula como un problema de diseño de clasificadores multiclase. En este contexto, se plantea el diseño de W-operadores utilizando modelos de clasificación basados en regresión logística y redes neuronales tipo feed-forward de tres capas (entrada, oculta, y salida). Se analiza y resuelve el problema práctico de desbalanceo de frecuencias. Este problema ocasiona que un clasificador tenga una muy baja, o incluso nula, capacidad de discriminación de las clases minoritarias frente a la clase o clases de mayor frecuencia. Para problemas de diseño de W-operadores que involucran gran cantidad de imágenes de entrenamiento, se plantea y analiza el diseño de ensambles de clasificadores. Esta solución se propone para reducir la complejidad computacional del diseño de clasificadores complejos utilizando un único conjunto de entrenamiento de gran tamaño con respecto a la cantidad de memoria RAM disponible. El diseño y aplicación de los clasificadores paramétricos propuestos es analizada como un diseño automático de un banco de filtros espaciales y transformaciones píxel a píxel. Las máscaras de los filtros son construidas a partir de los parámetros de cada clasificador. También se presenta un análisis que permite la visualización de los patrones de imágenes que maximizan la respuesta de un clasificador dado un problema de PDI. Además se proponen dos métodos de preprocesamiento para los vectores de características previo al diseño de los clasificadores. En base a la teoría desarrollada se proponen nuevos métodos para la solución de tres problemas reales de segmentación de imágenes médicas: segmentación de vasos sanguíneos y exudados en imágenes oculares, y segmentación de la glándula prostática en resonancias magnéticas. En todos estos casos se analizan y se hacen comparaciones de los resultados obtenidos con los resultados de otros métodos propuestos en la literatura científica para la solución de estos problemas.

5.1. Introducción

En este capítulo se propone un nuevo método de diseño automático de W-operadores para procesamiento de imágenes en escala de grises. En este caso tanto la entrada, o imagen observada, como la salida, o imagen ideal, son imágenes en escala de grises. Sean $L_1^{\boldsymbol{E}}$ y $L_2^{\boldsymbol{E}}$ los espacios que contienen a todas las posibles imágenes observadas y todas las posibles imágenes ideales, respectivamente, con $L_1 = \{lmin_1,\ldots,lmax_1\}$, $L_2 = \{lmin_2,\ldots,lmax_2\}$, y $\boldsymbol{E}\subset\mathbb{Z}^2$. Para el diseño automático de W-operadores en escala de grises se considera un par de procesos estocásticos conjuntamente estacionarios ($\mathbf{O}$,$\mathbf{I}$), donde el proceso $\mathbf{O}$ genera imágenes observadas y el proceso $\mathbf{I}$ genera imágenes ideales. Dado un par de procesos ($\mathbf{O}$,$\mathbf{I}$), el objetivo del diseño automático es encontrar un W-operador $\boldsymbol{\Psi}$: $L_1^{\boldsymbol{E}} \rightarrow L_2^{\boldsymbol{E}}$ tal que la imagen resultante $\boldsymbol{\Psi}(\mathbf{O})$ del procesamiento sea una buena estimación de la imagen ideal $\mathbf{I}$ en términos del error cuadrático medio $\mathbb{E}[(\boldsymbol{\Psi}(\mathbf{O})(t) - \mathbf{I}(t))^2]$ en un píxel arbitrario $t\in\boldsymbol{E}$.

Sea una ventana $W = \{w_1,\ldots,(0,0),\ldots,w_n\}\subset\mathbb{Z}^2$ formada por $n = |W|$ elementos. Un operador de ventana, o W-operador, $\boldsymbol{\Psi}$ para imágenes en escala de grises se define y representa mediante una función característica ψ: $L_1^{W} \rightarrow L_2$ que asigna un nivel de gris del conjunto L_2 a una configuración, observada a través de la ventana W, de la imagen $\mathbf{O}$ trasladada espacialmente por un punto arbitrario $t\in\boldsymbol{E}$: $\boldsymbol{\Psi}(\mathbf{O})(t) = \psi(\mathbf{O}_t(W))$. Se asume que el centro de la ventana W es el punto (0,0). Si se define el vector de variables aleatorias $\mathbf{X} = (X_1,\ldots,X_n)$, donde $X_i = \mathbf{O}_t(w_i)$, con $w_i\in W$ e $i = 1,\ldots,n$, y la variable aleatoria $Y = \mathbf{I}(t)$, entonces $\psi(\mathbf{X})$ es un estimador de Y. El conjunto de todos los posibles vectores $\mathbf{X}$ que se pueden observar en

O a través de W se denota mediante $\mathcal{X} = L_1^n$; mientras que $\mathcal{Y} = L_2$ es el conjunto de todos los posibles niveles de gris que se pueden observar en **I**. En base a lo anterior, el diseño automático de un W-operador en escala de grises consiste en encontrar una función característica ψ que minimice el error cuadrático medio $\mathbb{E}[(\psi(\mathbf{X}) - Y)^2]$. Aquí, la esperanza se calcula sobre la distribución conjunta $\Pr(\mathbf{X},Y)$ del espacio $\{\mathcal{X}\times\mathcal{Y}\}$.

Para el caso donde cada píxel perteneciente a la imagen ideal **I** toma una etiqueta del conjunto $\mathcal{Y} = \{0,1,\ldots,c - 1\}$, con $c\in\mathbb{Z}^+$, la función ψ: $\mathcal{X} \rightarrow \mathcal{Y}$ es un clasificador multiclase. Este clasificador asigna una etiqueta del conjunto $\mathcal{Y}$ a un vector $\mathbf{X}\in\mathcal{X}$. En este contexto, el diseño de un W-operador consiste en encontrar un clasificador multiclase ψ que minimice la probabilidad $\mathbb{P}(\psi(\mathbf{X}) \neq Y)$ de clasificar erróneamente una configuración de ventana $\mathbf{X}\in\mathcal{X}$ observada con la etiqueta Y. Para el caso particular donde ψ es un clasificador binario se cumple que $\mathbb{E}[(\psi(\mathbf{X}) - Y)^2] = \mathbb{P}(\psi(\mathbf{X}) \neq Y)$. En este capítulo se restringe el análisis para el caso donde un W-operador actúa como clasificador.

5.2. W-operadores en Escala de Grises y Regresión Logística

En esta sección se restringe el análisis al caso donde las imágenes observadas son imágenes en escala de grises, del conjunto $L_1^{\boldsymbol{E}}$, y las imágenes ideales son imágenes binarias, del conjunto $\{0,1\}^{\boldsymbol{E}}$. Por lo tanto, el diseño de clasificadores se limita al caso donde $c = 2$, con lo cual la función ψ: $\mathcal{X} \rightarrow \mathcal{Y}$ es un clasificador binario, donde $\mathcal{X} = L_1^n$ e $\mathcal{Y} = \{0,1\}$.

La respuesta de una red neuronal tipo feed-forward (Capítulo III, Sección 3.5) formada por una única neurona con función de transferencia lineal a un vector de entrada $\mathbf{X}\in\mathcal{X}$ está dada por

$$T(\mathbf{X};\boldsymbol{\beta}) = \beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n = \beta_0 + \sum_{i=1}^{n}\beta_i X_i\,, \tag{5.1}$$

donde $\boldsymbol{\beta} = (\beta_0,\beta_1,\ldots,\beta_n)\in\mathbb{R}^{n+1}$ es un vector formado por $n + 1$ parámetros. La clase de todas las funciones lineales que se pueden implementar usando esta red neuronal básica se denota mediante C_n, donde $C_n = \{T(\mathbf{X};\boldsymbol{\beta}) \mid \boldsymbol{\beta}\in\mathbb{R}^{n+1}\}$. Si se modela el logaritmo natural del cociente entre las probabilidades condicionales de clase $\mathbb{P}(Y = 1|\mathbf{X})$ y $\mathbb{P}(Y = 0|\mathbf{X}) = 1 - \mathbb{P}(Y = 1|\mathbf{X})$ mediante la red neuronal C_n, se obtiene la siguiente regresión dada por:

$$ln\left(\frac{\mathbb{P}(Y=1|\,\mathbf{X})}{1-\mathbb{P}(Y=1|\,\mathbf{X})}\right) = T(\mathbf{X};\boldsymbol{\beta}) = \beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n. \tag{5.2}$$

La ecuación 5.2 define una regresión lineal que, en aprendizaje computacional supervisado, toma el nombre de *regresión logística* o en inglés *logistic regression* [Hastie *et al.*, 2001], [Dreiseitl y Ohno-Machado, 2002], [Mohri *et al.*, 2012], [Murphy, 2012].

Si se resuelve la ecuación 5.2 para $\mathbb{P}(Y = 1|\mathbf{X})$ se obtiene la siguiente ecuación:

$$\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta}) = 1/(1 + exp(-T(\mathbf{X};\boldsymbol{\beta}))), \tag{5.3}$$

donde $\boldsymbol{\beta}$ en la probabilidad condicional de la clase 1 $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ dado el vector $\mathbf{X}$ enfatiza la aproximación realizada de la probabilidad $\mathbb{P}(Y = 1|\mathbf{X})$ usando la clase de funciones C_n y la ecuación 5.3. La función sigmoidea f: $\mathbb{R} \rightarrow [0,1]$ tal que $f(z) = 1/(1 + exp(-z))$, del lado derecho de la ecuación 5.3, se denomina *función logsig* (Figura 5.1). Se puede comprobar que la derivada de la función *logsig* es $df(z)/dz = (1 - f(z))f(z)$. Por lo tanto, la regresión logística permite aproximar, o modelar, la probabilidad condicional de clase $\mathbb{P}(Y = 1|\mathbf{X})$ mediante funciones *logsig* cuyo argumento es el valor de una función lineal (ecuación 5.1). Esto implica que un clasificador basado en regresión logística particiona el espacio de características $\boldsymbol{\mathcal{X}}$ en dos regiones usando un hiperplano cuyos coeficientes se obtienen al resolver la ecuación 5.1 para 0: $T(\mathbf{X};\boldsymbol{\beta}) = 0$. Cada región de $\boldsymbol{\mathcal{X}}$ define una clase, donde $T(\mathbf{X};\boldsymbol{\beta}) \geq 0$ es la región perteneciente a la clase 1 y $T(\mathbf{X};\boldsymbol{\beta}) < 0$ es la región de la clase 0. Esto es completamente equivalente a umbralar el valor de la probabilidad condicional estimada $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ en 0.5. La dimensión VC de este modelo de clasificación es igual al número total de parámetros que se pueden ajustar: $VC = n + 1$ [Devroye *et al.*, 1996]. El uso del modelo de regresión logística involucra un costo de restricción si las distribuciones condicionales *a priori* $\Pr(\mathbf{X}|Y = 0)$ y $\Pr(\mathbf{X}|Y = 1)$ no son distribuciones Gausianas multivariable con igual matriz de covarianza [Duda *et al.*, 2001].

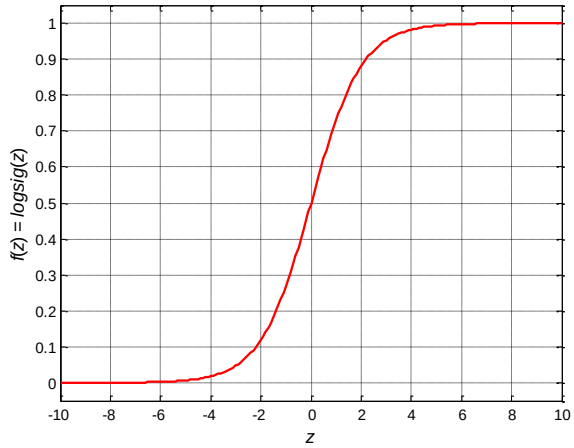


Figura 5.1. Forma de la función sigmoidea *logsig*.

Para un contexto práctico de diseño de un W-operador, sea un conjunto de ejemplos de entrenamiento $\boldsymbol{\mathcal{D}} = \{(\mathbf{X}_i,freq(\mathbf{X}_i,Y = 0),freq(\mathbf{X}_i,Y = 1))\}$, donde cada par vector-etiqueta $(\mathbf{X},Y)$ es una realización independiente de la distribución conjunta y fija, pero desconocida, $\Pr(\mathbf{X},Y)$ e $i = 1,\ldots,N$. El ajuste de parámetros de la regresión logística se puede realizar maximizando la función de verosimilitud $\mathbb{P}(\boldsymbol{\mathcal{D}}|\boldsymbol{\beta})$ del conjunto de entrenamiento $\boldsymbol{\mathcal{D}}$ dado el vector de coeficientes $\boldsymbol{\beta} = (\beta_0,\beta_1,\ldots,\beta_n)$:

$$\mathbb{P}(\boldsymbol{\mathcal{D}}|\boldsymbol{\beta}) = \prod_{i=1}^{N}(1-\mathbb{P}(Y=1|\,\mathbf{X}_i;\boldsymbol{\beta}))^{freq(\mathbf{X}_i,Y=0)}\,\mathbb{P}(Y=1|\,\mathbf{X}_i;\boldsymbol{\beta})^{freq(\mathbf{X}_i,Y=1)}. \tag{5.4}$$

A este método se lo conoce como estimación de parámetros por *máxima verosimilitud* [Devroye *et al.*, 1996]. Maximizar la ecuación 5.4 es equivalente a minimizar el negativo del logaritmo natural de la función de verosimilitud $\mathbb{P}(\boldsymbol{\mathcal{D}}|\boldsymbol{\beta})$:

$$-ln(\mathbb{P}(\boldsymbol{\mathcal{D}}\,|\,\boldsymbol{\beta})) = -\sum_{i=1}^{N}(freq(\mathbf{X}_i, Y=0) ln(1-\mathbb{P}(Y=1\,|\,\mathbf{X}_i;\boldsymbol{\beta})) + freq(\mathbf{X}_i, Y=1) ln(\mathbb{P}(Y=1\,|\,\mathbf{X};\boldsymbol{\beta}))) \quad (5.5)$$

Si se dividen los dos miembros de la ecuación 5.5 por la suma de todas las frecuencias de las observaciones del conjunto $\boldsymbol{\mathcal{D}}$: $p = \sum_{i=1}^{N}(freq(\mathbf{X}_i, Y=0) + freq(\mathbf{X}_i, Y=1))$, entonces se obtiene la siguiente función de costo para la estimación de parámetros de la regresión:

$$J(\boldsymbol{\beta}) = -(1/p)\sum_{i=1}^{N}(freq(\mathbf{X}_i, Y=0) ln(1-\mathbb{P}(Y=1\,|\,\mathbf{X}_i;\boldsymbol{\beta})) + freq(\mathbf{X}_i, Y=1) ln(\mathbb{P}(Y=1\,|\,\mathbf{X};\boldsymbol{\beta}))) \quad (5.6)$$

Una propiedad importante de la función de costo $J(\boldsymbol{\beta})$ de la ecuación 5.6 es la de ser una función convexa [Mohri *et al.*, 2012]. Esto garantiza la existencia de una única solución o mínimo global. En la Figura 5.2 se presenta un ejemplo de las isolíneas de una función de costo $J(\boldsymbol{\beta})$ para un espacio de parámetros $\boldsymbol{\beta} = (\beta_0,\beta_1,\beta_2)$, donde se asume que $\beta_0 = 0$. En esta figura se puede observar que, a medida que se aproxima al centro del gráfico, el valor de la función de costo disminuye hasta llegar al mínimo global.

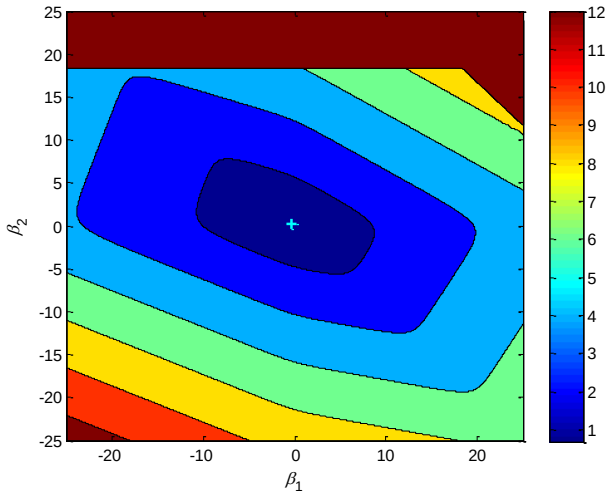


Figura 5.2. Isolíneas de la función de costo $J(\boldsymbol{\beta})$ de un modelo de regresión logística. El mínimo global de la función $J(\boldsymbol{\beta})$ se indica usando el símbolo "+".

Otra alternativa para realizar el ajuste de parámetros de la regresión logística consiste en utilizar el mismo procedimiento utilizado para el diseño automático de W-operadores binarios. Esto es, primero estimar la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$, usando la regla plug-in, para cada $\mathbf{X}\in\boldsymbol{\mathcal{D}}$ en base a las frecuencias $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$. Segundo, calcular el valor del cociente $\phi(\mathbf{X})$ mediante el lado izquierdo de la ecuación 5.2 utilizando un parámetro infinitesimal δ positivo. Tercero, resolver el problema de regresión lineal dado por $T(\mathbf{X};\boldsymbol{\beta}) = \phi(\mathbf{X})$ minimizando $\varepsilon_{ECM,N}(\boldsymbol{\beta}) = (1/2N)\sum_{i=1}^{N}(T(\mathbf{X}_i;\beta) - \phi(\mathbf{X}_i))^2$. Dado que

esta función de costo es convexa, se puede encontrar una solución exacta o mínimo global como resultado del proceso de minimización. En este caso $\boldsymbol{\beta}^{\mathrm{T}} = [\mathbf{D}^{\mathrm{T}}\mathbf{D}]^{-1}\mathbf{D}^{\mathrm{T}}\Phi$, siendo $\mathbf{D}$ una matriz de $N\times(n + 1)$ elementos, donde cada fila contiene un vector $(1,X_{i,1},...,X_{i,n})$ que se obtiene al concatenar las componentes de la configuración de ventana $\mathbf{X}_i = (X_{i,1},...,X_{i,n})$ con el escalar 1. El término Φ es un vector columna de $N\times 1$ elementos conteniendo los valores de $\phi(\mathbf{X}_i)$, con $i = 1,\ldots,N$.

Existen tres inconvenientes con el método descrito en el párrafo anterior. Primero, para ventanas de gran tamaño (n grande, por ejemplo 30×30 píxeles), el cálculo de la matriz $[\mathbf{D}^{\mathrm{T}}\mathbf{D}]^{-1}$ puede demandar de un alto costo computacional. Segundo, para el cálculo de la inversa $[\mathbf{D}^{\mathrm{T}}\mathbf{D}]^{-1}$, la matriz $\mathbf{D}^{\mathrm{T}}\mathbf{D}$ debe ser no singular. Este problema se puede solventar utilizando la matriz pseudoinversa para los casos donde $\mathbf{D}^{\mathrm{T}}\mathbf{D}$ no sea invertible. Tercero, el inconveniente más grave de este método es que, para ventanas de mediano y gran tamaño e imágenes observadas en escala de grises, los valores de las frecuencias $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$ son, usualmente, muy bajos. Esto dificulta obtener una buena estimación, usando la regla plug-in, de $\mathbb{P}(Y = 1|\mathbf{X})$ para las configuraciones de ventana $\mathbf{X}$ del conjunto de entrenamiento. Para evitar estos inconvenientes, en esta tesis se utiliza el método de máxima verosimilitud para el entrenamiento de la regresión logística, con el procedimiento antes descrito.

La minimización de la función de costo de la ecuación 5.6 consiste en resolver un problema de optimización, lo cual se puede realizar mediante el empleo de un método de descenso de gradiente. Para esto, las derivadas parciales de la función de costo $J(\boldsymbol{\beta})$, con respecto a cada uno de los parámetros a ajustar, se calculan mediante las siguientes ecuaciones:

$$\begin{aligned}\frac{\partial J(\boldsymbol{\beta})}{\partial \beta_0} &= (1/p)\textstyle\sum_{i=1}^{N}\left(freq(\mathbf{X}_i)\mathbb{P}(Y=1\,|\,\mathbf{X}_i;\boldsymbol{\beta}) - freq(\mathbf{X}_i, Y=1)\right)\\ \frac{\partial J(\boldsymbol{\beta})}{\partial \beta_j} &= (1/p)\textstyle\sum_{i=1}^{N}\left(\left(freq(\mathbf{X}_i)\mathbb{P}(Y=1\,|\,\mathbf{X}_i;\boldsymbol{\beta}) - freq(\mathbf{X}_i, Y=1)\right)X_{i,j}\right) \quad \forall j = 1,...,n\end{aligned}, \qquad (5.7)$$

donde $freq(\mathbf{X}_i) = freq(\mathbf{X}_i,Y = 0) + freq(\mathbf{X}_i,Y = 1)$, con $i = 1,\ldots,N$. Al dividir las frecuencias marginal $freq(\mathbf{X}_i)$ y conjunta $freq(\mathbf{X}_i,Y = 1)$ por la suma total de frecuencias p del conjunto de entrenamiento $\mathcal{D}$, se obtiene el peso con el que la configuración $\mathbf{X}_i$ y el par $(\mathbf{X}_i,Y = 1)$, respectivamente, influyen en la minimización de la función de costo $J(\boldsymbol{\beta})$. Si $freq(\mathbf{X}_i)$ es grande, $freq(\mathbf{X}_i,Y = 1)$ es pequeño, y $\mathbb{P}(Y = 1|\mathbf{X}_i;\boldsymbol{\beta}) = 1$, entonces la contribución de la configuración $\mathbf{X}_i$ al valor del gradiente es grande. Si $freq(\mathbf{X}_i)$ y $freq(\mathbf{X}_i,Y = 1)$ son grandes, y $\mathbb{P}(Y = 1|\mathbf{X}_i;\boldsymbol{\beta}) = 1$, entonces la contribución de la configuración de ventana $\mathbf{X}_i$ al valor del gradiente es baja.

Para la minimización de la ecuación 5.6, en esta tesis se utiliza un método numérico de optimización no lineal sin restricciones basado en la definición de regiones de confianza combinado con el método reflectivo de Newton [Coleman y Li, 1994, 1996]. La idea básica de este método es, en cada época, aproximar la función de costo $J(\boldsymbol{\beta})$ mediante una función

más simple $g(\boldsymbol{\beta})$ que refleja su comportamiento en una vecindad, o región de confianza, alrededor del valor actual para $\boldsymbol{\beta}$. La función $g(\boldsymbol{\beta})$ se utiliza para actualizar los parámetros y está basada en una aproximación de Taylor para $J(\boldsymbol{\beta})$ en el valor actual de $\boldsymbol{\beta}$. En cada época de la optimización, se evalúa el valor de la función $J(\boldsymbol{\beta})$ y su gradiente utilizando los ejemplos del conjunto de entrenamiento $\mathcal{D}$. La optimización finaliza cuando la reducción del valor de $J(\boldsymbol{\beta})$, o la magnitud de su gradiente, es menor que un cierto umbral, o cuando no existe cambio alguno en cualquiera de estos dos valores durante 5 épocas consecutivas. También se puede definir un número máximo de épocas de entrenamiento.

El costo computacional de entrenamiento de un clasificador basado en regresión logística depende, en gran medida, del número de ejemplos de entrenamiento, N, la dimensión, n, de las configuraciones de ventana, y el algoritmo de optimización. El valor de N depende del tamaño de la ventana y el número de imágenes de entrenamiento. Una forma de acelerar el entrenamiento de la regresión logística, por cada época de entrenamiento, consiste en el cálculo del gradiente de $J(\boldsymbol{\beta})$ de forma matricial: $\mathbf{G} = (1/p)[\mathbf{F}\bullet\times\mathbf{P} - \mathbf{B}]^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}$. $\mathbf{G}$ es un vector fila que contiene los valores de las $n + 1$ derivadas parciales $\partial J(\boldsymbol{\beta})/\partial\beta_0,\ldots,\partial J(\boldsymbol{\beta})/\partial\beta_n$, o gradiente, de la función de costo $J(\boldsymbol{\beta})$ con respecto a $\boldsymbol{\beta} = (\beta_0,\ldots,\beta_n)$. $\mathbf{F}$ y $\mathbf{P}$ son vectores columna, de $N\times 1$ elementos cada uno, conteniendo la frecuencia marginal $freq(\mathbf{X}_i)$ y la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X}_i;\boldsymbol{\beta})$ para cada configuración de ventana $\mathbf{X}_i$ del conjunto $\mathcal{D}$, respectivamente. El símbolo $\bullet\times$ denota la multiplicación elemento a elemento entre los vectores $\mathbf{F}$ y $\mathbf{P}$. $\mathbf{B}$ es un vector columna de $N\times 1$ elementos con la frecuencia $freq(\mathbf{X}_i,Y = 1)$ de $\mathbf{X}_i$ y $\mathbf{D}$ es una matriz de $N\times(n + 1)$ elementos formada al concatenar las componentes de la configuración $\mathbf{X}_i = (X_{i,1},...,X_{i,n})$ con el escalar 1: $(1,X_{i,1},...,X_{i,n})$, con $i = 1,\ldots,N$.

5.3. Diseño de Clasificadores y Segmentación de Imágenes

En la Sección 1.4.1 del Capítulo I se vio que la segmentación binaria de una imagen en escala de grises O consiste en una partición de su dominio en dos subconjuntos disjuntos S_1 y S_2. Utilizando un clasificador binario, la segmentación de O puede ser considerada como un problema de clasificación o etiquetado de sus píxeles. En este caso, lo usual es asignar la etiqueta 1, que en la imagen segmentada aparece como blanco, a los píxeles pertenecientes a los objetos de interés y 0, que en la imagen segmentada aparece como negro, a los píxeles pertenecientes al fondo. Por lo tanto, el problema de segmentación binaria de imágenes en escala de grises puede ser considerado como un problema de diseño de un clasificador binario. Este clasificador induce una partición del dominio de O en dos subconjuntos disjuntos S_1 y S_2 que contienen los píxeles etiquetados con 0 y 1, respectivamente. Para un problema de segmentación que involucre a más de un tipo de objeto de interés, aparte del fondo de la imagen, el problema consiste en el diseño de un clasificador multiclase. Dicho clasificador induce una partición del dominio de la imagen a segmentar en tres o más subconjuntos disjuntos. Uno de los subconjuntos contiene los píxeles del fondo y los demás contienen los píxeles de cada tipo de objeto segmentado.

5.4. Regresión Logística y Segmentación Binaria de Imágenes

En esta sección se analizan aspectos prácticos concernientes a la segmentación binaria de imágenes en escala de grises usando W-operadores definidos con regresión logística.

5.4.1. Análisis y Aplicación de la Regresión Logística mediante Correlación y Convolución

Una forma básica de segmentar una imagen en escala de grises es umbralando la imagen resultante de aplicar un determinado filtro espacial mediante una convolución o correlación de imágenes. En este caso, el problema de segmentación consiste en diseñar la forma y encontrar los coeficientes del kernel o máscara que define al filtro espacial. También se debe definir un umbral para obtener la imagen binaria. Si se analiza la respuesta de la regresión logística $T(\mathbf{X};\boldsymbol{\beta}) = \beta_0 + \sum_{i=1}^{n}\beta_i X_i$ para un vector de configuración $\mathbf{X} = (X_1,\ldots,X_n)$, se puede notar que la sumatoria $\sum_{i=1}^{n}\beta_i X_i$ es equivalente a la respuesta de la correlación entre la imagen a segmentar O y una máscara o kernel B para un píxel arbitrario t. $X_1,\ldots,X_n$ son los niveles de gris de la observación, a través de una ventana $W = \{w_1,\ldots,w_n\}$, del píxel t de la imagen a segmentar O. La máscara B está formada por los coeficientes $\beta_1,\ldots,\beta_n$ de la regresión logística con una estructura (posición de los coeficientes) definida por la estructura creada para la ventana W. El valor del parámetro β_0 puede ser considerado como un factor que aumenta o disminuye la respuesta de la correlación de imágenes dependiendo de si es un valor positivo o negativo, respectivamente. En este contexto, $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$, calculado usando la ecuación 5.3, cuantifica mediante un valor de probabilidad el valor, para un píxel t, de la correlación entre las imágenes O y B. Lo anterior se extiende directamente para la operación de convolución si se rota la máscara B en un valor de 180º alrededor de su origen.

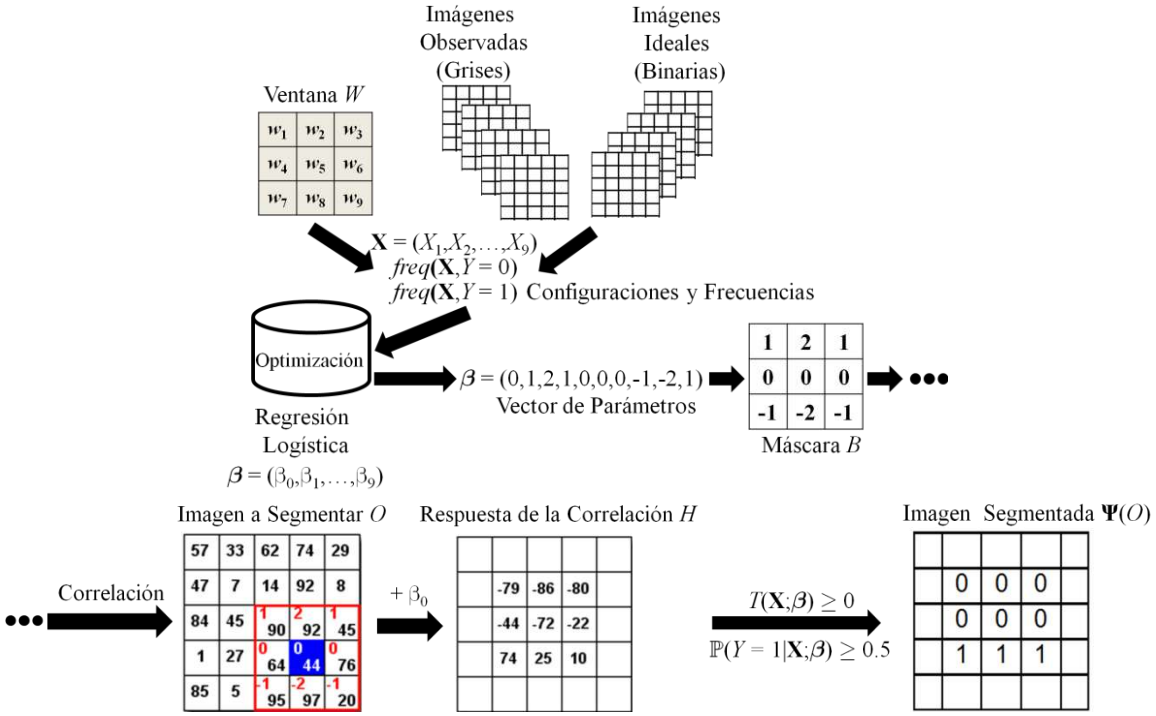


Figura 5.3. Ilustración de la segmentación binaria de imágenes en escala de grises utilizando un clasificador binario basado regresión logística. La aplicación del clasificador se realiza mediante las operaciones de correlación y umbralamiento de imágenes.

Ejemplo 5.1. En la Figura 5.3 se ilustra la segmentación binaria de imágenes en escala de grises usando un clasificador basado en regresión logística y una ventana W de 3×3 píxeles. El procedimiento se inicia con el ajuste del vector de parámetros $\boldsymbol{\beta} = (\beta_0,\beta_1,\ldots,\beta_9)$, o entrenamiento de la regresión logística, mediante la optimización de la función de costo dada por la ecuación 5.6. Para la optimización se utiliza un conjunto de N configuraciones de ventana $\mathbf{X} = (X_1,\ldots,X_9)$ con sus respectivas frecuencias $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$. Las configuraciones y frecuencias se obtienen mediante un escaneo de las imágenes observadas e ideales de entrenamiento utilizando la ventana W. Luego, en base a los parámetros estimados para la regresión logística $\beta_1,\ldots,\beta_9$, se define una máscara B de 3×3 píxeles. La estructura de la máscara B es igual a la estructura definida para la ventana W. Como resultado de la correlación entre las imágenes O y B se obtiene una nueva imagen, a cuyos píxeles se les suma el valor del parámetro β_0, resultando en la imagen H. Como paso final, se obtiene la imagen segmentada $\boldsymbol{\Psi}(O)$ umbralando en 0 los valores de los píxeles de la imagen H. Nótese que en este caso no se procesan los píxeles de borde donde la máscara no está contenida totalmente en el dominio de la imagen a procesar.

El último paso del procedimiento antes descrito es equivalente a aplicar a cada píxel de la imagen H la transformación $f(z) = 1/(1 + exp(-z))$. Como resultado de esta transformación puntual se obtiene un mapa con las probabilidades condicionales $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ de que cada píxel de la imagen observada O sea parte del objeto a segmentar dada la configuración $\mathbf{X}$ observada en dicho píxel. Luego se puede obtener la imagen segmentada $\boldsymbol{\Psi}(O)$ umbralando el mapa de probabilidades. El resultado del umbralamiento de H en 0 es igual al resultado del umbralamiento del mapa de probabilidades en 0.5. Dependiendo de la aplicación, se puede utilizar un umbral para la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ distinto a 0.5. Aparte de su función ilustrativa, este procedimiento también hace posible que la aplicación de un W-operador definido mediante un clasificador basado en regresión logística sea una tarea computacionalmente simple. Esto es debido a que existen algoritmos de PDI que han sido optimizados para realizar las operaciones de correlación y convolución de imágenes [Gonzalez y Woods, 2002], [Shih, 2009].

5.4.2. Cálculo del Patrón que Maximiza la Regresión Logística

Después del análisis de la segmentación binaria de imágenes en escala de grises, utilizando W-operadores basados en regresión logística, surge la pregunta *¿qué patrón o características busca el clasificador en las imágenes a procesar*? Para responder a esta pregunta se debe encontrar la observación de ventana $\mathbf{X}$ que maximiza $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$. Encontrar la imagen que maximiza dicha probabilidad es equivalente a encontrar el vector de configuración $\mathbf{X} = (X_1,\ldots,X_n)$ que maximiza la función $T(\mathbf{X};\boldsymbol{\beta}) = \beta_0 + \sum_{i=1}^{n}\beta_i X_i$. Para no obtener una respuesta trivial, donde los valores de todas las componentes $X_i\in\mathbf{X}$ tienden a infinito, se puede asumir que el cuadrado de la norma del vector $\mathbf{X}$ está acotado superiormente por una cantidad positiva y finita s: $\|\mathbf{X}\|^2 = \sum_{i=1}^{n}(X_i)^2 \leq s$.

La maximización de la función $T(\mathbf{X};\boldsymbol{\beta})$, sujeta a la restricción $\sum_{i=1}^{n}(X_i)^2 - s \leq 0$, se puede resolver utilizando el *método de los multiplicadores de Lagrange* [Bertsekas, 1982],

[Bishop, 2005], [Mohri *et al.*, 2012]. Esta maximización equivale a encontrar el punto de silla de la función

$$L(\mathbf{X},\propto) = (\beta_0 + \sum_{i=1}^{n}\beta_i X_i) + \alpha(-\sum_{i=1}^{n}(X_i)^2 + s), \tag{5.8}$$

donde $\alpha \geq 0$ es un *multiplicador de Lagrange*. El punto de silla de la función $L(\mathbf{X},\alpha)$ se puede obtener resolviendo el siguiente sistema de ecuaciones:

$$\beta_i X_i - 2\alpha = 0 \;\forall i = 1,\ldots,n, \qquad \text{y} \qquad \alpha(-\sum_{i=1}^{n}(X_i)^2 + s) = 0. \tag{5.9}$$

Resolviendo lo anterior se tiene que los valores de los píxeles de la observación de ventana que maximiza la respuesta de la regresión logística están dados por

$$X_i = \frac{\beta_i\sqrt{s}}{\sqrt{\sum_{j=1}^{n}(\beta_j)^2}}. \tag{5.10}$$

Ejemplo 5.2. Sea β = (0,-1,0,1,-1,0,1,-1,0,1) un vector que define un clasificador basado en regresión logística entrenado con configuraciones $\mathbf{X} = (X_1,\ldots,X_9)$ obtenidas mediante una ventana W de 3×3 píxeles (Figura 5.4-a). En base a este vector se define una máscara B, cuya estructura es igual a la estructura de la ventana W (Figura 5.4-b). Asumiendo que cada píxel de las imágenes a procesar toma un valor en el intervalo [0,1], se puede comprobar fácilmente que el máximo de $\|\mathbf{X}\|^2$ es s = 9. En este caso, el vector o patrón que maximiza la respuesta de la regresión es **X** = (-1.22,0,1.22,-1.22,0,1.22,-1.22,0,122). Las componentes de este vector han sido calculadas utilizando la ecuación 5.10. En la Figura 5.4-c se presenta un mapa de intensidades construido a partir de las componentes de **X**.

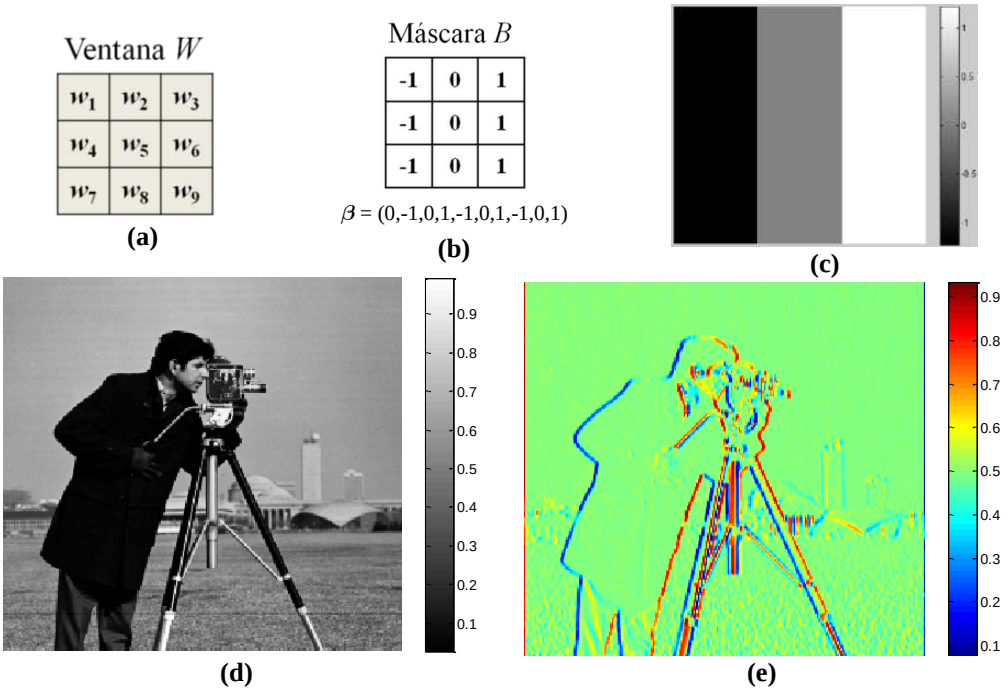


Figura 5.4. Aplicación de la regresión logística mediante una correlación de imágenes. (a) Ventana utilizada para el diseño del clasificador. (**b**) Máscara definida a partir del vector de coeficientes de la regresión logística y usando la estructura de la ventana de la imagen a. (**c**) Mapa de intensidades que maximizan la respuesta de la regresión logística. (**d**) Imagen original con niveles de intensidad en el rango [0,1]. (**e**) Mapa de probabilidades que muestra el nivel de correlación entre la máscara de la imagen b y la imagen d.

Dado que los valores de algunos coeficientes de la máscara presentada en la Figura 5.4-b están fuera del rango [0,1], es imposible encontrar este patrón en la imagen de la Figura 5.4-d. Sin embargo, al realizar la correlación de imágenes entre *O* (imagen original), con rango [0,1], y la máscara *B*, los valores resultantes serán altos en las regiones de *O* donde la variación de intensidad sea creciente de izquierda a derecha. En la Figura 5.4-e se presenta un mapa de probabilidades calculado a partir de los resultados de la correlación entre cada píxel (y sus vecinos) de la imagen de la Figura 5.4-d y la máscara B de la Figura 5.4-b. El valor de cada píxel del mapa de probabilidades está calculado aplicando la función *logsig* a cada valor de la correlación. Es importante recordar que *logsig*($-\infty$) = 0, *logsig*(0) = 0.5, y *logsig*($+\infty$) = 1 (Figura 5.1). Se puede notar que el nivel de correlación es alto en el sentido negativo, resultando en una probabilidad cercana a 0, para aquellas regiones de la imagen *O* donde la variación de intensidad es decreciente de izquierda a derecha. La correlación es aproximadamente nula, resultando en una probabilidad cercana a 0.5, en aquellas regiones de la imagen *O* donde no existe variación de intensidad. El valor de la correlación es alta en el sentido positivo, resultando en una probabilidad cercana a 1, en las regiones donde la variación de intensidad de la imagen *O* es creciente de izquierda a derecha. El mínimo y máximo del mapa de probabilidades (Figura 5.4-e) son 0.0712 y 0.9324, respectivamente.

5.5. Diseño Balanceado de Clasificadores

En la Sección 1.5.2 del Capítulo I se mostró que el error verdadero $\varepsilon(\psi)$ de la función característica, o clasificador, ψ que define a un W-operador puede ser expresada en términos de su tasa de falsos positivos, FPR, y tasa de falsos negativos, FNR, dado por la siguiente ecuación:

$$\begin{aligned}\varepsilon(\psi) &= \mathbb{P}(\psi(\mathbf{X})=1\,|\,Y=0)\mathbb{P}(Y=0)+\mathbb{P}(\psi(\mathbf{X})=0\,|\,Y=1)\mathbb{P}(Y=1)\\ &= \mathrm{FPR}\times\mathbb{P}(Y=0)+\mathrm{FNR}\times\mathbb{P}(Y=1)\end{aligned}. \qquad (5.11)$$

A nivel teórico, si la clase 0 tiene una probabilidad *a priori* $\mathbb{P}(Y = 0)$ mucho más grande que $\mathbb{P}(Y = 1)$, entonces la forma de obtener un valor de $\varepsilon(\psi)$ pequeño es haciendo que el producto $\mathrm{FPR}\times\mathbb{P}(Y = 0)$ sea también pequeño. Dado que para un problema $\mathbb{P}(Y = 0)$ es fijo, esto se consigue reduciendo el valor de FPR. Sin embargo, esto a su vez causa un gran incremento del valor de FNR, el cual se ve compensado al ser multiplicado por un valor pequeño de $\mathbb{P}(Y = 1)$. En este caso, a pesar de que $\varepsilon(\psi)$ es bajo, se puede notar que existe una gran diferencia, o un desbalance, entre los valores de FNR y FPR: FNR $>>$ FPR. Una situación parecida ocurre cuando se tiene que $\mathbb{P}(Y = 1) >> \mathbb{P}(Y = 0)$, lo cual obliga a procurar que FPR $>>$ FNR para que el error $\varepsilon(\psi)$ sea bajo.

A nivel práctico, si la frecuencia total de una de las clases es mucho mayor que la frecuencia total de la otra clase, entonces el ajuste de parámetros durante el entrenamiento de un clasificador se realiza de tal modo que los valores de $\mathbb{P}(Y|\mathbf{X};\boldsymbol{\beta})$ sean altos para la clase mayoritaria. Esto se puede comprobar fácilmente para el caso de regresión logística al analizar la función de costo dada por la ecuación 5.6. En estas condiciones, es usual que el clasificador diseñado prediga casi siempre la etiqueta de la clase con mayor frecuencia. En este escenario, a pesar de que el error empírico del clasificador diseñado puede ser muy

bajo, existe un gran desbalance entre sus valores de FNR y FPR tal como se evidencia a través del siguiente ejemplo.

Ejemplo 5.3. Sea el problema de clasificación de la Tabla 5.1, donde la primera columna contiene a todos los posibles vectores $\mathbf{X} = (X_1,X_2)$ para el problema en cuestión. En la segunda y tercera columnas se presentan las probabilidades conjuntas para todos los pares $(\mathbf{X},Y = 0)$ y $(\mathbf{X},Y = 1)$, respectivamente. En este problema, la clase predominante es la clase 0 con una probabilidad $\mathbb{P}(Y = 0) = 0.9105$; mientras que la clase 1 tiene una probabilidad $\mathbb{P}(Y = 1) = 0.0895$. La cuarta columna contiene las etiquetas del clasificador óptimo ψ_{opt} para cada vector $\mathbf{X}$. En la quinta columna se presentan las predicciones de un clasificador ψ, diseñado empíricamente, el cual predice constantemente la etiqueta 0. El error de este clasificador, $\varepsilon(\psi) = 0.0895$, es cercano al error del clasificador óptimo, $\varepsilon(\psi_{opt}) = 0.0703$. Los valores de FPR y FNR del clasificador ψ son 0 y 1, respectivamente; mientras que los valores de FPR y FNR del clasificador óptimo ψ_{opt} son 0.0351 y 0.4286, respectivamente. Lo anterior muestra que, aunque el error del clasificador diseñado ψ es bajo, existe un desbalance extremadamente grande entre sus valores de FPR y FNR. Por lo tanto, pese a su bajo error, el clasificador ψ no tiene utilidad práctica, pues no reconoce ningún caso de la clase minoritaria (clase 1).

Tabla 5.1. Ejemplo de diseño desbalanceado de clasificadores.

$\mathbf{X} = (X_1,X_2)$	$\mathbb{P}(\mathbf{X},Y = 0)$	$\mathbb{P}(\mathbf{X},Y = 1)$	$\psi_{opt}(\mathbf{X})$	$\psi(\mathbf{X})$
0 0	0.1757	0.0096	0	0
0 1	0.1278	0.0064	0	0
0 2	0.0958	0.0064	0	0
1 0	0.0160	0.0224	1	0
1 1	0.0160	0.0288	1	0
1 2	0.0479	0.0032	0	0
2 0	0.0319	0.0032	0	0
2 1	0.2396	0.0064	0	0
2 2	0.1597	0.0032	0	0

Para el diseño de clasificadores para la segmentación de imágenes, un desbalance en los valores de las frecuencias a favor de la clase 0 haría que la imagen resultante de la segmentación contenga únicamente fondo. Si la frecuencia predominante es la de la clase 1, entonces todos los píxeles serían clasificados como parte del objeto de interés, sin que se reconozca el fondo. Es claro que ninguno de estos dos escenarios posee utilidad práctica dentro de PDI. En estas condiciones, lo ideal sería minimizar el error empírico del clasificador diseñado estableciendo al mismo tiempo un equilibrio, o balance, entre sus valores de FNR y FPR. Desafortunadamente, esto no es posible en la práctica usando la optimización propuesta debido a que el ajuste de parámetros, o entrenamiento de un clasificador, se realiza minimizando el error de clasificación.

Si para un problema de clasificación determinado se tiene que $\mathbb{P}(Y = 0) = \mathbb{P}(Y = 1) = 0.5$, entonces la ecuación 5.11 resulta en

$$\varepsilon(\psi) = (\text{FPR} + \text{FNR})/2. \tag{5.12}$$

Por lo tanto, si las probabilidades *a priori* de clase son iguales, entonces el error verdadero de un clasificador es igual al promedio entre sus tasas de falsos positivos y falsos negativos. Esto implica que la minimización del error $\varepsilon(\psi)$ de un clasificador ψ para un problema donde las probabilidades de clase son iguales equivale a minimizar el promedio entre sus valores de FPR y FNR. En la práctica, este escenario se puede conseguir balanceando artificialmente las frecuencias de todos los ejemplos de entrenamiento, con lo cual se tiene $\sum_{i=1}^{N} freq_{Bal}(\mathbf{X}_i, Y=0) = \sum_{i=1}^{N} freq_{Bal}(\mathbf{X}_i, Y=1)$ resultando en $\hat{\mathbb{P}}(Y=0) = \hat{\mathbb{P}}(Y=1) = 0.5$. En función de este análisis se tiene que para un problema de clasificación binaria, las frecuencias balanceadas para la configuración $\mathbf{X}_i$, con $i = 1,\ldots,N$, se calculan mediante la ecuación 5.13.

$$\begin{aligned} freq_{Bal}(\mathbf{X}_i, Y=0) &= \frac{1}{2}\left(\frac{\sum_{j=1}^{N}\sum_{u=0}^{1} freq(\mathbf{X}_j, Y=u)}{\sum_{j=1}^{N} freq(\mathbf{X}_j, Y=0)}\right) freq(\mathbf{X}_i, Y=0) \\ freq_{Bal}(\mathbf{X}_i, Y=1) &= \frac{1}{2}\left(\frac{\sum_{j=1}^{N}\sum_{u=0}^{1} freq(\mathbf{X}_j, Y=u)}{\sum_{j=1}^{N} freq(\mathbf{X}_j, Y=1)}\right) freq(\mathbf{X}_i, Y=1) \end{aligned}. \tag{5.13}$$

En general, para un problema de clasificación que involucra a $c > 2$ clases, el balanceo artificial de las frecuencias se puede realizar mediante

$$freq_{Bal}(\mathbf{X}_i, Y=k) = \frac{1}{c}\left(\frac{\sum_{j=1}^{N}\sum_{u=0}^{c-1} freq(\mathbf{X}_j, Y=u)}{\sum_{j=1}^{N} freq(\mathbf{X}_j, Y=k)}\right) freq(\mathbf{X}_i, Y=k), \tag{5.14}$$

donde $k = 0,1,\ldots,(c-1)$.

Dado que en el anterior análisis no se modifican las probabilidades condicionales *a priori* $\mathbb{P}(\mathbf{X}|Y)$, sino solamente de las probabilidades marginales *a priori* $\mathbb{P}(Y)$, el problema de clasificación es esencialmente el mismo, con respecto a los valores de FPR y FNR. Esto es, el clasificador que minimiza el error $\varepsilon(\psi) = (\text{FPR} + \text{FNR})/2$, utilizando el balanceo artificial de frecuencias, es el mismo que minimiza el promedio entre la tasa de falsos positivos y falsos negativos para el problema original, donde las frecuencias están desbalanceadas. La ventaja del balanceo artificial radica en que no se requiere modificar el algoritmo de entrenamiento de un clasificador, que usualmente no es una tarea trivial, para conseguir minimizar el promedio entre su tasa de falsos positivos y falsos negativos. Por otra parte, el balanceo artificial de frecuencias hace que, para el problema original, la frontera de decisión se desplace hacia la región perteneciente a la clase con mayor probabilidad marginal estimada $\hat{\mathbb{P}}(Y)$. Esto a su vez provoca un incremento en el error del clasificador diseñado, el mismo que se ve compensado al lograr un balance entre su tasa de falsos positivos y falsos negativos.

La forma clásica de realizar el balanceo de muestras o conjuntos de entrenamiento es mediante un submuestreo de la clase mayoritaria o un sobremuestreo de la clase minoritaria [Maloof, 2003], [Weiss, 2004], [Jiang *et al.*, 2011]. El submuestreo provoca una pérdida de información debido a que se descartan algunas muestras. El sobremuestreo aumenta la carga computacional del problema debido a que se incrementa el número total de muestras. El balanceo artificial usado en esta tesis no presenta ninguno de estos problemas debido a que, previo al ajuste de parámetros de los clasificadores, sólo se modifican los valores de las frecuencias de cada ejemplo de entrenamiento [Brun *et al.*, 2010], [Brun *et al.*, 2013].

Ejemplo 5.4. Para mostrar el efecto del balanceo artificial de frecuencias se considera un problema de clasificación binaria con un espacio de características bidimensional continuo. Las muestras de cada clase se generan a partir de un modelo con distribuciones condicionales *a priori* $\Pr(\mathbf{X}|Y = 0)$ y $\Pr(\mathbf{X}|Y = 1)$ Gausianas con medias $\boldsymbol{\mu}_0 = \alpha\mathbf{a}$ y $\boldsymbol{\mu}_1 = -\alpha\mathbf{a}$, respectivamente, donde $\alpha = 1$ y $\mathbf{a} = (2/\sqrt{8}, 2/\sqrt{8})$. Para las matrices de covarianza se asume que $\Sigma_0 = \Sigma_1 = \mathbf{I}$, donde $\mathbf{I}$ es la matriz identidad de 2×2 elementos. Para este problema de clasificación Gausiano, la frontera de decisión óptima es una línea recta [Duda *et al.*, 2001], [Murphy, 2012]. El diseño de clasificadores se realiza usando regresión logística y un conjunto de entrenamiento de $N = 2500$ muestras, con 2250 y 250 realizaciones generadas a partir de las distribuciones $\mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$ y $\mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1)$, respectivamente. En estas condiciones se tiene que $\mathbb{P}(Y = 0) = 0.9$ y $\mathbb{P}(Y = 1) = 0.1$. En la Tabla 5.2 se reportan los valores estimados de ER, FNR, y FPR calculados antes y después de aplicar el balanceo artificial de frecuencias previo al entrenamiento de los clasificadores. Estos valores han sido estimados al repetir el experimento 50 veces utilizando en cada iteración un conjunto de testeo formado por 1000 muestras (750 de clase 0 y 250 de la clase 1).

Tabla 5.2. Ejemplo del efecto del diseño balanceado y no balanceado de clasificadores binarios basados regresión logística para un modelo con distribuciones condicionales *a priori* Gausianas.

	Sin Balanceo	**Con Balanceo**
ER [%]	7.05	15.89
FPR [%]	1.90	15.85
FNR [%]	85.38	7.27

Los resultados de la Tabla 5.2 muestran que el balanceo artificial de frecuencias produce un equilibrio entre los valores de FPR y FNR, a costa de un incremento de la tasa de error, ER. En la Figura 5.5 se muestra un ejemplo de las fronteras de decisión implementadas por la regresión logística antes y después de aplicar el balanceo de frecuencias. Se puede observar que el balanceo frecuencias hace que la frontera de decisión se mueva hacia la región correspondiente a la clase mayoritaria (clase 0). Esta es la razón por la cual se incrementa el valor de ER del clasificador balanceado con respecto al clasificador sin balanceo. En contraste con este incremento de la tasa de error, el beneficio del balanceo de frecuencias radica en que permite obtener una disminución dramática de la tasa de falsos negativos.

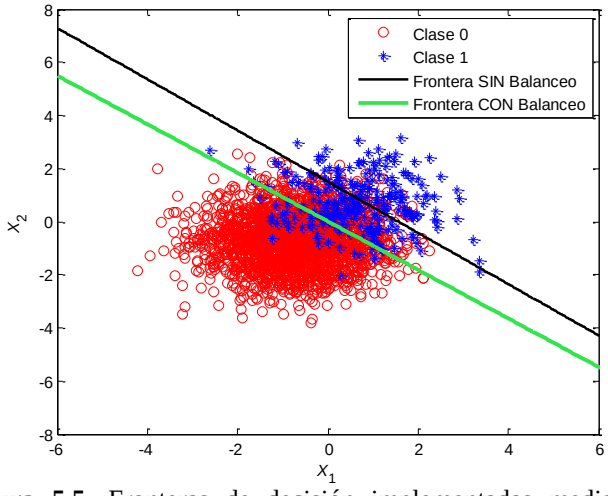


Figura 5.5. Fronteras de decisión implementadas mediante regresión logística con y sin balanceo artificial de frecuencias.

5.6. Costo Computacional y Ensambles de Clasificadores

A nivel teórico, grandes conjuntos de ejemplos de entrenamiento $\mathcal{D}$ (N grande) permiten reducir el costo de diseño de un clasificador. El costo de restricción se puede reducir aumentando la complejidad, o dimensión VC, de la clase de funciones $C = \{\mathbb{P}(Y = 1|\mathbf{X};\beta)\}$ usada para aproximar la probabilidad condicional de clase $\mathbb{P}(Y = 1|\mathbf{X})$. Si la dimensión VC de la clase de funciones C es mucho menor que el número de ejemplos de entrenamiento disponibles N, por ejemplo en una relación $N/VC > 10$, entonces el riesgo de sobreentrenamiento u overfitting es mínimo. La reducción del costo de diseño se da porque el número de ejemplos de entrenamiento permite encontrar la mejor función dentro de la clase C. Todo lo anterior implica que, para reducir el error de clasificación, se debe aumentar la complejidad del modelo utilizado en la medida en que su dimensión VC sea mucho menor que el valor de N. Lamentablemente, cuando la capacidad computacional disponible es limitada, el manejo de grandes conjuntos de entrenamiento y el uso de modelos de clasificación con dimensión VC alta representan un problema a nivel práctico.

Si el diseño de W-operadores involucra el uso de múltiples pares de imágenes de entrenamiento, lo cual ocurre frecuentemente sobre todo en aplicaciones reales, entonces el costo computacional del diseño de los clasificadores aumenta de manera significativa. Este problema se vuelve muy crítico cuando se utilizan ventanas de mediano y gran tamaño e imágenes de entrenamiento formadas por una gran cantidad de píxeles con un número elevado de niveles de gris. Esto se debe a que, para ventanas de mediano y gran tamaño y gran cantidad de niveles de gris, es usual que una configuración se observe una única vez durante el escaneo de las imágenes de entrenamiento. Esto a su vez resulta en un número de ejemplos de entrenamiento, N, aproximadamente igual al número total de píxeles escaneados. En estas condiciones, tanto el almacenamiento en memoria del conjunto de entrenamiento $\mathcal{D}$ como la optimización para el ajuste de parámetros de un clasificador se convierten en tareas computacionalmente complejas. Es importante aclarar que, este problema práctico existe si la capacidad computacional disponible para el diseño de clasificadores es limitada.

Ejemplo 5.5. Sea el problema de la segmentación de los vasos sanguíneos en las imágenes oculares de la base de datos pública DRIVE [Staal *et al.*, 2004]. En esta base de datos se disponen de 20 pares de imágenes, observadas e ideales, para el diseño de un W-operador. Cada imagen observada está formada por un total de 584×565 píxeles y 256 niveles de intensidad por cada canal de color. Para el diseño de los clasificadores se puede utilizar el canal verde, descartando los canales rojo y azul, debido a que éste presenta el menor nivel de ruido y el mayor contraste entre los vasos sanguíneos y el fondo retiniano [Staal *et al.*, 2004], [Fraz *et al.*, 2012]. Si se usan ventanas de 15×15 píxeles, que permiten observar tanto a los vasos sanguíneos gruesos como a los vasos sanguíneos delgados, y se asume que se obtiene una configuración diferente por cada píxel escaneado, entonces se obtiene un total de $N = (584 - 14)\times(565 - 14) = 1425.9$ millones de configuraciones de entrenamiento. Nótese que para este cálculo se descartan los píxeles de los bordes donde la ventana no está totalmente contenida en la imagen a escanear. Asumir que se observa una configuración diferente por cada píxel escaneado no es un criterio pesimista si se toma en cuenta el tamaño del espacio de configuraciones que es $256^{15\times15}$.

En estas condiciones, para el diseño de un W-operador se requiere el almacenamiento en memoria de 1425.9 millones de configuraciones de ventana, donde cada configuración es un vector $\mathbf{X}$ formado por un total de $n = 15\times15 = 225$ componentes. Adicionalmente, también se deben almacenar las dos frecuencias de clase $freq(\mathbf{X},Y = 0)$ y $freq(\mathbf{X},Y = 1)$ de cada configuración. Dado que el entrenamiento de los clasificadores involucra el manejo de números reales, entonces es necesario representar computacionalmente tanto a las componentes de las configuraciones de ventana como a sus frecuencias mediante números de doble precisión; es decir, usando 8 bytes por elemento. En estas condiciones, sólo para el almacenamiento del conjunto de ejemplos de entrenamiento se requiere un mínimo de $1425.9\times8 = 11.40$ GB de memoria RAM. A esto se debe aumentar el espacio de memoria requerido para la tarea de optimización. Si la capacidad de memoria disponible es limitada, por ejemplo 8 GB, entonces el problema se vuelve intratable desde el punto de vista práctico. Por el contrario, si los recursos computacionales son suficientes a nivel de memoria y capacidad de cálculo, por ejemplo utilizando un clúster de computadores, entonces no existe una limitante práctica. Esto es debido a que el entrenamiento de los clasificadores aquí utilizados puede ser paralelizado.

La solución propuesta en esta tesis para resolver el problema computacional que involucra el manejo de grandes conjuntos de entrenamiento y clasificadores complejos consiste en el diseño de un *ensamble* o conjunto de varios de clasificadores más simples [Bishop, 2005], [Mohri *et al.*, 2012]. Esto es, se propone diseñar tantos clasificadores como pares de imágenes, o volúmenes, se dispongan para la etapa de entrenamiento. A cada clasificador que forma el ensamble se lo denomina *clasificador de base*. Esta solución implica el uso de clasificadores de base cuya complejidad, o dimensión VC_1, debe estar acorde con la cantidad de configuraciones que se pueden extraer a partir de cada par de imágenes de entrenamiento. Evidentemente, para evitar el overfitting, el valor de la dimensión VC_1 de cada clasificador de base debe ser menor que la dimensión VC_0 del clasificador que se podría diseñar si se pudieran utilizar simultáneamente todas las configuraciones de ventana de todas las imágenes de entrenamiento.

Si el clasificador óptimo no está contenido dentro del espacio de búsqueda del ensamble, entonces existe un costo de restricción. Sin embargo, es usual que la frontera de decisión del ensamble sea más compleja que la frontera de decisión de cada clasificador de base [Oza y Tumer, 2008], [Krogh y Vedelsby, 1995]. Esto especialmente para el caso donde la correlación entre los errores de los clasificadores de base es baja (Ejemplo 5.6). Por lo tanto, la capacidad de aproximación del ensamble puede ser mayor que la capacidad de los clasificadores de base. Es decir, con un ensamble se puede implementar un clasificador complejo diseñando únicamente clasificadores simples. El diseño de clasificadores simples, a su vez, involucra un bajo costo de estimación y también bajo costo computacional que es lo que se persigue en esta sección.

Ejemplo 5.6. Sea el caso donde se diseñan 3 clasificadores lineales basados en regresión logística para un espacio de configuraciones bidimensional. Para el diseño de los clasificadores se consideran tres conjuntos de ejemplos de entrenamiento diferentes. Las fronteras de decisión de cada clasificador se muestran en la Figura 5.6. Cada clasificador etiqueta un vector de características con $Y = 1$ si dicho vector se localiza por encima de la frontera de decisión. Nótese que las predicciones de los clasificadores difieren en ciertas regiones del espacio de características. La respuesta o etiqueta del ensamble para un vector dado se obtiene mediante una votación entre las etiquetas calculadas usando los tres clasificadores de base. Esto resulta en un ensamble de clasificadores que particiona el

espacio de características, en dos regiones, mediante una función no lineal construida a partir de tres fronteras lineales.

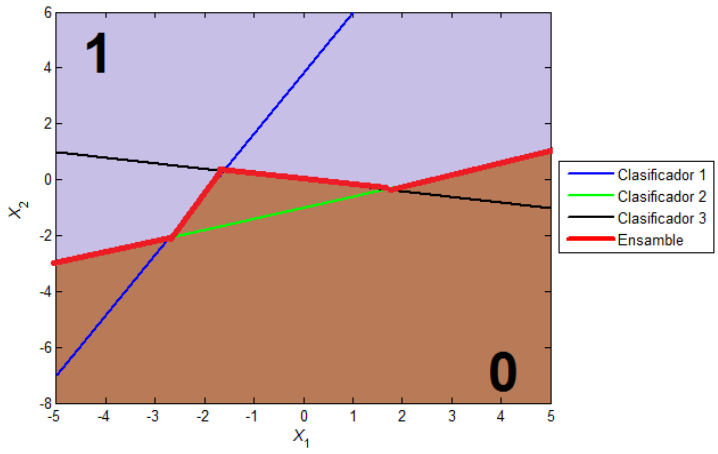


Figura 5.6. Ilustración de la frontera de decisión resultante al combinar, en un ensamble, tres clasificadores lineales de base mediante una votación entre sus predicciones.

En esta sección se analiza el error de aproximación de un ensamble (Figura 5.7) en función de los errores de aproximación de sus clasificadores de base. Las probabilidades condicionales del ensamble y de los clasificadores de base, dada una configuración $\mathbf{X}$, se denotan mediante $\mathbb{P}_{\mathrm{E}}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ y $\mathbb{P}_1(Y = 1|\mathbf{X};\boldsymbol{\beta}),\ldots,\mathbb{P}_M(Y = 1|\mathbf{X};\boldsymbol{\beta})$, respectivamente. El valor de la probabilidad condicional $\mathbb{P}_{\mathrm{E}}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ para una configuración $\mathbf{X}$ se calcula mediante

$$\mathbb{P}_{\mathrm{E}}(Y=1|\,\mathbf{X};\boldsymbol{\beta}) = \sum_{i=1}^{M}(Pe_i(\mathbf{X};\boldsymbol{\beta})\mathbb{P}_i(Y=1|\,\mathbf{X};\boldsymbol{\beta}))\,, \tag{5.15}$$

donde $Pe_i(\mathbf{X};\boldsymbol{\beta}) \geq 0$ denota el peso con el que la probabilidad condicional $\mathbb{P}_i(Y = 1|\mathbf{X};\boldsymbol{\beta})$, dado el vector $\mathbf{X}$, contribuye a la predicción del ensamble. Adicionalmente, para cada configuración $\mathbf{X}\in\boldsymbol{\mathcal{X}}$ se considera la restricción para los pesos $\sum_{i=1}^{M} Pe_i(\mathbf{X};\boldsymbol{\beta}) = 1$, la cual permite la realización del siguiente análisis. Nótese que la definición de la ecuación 5.15 es válida para cualquier modelo de clasificación que permita estimar $\mathbb{P}(Y|\mathbf{X})$.

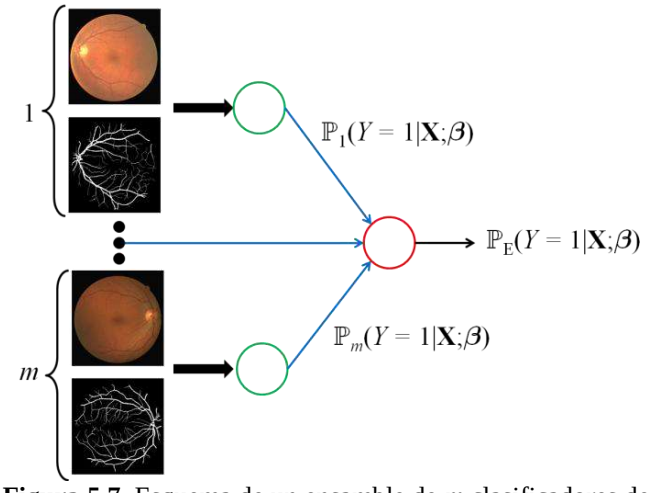


Figura 5.7. Esquema de un ensamble de m clasificadores de base para la segmentación de vasos sanguíneos en imágenes oculares.

La calidad de la aproximación de $g(\mathbf{X}) = \mathbb{P}(Y = 1|\mathbf{X})$ mediante el ensamble de clasificadores $h_E(\mathbf{X};\boldsymbol{\beta}) = \mathbb{P}_E(Y = 1|\mathbf{X};\boldsymbol{\beta})$ está dada por el valor de $\varepsilon_E = \mathbb{E}[(h_E(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X}))^2]$, donde la esperanza se calcula con respecto a la distribución marginal de $\mathbf{X}$. La varianza de las predicciones de los clasificadores de base con respecto a la predicción del ensamble está dada por

$$Var_E = \mathbb{E}[\sum_{i=1}^{M} Pe_i(\mathbf{X};\boldsymbol{\beta})\left(h_i(\mathbf{X};\boldsymbol{\beta}) - h_E(\mathbf{X};\boldsymbol{\beta})\right)^2], \tag{5.16}$$

donde $h_i(\mathbf{X};\boldsymbol{\beta}) = \mathbb{P}_i(Y = 1|\mathbf{X};\boldsymbol{\beta})$. La ecuación 5.16 puede ser expresada como

$$Var_E = \mathbb{E}[\sum_{i=1}^{M} Pe_i(\mathbf{X};\boldsymbol{\beta})\left(h_i(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X})\right)^2 + A], \tag{5.17}$$

donde

$$A = \sum_{i=1}^{M} Pe_i(\mathbf{X};\boldsymbol{\beta})[(h_E(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X}))^2 - 2(h_i(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X}))(h_E(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X}))]. \tag{5.18}$$

Operando la ecuación 5.18 se puede demostrar que

$$A = -\left(h_E(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X})\right)^2. \tag{5.19}$$

Combinando las ecuaciones 5.16, 5.17, y 5.19 se tiene que

$$\begin{aligned}\varepsilon_E = \mathbb{E}[\left(h_E(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X})\right)^2] = \sum_{i=1}^{M} \mathbb{E}[Pe_i(\mathbf{X};\boldsymbol{\beta})\left(h_i(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X})\right)^2] \\ - \sum_{i=1}^{M} \mathbb{E}[Pe_i(\mathbf{X};\boldsymbol{\beta})\left(h_i(\mathbf{X};\boldsymbol{\beta}) - h_E(\mathbf{X};\boldsymbol{\beta})\right)^2]\end{aligned}. \tag{5.20}$$

ε_E = *Error de aproximación del ensamble* =

Error de aproximación de los clasificadores de base - *Varianza*

Esto implica que el grado de aproximación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ usando la probabilidad condicional del ensamble, $\mathbb{P}_E(Y = 1|\mathbf{X};\boldsymbol{\beta})$, es función tanto de la capacidad de aproximación de los clasificadores de base como de su nivel de varianza con respecto a la predicción del ensamble. A mayor capacidad de aproximación y mayor nivel de varianza de los clasificadores de base, mejor será la aproximación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ realizada mediante el ensamble $\mathbb{P}_E(Y = 1|\mathbf{X};\boldsymbol{\beta})$.

Un aumento de la capacidad de aproximación de los clasificadores de base se consigue incrementando la complejidad, o dimensión VC, del modelo de clasificación utilizado para su diseño. Para esta parte se debe tener en cuenta el número de ejemplos de entrenamiento disponibles para evitar el overfitting. Por otra parte, en problemas de PDI es usual que los niveles de intensidad, contraste, textura, posición, y orientación de los objetos a segmentar, entre otras propiedades, varíen entre cada par de imágenes de entrenamiento. Por lo tanto, dado que cada clasificador de base se ajusta a las características del par de imágenes a partir del cual fue diseñado, es razonable suponer que su varianza sea alta. En estas condiciones, se puede esperar que los resultados del ensamble no difieran significativamente de los resultados que se podrían alcanzar diseñando un único clasificador más complejo que los clasificadores de base y utilizando simultáneamente todo el conjunto de imágenes de entrenamiento.

El error promedio de los clasificadores de base está dado por

$$\varepsilon_{AV} = (1/M)\sum_{i=1}^{M}\mathbb{E}[(h_i(\mathbf{X};\boldsymbol{\beta}) - g(\mathbf{X}))^2]. \quad (5.21)$$

Si en la ecuación 5.20 se reemplaza $Pe_i(\mathbf{X};\boldsymbol{\beta}) = 1/M$, $\forall\mathbf{X}\in\boldsymbol{\mathcal{X}}$ e $i = 1,\ldots,M$, se tiene que

$$\varepsilon_E = \varepsilon_{AV} - (1/M)\sum_{i=1}^{M}\mathbb{E}[(h_i(\mathbf{X};\boldsymbol{\beta}) - h_E(\mathbf{X};\boldsymbol{\beta}))^2]. \quad (5.22)$$

Lo anterior implica que $\varepsilon_E \leq \varepsilon_{AV}$; es decir, el error de aproximación del ensamble analizado anteriormente mejora o, a lo sumo, es igual al promedio de los errores de aproximación de los clasificadores de base. Si se define el clasificador del ensamble como $\psi_E(\mathbf{X}) = 1$ para cuando $\mathbb{P}_E(Y = 1|\mathbf{X};\boldsymbol{\beta}) \geq 0.5$ y $\psi_E(\mathbf{X}) = 0$ para el caso contrario, entonces se puede mostrar que $\varepsilon(\psi_E) - \varepsilon(\psi_{opt}) \leq 2\sqrt{\mathbb{E}[(\mathbb{P}_E(Y=1|\mathbf{X};\boldsymbol{\beta}) - \mathbb{P}(Y=1|\mathbf{X}))^2]}$ [Devroye *et al.*, 1996]. Los términos $\varepsilon(\psi_E)$ y $\varepsilon(\psi_{opt})$ denotan los errores de clasificación del ensamble y el clasificador óptimo para la distribución conjunta de los pares $(\mathbf{X},Y)$, respectivamente. La esperanza se calcula con respecto a la distribución marginal de los vectores $\mathbf{X}$. Sin ninguna sorpresa, lo anterior implica que la cota superior para la diferencia $\varepsilon(\psi_E) - \varepsilon(\psi_{opt})$ disminuye a medida que el ensamble mejora la aproximación de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$.

Para una configuración cualquiera $\mathbf{X}$ y un ensamble de clasificadores, es posible que las predicciones de unos clasificadores de base sean más informativas, o menos ambiguas, que las predicciones de los demás clasificadores. En este caso, para el cálculo de la probabilidad condicional del ensamble $\mathbb{P}_E(Y = 1|\mathbf{X};\boldsymbol{\beta})$ dado un $\mathbf{X}$ cualquiera, resulta apropiado asignar un mayor peso a aquellos clasificadores de base que predicen $\mathbb{P}(Y = 1|\mathbf{X})$, dado $\mathbf{X}$, sin ambigüedad. En esta tesis se propone definir el peso, $Pe_i(\mathbf{X};\boldsymbol{\beta})$, para la predicción realizada por el i-ésimo clasificador de base, $\mathbb{P}_i(Y = 1|\mathbf{X};\boldsymbol{\beta})$, dada una configuración arbitraria $\mathbf{X}$, mediante la siguiente expresión:

$$Pe_i(\mathbf{X};\boldsymbol{\beta}) = \frac{1 - \mathbb{H}_i(\mathbf{X};\boldsymbol{\beta})}{M - \sum_{i=1}^{M}\mathbb{H}_i(\mathbf{X};\boldsymbol{\beta})}, \quad (5.23)$$

donde $\mathbb{H}_i(\mathbf{X};\boldsymbol{\beta})$ es la Entropía de Shannon [Bishop, 2005] dada por

$$\mathbb{H}_i(\mathbf{X};\boldsymbol{\beta}) = -\sum_{j=0}^{1}\mathbb{P}_i(Y = j|\mathbf{X};\boldsymbol{\beta})log_2\mathbb{P}_i(Y = j|\mathbf{X};\boldsymbol{\beta}), \quad (5.24)$$

con $0log_2(0) = 0$.

En general, para un problema de clasificación con $c > 2$ clases, la probabilidad del ensamble para la clase $Y = k$, dada una configuración arbitraria $\mathbf{X}$, se calcula como

$$\mathbb{P}_E(Y = k|\mathbf{X};\boldsymbol{\beta}) = \sum_{i=1}^{M}(Pe_i(\mathbf{X};\boldsymbol{\beta})\mathbb{P}_i(Y = k|\mathbf{X};\boldsymbol{\beta})), \quad (5.25)$$

con $k = 0,\ldots,(c - 1)$ y $Pe_i(\mathbf{X};\boldsymbol{\beta})$ está dado por la ecuación 5.23. Para este caso general de clasificación multiclase, la entropía de Shannon del i-ésimo clasificador se calcula mediante

$$\mathbb{H}_i(\mathbf{X};\boldsymbol{\beta}) = -\sum_{j=0}^{c-1}\mathbb{P}_i(Y = j|\mathbf{X};\boldsymbol{\beta})log_c\mathbb{P}_i(Y = j|\mathbf{X};\boldsymbol{\beta}), \quad (5.26)$$

con $0log_c(0) = 0$. Para el i-ésimo clasificador de base y una configuración arbitraria $\mathbf{X}$, la entropía de Shannon $\mathbb{H}_i(\mathbf{X};\boldsymbol{\beta})$ es máxima cuando $\mathbb{P}_i(Y = k|\mathbf{X};\boldsymbol{\beta}) = 1/c$, $k = 0,\ldots,(c - 1)$. Para este caso, el valor del peso $Pe_i(\mathbf{X};\boldsymbol{\beta})$ es nulo: $Pe_i(\mathbf{X};\boldsymbol{\beta}) = 0$. Por lo tanto, el peso con el que un clasificador de base contribuye a la predicción del ensamble, dada una configuración arbitraria, es bajo si su predicción no es informativa o es ambigua. Para el caso donde $\mathbb{P}_1(Y|\mathbf{X};\boldsymbol{\beta}) = \ldots = \mathbb{P}_M(Y|\mathbf{X};\boldsymbol{\beta})$, se tiene que $Pe_i(\mathbf{X};\boldsymbol{\beta}) = 1/M$, con $i = 1,\ldots,M$.

Ejemplo 5.7. En la Tabla 5.3 se presentan las probabilidades condicionales de las tres clases a las que puede pertenecer una configuración de ventana $\mathbf{X}$. Se asume que estas probabilidades fueron calculadas a partir de $M = 3$ clasificadores de base diseñados usando un mismo modelo de clasificación, pero entrenados con diferentes ejemplos. En la quinta columna de esta tabla se presentan los pesos con que cada clasificador influye en las predicciones del ensamble. Las predicciones de las probabilidades condicionales de los clasificadores 1 y 3 son mayores para la clase $Y = 2$. El clasificador 2 predice uniformemente las probabilidades condicionales para las 3 clases en cuestión. Es esta la razón por la que su contribución, o peso, en las predicciones del ensamble es nula.

Tabla 5.3. Predicciones de un ensamble y sus clasificadores de base.

	$\mathbb{P}_i(Y = 0\|\mathbf{X};\boldsymbol{\beta})$	$\mathbb{P}_i(Y = 1\|\mathbf{X};\boldsymbol{\beta})$	$\mathbb{P}_i(Y = 2\|\mathbf{X};\boldsymbol{\beta})$	Peso, $Pe_i(\mathbf{X};\boldsymbol{\beta})$
Clasificador 1	0.10	0.20	0.70	0.67
Clasificador 2	0.33	0.33	0.33	0
Clasificador 3	0.20	0.20	0.60	0.33
Ensamble	0.13	0.20	0.67	

5.7. Clasificación Multiclase y Redes Neuronales Tipo Feed Forward

Una vez analizada la versión más simple de una red neuronal tipo feed-forward, como es el caso de la regresión logística, y los problemas de diseño balanceado y ensambles de clasificadores, en esta sección se considera el diseño de redes neuronales más complejas y problemas de clasificación multiclase. El incremento de la complejidad, o dimensión VC, de una red neuronal hace que su capacidad de aproximación mejore. Para esta sección se considera una arquitectura de red neuronal tipo feed-forward de tres capas: entrada, oculta, y salida (Figura 5.8).

Sea la arquitectura de red neuronal tipo feed-forward de la Figura 5.8. En este caso la capa de entrada recibe un vector $\mathbf{X}$ de n componentes, $X_1,\ldots,X_n$. La capa oculta está formada por m neuronas $S_1^{(1)} \ldots, S_m^{(1)}$ con funciones de transferencia *logsig*: $f^{(1)}(z) = 1/(1 + exp(-z))$. La capa de salida está formada por c neuronas $S_1^{(2)}, \ldots, S_c^{(2)}$. Para el caso donde $c = 2$, clasificación binaria, sólo es necesario utilizar una única neurona de salida. Esto es debido a que $\mathbb{P}(Y = 0|\mathbf{X}) = 1 - \mathbb{P}(Y = 1|\mathbf{X})$. En este caso se puede utilizar también la función *logsig* para la neurona de la capa de salida. Para $c > 2$, la respuesta $\mathbb{P}(Y = i|\mathbf{X};\boldsymbol{\beta})$ de la i-ésima neurona de la capa de salida predice la probabilidad condicional de que un vector $\mathbf{X}$ pertenezca a la clase $i \in \mathcal{Y}$, con $\mathcal{Y} = \{1,2,\ldots,c\}$. Por conveniencia del análisis de esta sección se consideran las etiquetas $1,\ldots,c$ en lugar de $0,\ldots,(c - 1)$.

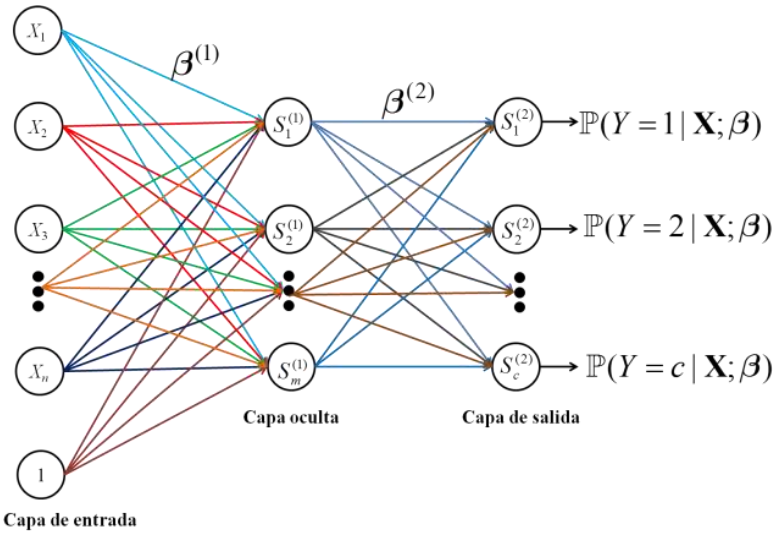


Figura 5.8. Arquitectura de red una neuronal tipo feed-forward para clasificación multiclase con tres capas: entrada $S^{(0)}$, oculta $S^{(1)}$, y salida $S^{(2)}$.

El valor de $\mathbb{P}(Y = i|\mathbf{X};\boldsymbol{\beta})$, que es un estimado de $\mathbb{P}(Y = i|\mathbf{X})$ dado el vector $\mathbf{X}$, se calcula mediante funciones de transferencia *softmax* [Bridle, 1990], [Bishop, 2005]:

$$\mathbb{P}(Y=i\,|\,\mathbf{X};\boldsymbol{\beta}) = \frac{exp\left(\sum_{j=1}^{m}\beta_{i,j}^{(2)}R_j^{(1)}\right)}{\sum_{k=1}^{c}exp\left(\sum_{j=1}^{m}\beta_{k,j}^{(2)}R_j^{(1)}\right)}, \tag{5.27}$$

donde $R_j^{(1)} = f^{(1)}\left(T_j(\mathbf{X};\boldsymbol{\beta}_j^{(1)})\right)$ denota la respuesta de la neurona $S_j^{(1)}$ de la capa oculta para el vector de entrada $\mathbf{X}$. La función de transferencia softmax permite generalizar la definición de regresión logística para un problema de clasificación con más de dos clases.

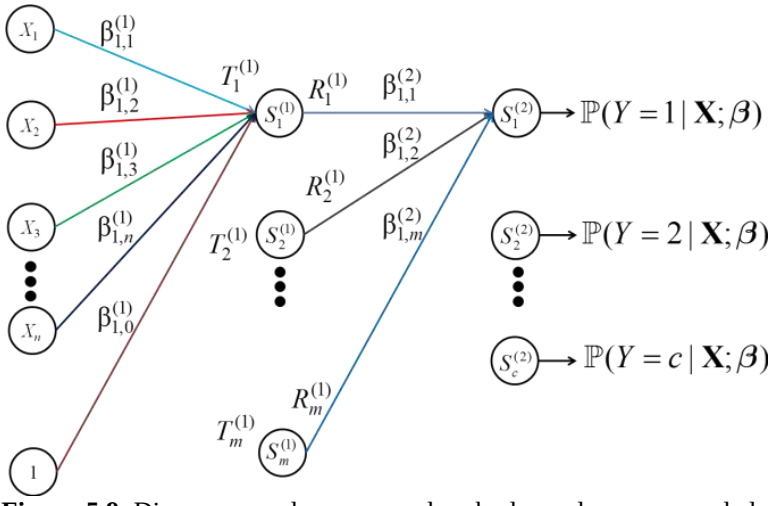


Figura 5.9. Diagrama con los pesos y el umbral para las neuronas de la parte superior de la capa oculta $S^{(1)}$ y la capa de salida $S^{(2)}$.

En la Figura 5.9 se puede observar que $\beta_{i,j}^{(2)}$ denota el peso de la conexión entre las neuronas $S_j^{(1)}$ y $S_i^{(2)}$ de la capa oculta y la capa de salida, respectivamente. La respuesta de la neurona $S_j^{(1)}$ de la capa oculta para un vector de entrada $\mathbf{X}$ se calcula como

$$T_j(\mathbf{X};\boldsymbol{\beta}_j^{(1)}) = \sum_{l=1}^{n}\beta_{j,l}^{(1)}X_l + \beta_{j,0}^{(1)}, \tag{5.28}$$

donde $\beta_{j,l}^{(1)}$ denota el peso de la conexión entre la entrada $S_l^{(0)} = X_l$ y la neurona $S_j^{(1)}$ de la capa oculta, con $l = 1,\ldots,n$ y $j = 1,\ldots,m$. $\beta_{j,0}^{(1)}$ denota el umbral de la neurona $S_j^{(1)}$. El vector $\boldsymbol{\beta}_j^{(1)} = (\beta_{j,0}^{(1)},\ldots,\beta_{j,n}^{(1)})$ contiene los parámetros de la neurona $S_j^{(1)}$. Para PDI, la capa oculta de la red de la Figura 5.9 puede ser vista como un banco de filtros espaciales (Sección 5.4). Es decir, los pesos de la neurona $S_j^{(1)}$ de la capa oculta pueden ser interpretados como los coeficientes de una máscara de convolución o correlación de imágenes que busca un determinado patrón o configuración de ventana que se puede obtener con la ecuación 5.10.

Las funciones de activación *logsig* de la capa oculta miden el nivel de correlación entre una configuración de entrada $\mathbf{X}$ y los coeficientes de la máscara que se puede implementar con los pesos de cada neurona de la capa oculta. En función de las respuestas de las m neuronas de la capa oculta, se obtiene un vector $\mathbf{Z} = (Z_1,\ldots,Z_m)$ dada una configuración de entrada $\mathbf{X}$. El término $Z_i \in \mathbf{Z}$ denota la respuesta del filtro (neurona) $S_i^{(1)}$, en un rango [0,1], para una entrada $\mathbf{X}$, con $i = 1,\ldots,m$. La capa de salida está compuesta por c clasificadores lineales basados en el modelo de regresión logística. Cada clasificador de la capa de salida predice la probabilidad condicional de clase $\mathbb{P}(Y|\mathbf{X};\boldsymbol{\beta})$ para el píxel donde se observó $\mathbf{X}$ a partir del vector de respuestas del banco filtros de la capa oculta, $\mathbf{Z}$. La etiqueta para $\mathbf{X}$ está dada por la clase de mayor probabilidad condicional estimada. Nótese que $\sum_{i=1}^{c} \mathbb{P}(Y = i \mid \mathbf{X};\boldsymbol{\beta}) = 1$, lo cual implica asumir que las clases para una configuración $\mathbf{X}$ son mutuamente excluyentes. Dado que cada píxel de una imagen toma un único valor, o un único vector para el caso de las imágenes multicanal, este criterio de exclusividad entre las clases es válido para problemas de PDI.

En la práctica, para el entrenamiento de una red neuronal, se utiliza una tabla de frecuencias $\boldsymbol{\mathcal{D}} = \{(\mathbf{X}_i, freq(\mathbf{X}_i, Y = 1),\ldots, freq(\mathbf{X}_i, Y = c))\}$, con $i = 1,\ldots,N$. Esta tabla está formada por N configuraciones de ventana con sus correspondientes frecuencias de clase. Las configuraciones de ventana y sus frecuencias son obtenidas a partir de un par de imágenes de entrenamiento usando una ventana W formada por n píxeles. El ajuste de los parámetros de la red neuronal se puede realizar maximizando la siguiente función de verosimilitud:

$$\mathbb{P}(\boldsymbol{\mathcal{D}}|\boldsymbol{\beta}) = \prod_{i=1}^{N}\prod_{j=1}^{c} \mathbb{P}(Y = j \mid \mathbf{X}_i;\boldsymbol{\beta})^{freq(\mathbf{X}_i, Y=j)} . \tag{5.29}$$

Maximizar la ecuación 5.29 equivale a encontrar un mínimo local para la función de costo

$$J(\boldsymbol{\beta}) = -(1/p)\sum_{i=1}^{N}\sum_{j=1}^{c} freq(\mathbf{X}_i, Y = j) ln(\mathbb{P}(Y = j \mid \mathbf{X};\boldsymbol{\beta})) , \tag{5.30}$$

donde $p = \sum_{i=1}^{N}\sum_{j=1}^{c} freq(\mathbf{X}_i, Y = j)$. Si para el entrenamiento de la red neuronal se requiere realizar un balanceo artificial de frecuencias, entonces en la ecuación 5.30 se puede utilizar $freq_{Bal}(\mathbf{X}_i, Y = j)$ en lugar de $freq(\mathbf{X}_i, Y = j)$ (ecuación 5.14) para cada $\mathbf{X}_i$, con i = 1,…,N.

Para problemas de clasificación con más de dos clases ($c > 2$), en la capa de salida de la arquitectura de red presentada en la Figura 5.9 se tienen tantas neuronas como número de clases. Si se analiza para este caso la función de transferencia softmax utilizada para calcular $\mathbb{P}(Y|\mathbf{X};\boldsymbol{\beta})$ (ecuación 5.27), se puede notar que el conjunto de parámetros de las neuronas de la capa de salida son redundantes. Esto significa que, tanto $(\boldsymbol{\beta}_1^{(2)},\ldots,\boldsymbol{\beta}_c^{(2)})$

como $((\boldsymbol{\beta}_1^{(2)} - \boldsymbol{\gamma}),\ldots,(\boldsymbol{\beta}_c^{(2)} - \boldsymbol{\gamma}))$ resultan en idénticas funciones para $\mathbb{P}(Y|\mathbf{X};\boldsymbol{\beta})$, donde el vector $\boldsymbol{\beta}_i^{(2)} = (\beta_{i,1}^{(2)},...,\beta_{i,m}^{(2)})$ contiene los pesos para la neurona $S_i^{(2)}$ de la capa de salida, con $i = 1,\ldots,c$ y $\boldsymbol{\gamma} = (\gamma_1,\ldots,\gamma_m) \in \mathbb{R}^m$ es un vector cualquiera. Una solución para este problema consiste en igualar a 0 todos los pesos de cualquier neurona de la capa oculta. Sin embargo, esto complica la implementación del algoritmo de retro-propagación del error para el cálculo del gradiente de la función $J(\boldsymbol{\beta})$. En su lugar, lo más conveniente es manejar esta redundancia mediante el uso de alguna técnica de *regularización* [Bishop, 2005].

Si se utiliza *regularización por decaimiento de pesos*, entonces la función de costo que se debe optimizar para el entrenamiento de la red neuronal está dada por

$$J_R(\boldsymbol{\beta}) = J(\boldsymbol{\beta}) + (\lambda/2)Reg(\boldsymbol{\beta}), \tag{5.31}$$

donde $Reg(\boldsymbol{\beta}) = \sum_{i=1}^{m}\sum_{j=1}^{n}(\beta_{i,j}^{(1)})^2 + \sum_{i=1}^{c}\sum_{j=1}^{m}(\beta_{i,j}^{(2)})^2$ y $\lambda \geq 0$ es un parámetro que controla el grado de regularización en el entrenamiento de la red neuronal. El valor de λ se debe fijar como parte del diseño de la arquitectura de red. Valores de λ grandes, por ejemplo $\lambda > 1$, hacen que la optimización se concentre en la minimización de $Reg(\boldsymbol{\beta})$, lo cual resulta en valores pequeños para los pesos de la capa oculta y la capa de salida [Duda *et al.*, 2001], [Abu-Mostafa *et al.*, 2012], [Mohri *et al.*, 2012]. Esto implica que, durante el entrenamiento de la red neuronal, se favorece a las funciones (fronteras de decisión) con variación suave y se penaliza a aquellas funciones con variaciones bruscas. Para cuando $\lambda = 0$ no existe regularización y la función de costo de la ecuación 5.31 es idéntica a la de la ecuación 5.30. Para valores de λ pequeños, por ejemplo $\lambda << 1$, la optimización prioriza la minimización de $J(\boldsymbol{\beta})$ sobre el término de regularización $Reg(\boldsymbol{\beta})$.

La regularización por decaimiento de pesos no solamente resuelve el problema de redundancia de parámetros, sino que también disminuye el riesgo de overfitting. Esto sucede, especialmente, cuando el número de neuronas en la capa oculta es grande (red neuronal compleja) y se disponen de pocos ejemplos de entrenamiento. Para la regularización no se toman en cuenta los umbrales de las neuronas de la capa oculta. Esto se debe a que su regularización, usualmente, no ejerce influencia sobre la forma de la frontera de decisión que se implementa mediante una red neuronal [Bishop, 2005].

5.8. Preprocesamiento de Imágenes y Configuraciones de Ventana

Para mejorar el desempeño de los clasificadores, como parte de su diseño, se puede aplicar un preprocesamiento tanto a las imágenes del conjunto de entrenamiento como a las configuraciones de ventana. El preprocesamiento a nivel de imágenes puede consistir en la aplicación de transformaciones para realce de contraste, reducción del nivel de brillo, filtrado de ruido, deblurring, entre otras. Si las imágenes a procesar son imágenes color, se puede también aplicar una transformación para obtener una imagen en escala de grises con el objetivo de simplificar el diseño de los clasificadores. En todos estos casos, se debe tener presente que las transformaciones aplicadas al conjunto de imágenes de entrenamiento son parte de la estructura del clasificador. Por lo tanto, dichas transformaciones deben ser también aplicadas a las imágenes de testeo utilizando la misma secuencia y parámetros que

fueron usados en el conjunto de entrenamiento. Así mismo, el preprocesamiento debe ser realizado de tal modo de no eliminar información importante para la etapa de clasificación.

A nivel de configuraciones de ventana, el preprocesamiento puede estar orientado a reducir el nivel de correlación de las intensidades o niveles de gris entre las componentes de una configuración de ventana **X**. La reducción de la correlación entre las variables con las que se diseñan un clasificador contribuye mejorar su poder de discriminación entre las clases [Duda *et al.*, 2001], [Bishop, 2005], [Murphy, 2012]. También se pueden eliminar ciertas componentes de **X** para reducir el costo de diseño o mejorar la estimación de los clasificadores. Este último tipo de preprocesamiento fue analizado en el Capítulo II, Sección 2.8, bajo el criterio de restricciones. Adicionalmente, en reconocimiento de patrones es usual, previo al entrenamiento de un clasificador, el aplicar una normalización de las medias y las varianzas de las características de los ejemplos de entrenamiento a 0 y 1, respectivamente [Bishop, 2006], [Duda *et al.*, 2001], [Mohri *et al.*, 2012]. Sin embargo, debido a la estacionaridad de las configuraciones de ventana, el uso de este tipo de normalización, que implica el cálculo de la media y varianza para cada característica, no representa un gran beneficio práctico. Esto porque, teóricamente, las estadísticas entre las configuraciones de ventana son las mismas.

5.8.1. Aperture

Para el preprocesamiento de las configuraciones de ventana, en esta tesis se propone utilizar el mismo preprocesamiento utilizado para el diseño de filtros aperture (Sección 2.8 del Capítulo II) [Hirata *et al.*, 2000]. En este caso, dada una configuración $\mathbf{X} = (X_1,\ldots,X_n)$, primero se obtiene la mediana $z = median(X_1,\ldots,X_n)$ de sus componentes. Posteriormente, se obtiene el vector $(X_1 - z,\ldots,X_n - z)$. Finalmente, se limita o trunca el rango de cada componente de este vector a un conjunto $K = \{-k,-k + 1,\ldots,0,\ldots,k - 1,k\}$ conteniendo $2k + 1$ niveles de gris, con $k \in \mathbb{Z}^+$ (ecuación 2.26 del capítulo II). Este procedimiento se aplica a cada una de las N configuraciones de ventana que forman el conjunto de ejemplos de entrenamiento de un clasificador. A los operadores diseñados mediante este tipo de preprocesamiento se los denomina filtros aperture.

En esta tesis se evidencia, empíricamente, que el preprocesamiento de los filtros aperture permite reducir la correlación entre las variables que forman las configuraciones de ventana teniendo en cuenta su estacionaridad. Otra ventaja de este preprocesamiento consiste en la disminución del nivel de ruido de cada configuración debido a que se aprovechan las propiedades estadísticas del filtro mediana [Gonzalez y Woods, 2002]. Sin embargo, dado que en este preprocesamiento el número de niveles de gris que forman el conjunto K es, usualmente, menor que el número de niveles de gris de las imágenes observadas, entonces se impone una restricción en el espacio de configuraciones. Esto es, dos o más configuraciones de ventana diferentes pueden resultar en una misma configuración de aperture. Adicionalmente, este tipo de preprocesamiento no es el más indicado para problemas de segmentación donde la intensidad y textura son las características más importantes para discriminar los objetos de interés. En contraste a este defecto, los filtros aperture permiten capturar información de formas siempre y cuando su variación de intensidad esté contenida dentro del conjunto K.

5.8.2. Wavelets

Una alternativa propuesta en esta tesis para la reducción de variables en las configuraciones de ventana es el uso de wavelets. Wavelets es una familia de transformaciones generadas por medio de traslaciones y dilataciones de una función primitiva llamada wavelet madre [Mallat, 1989], [Daubechies, 2004], [Gonzalez y Woods, 2002]. La transformada wavelet discreta (TWD) 2D de Haar de nivel 1 puede ser representada mediante un banco de filtros espaciales, cuyas máscaras se presentan en la Figura 5.10-a. La aplicación de esta transformada de nivel 1 a una imagen u observación de ventana formada por $2^u \times 2^v$ píxeles, con $u,v \in \mathbb{Z}^+$, consiste de la operación de correlación de imágenes seguido de un submuestreo de filas y columnas por un factor de 2 (Figura 5.10-b). Como resultado se obtiene una nueva imagen formada por 4 canales con $2^{u-1} \times 2^{v-1}$ píxeles por cada canal. Estos canales contienen la aproximación y los detalles horizontales, verticales, y diagonales de la imagen original (Figura 5.10-c). La aproximación resulta de la aplicación de un filtro pasabajos y el submuestreo de filas y columnas de la imagen original en un factor de 2. El número de filas y columnas de la imagen a procesar con la TWD de Haar deben ser potencias de 2.

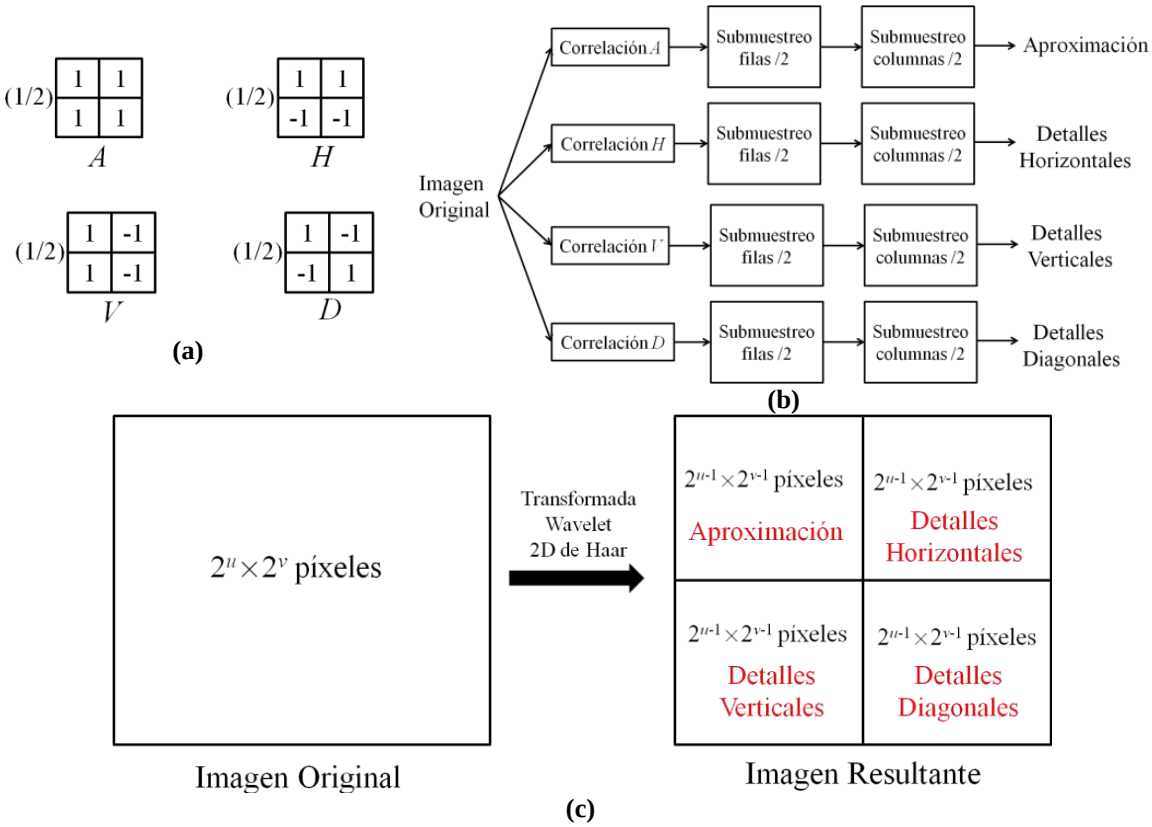


Figura 5.10. Esquema de la transformada Wavelet 2D de Haar de nivel 1. (**a**) Máscaras del banco de filtros espaciales. (**b**) Diagrama de las operaciones de la transformada, y (**c**) Imagen de entrada y resultado.

Para un problema de PDI, los vectores de características para el diseño de los clasificadores pueden estar formados por las intensidades del canal de aproximación de la TWD 2D de Haar de nivel 1 (Figura 5.11). En este caso se pueden descartar todos los detalles si éstos contienen un elevado nivel de ruido o si alguno de ellos no es importante para la discriminación entre las clases. Otra alternativa consiste en utilizar sólo la información de los tres canales de detalles o también se puede usar uno o dos canales de detalles más el

canal de aproximación. En cualquiera de estos casos, la reducción de la cantidad de información utilizada para el diseño de los clasificadores involucra un costo de restricción. Sin embargo, esta forma de reducción de características le permite al usuario tener control sobre el tipo de información que se conserva y la que se descarta durante el diseño de los clasificadores. Adicionalmente, esta transformada permite el uso de ventanas de gran tamaño compensando de esta manera la información que se descarta al no usar todos sus canales.

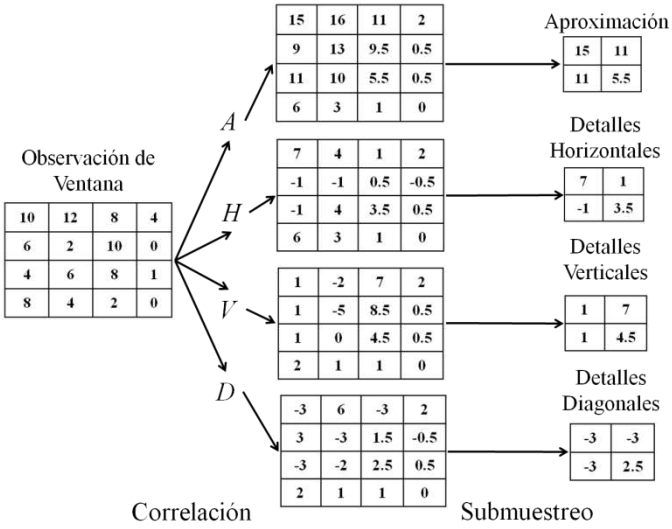


Figura 5.11. Resultado de aplicar la TWD 2D de Haar de nivel 1 a una configuración de ventana de 4×4 píxeles.

5.9. Experimentos

En esta sección se presentan tres casos de aplicación de las nuevas propuestas desarrolladas en las secciones anteriores a problemas de segmentación de imágenes médicas. Los problemas abordados usando el diseño automático de W-operadores son la segmentación de vasos sanguíneos [Benalcázar *et al.*, 2011; Benalcázar *et al.*, 2012; Benalcázar *et al.*, 2014b; Benalcázar *et al.*, 2014c] y exudados en imágenes oculares [Benalcázar *et al.*, 2013], y la segmentación de la glándula prostática y su zona periférica en imágenes de resonancias magnéticas [Benalcázar *et al.*, 2014a]. Las imágenes y volúmenes provienen de bases de datos públicas, las cuales se indican en los detalles de cada experimento.

5.9.1. Segmentación de Vasos Sanguíneos en Imágenes Oculares

La segmentación de vasos sanguíneos en imágenes oculares consiste en una partición del dominio de dichas imágenes en dos subconjuntos disjuntos que son: el árbol arterial retiniano y el fondo de la imagen formado por el contorno retiniano, el disco óptico y, en ciertos casos, algunas patologías como sangrados, exudados, y cambios de pigmentación del epitelio. Este proceso permite extraer características importantes de los vasos sanguíneos como son: longitud, ancho, tortuosidad, patrón morfológico del árbol arterial, número de bifurcaciones, entre otras. Estas características son útiles para diagnóstico, búsqueda, tratamiento, y evaluación de varias enfermedades cardiovasculares, y oftalmológicas, entre las cuales están: hipertensión arterial, retinopatía diabética, arterioesclerosis, entre otras [Fraz *et al.*, 2012]. Por otra parte, se ha encontrado que la morfología del árbol vascular retiniano es única para cada persona, por lo cual se puede

utilizar esta característica para la implementación de sistemas de identificación biométrica [Staal *et al.*, 2004], [Dougherty, 2009], [Fraz *et al.*, 2012].

Se deben tener en cuenta varias consideraciones al momento de segmentar las imágenes oculares: (1) los vasos sanguíneos retinales tienen atributos similares a los de las hemorragias en términos de color, brillo, contraste, y textura; (2) los vasos retinales tienen un amplio espectro de grosor; (3) los vasos sanguíneos delgados tienen un bajo nivel de contraste comparado con la retina; (4) las imágenes oculares poseen una alta variabilidad de color, iluminación, y contraste, tanto dentro de una imagen como entre imágenes; y (5) la orientación de los vasos sanguíneos (izquierda-derecha o derecha-izquierda) varía dependiendo del ojo analizado.

Imágenes: Las imágenes utilizadas en este experimento provienen de la base de datos pública DRIVE (Sección 2.3.2 del Capítulo II). Para el diseño de los operadores de imágenes se utilizaron los 20 pares de imágenes del conjunto de entrenamiento; mientras tanto que para evaluar su desempeño se utilizaron los 20 pares del conjunto testeo. Las imágenes oculares de los conjuntos de entrenamiento y testeo tienen un tamaño de 565×584 píxeles. Cada par está formado por una imagen ocular en color y su respectiva imagen binaria conteniendo los vasos sanguíneos segmentados manualmente por expertos. [Staal *et al.*, 2004] Cada canal de las imágenes color tiene un rango de niveles de grises entre 0 y 255.

Preprocesamiento de Imágenes: Para todos los métodos testeados en este experimento se utilizó una transformación de color a niveles de gris. Esta transformación aplicada a una imagen color RGB $\underline{O}$: $\boldsymbol{E} \rightarrow L^3$ permite obtener una imagen en escala de grises O: $\boldsymbol{E} \rightarrow L$ usando la función Ω: $(L^3)^{\boldsymbol{E}} \rightarrow L^{\boldsymbol{E}}$ definida mediante

$$\Omega(\underline{O})(t) = O(t) = round(0.07O_{\mathrm{R}}(t) + 0.9O_{\mathrm{G}}(t) + 0.3O_{\mathrm{B}}(t)), \tag{5.32}$$

donde $\boldsymbol{E} \subset \mathbb{Z}^2$ y $L = \{0,\ldots,255\}$ es un conjunto de niveles de gris. Los coeficientes de esta transformación fueron elegidos tomando en cuenta que, en las imágenes oculares, el canal verde (O_{G}) tiene el mayor nivel de contraste y el menor nivel de ruido entre los vasos sanguíneos y el fondo. Los canales rojo (O_{R}) y azul (O_{B}) tienen mayor nivel de ruido y menor nivel de contraste [Staal *et al.*, 2004] (Figura 5.12-b, -c, y -d). Un ejemplo de la aplicación de esta transformación de color a niveles de gris se presenta en la Figura 5.12-e.

Para el diseño de W-operadores en base a redes neuronales tipo feed-forward usando configuraciones preprocesadas con la TWD 2D de Haar se aplicó un realce de contraste a las imágenes oculares en escala de grises. El realce de contraste también se utilizó para un método se segmentación basado en el umbralamiento local de Otsu. Este procedimiento permite mejorar la calidad de la información del canal de aproximación de esta transformada wavelet. Para esto se utilizó el método de ecualización de histograma adaptativo por regiones con contraste limitado (CLAHE del inglés *contrast-limited adaptive histogram equalization*) [Zuiderveld, 1994]. Para este experimento, el método CLAHE opera en regiones de 19×19 píxeles. El contraste de cada región se realza de forma tal que el histograma resultante de cada región es aproximadamente uniforme o plano. Luego, regiones vecinas se combinan usando interpolación bilineal [Gonzalez y Woods, 2002] para eliminar los bordes introducidos artificialmente. Para evitar la amplificación de

ruido, la mejora de contraste de cada región se limita a 0.0075. Un ejemplo de la aplicación de este realce se contraste se presenta en la Figura 5.12-f.

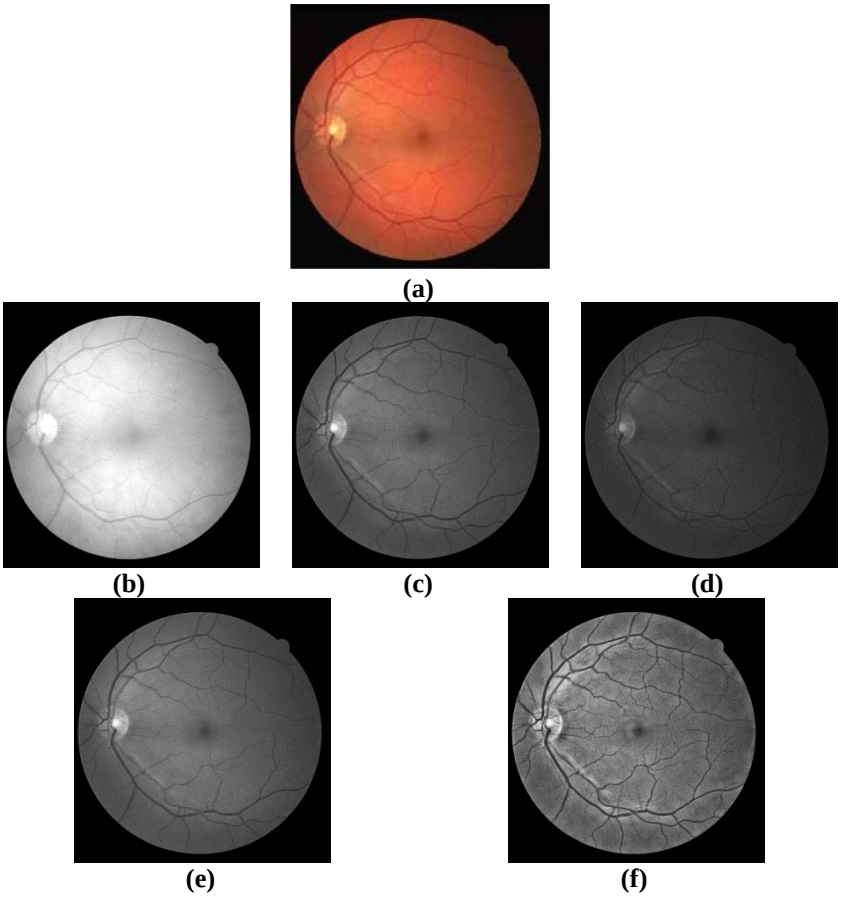


Figura 5.12. Muestra de una imagen ocular de la base de datos DRIVE. (**a**) Imagen color y canales RGB (**b**) rojo, (**c**) verde, y (**d**) azul. Se puede observar que el canal verde es el que tiene el mayor nivel de contraste entre los vasos sanguíneos y el fondo de la imagen. (**e**) Imagen resultante de la transformación de color a niveles de gris de (ecuación 5.32). (**f**) Imagen resultante de aplicar el método para realce de contraste CLAHE a la imagen e.

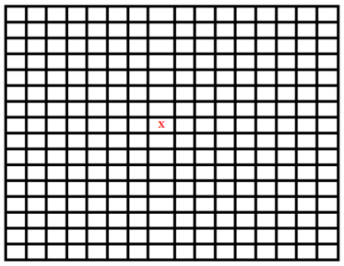
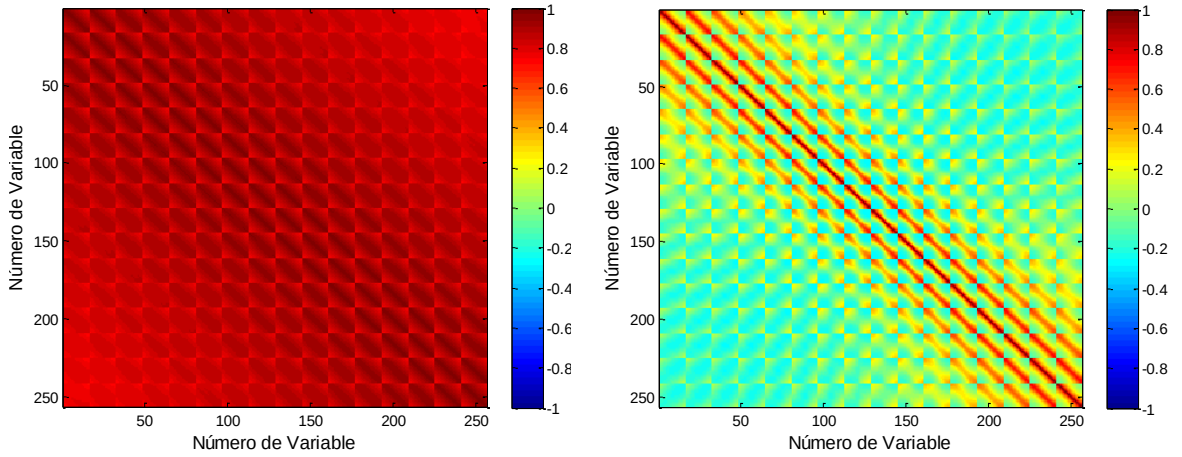


Figura 5.13. Ventana utilizada para el diseño automático de W-operadores para la segmentación de imágenes oculares. El centro se marca con una "x".

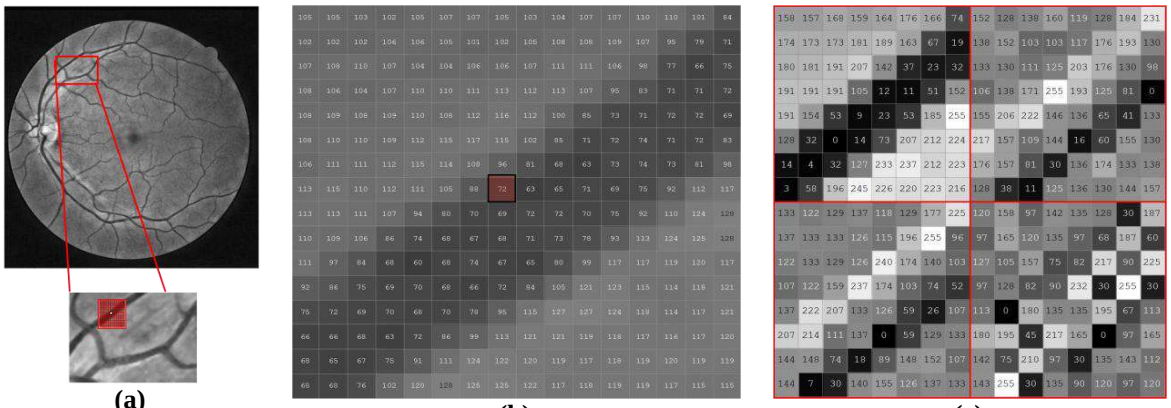
Preprocesamiento de Configuraciones: En este experimento se utilizan ventanas formadas por 16×16 píxeles, en cuyos centros se localiza el píxel a ser procesado (Figura 5.13). Este tamaño de ventana fue seleccionado para contener en su totalidad a los píxeles pertenecientes a los vasos sanguíneos gruesos y delgados de las imágenes del conjunto de entrenamiento. Las configuraciones observadas a través de estas ventanas fueron preprocesdas utilizando aperture y la TWD 2D de Haar de nivel 1, siendo este último tipo

preprocesamiento la razón por la que se utilizaron ventanas de tamaño par. Para el aperture se utilizó un conjunto $K = \{-k,-k+1,\ldots,0,\ldots,k-1,k\}$ de 21 niveles de gris, con $k = 10$. Este conjunto de niveles de gris fue seleccionado teniendo en cuenta los niveles de intensidad de los vasos sanguíneos de las imágenes en escala de grises.



(a) **(b)**

Figura 5.14. Ejemplo de las matrices de correlación de las configuraciones de ventana para la segmentación de los vasos sanguíneos oculares de la base de datos DRIVE. Las configuraciones son obtenidas a partir de las imágenes oculares en escala de grises sin realce de contraste. La matriz de correlación se presenta como un mapa de colores, donde el color de cada píxel indica el nivel de correlación entre las variables de sus coordenadas. (**a**) Matriz de correlación de las configuraciones de ventana. (**b**) Matriz de correlación de las configuraciones de aperture. Comparando los dos mapas de colores se puede observar claramente como el nivel de correlación disminuye al aplicar el preprocesamiento aperture. Las dos matrices están calculadas a partir de 269360 vectores formados por $16 \times 16 = 256$ variables, $\mathbf{X} = (X_1,\ldots,X_{256})$.



(a) **(b)** **(c)**

Figura 5.15. Ilustración de la aplicación de la TWD 2D de Haar de nivel 1 a una observación de ventana. (**a**) Imagen original y zoom del área escaneada con una ventana de 16×16 píxeles. (**b**) Observación de ventana, donde el píxel a ser procesado está encerrado con un cuadrado negro. (**c**) Resultado de la transformada wavelet 2D de Haar de nivel 1 con un rango entre 0 y 255. Nótese como el canal de aproximación (esquina superior izquierda de la imagen c) preserva la forma del vaso sanguíneo.

En la Figura 5.14 se puede observar como el aperture disminuye el nivel de correlación entre las componentes de los vectores de características. Para el caso de la transformada de Haar se utilizó sólo la información del canal de aproximación, descartando la información de todos los canales de detalle. En este último caso, los vectores para el diseño de los clasificadores están formados por 64 características con un rango entre 0 y 255. Se utilizó el canal de aproximación debido a que éste preserva en su mayoría la forma de los vasos sanguíneos (Figura 5.15).

Métodos de Clasificación: En este experimento se testearon 8 métodos de segmentación basados en la clasificación de los píxeles de las imágenes a procesar. Cada clasificador debe predecir la etiqueta para un píxel en función de su configuración de ventana, donde 1 indica vaso sanguíneo y 0 representa fondo. Se diseñaron clasificadores en base a redes neuronales tipo feed-forward entrenadas con configuraciones de ventana (1) sin preprocesamiento (RN-Ventana), y preprocesadas tanto con (2) aperture (RN-Aperture) como aplicando (3) la TWD 2D de Haar (RN-Ventana-Wavelet-A). Se diseñaron también clasificadores con regresión logística (4) sin preprocesamiento de las configuraciones de ventana y con (5) preprocesamiento utilizando aperture. Otro método testeado es (6) SVM entrenado con configuraciones aperture (SVM-Aperture). También se usaron W-operadores basados en (7) multi-resolución piramidal combinada con aperture, y (8) una segmentación rápida basado en el umbralamiento local de Otsu [Otsu, 1979], [Gonzalez y Woods, 2002].

El número de nodos de la capa de entrada de las redes neuronales está dado por la longitud de los vectores de configuraciones. Esto es 256 entradas para RN-Ventana y RN-Aperture, y 64 entradas para RN-Ventana-Wavelet-A. La capa de salida en todos estos casos tiene una única neurona que retorna un estimado $\mathbb{P}(Y = 1|\mathbf{X};\boldsymbol{\beta})$ de la probabilidad condicional $\mathbb{P}(Y = 1|\mathbf{X})$ dada una configuración de ventana $\mathbf{X}$. El número de neuronas de la capa oculta se determinó utilizando un procedimiento de validación cruzada usando 5 pares de imágenes del conjunto de entrenamiento. Este procedimiento es similar al utilizado para el cálculo de los errores en los experimentos presentados en el capítulo anterior. En base a este procedimiento se compararon los resultados de redes neuronales formadas por diferente número de neuronas en la capa oculta. Los mejores resultados para los clasificadores basados en RN-Ventana, RN-Aperture, y RN-Ventana-Wavelet-A se obtuvieron usando 30 neuronas para la capa oculta.

El número de entradas para la regresión logística está dado por la dimensión de los vectores de características; es decir, 256 entradas. Para el diseño de los SVMs se utilizó la función de kernel más simple como es el caso del kernel lineal. Adicionalmente, para su entrenamiento sólo se utilizó sólo un 25% del total de configuraciones utilizadas para los demás métodos. Esto con el objetivo de reducir la gran demanda de memoria y el tiempo de cálculo que implica encontrar los vectores de soporte. Para la segmentación basada en el método de Otsu, primero se normalizó el rango de cada píxel a procesar al intervalo [0,1]. Es conveniente recordar que las imágenes de entrada para este método tienen un realce de contraste utilizando el método CLAHE. Para cada píxel a ser clasificado, se calculó un umbral usando el método de Otsu aplicado a las intensidades de sus vecinos en un entorno de 16×16 píxeles.

Entrenamiento de Clasificadores: Todos los métodos utilizados en este experimento, excepto el método de segmentación rápida de Otsu, involucran una etapa de entrenamiento. Las redes neuronales, la regresión logística, y la multi-resolución piramidal predicen las probabilidades de clase dada la configuración de ventana para un píxel. Los pesos de las redes neuronales y la regresión logística fueron ajustados minimizando las funciones de costo presentadas en su análisis teórico (ecuaciones 5.6 y 5.30). Para el entrenamiento de las redes neuronales no se utilizó regularización debido a que en este caso no existe el problema de redundancia de parámetros de la capa de salida. Los parámetros de las redes

neuronales y la regresión logística fueron ajustados utilizando las configuraciones de ventana obtenidas a partir de cada par de imágenes de entrenamiento. La estimación de las probabilidades condicionales mediante multi-resolución piramidal se realizó utilizando un conjunto de 5 apertures $\{W_1 \times K,...,W_5 \times K\}$, donde $W_1,...,W_5$ son ventanas de 16×16, 8×8, 4×4, 2×2, y 1 píxeles, respectivamente.

La combinación de las predicciones de redes neuronales, regresión logística, y multi-resolución piramidal se realizó utilizando el método de ensamble descrito en la Sección 5.6 del presente capítulo. Debido a que SVM predice directamente la etiqueta de cada píxel, para definir la probabilidad del ensamble para una configuración de ventana se aplicó una votación entre las etiquetas de los 20 clasificadores SVM diseñados.

Testeo y Comparación de Métodos: Cada ensamble de clasificadores diseñado fue evaluado utilizando los 20 pares de imágenes del conjunto de testeo de la base de datos DRIVE. Para la comparación de los métodos testeados se utilizó la curva ROC del inglés *receiver operating characteristic* y el valor de área bajo esta curva, denominado AUC del inglés *area under the curve* [Murphy, 2012]. Las curvas ROC se basan en las estimaciones de las tasas de falsos negativos FNR = $FN/(TP + FN)$ y falsos positivos FPR = $FP/(FP + TN)$. Mientras más cercano sea a 1 el valor de AUC de un clasificador mejor será su desempeño. Valores de AUC cercanos a 0.5 (performance del algoritmo aleatorio) indican una pobre performance de los clasificadores diseñados. El término *TP* es el número de píxeles correctamente clasificados como vasos sanguíneos; *FN* es el número de píxeles incorrectamente clasificados como fondo; *FP* es el número de píxeles incorrectamente clasificados como vasos sanguíneos; y *TN* es el número de píxeles correctamente clasificados como fondo.

Resultados, Comparaciones, y Discusión: La curva de ROC para cada ensamble de clasificadores se obtuvo en base a las estimaciones de (1 - FNR,FPR) = (TPR,FPR). Estos pares fueron calculados umbralando el valor de la probabilidad condicional $\mathbb{P}_E(Y = 1|\mathbf{X};\boldsymbol{\beta})$ en cada valor del conjunto $\tau = \{0,0.05,...,1\}$. TPR denota la tasa de verdaderos positivos. Para el método de segmentación rápida se compararon los umbrales locales de Otsu para cada píxel a clasificar con cada valor del conjunto τ. En la Figura 5.16 se presentan las curvas ROC y los valores de AUC para cada uno de los ensambles testeados. Las curvas ROC fueron obtenidas a partir de los promedios de FNR y FPR calculados sobre las 20 imágenes de testeo.

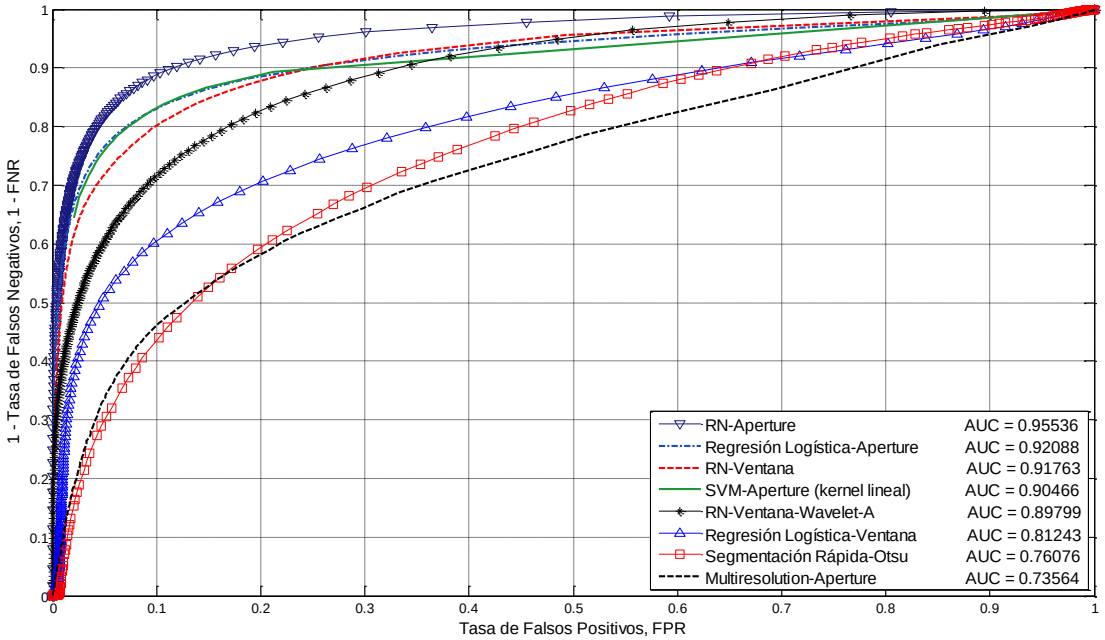


Figura 5.16. Curvas ROC y valores de AUC para los ensambles testeados para la segmentación de los vasos sanguíneos en imágenes oculares de la base de datos DRIVE.

Según los valores de AUC de la Figura 5.16, las redes neuronales tipo feed-forward entrenadas con configuraciones aperture son las que mejor desempeño tienen en la tarea de segmentación de los vasos sanguíneos. Este método de clasificación mejora los resultados de la regresión logística-Aperture, RN-Ventana, SVM-Aperture, RN-Ventana-Wavelet-A, regresión logística-Ventana, segmentación rápida de Otsu, y Multi-resolución-Aperture en 3.8%, 4.1%, 5,6%, 6.4%, 17.6%, y 25.6%, respectivamente. Es importante también observar que no existe una diferencia significativa entre los resultados obtenidos en base a los clasificadores diseñados utilizando regresión logística-Aperture, RN-Ventana, SVM-Aperture, y RN-Ventana-Wavelet-A. Este resultado es interesante teniendo en cuenta que las redes neuronales permiten implementar fronteras de decisión más complejas que la regresión logística y los SVM con kernel lineal.

Para el preprocesamiento de las configuraciones de ventana utilizando aperture y el diseño de clasificadores con regresión logística y redes neuronales se puede observar que existe una mejora de los resultados. Esto se corrobora observando las curvas ROC de regresión logística-Aperture y -Ventana, y RN-Aperture y RN-Ventana. Los resultados de RN-Ventana y RN-Ventana-Wavelet-A son aproximadamente iguales. Esto implica que para mejorar la segmentación de los vasos sanguíneos usando la TWD 2D de Haar de nivel 1 se requiere añadir información de los canales de detalles para el diseño de los clasificadores. Sin embargo, dado que en RN-Ventana-Wavelet-A se utilizan vectores de características de menor dimensión que para RN-Ventana y RN-Aperture, el entrenamiento de las redes neuronales en el primer caso involucra un menor costo computacional.

La peor segmentación se obtiene utilizando Multi-resolución-Aperture. Es interesante observar que, pese a su complejidad computacional, Multi-resolución-Aperture no mejora la calidad de los resultados del método de Otsu. Esto evidencia que el costo de diseño, o estimación, de W-operadores en base a multi-resolución piramidal para la segmentación de los vasos sanguíneos oculares es alto. Esto sucede a pesar de que se disminuyen el número de niveles de gris de las configuraciones de ventana utilizando el preprocesamiento aperture. Por lo tanto, para mejorar la calidad de la estimación de las probabilidades

condicionales usando multi-resolución se requieren más configuraciones de ventana. Desafortunadamente, esto acarrea un incremento significativo del costo computacional tanto de entrenamiento como de testeo.

En general, los resultados de RN-Aperture y RN-Ventana, y regresión logística-Aperture y -Ventana, evidencian que el preprocesamiento de las configuraciones de ventana usando aperture contribuye a mejorar el desempeño de los clasificadores. Esto se debe a que el aperture por mediana ayuda reducir tanto el nivel de ruido como la correlación de las variables que intervienen en el diseño de los clasificadores. Además, para la segmentación de los vasos sanguíneos, de entre todos los métodos testeados, los métodos paramétricos son los que permiten obtener los mejores resultados. Esto evidencia nuevamente que, ante una cantidad limitada de datos de entrenamiento, los métodos paramétricos permiten obtener, usualmente, mejores resultados que los métodos no paramétricos. Esto se suma al alto costo computacional que demanda el uso de métodos no paramétricos como por ejemplo multi-resolución piramidal.

Tabla 5.4. Lista de los 5 mejores métodos propuestos en la literatura científica para la segmentación de vasos sanguíneos en imágenes oculares. Resultados tomados de [Fraz *et al.*, 2012].

Publicación	Método	AUC
[Osareh y Shadgar, 2009]	Clasificadores basados en un modelo de dos Gausianas y SVMs. Los vectores de características se obtienen en base a filtros multiescala de Gabor.	0.9650
[Soares *et al.*, 2006]	Clasificador basado en un modelo de dos Gausianas y vectores de características obtenidos a partir de la transformada 2D de Gabor.	0.9614
[Lam *et al.*, 2010]	Clasificación basada en un modelo de multiconcavidad para los vasos sanguíneos.	0.9614
[Ricci y Perfetti, 2007]	Clasificador basado en SVMs. Los vectores de características están formados por las respuestas de un conjunto de detectores de líneas que atraviesan el píxel a clasificar en diferentes direcciones.	0.9558
[Staal *et al.*, 2004]	Clasificador basado en *k*NN. Los vectores de características están basados en medidas de curvatura en la vecindad de cada píxel a clasificar.	0.9520

En el trabajo de [Fraz *et al.*, 2012] se realiza una revisión bibliográfica exhaustiva de los métodos de segmentación de vasos sanguíneos retinales publicados en la literatura científica. Esta revisión está basada en el análisis de aproximadamente 135 trabajos que abordan este problema de segmentación desde diferentes ópticas, incluyendo aprendizaje computacional supervisado, aprendizaje no supervisado, morfología matemática, modelado de las formas de los vasos sanguíneos, entre otros. En la Tabla 5.4 se resumen los 5 mejores resultados calculados utilizando las imágenes de la base de datos DRIVE y presentados en dicha revisión bibliográfica. Los resultados obtenidos en este experimento utilizando el método RN-Aperture, con AUC = 0.9553, son comparables con los 5 mejores trabajos analizados en la revisión bibliográfica antes descrita. Es interesante notar que algunos métodos utilizados para la comparación están basados en los mismos clasificadores testeados en este experimento. Sin embargo, la diferencia radica en la forma de extracción de los vectores de características. Es importante tener presente que en esta tesis se desarrolla una nueva metodología general para diseño de operadores morfológicos para PDI, y no solamente la solución para un problema de procesamiento específico.

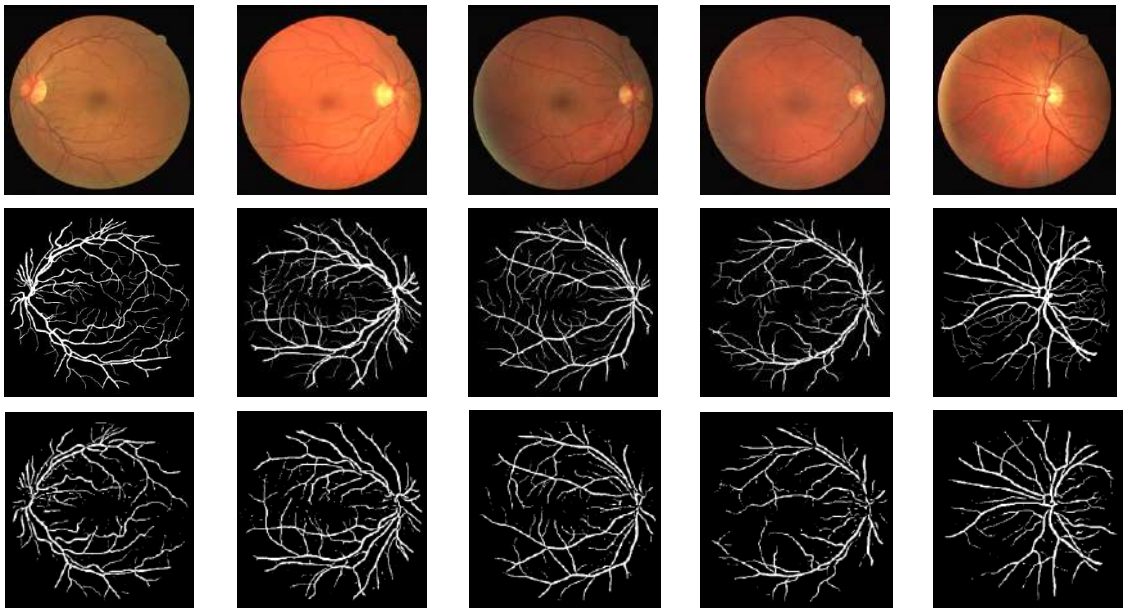


Figura 5.17. Resultados de la segmentación obtenidos en base al método RN-Aperture. La primera fila muestra las imágenes originales color. La segunda fila muestra las imágenes ideales segmentadas manualmente. La tercera fila muestra los resultados de RN-Aperture umbralados en 0.3.

En la Figura 5.17 se presentan algunos resultados obtenidos en base al mejor método (RN-Aperture) de entre todos los métodos testeados en este experimento. En estas imágenes se puede observar que los vasos sanguíneos gruesos son segmentados completamente. Adicionalmente, se puede notar una cierta robustez del método RN-Aperture ante las variaciones de intensidad y color tanto entre imágenes como dentro de una misma imagen. También existe cierta robustez de la segmentación ante cambios de la orientación del árbol arterial. Sin embargo, los vasos sanguíneos delgados son clasificados como fondo en su mayoría. Esto se debe principalmente a que el nivel de contraste entre estos vasos sanguíneos y el fondo es muy bajo. Por fortuna, para diagnóstico médico y sistemas biométricos, este tipo de vasos sanguíneos no revisten mayor importancia [Fraz *et al.*, 2012].

En la Figura 5.18 se presentan las observaciones de ventana, o patrones, que maximizan la respuesta de cada una de las neuronas de la capa oculta de una de las redes neuronales entrenadas en base a RN-Aperture. En esta figura se puede notar que los píxeles que ejercen mayor influencia sobre la clasificación son aquellos localizados cerca del píxel central. Adicionalmente, se puede ver que algunas neuronas buscan líneas diagonales y horizontales que atraviesan el centro de cada patrón.

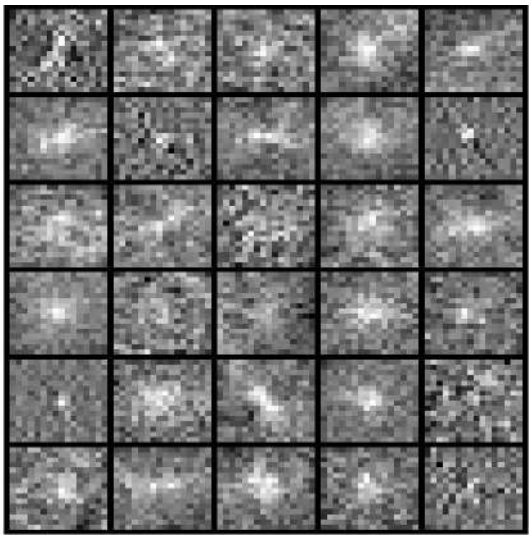


Figura 5.18. Observaciones de ventana que maximizan las respuestas de las neuronas de la capa oculta de una red neuronal tipo feed-forward entrenada usando el método RN-Aperture.

En la Tabla 5.5 se resumen los datos relacionados con el costo computacional en términos del tiempo de escaneo, entrenamiento, y testeo de cada método evaluado en la segmentación de los vasos sanguíneos oculares. La segunda columna muestra que el escaneo de cada imagen de entrenamiento para extraer configuraciones de ventana y configuraciones de aperture toma aproximadamente 30 minutos. El tiempo más alto (2.7 horas) se da para multi-resolución piramidal debido a que en este caso las imágenes se escanean usando 5 ventanas de diferentes tamaños.

Tabla 5.5. Tiempos promedios de escaneo de las imágenes de entrenamiento, ajuste de parámetros de los clasificadores, y aplicación de los ensambles de 20 clasificadores de base para la segmentación de vasos sanguíneos oculares usando imágenes en escala de grises con 565×584 píxeles.

Método	**Tiempo de escaneo por cada par de imágenes de entrenamiento [h]**	**Tiempo de entrenamiento por cada clasificador de base [h]**	**Tiempo de aplicación del ensamble por cada imagen de testeo [h]**
RN-Aperture	0.47	2.86	0.08
Regresión logística-Aperture	0.47	0.26	0.02
RN-Ventana	0.44	3.09	0.06
SVM-Aperture	0.47	0.08	0.74
RN-Wavelet-A	0.69	2.58	0.04
Regresión logística-Ventana	0.44	0.23	0.01
Segmentación rápida-Otsu	--------	--------	0.13
Multi-resolución-Aperture	2.71	--------	7.07

Los valores de tiempo de la tercera columna de la Tabla 5.5 muestran que, en el diseño de W-operadores utilizando clasificadores, la tarea más intensa desde el punto de vista computacional es el ajuste de parámetros. Debido a que las redes neuronales tienen mayor número de parámetros que los demás métodos paramétricos testeados, su entrenamiento toma más tiempo. De todos los métodos que usan redes neuronales, el entrenamiento utilizando configuraciones con la TWD 2D de Haar de nivel 1 toma menos tiempo debido a que la longitud de los vectores de características es menor que para las configuraciones de ventana y de aperture. El entrenamiento de los SVMs toma menor tiempo que el entrenamiento de la regresión logística debido a que para el primero sólo se utiliza un 25% del total de las configuraciones de entrenamiento disponibles. Se debe tener presente que

esto se realizó para reducir el costo computacional que representa encontrar los vectores de soporte y el ajuste de parámetros en SVM.

Según los valores de la cuarta columna de la Tabla 5.5, el procesamiento de cada imagen de testeo usando los ensambles de 20 clasificadores basados en regresión logística y redes neuronales toma aproximadamente 1 y 5 minutos, respectivamente. Estos tiempos son menores que el tiempo que toma segmentar una imagen aplicando el método de umbralamiento local de Otsu y SVM. El tiempo de procesamiento usando SVMs es alto comparado con las redes neuronales y la regresión logística debido a que, por cada píxel a clasificar y por cada vector de soporte, se debe calcular una combinación lineal. Este proceso se realiza por cada uno de los 20 SVMs de base. El tiempo de procesamiento de la multi-resolución piramidal es extremadamente alto, situación que no es compensada por sus valores de error. Todos los experimentos fueron realizados usando un computador con procesador Intel core i-5, con velocidad de CPU de 2.30 GHz, y 8 GB de memoria RAM. Los algoritmos fueron implementados en Matlab®.

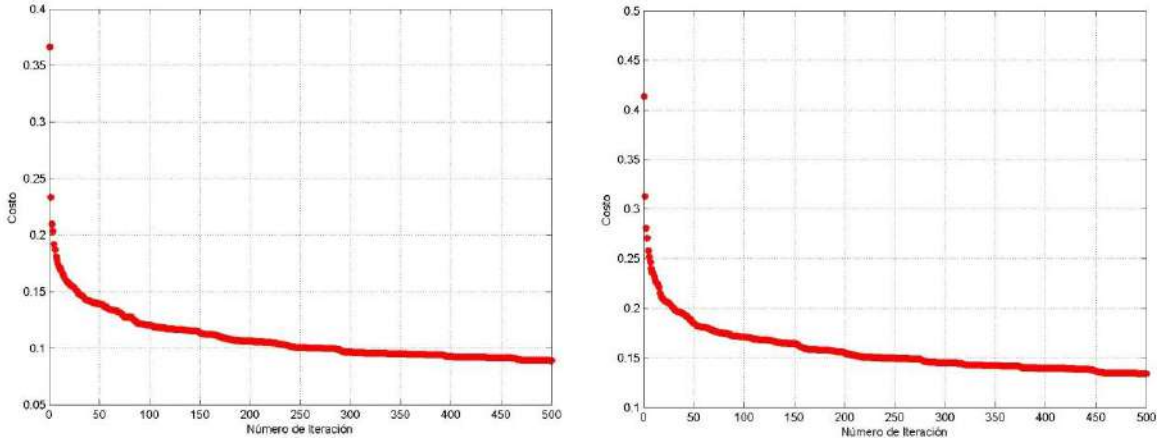


Figura 5.19. Ejemplo de curvas de entrenamiento de las redes neuronales del método RN-Aperture. Cada curva corresponde a una red neuronal diferente. El número máximo de épocas de entrenamiento en cada caso es 500. En ambas curvas, el valor de la función de costo disminuye rápidamente en las primeras épocas y luego disminuye lentamente a medida que el entrenamiento se acerca a un mínimo local.

En la Figura 5.19 se presentan, a modo de ejemplo, dos curvas de entrenamiento de las redes neuronales del método RN-Aperture. En este caso el ajuste de los parámetros de las redes neuronales se realiza utilizando 500 iteraciones o épocas de entrenamiento. Este mismo criterio fue también utilizado para el entrenamiento de la regresión logística. En esta figura se puede notar como el valor de la función a minimizar, o costo, disminuye con cada época de entrenamiento. La velocidad de la disminución del valor del costo es alta en las primeras épocas de entrenamiento y se reduce conforme aumenta el número de épocas de entrenamiento. Este se debe a que a medida que progresa el entrenamiento, la red neuronal converge a un mínimo local.

5.9.2. Segmentación de Exudados en Imágenes Oculares

La presencia de exudados es uno de los principales signos del edema macular diabético (DME del inglés *diabetic macular edema*). Esta patología ocurre cuando la retina se hincha como una complicación de la retinopatía diabética. Hay dos tipos de exudados: duros y blandos. Los *exudados duros* están compuestos por lípidos y depósitos de proteínas que se filtran desde el torrente sanguíneo hacia la retina. Los *exudados blandos* son microinfartos que ocurren en la superficie de la retina. Ambos tipos de exudados son consecuencias

directas de la obstrucción de los vasos sanguíneos oculares debido a los niveles elevados de azúcar en sangre ocasionados por la diabetes. La presencia de exudados dentro de la mácula central, donde se concentran la mayoría de los fotoreceptores oculares, puede causar pérdida de visión, o incluso ceguera, en pacientes diabéticos. En las imágenes oculares, los exudados aparecen como estructuras algodonosas de color amarillento con bordes bien definidos y formas variables [Winder *et al.*, 2009], [Giancardo *et al.*, 2011], [Giancardo *et al.*, 2012], [Youssef y Solouma, 2012].

Con un diagnóstico temprano y un tratamiento adecuado se puede inhibir los efectos de la retinopatía diabética, y en consecuencia los efectos del DME. Sin embargo, la retinopatía diabética es asintomática en sus primeras etapas. Por esta razón muchas personas permanecen sin ser diagnosticadas hasta que empiezan a perder su visión. Esto último se debe a la creciente cantidad de exudados duros y blandos y otras patologías que acompañan a la retinopatía diabética como los microaneurismas y el continuo aparecimiento de nuevos vasos sanguíneos. Idealmente, un escaneo masivo de todos los pacientes diabéticos, incluso de aquellos que no experimentan problemas de visión, ayudaría a diagnosticar el DME en etapas tempranas [Winder *et al.*, 2009]. Desafortunadamente, esta tarea demanda de un trabajo intenso que toma un largo período de tiempo ya que cada imagen de cada paciente debe ser analizada por un especialista. Para cuando la retinopatía diabética ya ha sido diagnosticada, la segmentación y localización de los exudados duros y blandos ayuda a la planificación de tratamientos de fotocoagulación con láser. Estos tratamientos permiten detener el crecimiento de los exudados previniendo así la pérdida de visión o ceguera [Giancardo *et al.*, 2012].

Lo anterior evidencia que la segmentación automática de exudados en imágenes oculares es un campo de investigación abierto dentro de PDI aplicado a la medicina. Esto, sobre todo, cuando se desea implementar algoritmos para un amplio espectro de aplicación. En este experimento se diseñan W-operadores para la segmentación automática de exudados duros y blandos en imágenes oculares. Es importante mencionar que, en esta segmentación, los exudados duros y blandos son considerados como una sola clase. Una de las razones para esta consideración es porque su clasificación es problemática incluso para los especialistas médicos. Adicionalmente, ambos tipos de exudados son consecuencias directas de la retinopatía diabética. Por lo tanto, su diferenciación no aporta una ventaja clínica significativa para el diagnóstico de DME [Giancardo *et al.*, 2012]. En lo posterior, tanto a los exudados duros como a los exudados blandos se los llama simplemente exudados.

Imágenes: Las imágenes para este experimento provienen de la base de datos pública *Hamilton Eye Institute Macular Edema Database* (HEI-MED) [Giancardo *et al.*, 2011]. Esta base de datos se compone de 169 imágenes color de fondo ocular, con 256 niveles de intensidad por cada canal, acompañadas de sus respectivas imágenes binarias ideales o de gold estándar. Estas imágenes tienen un tamaño de 2196×1958 píxeles. Las imágenes binarias de gold estándar contienen las segmentaciones manuales, realizadas por expertos, de los exudados presentes en las imágenes oculares color. En las segmentaciones manuales, los exudados de pequeño tamaño, cuya distinción individual no resulta clara, resultan combinados en un único grupo. La base de datos HEI-MED contiene imágenes de pacientes provenientes de diferentes etnias y etapas de la retinopatía diabética. Se incluyen también imágenes de personas sin DME. Para este experimento, se dividió aleatoriamente las 169 imágenes en 3 subconjuntos: TrS_1, TrS_2, y TtS. Los subconjuntos TrS_1 y TrS_2 contienen 30 imágenes cada uno y se utilizan para el diseño del método propuesto. El subconjunto TtS contiene las 109 imágenes restantes que son utilizadas para la etapa de testeo.

Método Propuesto: El método utilizado para la segmentación de los exudados en imágenes oculares se compone de los siguientes pasos:

a) *Transformación de Color a Niveles de Gris*: Para este paso se utilizó la misma transformación de color a niveles de gris definida en el experimento anterior. Las imágenes en color y en escala de grises de este paso se denotan mediante $\underline{O}$ y O_1, respectivamente (Figura 5.20-a y -b),

b) *Detección de Marcadores de Candidatos a Exudados:* Este paso empieza con la obtención de un mapa con las probabilidades condicionales de que cada píxel de la imagen O_1 sea parte de un exudado dada su configuración de ventana (Figura 5.20-c). Para esto se utilizó un ensamble de W-operadores basados en regresión logística y aperture. Se utilizó regresión logística debido al bajo costo computacional que involucra su entrenamiento y aplicación, teniendo en cuenta el gran tamaño de las imágenes de este experimento. Las ventanas usadas se componen de $n = |W| = 15\times15$ píxeles, donde el píxel a procesar corresponde al centro de la ventana (fila 8, columna 8). El conjunto para el aperture está formado por 21 niveles de gris: $K = \{-10,-9,\ldots,0,...,9,10\}$. Para el diseño de los W-operadores se utilizó un balanceo artificial de frecuencias (Sección 5.5) debido a que el número de configuraciones de los exudados es aproximadamente la mitad de las configuraciones del fondo: $\hat{\mathbb{P}}(Y = 0) = 0.6331$ y $\hat{\mathbb{P}}(Y = 1) = 0.3669$, donde 0 es fondo y 1 corresponde a exudados. El entrenamiento del ensamble de clasificadores se realizó utilizando las imágenes del conjunto TrS_1. La imagen binaria con los marcadores de los candidatos a exudados, H_1, se obtuvo umbralando el mapa de probabilidades en 0.2. Este valor de umbral fue escogido heurísticamente de tal modo de detectar todas las marcas correspondientes a los exudados presentes en las imágenes de entrenamiento. En la Figura 5.20-d se muestra la imagen con etiquetas $\underline{H_1}$ resultante de agrupar y etiquetar los píxeles de cada objeto presente en la imagen Binaria H_1. Para este agrupamiento y etiquetado se utilizó un criterio de conectividad 8.

c) *Remoción de Vasos Sanguíneos y Filtrado de Ruido:* En este paso se eliminaron los grupos de píxeles de la imagen binaria H_1 que forman parte de vasos sanguíneos. La segmentación de los vasos sanguíneos de la imagen en escala de grises O_1 se realizó utilizando el ensamble de W-operadores basados en regresión logística y aperture del anterior experimento con una ventana de 16×16 píxeles. Para esto se utilizó un umbral para las probabilidades condicionales de 0.5 (umbral del operador óptimo). El resultado de la segmentación de los vasos sanguíneos (Figura 5.20-e) no es perfecto debido a la diferencia estadística, especialmente de tamaño, que existe entre las imágenes usadas para el entrenamiento y las imágenes de testeo. Las imágenes de entrenamiento son las imágenes de la base de datos DRIVE que tienen un tamaño de 565×584 píxeles; mientras que las imágenes de testeo de HEI-MED tienen un tamaño de 2196×1958 píxeles. A pesar de esta diferencia, la segmentación obtenida permite remover de H_1 algunos grupos de píxeles que, siendo parte de vasos sanguíneos, fueron detectados como marcadores de candidatos a exudados. Posteriormente, a la imagen resultante de esta remoción de vasos sanguíneos se le aplicó un filtrado de objetos con tamaño menor a 100 píxeles, considerados como ruido, resultando en una nueva imagen binaria H_2. La imagen con etiquetas $\underline{H_2}$ resultante de agrupar y etiquetar los píxeles de cada objeto presente en la imagen Binaria H_2 se presenta en la Figura 5.20-f. Para este agrupamiento y etiquetado se utilizó un criterio de conectividad 8.

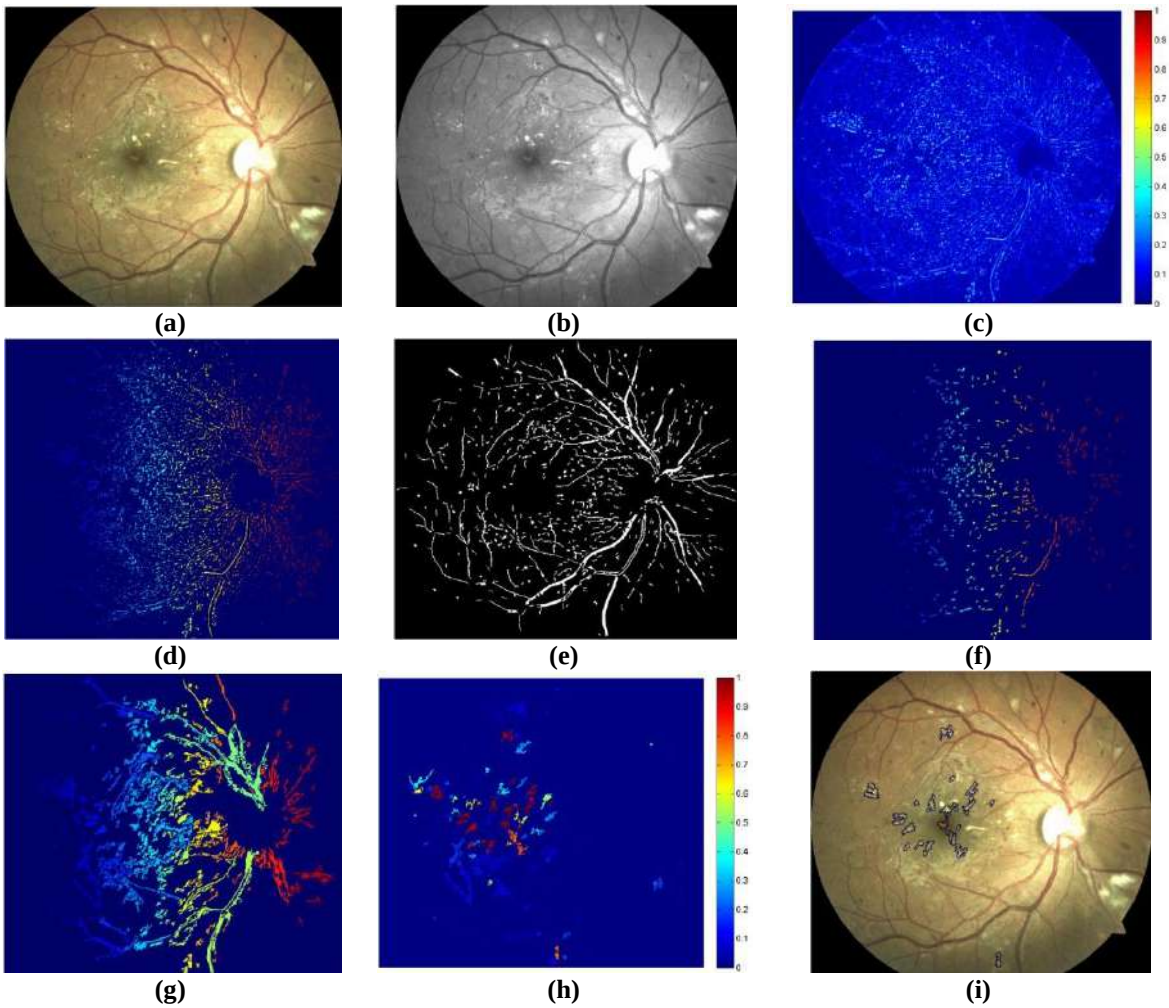


Figura 5.20. Ilustración de la segmentación automática de exudados en imágenes oculares de la base de datos HEI-MED. (**a**) Imagen ocular color RGB, $\underline{O}$. (**b**) Imagen en escala de grises, O_1. (**c**) Mapa de probabilidades de los píxeles candidatos a exudados. (**d**) Marcadores de los candidatos a exudados (cada marcador se muestra con un color diferente), $\underline{H}_1$. (**e**) Vasos sanguíneos segmentados automáticamente usando W-operadores basados en regresión logística y aperture con ventanas de 16×16 píxeles. (**f**) Imagen resultante de la remoción de los vasos sanguíneos y el filtrado de ruido de la imagen d (cada objeto se muestra con un color diferente), $\underline{H}_2$. (**g**) Imagen con los candidatos a exudados (cada candidato se muestra con un color diferente), $\underline{H}_3$. (**h**) Mapa de probabilidades para la clasificación de los candidatos a exudados. (**i**) Bordes de los exudados segmentados (umbralando el mapa de probabilidades en 0.7) superpuestos en la imagen original.

d) *Detección de Candidatos a Exudados:* En este paso se obtuvo una nueva imagen binaria H_3 formada por grupos de píxeles que son potenciales exudados. Esta imagen fue obtenida aplicando un procedimiento de reconstrucción morfológica [Vincent, 1993] a la imagen binaria H_2 resultante del paso anterior. Para la reconstrucción morfológica, H_2 fue utilizada como imagen marcador y la imagen binaria resultante del umbralamiento en 0.1 del mapa de probabilidades obtenido en el paso b fue utilizada como máscara. Este umbral fue seleccionado heurísticamente de tal modo que los candidatos detectados incluyan a todos los exudados. La imagen con etiquetas $\underline{H}_3$ obtenida al agrupar y etiquetar los objetos de la imagen binaria H_3, usando un criterio de conectividad 8, se presenta en la Figura 5.20-g.

e) *Extracción de Características y Clasificación de Candidatos a Exudados:* La imagen binaria H_3 contiene grupos de píxeles que no son exudados, los cuales fueron detectados como consecuencia de los marcadores espurios encontrados en el paso b. En este paso se realiza una clasificación de cada grupo de píxeles de H_3 utilizando regresión logística. La

regresión logística fue entrenada utilizando un conjunto de pares $(\mathbf{X}_1,Y_1),\ldots,(\mathbf{X}_q,Y_q)$, donde q denota el número total de candidatos a exudados detectados en las imágenes del conjunto TrS_2. Para esto se aplicó a cada imagen color de TrS_2 los pasos descritos anteriormente. $\mathbf{X}\in\mathbb{R}^{31}$ es un vector de 31 características extraídas a partir de cada candidato a exudado y la variable Y denota su etiqueta. La etiqueta es $Y = 1$ si la región detectada como candidato a exudado interseca con un grupo de píxeles marcados como exudados en las imágenes ideales o de gold estándar del grupo TrS_2.

Las características utilizadas para la clasificación de cada candidato a exudado son:

- *Forma* (extraídas a partir de $\underline{H}_3$): área, perímetro, excentricidad, y longitud del eje mayor y menor de cada grupo de píxeles;
- *Textura* (extraídas a partir de O_1): intensidad media de los bordes (valor medio de las intensidades de los píxeles que se encuentran en los bordes extraídos con un gradiente morfológico por erosión utilizando un elemento estructurante tipo disco de radio 1 píxel), media y desviación estándar de las intensidades de los píxeles que forman los candidatos, y contraste local entre los píxeles de los candidatos y el fondo (media/desviación estándar de las intensidades de los píxeles dentro de los candidatos menos la media/desviación estándar de las intensidades de los píxeles que se encuentran en un anillo circundante a los candidatos con un ancho medio de 6 píxeles);
- *Color* (extraídas a partir de $\underline{O}$): mediana, media, y desviación estándar de los píxeles de los candidatos a exudados en los modelos de color LUV (canales U y V), RGB (todos los canales), y HSV (canales H y S) de la imagen $\underline{O}$.

El clasificador lineal entrenado en base a estas 31 características se aplicó a cada uno de los candidatos a exudados detectados en la imagen binaria H_3. De esta manera se obtuvo un mapa de probabilidades de exudados como el presentado en la Figura 5.20-h. El umbralamiento de este mapa resulta en la segmentación de los exudados (Figura 5.20-i).

Resultados, Comparación, y Discusión: La evaluación de los resultados del método propuesto para la segmentación de exudados se realizó utilizando las 109 imágenes del conjunto TtS. Para fines de comparación, la evaluación del método propuesto se realizó en base a la clasificación de las imágenes del conjunto TtS como pertenecientes a personas con (etiqueta 1) y sin DME (etiqueta 0). Se clasifica positivamente a una persona con DME cuando en la imagen resultante al umbralar el mapa de probabilidades del paso e se encuentra al menos un exudado. En la Figura 5.21 se presenta la curva ROC en base a los estimados de las tasas de falsos negativos y falsos positivos mediante FNR = $FN/(TP + FN)$ y FPR = $FP/(FP + TN)$, respectivamente. TP denota el número de personas correctamente clasificadas con DME. FN denota el número de personas incorrectamente clasificadas sin DME. TN es el número de personas correctamente clasificadas sin DME. FP es el número de personas incorrectamente clasificadas con DME. El valor de área bajo la curva ROC es AUC = 0.77. Este valor y la forma de la curva ROC indican que el algoritmo propuesto tiende a detectar exudados en algunas imágenes que pertenecen a personas sin DME (falsos positivos).

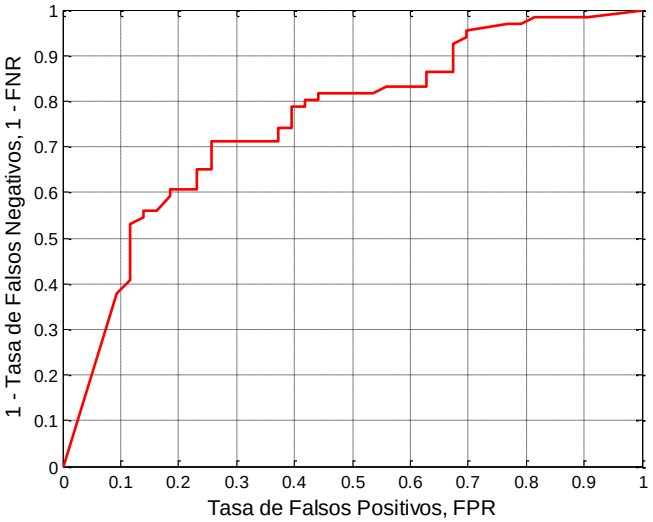


Figura 5.21. Curva ROC de la clasificación de imágenes con DME en base a la segmentación de exudados. El valor de área bajo la curva es AUC = 0.77.

En la literatura científica se han propuesto varios algoritmos para la segmentación automática de exudados en imágenes oculares [Winder *et al.*, 2009], [Youssef y Solouma, 2012]. Sin embargo, en la mayoría de los casos, la evaluación del desempeño de los métodos propuestos se hace utilizando bases de datos privadas con pocas imágenes de testeo o bases de datos públicas que no tienen imágenes de gold estándar, ni datos referentes al diagnóstico de DME en las personas examinadas. Por esta razón, resulta difícil realizar una comparación de los resultados obtenidos en este experimento como aquella desarrollada en la segmentación de los vasos sanguíneos. De la revisión bibliográfica llevada a cabo previo a la realización de este experimento se encontró que sólo en [Giancardo *et al.*, 2011] y [Giancardo *et al.*, 2012] se realizó la evaluación de los algoritmos propuestos para diagnóstico automático de DME utilizando la base de datos HEI-MED. En estos trabajos se reportan valores de AUC de 0.81 y 0.82, respectivamente. Esto implica una superioridad de estos algoritmos en un 5% y 6% respecto al método aquí propuesto. Sin embargo, los algoritmos propuestos en estos trabajos no muestran los resultados de la segmentación de los exudados.

En la Figura 5.22 se presentan algunos resultados obtenidos en este experimento. Estas imágenes evidencian que el método propuesto es robusto a la variación de los niveles de intensidad y color dentro de una imagen y entre las imágenes oculares procesadas. Adicionalmente, también se puede notar que las hemorragias no aparecen como falsos positivos (imágenes de la segunda columna). En general, los resultados obtenidos en esta tesis, y en las publicaciones utilizadas para la comparación, demuestran que es posible realizar el diagnóstico automático de DME. Sin embargo, se requiere mejorar la performance de los métodos propuestos utilizando otros descriptores de la presencia de DME aparte de los exudados. Por lo tanto, el diagnóstico automático de DME sigue siendo una problemática no resuelta que requiere de nueva investigación.

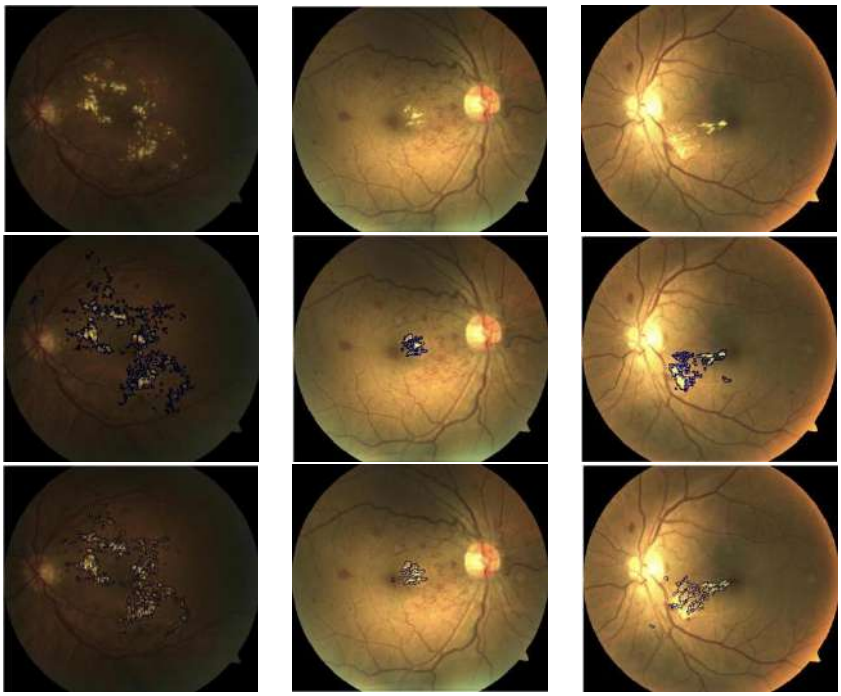


Figura 5.22. Resultados de la segmentación automática de exudados usando W-operadores diseñados en base a regresión logística y aperture. Las imágenes utilizadas provienen de la base de datos pública HEI-MED. Cada columna contiene información de una imagen diferente. La primera fila muestra las imágenes originales. La segunda fila muestra las imágenes de gold estándar con los bordes de los exudados segmentados manualmente. La tercera fila muestra los resultados del método propuesto en esta tesis.

El tiempo promedio de escaneo de cada par de imágenes de entrenamiento para la extracción de configuraciones aperture es de 30 minutos. El tiempo de entrenamiento de cada modelo de regresión logística utilizado en el paso b es de aproximadamente 15 minutos. El criterio de finalización del entrenamiento de la regresión logística está basado tanto en la variación del valor de la función de costo como en un número máximo de 100 épocas. Si el valor de la función de costo no cambia en 5 épocas consecutivas, entonces se detiene el entrenamiento. El tiempo promedio de procesamiento de cada imagen de testeo es de aproximadamente 7 minutos. Este experimento fue realizado usando un computador con procesador Intel core i-5, con velocidad de CPU de 2.30 GHz, y 8 GB de memoria RAM. Los algoritmos fueron implementados en Matlab®.

5.9.3. Segmentación de la Próstata en Imágenes de Resonancia Magnética

En este experimento se aplica la metodología propuesta para el diseño automático de W-operadores al problema de segmentación de la glándula prostática en imágenes de resonancia magnética tipo T2W (MRIs del inglés *magnetic resonance images*) [Dougherty, 2009]. Este problema de segmentación consiste en detectar las dos partes principales de la próstata: la zona periférica y la glándula central. La anatomía de la próstata se asemeja a un cono conteniendo una bola de helado, en la que el cono corresponde a la zona periférica y el helado corresponde a la glándula central [Verma y Rajesh, 2011]. Algunas biopsias de la próstata guiadas por ultrasonido retornan resultados negativos a pesar de los altos niveles del marcador PSA (del inglés *prostate-specific antigen*) en exámenes de sangre y la presencia real de cáncer. La distinción entre estas dos partes de la próstata es importante debido a que, tales fallas de diagnóstico, se han atribuido a la falta de extracción de

muestras de la glándula central, donde surgen alrededor del 30% de los adenocarcinomas [Claus *et al.*, 2004].

La segmentación de la próstata proporciona información para varias tareas médicas incluyendo: (1) la localización de los límites de la próstata tanto para radioterapias como para guiar biopsias; (2) la estimación de su volumen para tareas de seguimiento del progreso de enfermedades prostáticas; (3) la inicialización o definición de marcadores para algoritmos de registración multimodales; y (4) el establecimiento de regiones de interés para los métodos automáticos de detección y evaluación del cáncer de próstata. La segmentación automática de la glándula prostática es importante porque su segmentación manual es una tarea tediosa, demandante de tiempo, y además se requiere de entrenamiento y experiencia médica.

Para la segmentación automática de la glándula de la prostática en MRIs T2W, se debe tener en cuenta que los píxeles/voxels que pertenecen a la zona periférica tienen, usualmente, valores de intensidad altos. Los píxeles/voxels que pertenecen a la glándula central tienen, usualmente, valores de intensidad inferiores a los de la zona periférica. Sin embargo, la primera de estas características puede cambiar con la presencia de cáncer prostático o con la existencia de patologías benignas tales como hemorragias, prostatitis, hiperplasia, y secuelas pos-tratamiento de radioterapias. Por estas razones, la segmentación automática de la próstata es un problema difícil de PDI. Además, se debe tener en cuenta la presencia de ruido, la falta de homogeneidad en los MRIs, y también las complejas estructuras anatómicas de la próstata y de las glándulas y órganos circundantes.

Imágenes: Las resonancias magnéticas utilizadas en este experimento proceden de la colección-3T del *NCI-ISBI 2013 Challenge* [Kirby y Montagnese, 2013]. Estos MRIs son públicos y destinados al desarrollo de métodos para la segmentación automática de la glándula prostática. Esta colección de MRIs consta de 30 volúmenes transversales T2W acompañados de sus respectivas segmentaciones manuales o volúmenes de gold estándar. El número de cortes por cada volumen de esta base de datos varía entre 15 y 24. Cada corte contiene 320×320 píxeles, los cuales toman un valor de intensidad del conjunto {0,1,...,2965}. Para este experimento, se dividió aleatoriamente el conjunto de 30 volúmenes en dos subconjuntos: uno para entrenamiento y el otro para testeo, donde cada subconjunto contiene 15 volúmenes.

Método Propuesto: En este experimento se diseñaron W-operadores en base a redes neuronales tipo feed-forward de tres capas: entrada, oculta y salida (Sección 5.7). La capa de salida de las redes neuronales está formada por tres neuronas que predicen las probabilidades condicionales de clase $\mathbb{P}(Y = i|\mathbf{X};\boldsymbol{\beta})$ dado un vector de características $\mathbf{X}$ para cada píxel a clasificar. Las clases se denotan con las etiquetas i = 1, 2, y 3 que corresponden a fondo, glándula central, y zona periférica, respectivamente. Para el escaneo de cada corte y la obtención de configuraciones se utilizaron ventanas ralas de 37×37 píxeles, con una cantidad efectiva de 19×19 píxeles. Los píxeles tomados en cuenta son los pertenecientes a las filas y columnas impares. El píxel a procesar corresponde al centro de la ventana (fila 19 y columna 19 de la ventana rala). Este tamaño de ventana fue seleccionado en base al tamaño promedio de la próstata en los cortes de los volúmenes de entrenamiento.

Los vectores de características para el entrenamiento de las redes neuronales están formados por 722 componentes. Las primeras 361 componentes corresponden a los niveles de intensidad observados a través de la ventana de escaneo sin aplicar ningún tipo de

preprocesamiento. Las 361 componentes restantes corresponden a la observación de ventana preprocesada con aperture. Para esto se utilizó un conjunto de 301 niveles de intensidad: $K = \{-150,\ldots,0,...,150\}$. Las configuraciones de ventana, sin preprocesamiento, están destinadas a capturar información de intensidad y texturas, que son características importantes para la segmentación de los MRIs. Las configuraciones de aperture están destinadas a capturar información de formas, realizando un filtrado del ruido e independientemente del nivel de intensidad local de la configuración asociada con cada píxel a clasificar.

Para la capa oculta de las redes neuronales utilizadas en este experimento se utilizaron 100 neuronas con funciones de transferencia *logsig*. Este número de neuronas fue determinado en base a pruebas preliminares realizadas sobre 5 volúmenes seleccionados aleatoriamente de entre todos los 15 volúmenes del conjunto de entrenamiento. Una vez determinada la arquitectura de las redes, éstas fueron entrenadas utilizando los vectores de características extraídos a partir de todos los cortes de cada volumen de entrenamiento. Esto implica que se entrenaron un total de 15 redes neuronales tipo feed-forward de tres capas.

Para el entrenamiento de las redes neuronales se utilizó regularización por decaimiento de pesos con un factor $\lambda = 10^{-2}$. Este valor fue determinado heurísticamente como parte de las pruebas preliminares para la selección del número de neuronas de la capa oculta. Se utilizó regularización para el ajuste de pesos para resolver el problema de redundancia de los parámetros de la capa de salida. Adicionalmente, previo al entrenamiento de las redes neuronales se aplicó un balanceo artificial a las frecuencias de cada clase. Esto debido a que las configuraciones predominantes son las pertenecientes al fondo con una probabilidad de clase estimada sobre el conjunto de entrenamiento de $\hat{\mathbb{P}}(Y = 1) = 0.954$, seguido muy lejos por las configuraciones de la glándula central con $\hat{\mathbb{P}}(Y = 2) = 0.031$, y la zona periférica con $\hat{\mathbb{P}}(Y = 3) = 0.015$.

Resultados, Comparaciones, y Discusión: Para la evaluación del método se segmentación propuesto se utilizaron los 15 volúmenes del conjunto de testeo. La Tabla 5.6 muestra la matriz de confusión [Murphy, 2012] obtenida mediante la comparación vóxel a vóxel de los volúmenes segmentados con sus correspondientes volúmenes de gold estándar. Cada valor de esta tabla está normalizado con respecto al número total de vóxeles procesados (29237248). Las etiquetas debajo del número de cada casilla se utilizan para facilitar la explicación del cálculo de las medidas de calidad de la segmentación. La tasa de error (ER) de segmentación, calculada como el número de vóxeles incorrectamente clasificados sobre el total de vóxeles procesados, es ER = 0.0316. Este valor se obtiene sumando, y luego dividiendo para 100%, todos los valores de la Tabla 5.6 que se encuentran fuera de la diagonal principal. Es importante tener en cuenta que este valor es bastante bajo debido a que, gran parte de los vóxeles de fondo, son correctamente clasificados. Sin embargo, esta medida no es un buen indicador de la calidad de la segmentación de la próstata debido a que para su cálculo se incluye al fondo.

En la Tabla 5.7 se presentan los resultados obtenidos al normalizar los valores de cada columna de la Tabla 5.6. En esta nueva tabla los valores de cada columna suman 100%. Esta forma de presentación de los resultados permite evaluar la clasificación de los vóxeles por cada una de las clases analizadas. Los datos de la primera columna indican que, del total de vóxeles pertenecientes a fondo, 97.8892% fueron clasificados correctamente como fondo, mientras que 1.6813% y 0.4295% fueron incorrectamente clasificados como

glándula central y zona periférica, respectivamente. En el caso de los vóxeles pertenecientes a la glándula central, 72.0275% fueron correctamente clasificados, mientras que 23.7091% y 4.2634% fueron incorrectamente clasificados como fondo y zona periférica, respectivamente. Finalmente, en el caso de los vóxeles de la zona periférica, 76.3224% fueron clasificados correctamente, mientras que 18.7241% y 4.9534% fueron clasificados incorrectamente como fondo y glándula central, respectivamente.

Tabla 5.6. Matriz de confusión expresada en porcentajes de los resultados de la segmentación automática de la próstata en MRIs utilizando W-operadores diseñados con redes neuronales tipo feed-forward. Los porcentajes se han obtenido dividiendo cada valor de la matriz de confusión por 29237248.

		Etiquetas verdaderas [%]		
	Clase	Fondo	Glándula central	Zona periférica
Etiquetas estimadas [%]	Fondo	93.7229 ($a_{1,1}$)	0.7224 ($a_{1,2}$)	0.2264 ($a_{1,3}$)
	Glándula central	1.6098 ($a_{2,1}$)	2.1947 ($a_{2,2}$)	0.0599 ($a_{2,3}$)
	Zona periférica	0.4112 ($a_{3,1}$)	0.1299 ($a_{3,2}$)	0.9229 ($a_{3,3}$)

Tabla 5.7. Matriz de confusión normalizada por columnas de los resultados de la segmentación automática de la próstata en MRIs utilizando W-operadores diseñados con redes neuronales tipo feed-forward. Los valores de cada columna suman 100%.

		Etiquetas verdaderas [%]		
	Clase	Fondo	Glándula central	Zona periférica
Etiquetas estimadas [%]	Fondo	97.8892	23.7091	18.7241
	Glándula central	1.6813	72.0275	4.9534
	Zona periférica	0.4295	4.2634	76.3224

Tabla 5.8. Valores de sensibilidad y precisión de la segmentación automática de la glándula prostática y de sus dos componentes principales: la glándula central y la zona periférica.

Métrica	**Valor**
Sensibilidad (etiqueta estimada | etiqueta real)	
Próstata | Próstata $(a_{2,2}+ a_{2,3}+ a_{3,2}+ a_{3,3})/(a_{1,2}+ a_{1,3}+a_{2,2}+ a_{2,3}+ a_{3,2}+ a_{3,3})$	0.7770
Glándula central | Glándula central $(a_{2,2})/(a_{1,2}+a_{2,2}+ a_{3,2})$	0.7202
Zona periférica | Zona periférica $(a_{3,3})/(a_{1,3}+a_{2,3}+ a_{3,3})$	0.7632
Precisión (etiqueta real | etiqueta estimada)	
Próstata | Próstata $(a_{2,2}+ a_{2,3}+ a_{3,2}+ a_{3,3})/(a_{2,1}+a_{2,2}+ a_{2,3}+ a_{3,1}+ a_{3,2}+ a_{3,3})$	0.6207
Glándula central | Glándula central $(a_{2,2})/(a_{2,1}+a_{2,2}+ a_{2,3})$	0.5679
Zona periférica | Zona periférica $(a_{3,3})/(a_{3,1}+a_{3,2}+ a_{3,3})$	0.6303

En la Tabla 5.8 se calculan los valores de sensibilidad, o en inglés *recall*, y precisión para la segmentación de la próstata y sus dos componentes a partir de los datos presentados en la Tabla 5.6. La *sensibilidad* mide la proporción de vóxeles correctamente clasificados como parte de una clase dado el total de vóxeles que realmente pertenecen a dicha clase. La

precisión mide la proporción de vóxeles que realmente pertenecen a una clase dado el total de vóxeles clasificados como parte de dicha clase. Mientras más cercanos a 1 sean la sensibilidad y la precisión, mejor será el desempeño de un algoritmo identificando los vóxeles de cada clase. Para este experimento, los valores de sensibilidad indican que el método de segmentación propuesto es capaz de detectar la mayor parte de la próstata (glándula central y zona periférica). Sin embargo, existe una porción considerable de vóxeles dentro de cada clase que no son reconocidos, lo cual implica la existencia de falsos negativos dentro de cada clase, especialmente para el caso de la glándula central. Los valores de precisión indican que el método propuesto detecta incorrectamente una cantidad considerable de vóxeles de fondo como parte de la glándula central y la zona periférica. Este problema es más crítico para la glándula central. Esto implica la existencia de falsos positivos en cada clase de interés.

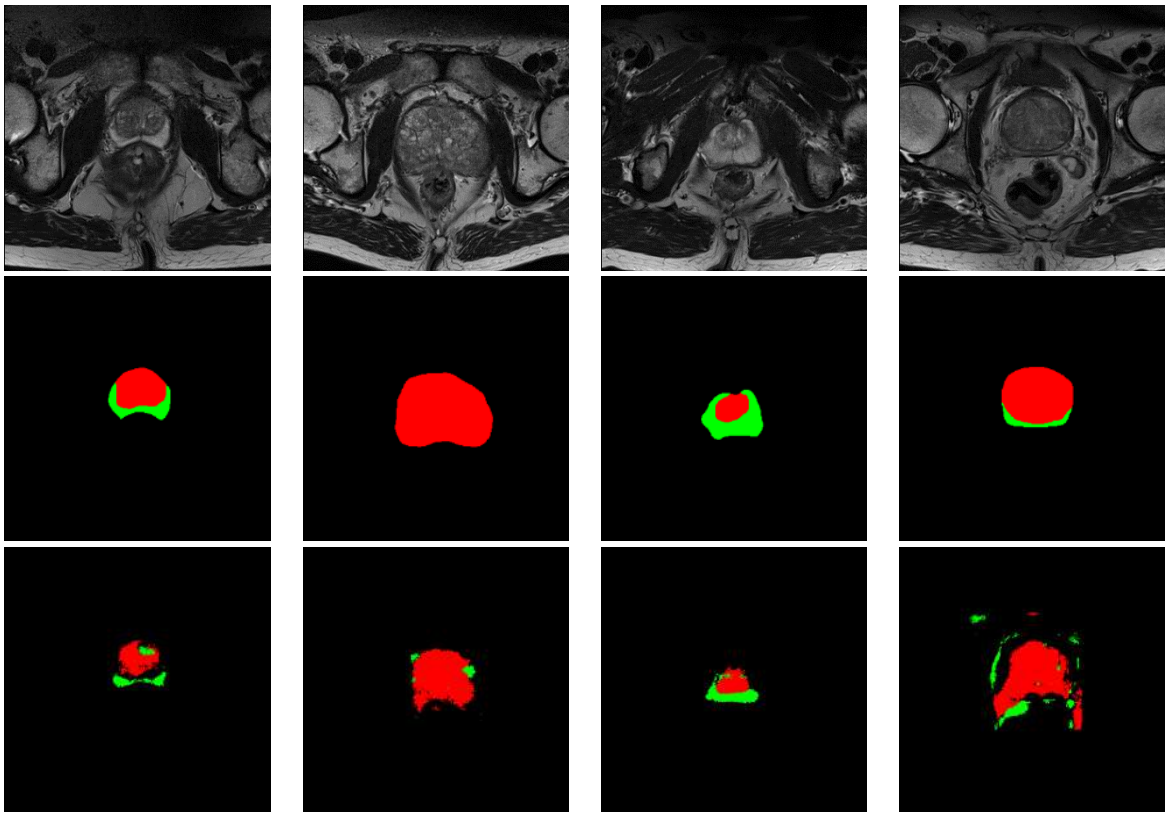


Figura 5.23. Resultados de la segmentación automática de la glándula prostática en imágenes resonancia magnética T2W del NCI-ISBI 2013 Challenge. Cada columna presenta la información de un corte diferente. La primera fila contiene los cortes originales. La segunda fila muestra los resultados de la segmentación manual (gold estándar). La Tercera fila muestra los resultados obtenidos con el método propuesto.

Los resultados numéricos de la Tabla 5.8 corroboran los resultados visuales que se presentan en la Figura 5.23. En los cortes mostrados en esta figura, la glándula central se colorea de rojo; mientras que la región de la zona periférica se colorea de verde. El valor seleccionado para el umbralamiento del mapa de probabilidades de cada corte fue 0.5 (umbral del clasificador óptimo). Cada corte presentado pertenece a un volumen diferente. En esta figura se puede observar que el método de segmentación propuesto es capaz de detectar correctamente, en todos los casos, la región central de la glándula prostática. Los errores ocurren por la incorrecta segmentación de las regiones de borde de la glándula central y también debido a que, en algunos casos, sólo se detectan partes de la zona periférica.

Para fines de comparación en la Tabla 5.9 se presentan los 5 mejores resultados obtenidos en el NCI-ISBI 2013 Challenge [Kirby y Montagnese, 2013]. Las comparaciones se hacen en función de los valores de sensibilidad y especificidad en la clasificación de los vóxeles de cada clase. La *especificidad* de cada clase es la proporción entre el total de verdaderos negativos sobre el total de verdaderos negativos y falsos positivos de dicha clase. Cuanto más cercana a 1 sea la especificidad, mejor será el desempeño de un algoritmo en la discriminación entre los vóxeles de cada clase. Se debe notar que, en este caso, se toma en cuenta la información de las dos clases de interés (glándula central y la zona periférica) más el fondo; siendo ésta la diferencia con respecto a la sensibilidad.

En términos de la sensibilidad de clasificación de la glándula central, el mejor método (CCIPD RobToth) tiene una ventaja de 11.81% respecto del método propuesto que ocupa el segundo lugar. Para la sensibilidad de la zona periférica, el método propuesto ocupa el cuarto lugar con una diferencia del 16% con respecto al método de CCIPD RobToth. En cuanto en la especificad de clasificación, los resultados obtenidos con el método propuesto muy similares a aquellos de los demás métodos. En ninguno de los casos de comparación se presentan detalles de los algoritmos de segmentación utilizados.

Tabla 5.9. Valores de sensibilidad y especificidad de la segmentación automática de la próstata del método propuesto en esta tesis y otros métodos cuyos resultados están publicados en [Kirby y Montagnese, 2013].

Método/Métrica	**Sensibilidad** Glándula central $(a_{2,2})/(a_{1,2}+a_{2,2}+a_{3,2})$	**Sensibilidad** Zona periférica $(a_{3,3})/(a_{1,3}+a_{2,3}+a_{3,3})$	**Especificidad** Glándula central $(a_{1,1}+a_{1,3}+a_{3,1}+a_{3,3})/(a_{1,1}+a_{1,3}+a_{2,1}+a_{2,3}+a_{3,1}+a_{3,3})$	**Especificidad** Zona periférica $(a_{1,1}+a_{1,2}+a_{2,1}+a_{2,2})/(a_{1,1}+a_{1,2}+a_{2,1}+a_{2,2}+a_{3,1}+a_{3,2})$
Propuesto	*0.720*	*0.763*	*0.983*	*0.995*
Geert Litjens 1	0.698	0.848	0.998	0.998
CCIPD RobToth	0.805	0.886	0.997	0.997
MD Anderson Cancer Center	0.339	0.697	0.998	0.997
Qinquan Gao	0.518	0.835	0.998	0.998
Geert Litjens 2	0.434	0.687	0.997	0.997

A nivel general, los resultados obtenidos en esta tesis y en los trabajos utilizados para la comparación evidencian que la segmentación automática de la próstata es un problema que todavía no está resuelto. Para mejorar la calidad de la segmentación se podría realizar el procesamiento los MRIs tratados como volúmenes y no corte a corte. Para el caso de los W-operadores, esto requiere el uso de ventanas 3D, con lo cual se incrementa el costo computacional del procesamiento, especialmente si las ventanas requeridas son de gran tamaño. Otra alternativa consiste en utilizar el método aquí propuesto como un método para definir marcadores que pueden ser utilizados en combinación con otras técnicas de segmentación.

Con respecto al costo computacional del método propuesto, el tiempo promedio de escaneo de cada volumen de entrenamiento es de aproximadamente 10 minutos. El tiempo promedio de entrenamiento de cada red neuronal es de aproximadamente 2.5 horas. El tiempo promedio de procesamiento de cada volumen utilizando el ensamble de 15 redes neuronales es de aproximadamente 5 minutos. Este experimento fue realizado usando un computador con procesador Intel core i-5, con velocidad de CPU de 2.30 GHz, y 8 GB de memoria RAM. Los algoritmos fueron implementados en Matlab®. El número de épocas de entrenamiento de cada red neuronal es 350.

En la Figura 5.24-a se presenta un ejemplo de una curva de entrenamiento de una red neuronal para este experimento. En este caso el valor del error de entrenamiento al inicio es

mayor que 1 debido a la regularización aplicada a los pesos de las neuronas de la capa oculta y la capa de salida. En la Figura 5.24-b y -c se presentan, a modo de ejemplo, los patrones de ventana y aperture, de 37×37 píxeles, que maximizan las respuestas de las 100 neuronas de la capa oculta. Se puede observar como los dos tipos de configuraciones contribuyen con diferente tipo de información para la clasificación. En particular se puede observar que las configuraciones de ventana actúan como una especie de detectores de bordes y texturas.

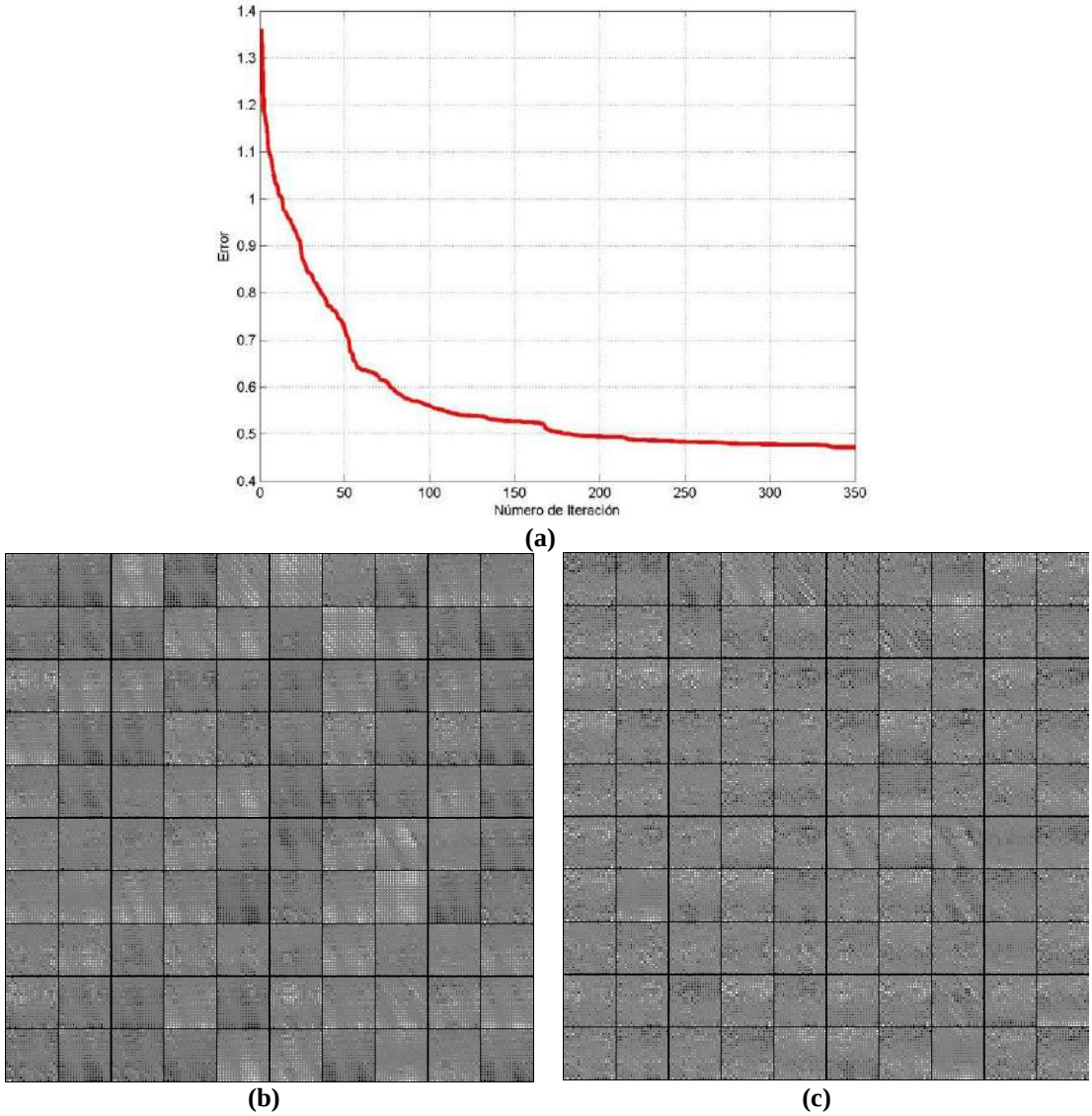


Figura 5.24. (**a**) Curva de entrenamiento con regularización y patrones de (**b**) ventana y (**c**) aperture de 37×37 píxeles que maximizan la respuesta de una red neuronal entrenada para la segmentación automática de la glándula prostática.

5.10. Anotaciones Finales

En este capítulo se han realizado las siguientes contribuciones:

- Se ha propuesto un nuevo método para el diseño automático de W-operadores para la clasificación de los píxeles de imágenes en escala de grises.

- El diseño propuesto se basa en el uso de dos modelos de clasificación que son la regresión logística, para clasificación binaria, y las redes neuronales artificiales tipo feed-forward, para clasificación binaria y clasificación multiclase.
- Se ha analizado el problema de diseño desbalanceado de clasificadores. En base a este análisis se ha propuesto el uso de un balanceo artificial de frecuencias previo al entrenamiento de los clasificadores en base a un conjunto de muestras.
- Se ha propuesto y analizado una solución, que consiste en un ensamble, para el diseño de clasificadores complejos, en términos de la dimensión VC, utilizando grandes cantidades de ejemplos de entrenamiento y recursos computacionales limitados.
- El análisis de los clasificadores propuestos, dado un problema de PDI, se ha realizado en función de las operaciones de convolución o correlación de imágenes y transformaciones píxel a píxel.
- Se ha propuesto el uso de aperture y de la transformada wavelet discreta 2D de Haar para el preprocesamiento de las configuraciones de ventana. El objetivo de estos dos tipos de preprocesamiento es reducir el espacio de búsqueda del W-operador óptimo. Además, se ha evidenciado empíricamente que el aperture permite reducir el nivel de correlación entre las componentes de las configuraciones de ventana.
- La nueva metodología propuesta en este capítulo ha sido aplicada a problemas reales de segmentación de imágenes médicas.

Los problemas abordados en este capítulo son la segmentación tanto de vasos sanguíneos como de exudados en imágenes de fondo ocular, y la segmentación de la glándula prostática en resonancias magnéticas. Para los experimentos realizados para cada problema, se han utilizado imágenes/volúmenes de bases de datos públicas, permitiendo de esta manera la comparación de los resultados obtenidos en esta tesis con otros métodos propuestos en la literatura científica. Esta es la primera vez que se ha aplicado el diseño automático de W-operadores para problemas reales de PDI que involucran gran cantidad de imágenes reales, y en particular, imágenes médicas. Los resultados obtenidos en base a los W-operadores diseñados con la metodología propuesta en esta tesis son comparables con los mejores métodos propuestos para cada uno de los tres problemas analizados. La diferencia con respecto a los métodos comparados es que en este capítulo se ha propuesto una metodología general y no un algoritmo particular para resolver un problema práctico específico.

El mayor costo computacional del diseño de W-operadores para la segmentación de imágenes en escala de grises, bajo el paradigma de reconocimiento de patrones, radica en el escaneo de las imágenes de entrenamiento y el posterior ajuste de los parámetros de los clasificadores. Una vez entrenados los clasificadores que definen a un W-operador su aplicación, o testeo, es relativamente rápida. Para la etapa de testeo se pueden utilizar las operaciones de convolución o correlación de imágenes siempre que el preprocesamiento de las configuraciones de ventana sea lineal. Las máscaras para cualquiera de estas operaciones se implementan a partir de los parámetros de los clasificadores entrenados, y utilizando la misma estructura definida para la ventana usada para la extracción de configuraciones.

En los experimentos realizados, dos de los tres problemas involucran el procesamiento de imágenes color. Para estas imágenes se utilizó una transformación de color a niveles de gris. Sin embargo, este preprocesamiento ocasiona la pérdida de la información contenida en el color. En el próximo capítulo se presentan los primeros avances realizados para el

diseño automático de W-operadores para el procesamiento directo de imágenes color y en general de imágenes multicanal.

5.11. Referencias

[1] Y.S. Abu-Mostafa, M. Magdon-Ismail and H.T. Lin, *Learning from data*, AMLBook, Pasadena CA, 2012.

[2] M. Benalcázar, J. Padín, A. Bouchet, M. Brun and V. Ballarin, "Diseño Automático de Operadores Morfológicos Aplicado a la Segmentación de Angiografías Retinales," *Proc. Congreso Argentino de Informática y Salud (CAIS 2011)*, Córdoba, Argentina, 137–47 (2011).

[3] M.E. Benalcázar, M. Brun and V.L. Ballarin, "Segmentación de Vasos Sanguíneos en Angiogrfías Retinales usando Ensambles de Filtros Aperture," *Proc. International Symposium on Innovation and Technology ISIT2012*, Perú, 125-9 (2012).

[4] M.E. Benalcázar, M. Brun and V.L. Ballarin, "Automatic Design of Window Operators for the Segmentation of the Prostate Gland in Magnetic Resonance Images," *Proc. Latin American Conference on Biomedical Engineering CLAIB 2014*, Paraná Argentina, 2014a.

[5] M.E. Benalcázar, M. Brun and V.L. Ballarín, "Automatic Segmentation of Exudates in Ocular Images using Ensembles of Aperture Filters and Logistic Regression," *Proc. 19th Argentinean Bioengineering Society Congress (SABI 2013)*, Tucumán Argentina, 2-11 (2013).

[6] M.E. Benalcázar, M. Brun and V.L. Ballarín, "Automatic Design of Aperture Filters Using Neural Networks Applied to Ocular Image Segmentation," *Proc. EUSIPCO 2014*, Lisboa, 1-5 (2014b).

[7] M.E. Benalcázar, M. Brun, V.L. Ballarin and R.M. Hidalgo, "Automatic Design of Ensembles of Window Operators for Ocular Image Segmentation," *Latin America Transactions, IEEE (Revista IEEE America Latina)*, vol. 12, pp. 935-41, 2014c.

[8] D. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, New York, 1982.

[9] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Cambridge, 2005.

[10] C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

[11] J.S. Bridle, "Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition," in *Neurocomputing*. vol. 68, F. Soulié and J. Hérault, Eds.: Springer Berlin Heidelberg, 1990, pp. 227-36.

[12] M. Brun, V.L. Ballarín and I.A. Pagnuco, "Diseño Balanceado de Clasificadores para Estudios de Asociación Poligenética," *Proc. 1er Congreso Argentino de Informática y Salud, 38 JAIIO*, Buenos Aires, 2010.

[13] M. Brun, M.E. Benalcázar, I.A. pagnuco and V.L.Ballarín, "New Balanced Logistic Regression Algorithm with application to Phenotype Classification," *Proc. 4ta. Conferencia Internacional de la Sociedad Iberoamericana de Bioinformática (SolBio)*, Rosario Argentina, 2013.

[14] F.G. Claus, H. Hricak and R. Hattery, "Pretreatment evaluation of prostate cancer: role of MR imaging and 1HMR spectroscopy," *Radiographics*, vol. 24, pp. 67-80, 2004.

[15] T. Coleman and Y. Li, "On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds," *Mathematical Programming*, vol. 67, pp. 189-224, 1994.

[16] T. Coleman and Y. Li, "An Interior Trust Region Approach for Nonlinear Minimization Subject to Bounds," *SIAM Journal on Optimization*, vol. 6, pp. 418-45, 1996.

[17] I. Daubechies, *Ten Lectures on Wavelets*, Rutgers University and AT&T Bell Laboratories, New York, 2004.

[18] L. Devroye, L. Györfi and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer, New York, 1996.

[19] G. Dougherty, *Digital image processing for medical applications*, Cambridge Press, New York, 2009.

[20] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: a methodology review," *Journal of Biomedical Informatics*, vol. 35, pp. 352-9, 2002.

[21] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification*, 2001.

[22] M.M. Fraz, P. Remagnino, A. Hoppe, B. Uyyanonvara, A.R. Rudnicka, C.G. Owen and S.A. Barman, "Blood vessel segmentation methodologies in retinal images – A survey," *Computer Methods and Programs in Biomedicine*, vol. 108, pp. 407-33, 2012.

[23] L. Giancardo, F. Meriaudeau, T. Karnowski, Y. Li, S. Garg, J.K. Tobin and E. Chaum, "Exudate-based diabetic macular edema detection in fundus images using publicly available datasets," *Medical image analysis*, vol. 16, pp. 216–26, 2012.

[24] L. Giancardo, F. Meriaudeau, T. Karnowski, Y. Li, K.W. Tobin and E. Chaum, "Automatic retina exudates segmentation without a manually labelled training set," *Proc. International symposium on biomedical imaging*, Chicago IL, 1396–400, 2011.
[25] R.C. Gonzalez and R.E. Woods, *Digital image processing*, Prentice Hall, Upper Saddle River, N. J., 2002.
[26] T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer, New York, 2001.
[27] R. Hirata, E.R. Dougherty and J. Barrera, "Aperture filters," *Signal Processing*, vol. 80, pp. 697-721, 2000.
[28] X. Jiang, R. El-Kareh and L. Ohno-Machado, "Improving Predictions in Imbalanced Data Using Pairwise Expanded Logistic Regression," *Proc. AMIA Annu*, 625–34 (2011).
[29] NCI-ISBI 2013 Challenge, *https://wiki.cancerimagingarchive.net/display/Public/Prostate-3T*,
[30] A. Krogh and J. Vedelsby, "Neural Network Ensembles, Cross Validation, and Active Learning," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D.S. Touretzky, and T.K. Leen, Eds. Cambridge MA: MIT Press, 1995, pp. 231-8.
[31] B.S.Y. Lam, G. Yongsheng and A.W.C. Liew, "General retinal vessel segmentation using regularization-based multiconcavity modeling," *IEEE Transactions on Medical Imaging*, vol. 29, pp. 1369–81, 2010.
[32] M.A. Maloof, "Learning when data sets are imbalanced and when costs are unequal and unknown " *Proc. CML-2003 Workshop on Learning from Imbalanced Data Sets II*, Washington DC,, 2003.
[33] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674–93, 1989.
[34] M. Mohri, A. Rostamizadeh and A. Talwalkar, *Foundations of Machine Learning*, MIT Press, London, 2012.
[35] K.P. Murphy, *Machine Learning: A probabilistic perspective*, MIT Press, Cambridge MA, 2012.
[36] A. Osareh and B. Shadgar, "Automatic blood vessel segmentation in color images of retina," *Iranian Journal Of Science And Technology Transaction B – Engineering* vol. 33, pp. 191–206, 2009.
[37] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, pp. 62-6, 1979.
[38] N.C. Oza and K. Tumer, "Classifier ensembles: Select real-world applications," *Information Fusion*, vol. 9, pp. 4-20, 2008.
[39] E. Ricci and R. Perfetti, "Retinal blood vessel segmentation using line operators and support vector classification," *IEEE Transactions on Medical Imaging*, vol. 26, pp. 1357–65, 2007.
[40] F.Y. Shih, *Image Processing and Mathematical Morphology*, CRC Press, New york, 2009.
[41] J.V.B. Soares, J.J.G. Leandro, R.M. Cesar, H.F. Jelinek and M.J. Cree, "Retinal vessel segmentation using the 2-D Gabor wavelet and supervised classification," *IEEE Transactions on Medical Imaging* vol. 25, pp. 1214–22, 2006.
[42] J. Staal, M.D. Abràmoff, M. Niemeijer, M.A. Viergever and B. Ginneken, "Ridge-Based Vessel Segmentation in Color Images of the Retina," *IEEE Transactions On Medical Imaging*, vol. 23, pp. 501-9, 2004.
[43] S. Verma and A. Rajesh, "A clinically relevant approach to imaging prostate cancer: review," *American Journal of Roentgenology*, vol. 196, pp. 1-10, 2011.
[44] L. Vincent, "Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms," *IEEE Transactions on Image Processing* vol. 2, pp. 176-201, 1993.
[45] G.M. Weiss, "Mining with rarity: a unifying framework," *ACM SIGKDD Explorations Newsletter*, vol. 6, pp. 7-19, 2004.
[46] R. Winder, P. Morrow, I. McRitchie, J. Bailie and P. Hart, "Algorithms for digital image processing in diabetic retinopathy," *Computerized medical imaging*, vol. 33, pp. 608–22, 2009.
[47] D. Youssef and H. Solouma, "Accurate detection of blood vessels improves the detection of exudates in color fundus images," *Computer methods and programs in biomedicine* vol. 108, pp. 1052–61, 2012.
[48] K. Zuiderveld, "Contrast Limited Adaptive Histogram Equalization," in *Graphic Gems IV*: Academic Press Professional, 1994, pp. 474-85.

CAPÍTULO VI

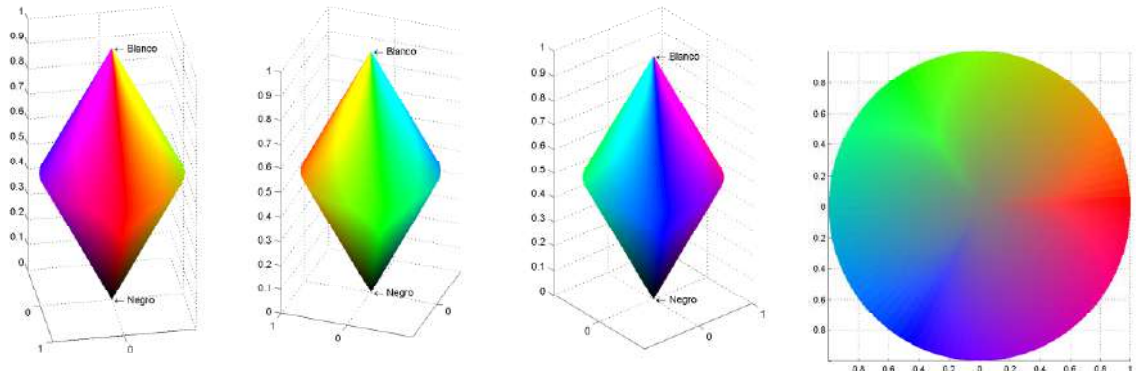
Método Propuesto para Diseño Automático de W-operadores en Color

Resumen: En este capítulo se propone un nuevo método para el diseño automático de W-operadores para el procesamiento de imágenes color. Estas imágenes se caracterizan por el hecho de que cada píxel toma un vector de tres valores y no un escalar como en las imágenes en escala de grises o binarias. En este contexto, se considera el problema particular donde las imágenes de entrada del procesamiento son imágenes color y las imágenes de salida son imágenes binarias. En función de esto se define a un W-operador mediante un clasificador binario. Para las imágenes color, y un tamaño de ventana dado, el número de componentes de una configuración de ventana es tres veces más grande que para el caso de imágenes en escala de grises o binarias. Esto implica un crecimiento exponencial del tamaño del espacio de búsqueda del clasificador o W-operador óptimo. A nivel práctico, para una cantidad fija y limitada de ejemplos de entrenamiento, esta situación obliga a definir restricciones para reducir el costo de estimación y el costo computacional de diseño de clasificadores. Con este antecedente, en este capítulo se extiende la definición del preprocesamiento aperture para reducir tanto el tamaño del espacio de configuraciones como el nivel de correlación entre las componentes de una configuración de ventana en color RGB. Finalmente, se presentan los resultados de un experimento que consiste en la aplicación de W-operadores para la segmentación automática de los vasos sanguíneos en las imágenes oculares color RGB de la base de datos DRIVE.

6.1. Introducción

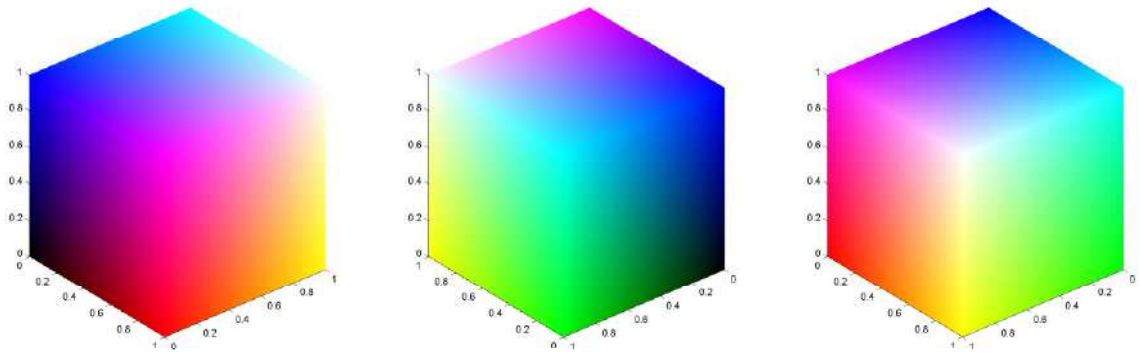
En este capítulo se propone un nuevo método para el diseño automático de W-operadores para el procesamiento de imágenes multicanal, y en particular, de imágenes color. El problema general es aquel donde tanto la entrada (imagen observada) como la salida (imagen ideal) del procesamiento son imágenes multicanal pertenecientes a los espacios de imágenes $(L_1 \times L_2 \times \ldots \times L_m)^{\boldsymbol{E}}$ y $(L'_1 \times L'_2 \times \ldots \times L'_{m'})^{\boldsymbol{E}}$, respectivamente. El conjunto $\boldsymbol{E} \subset \mathbb{Z}^2$ denota el dominio tanto de las imágenes observadas como de las ideales. L_i y L'_j denotan los conjuntos de intensidades del i-ésimo y j-ésimo canales de las imágenes observadas e ideales, respectivamente, con $i = 1,\ldots,m$ y $j = 1,\ldots,m'$. El análisis de este capítulo se restringe al caso particular donde las imágenes observadas son imágenes color, $m = 3$, y las imágenes ideales son imágenes binarias, $m' = 1$. Este análisis se extiende de manera directa al caso donde cada píxel de las imágenes ideales toma una etiqueta de un conjunto con más de dos etiquetas.

Cada píxel de una imagen color toma un vector de tres componentes. Cada componente representa un atributo, o medida del color, en el espacio de representación del rango de la imagen. Por ejemplo, para el modelo de color HSI, la primera componente del vector de un píxel representa el tono (H), la segunda componente representa la saturación (S), y la tercera componente representa la intensidad (I), tal como se puede observar en Figura 6.1. De igual manera para el espacio RGB, la primera componente representa la información de rojo, la segunda componente representa la información de verde, y la tercera componente representa la información de azul, tal como se puede observar en la Figura 6.2. [Gonzalez y Woods, 2002], [Koschan y Abidi, 2008], [Benalcázar *et al.*, 2011].



(a) **(b)** **(c)** **(d)**

Figura 6.1. Espacio de color HSI representado mediante un sólido de dos conos. Vista con **(a)** H = 0º (rojo), **(b)** H = 120º (verde), **(c)** H = 240º (azul), y **(d)** corte transversal del sólido en I = 0.5.



(a) **(b)** **(c)**

Figura 6.2. Espacio de color RGB representado mediante un cubo. Plano del (**a**) rojo, (**b**) verde, y (**c**) azul.

6.2. W-operadores Color y Clasificadores

Para el diseño automático de W-operadores en color se considera un par de procesos estocásticos conjuntamente estacionarios ($\underline{\mathbf{O}}$,$\mathbf{I}$), donde $\underline{\mathbf{O}}$ genera imágenes observadas color, que pertenecen al conjunto $(L_1 \times L_2 \times L_3)^{\boldsymbol{E}}$, e $\mathbf{I}$ genera imágenes ideales binarias, que pertenecen al conjunto $\{0,1\}^{\boldsymbol{E}}$. Dado el par de procesos estocásticos ($\underline{\mathbf{O}}$,$\mathbf{I}$), el objetivo del diseño automático es encontrar un W-operador $\boldsymbol{\Psi}$: $(L_1 \times L_2 \times L_3)^{\boldsymbol{E}} \rightarrow \{0,1\}^{\boldsymbol{E}}$ tal que $\boldsymbol{\Psi}(\underline{\mathbf{O}})$ sea una buena estimación de $\mathbf{I}$ en términos del error cuadrático medio $\mathbb{E}[(\boldsymbol{\Psi}(\underline{\mathbf{O}})(t) - \mathbf{I}(t))^2]$ evaluado en un píxel arbitrario $t \in \boldsymbol{E}$.

Sean una ventana $W = \{w_1,\ldots,(0,0),\ldots,w_n\} \subset \mathbb{Z}^2$, con $n = |W|$ elementos, una imagen observada color $\underline{\mathbf{O}} = (\mathbf{O}_1,\mathbf{O}_2,\mathbf{O}_3)$, y su respectiva imagen ideal $\mathbf{I}$ binaria. Las imágenes en escala de grises $\mathbf{O}_1 \in L_1^{\boldsymbol{E}}$, $\mathbf{O}_2 \in L_2^{\boldsymbol{E}}$, y $\mathbf{O}_3 \in L_3^{\boldsymbol{E}}$ son los tres canales de la imagen color $\underline{\mathbf{O}}$. Para un W-operador en color $\boldsymbol{\Psi}$: $(L_1 \times L_2 \times L_3)^{\boldsymbol{E}} \rightarrow \{0,1\}^{\boldsymbol{E}}$, el valor de la imagen resultante del procesamiento $\boldsymbol{\Psi}(\underline{\mathbf{O}})$, en un píxel arbitrario $t \in \boldsymbol{E}$, es igual al valor que su función característica ψ: $(L_1 \times L_2 \times L_3)^W \rightarrow \{0,1\}$ asigna a la configuración observada a través de la ventana W, centrada en (0,0), en la imagen color $\underline{\mathbf{O}}$ trasladada por t:

$$\boldsymbol{\Psi}(\underline{\mathbf{O}})(t) = \psi(\underline{\mathbf{O}}(w_1 + t),\ldots,\underline{\mathbf{O}}(w_n + t)). \qquad (6.1)$$

En la ecuación 6.1, se tiene que $\underline{\mathbf{O}}(w_i + t)$ es un vector de tres componentes: $\mathbf{O}_1(w_i + t)$, $\mathbf{O}_2(w_i + t)$, y $\mathbf{O}_3(w_i + t)$, con $i = 1,\ldots,n$ y $w_i \in W$. Sea $\mathbf{X} = (\underline{X}_1,\ldots,\underline{X}_n)$ un vector aleatorio que contiene la configuración observada en $\underline{\mathbf{O}}_t$, a través de W, con $t \in \boldsymbol{E}$:

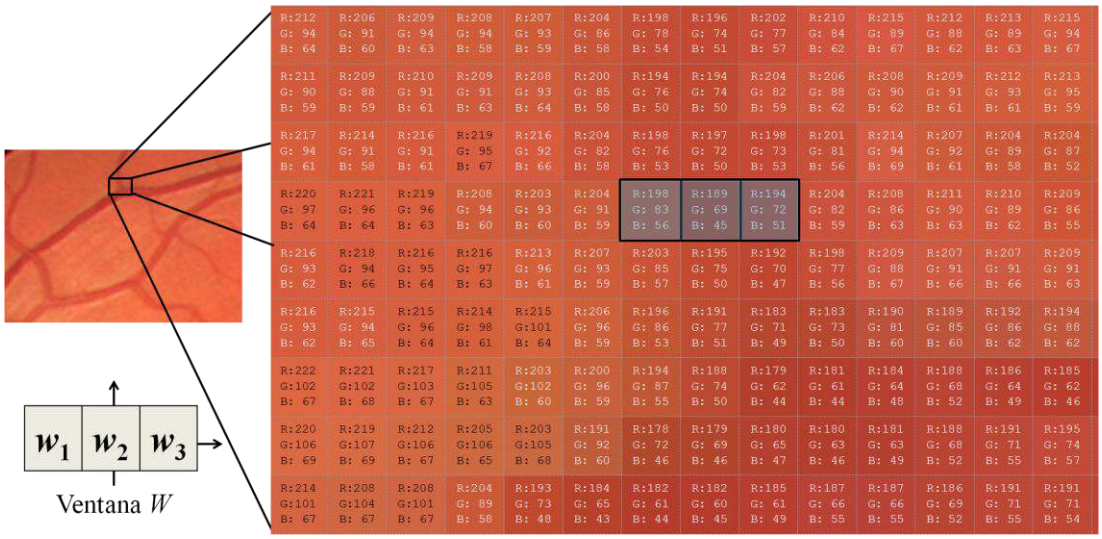
$$\mathbf{X} = (\underline{\mathbf{O}}(w_1 + t),\ldots,\underline{\mathbf{O}}(w_n + t)) = (\underline{X}_1,\ldots,\underline{X}_n). \tag{6.2}$$

En la ecuación 6.2, se tiene que $\underline{X}_i = (\mathbf{O}_1(w_i + t),\mathbf{O}_2(w_i + t),\mathbf{O}_3(w_i + t))$ es un vector de tres componentes, con $i = 1,\ldots,n$. El espacio de todos los posibles vectores $\mathbf{X}$, con $3n$ componentes, se denota mediante $\boldsymbol{\mathcal{X}}$. Se define también la variable aleatoria $Y = \mathbf{I}(t)$, cuyo rango es el conjunto $\mathcal{Y} = \{0,1\}$. En función de lo anterior, $\psi(\mathbf{X})$ es un estimador de Y, donde la calidad de la estimación está dada por $\mathbb{E}[(\psi(\mathbf{X}) - Y)^2]$. En este caso, la esperanza se calcula en base a la distribución conjunta $\Pr(\mathbf{X},Y)$ del espacio $\{\boldsymbol{\mathcal{X}} \times \mathcal{Y}\}$. Dado que $\psi(\mathbf{X}) \in \{0,1\}$, entonces $\mathbb{E}[(\psi(\mathbf{X}) - Y)^2] = \mathbb{P}(\psi(\mathbf{X}) \neq Y)$. Por lo tanto, el diseño de un operador de ventana, o W-operador, en color $\boldsymbol{\Psi}$ para imágenes ideales binarias consiste en el diseño de un clasificador binario ψ: $\boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$ que minimice la probabilidad $\mathbb{P}(\psi(\mathbf{X}) \neq Y)$ de clasificar erróneamente un vector $\mathbf{X}$ observado con la etiqueta Y. Si cada píxel perteneciente a las imágenes ideales toma una etiqueta del conjunto $\mathcal{Y} = \{0,\ldots,c - 1\}$, con $c \in \mathbb{Z}^+$ y $c > 2$, entonces la función ψ: $\boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$ es un clasificador multiclase.

6.2.1. W-operadores para Clasificación en el Modelo de Color RGB

Sea una imagen observada color RGB $\underline{O} = (O_R,O_G,O_B)$, donde O_R, O_G, y O_B son sus canales que contienen la información de rojo, verde, y azul, respectivamente. Para el rango, o vectores, de los píxeles de las imágenes color RGB es usual que $L_1 = L_2 = L_3 = L'$, donde $L' = \{lmin_1,\ldots,lmax_1\}$ es un conjunto de $l_1 = |L'|$ niveles de gris. En estas condiciones, un W-operador en color, para clasificación binaria, es una función $\boldsymbol{\Psi}$: $(L'^3)^{\boldsymbol{E}} \rightarrow \{0,1\}^{\boldsymbol{E}}$ que, aplicada a una imagen color RGB, retorna una imagen binaria. En este caso $\boldsymbol{\Psi}$ se define mediante un clasificador binario ψ: $\boldsymbol{\mathcal{X}} \rightarrow \mathcal{Y}$ que asigna una etiqueta del conjunto $\{0,1\}$ a un vector $\mathbf{X} = (\underline{X}_1,...,\underline{X}_n)$ que contiene $3n$ variables aleatorias (Figura 6.3). Para este caso, $\boldsymbol{\mathcal{X}}$ es el espacio que contiene a todos los $(l_1)^{3n}$ posibles vectores que se pueden observar a través de una ventana W, de $n = |W|$ puntos, en las imágenes observadas RGB.

En un contexto práctico, el clasificador binario ψ puede ser diseñado utilizando cualquiera de los métodos de clasificación presentados en los anteriores capítulos. Para esto se utiliza un conjunto de N ejemplos de entrenamiento $\boldsymbol{\mathcal{D}} = \{(\mathbf{X}_i,freq(\mathbf{X}_i,Y = 0),freq(\mathbf{X}_i,Y = 1))\}$, donde $i = 1,\ldots,N$. Para este conjunto se asume que los pares $(\mathbf{X},Y)$ son realizaciones independientes de la distribución conjunta $\Pr(\mathbf{X},Y)$ inducida por la distribución conjunta del proceso $(\underline{\mathbf{O}},\mathbf{I})$, donde $\underline{\mathbf{O}}$ genera imágenes color RGB e $\mathbf{I}$ genera imágenes binarias.



X = ((198,83,56),(189,69,45),(194,72,51))

Figura 6.3. Configuración en color RGB, **X**, observada a través de una ventana W de $n = 1\times3$ píxeles.

Para los experimentos del capítulo anterior se definió una transformación de color a niveles de gris Ω la cual, en este caso, permite reducir la dimensión del vector $\mathbf{X} = (\underline{X}_1,...,\underline{X}_n)$ de $3n$ a n componentes, resultando en un vector $\mathbf{Z} = \Omega(\mathbf{X}) = (Z_1,\ldots,Z_n)$. Se denota con $\mathcal{Z}$ al espacio de todos los posibles $(l_1)^n$ vectores $\mathbf{Z}$ de n variables, donde cada variable toma un valor del conjunto L'. Esta transformación implica una pérdida de la información del color debido a que cada vector $\underline{X}_i = (X_{i,1},X_{i,2},X_{i,3})\in\mathbf{X}$ es representado por el escalar $Z_i\in\mathbf{Z}$. Este escalar Z_i se obtiene mediante una suma ponderada de las componentes $X_{i,1}$, $X_{i,2}$, y $X_{i,3}$ del vector $\underline{X}_i$, con $i = 1,...,n$. Por lo tanto, el uso de la transformación de color a niveles de gris Ω implica una forma de restricción del espacio de configuraciones de ventana $\mathcal{X}$. Esto porque diferentes vectores $\mathbf{X}\in\mathcal{X}$ resultan en un mismo vector $\mathbf{Z}\in\mathcal{Z}$.

Debido a que el tamaño del espacio de configuraciones $\mathcal{Z}$, $(l_1)^n$, es mucho menor que el tamaño del espacio $\mathcal{X}$, $(l_1)^{3n}$, el costo de diseño de clasificadores usando $\mathcal{Z}$ es menor que el costo de diseño usando el espacio $\mathcal{X}$. Sin embargo, para problemas de segmentación donde el color es uno de los atributos importantes para discriminar a los objetos a segmentar del fondo, el uso de la transformación Ω puede acarrear un alto costo de restricción. Por lo tanto, para estos problemas, el diseño de los clasificadores debe ser realizado usando los vectores $\mathbf{X}$ en lugar de $\mathbf{Z}$. Esto a su vez implica un incremento de la dimensión VC de los modelos de clasificación a ser usados. Es conveniente recordar que la dimensión VC de los clasificadores paramétricos considerados en esta tesis (regresión logística, redes neuronales, y SVMs) es función de la dimensión de los vectores a clasificar.

En teoría, el vector $\mathbf{X}\in\mathcal{X}$ contiene 3 veces más información que el vector $\mathbf{Z} = \Omega(\mathbf{X})\in\mathcal{Z}$. Sin embargo, en la práctica es usual que exista una correlación tanto dentro de las componentes de cada vector $\underline{X}_i\in\mathbf{X}$ como entre las componentes de los vectores $\underline{X}_i$ y $\underline{X}_j$ que forman $\mathbf{X}$, donde $i,j = 1,\ldots,n$ e $i \neq j$. El primer problema ocurre, frecuentemente, para el caso de las imágenes color en el modelo RGB debido a la correlación de sus canales. El segundo problema ocurre para todos los modelos de color, especialmente, entre los vectores $\underline{X}_i$ y $\underline{X}_j$

observados en píxeles adyacentes dentro de la ventana *W*. Por esta razón, la proporción entre la cantidad de información efectiva que contiene la configuración en color **X** y la cantidad de información que contiene la configuración en niveles de gris **Z** es menor que 3. Por lo tanto, para problemas de segmentación que requieran el uso de la información de color, el diseño de clasificadores, usando las configuraciones de ventana en color **X**, puede demandar de un número mayor de ejemplos de entrenamiento que el calculado usando el análisis de Vapnik-Chervonenkis [Vapnik y Chervonenkis, 1971], [Vapnik, 2000]. Esto a pesar de que este análisis, usualmente, sobreestima la cantidad necesaria de ejemplos de entrenamiento debido a que es válido para cualquier distribución conjunta Pr(**X**,*Y*) de los pares (**X**,*Y*) (Capítulo III, Sección 3.3).

El requerimiento de un incremento en el número de ejemplos de entrenamiento *N* se da, no sólo por el gran tamaño del espacio de configuraciones de ventana $\mathcal{X}$, sino también por la correlación que existe entre las componentes de los vectores **X**. Este incremento de *N* permite disminuir tanto el riesgo de overfitting, o sobreajuste de parámetros, para los modelos de clasificación paramétricos como la maldición de la dimensión para modelos paramétricos y no paramétricos. Desafortunadamente, en la práctica el número de ejemplos de entrenamiento es fijo y limitado. Adicionalmente, en los casos donde sea posible disponer de un mayor número de ejemplos de entrenamiento, el ajuste de parámetros de los clasificadores, o entrenamiento, puede demandar de un alto costo computacional. En estas condiciones, resulta necesario imponer restricciones sobre el espacio de configuraciones en color $\mathcal{X}$ y la familia, o clase, de clasificadores a utilizar.

6.3. Preprocesamiento de Configuraciones de Ventana RGB

En esta sección se propone una extensión del preprocesamiento aperture para el caso de configuraciones de ventana en color RGB, **X**, formadas por *n* vectores, $\underline{X}_1,\ldots,\underline{X}_n$, de tres componentes cada uno. Para esta tarea se define el producto cartesiano $K = (K_1 \times K_2 \times K_3)$ que contiene a todos los vectores $\underline{Z}_i = (Z_{i,1},Z_{i,2},Z_{i,3})$ que pueden formar una configuración de aperture en color, **Z**, obtenida a partir de una configuración de ventana en color RGB, **X**, con $i = 1,\ldots,n$, $K_j = \{-k_j,-k_j + 1,\ldots,0,\ldots,k_j - 1,k_j\}$, y $j = 1,2,3$. En función de lo anterior, dada una configuración de ventana en color RGB $\mathbf{X} = (\underline{X}_1,\ldots,\underline{X})$, cada vector formado por tres componentes $\underline{Z}_i = (Z_{i,1},Z_{i,2},Z_{i,3})$ de la nueva configuración $\mathbf{Z} = (\underline{Z}_1,\ldots,\underline{Z}_n)$ puede ser calculado mediante

$$\underline{Z}_i = \begin{cases} Z_{i,1} = min(max(-k_1, X_{i,1} - z_1), k_1) \\ Z_{i,2} = min(max(-k_2, X_{i,2} - z_2), k_2) \\ Z_{i,3} = min(max(-k_3, X_{i,3} - z_3), k_3) \end{cases}, \tag{6.3}$$

con $i = 1,\ldots,n$. Nótese que la definición de la ecuación 6.3 implica asumir que el supremo e ínfimo, o máximo y mínimo, de los vectores del espacio *K* son (k_1,k_2,k_3) y $(-k_1,-k_2,-k_3)$, respectivamente. El vector $\mathbf{z} = (z_1,z_2,z_3)$ utilizado para calcular la traslación de rango de la configuración de ventana en color RGB **X** debe ser definido como una función de sus *n* vectores $\underline{X}_i$: $\mathbf{z} = \mathbf{z}(\underline{X}_i,\ldots,\underline{X}_n)$. Al igual que para el caso de imágenes en niveles de gris, el grado de restricción del preprocesamiento aperture en color está dado por los valores seleccionados para los parámetros k_1, k_2, y k_3.

6.3.1. Traslación de Rango por el Vector Perteneciente al Píxel Observado

Dada la imagen observada color $\underline{O} = (O_1,O_2,O_3)$ y la ventana $W = \{w_1,\ldots,(0,0),\ldots,w_n\}$, el vector **z** para una configuración **X** puede ser definido como $\mathbf{z} = (O_1(t),O_2(t),O_3(t))$, donde $t \in \boldsymbol{E}$ es el píxel de la imagen $\underline{O}_t$ que es observado a través del punto (0,0) de la ventana *W*. $\underline{O}_t$ denota la traslación espacial de $\underline{O}$ por *t*, y **X** es la configuración observada en $\underline{O}_t$ a través de la ventana *W*. Esta definición del vector **z** es sensible a la presencia de ruido en las imágenes observadas [Hirata *et al.*, 2000].

6.3.2. Traslación de Rango por el Vector Mediana de la Observación

Para ventanas *W* conteniendo un número *n* impar de píxeles, **z** puede ser definido como el vector mediana entre todos los vectores de la configuración **X**: $\mathbf{z} = median(\underline{X}_1,\ldots,\underline{X}_n)$. Este cálculo requiere de la definición de un criterio de orden total, o un orden parcial, para el producto cartesiano $L_1 \times L_2 \times L_3$, o L'^3, que constituye el rango de las imágenes observadas RGB. La necesidad de definir un criterio de orden entre vectores también aparece para cuando se quiere extender las definiciones de los operadores morfológicos binarios y en niveles de gris, y del filtro mediana, al caso general de las imágenes multicanal, y en particular, de las imágenes color (Capítulo I, Sección 1.4.3). Múltiples propuestas de criterios de orden para vectores han sido publicadas en la literatura científica, las cuales pueden ser clasificadas en los siguientes cuatro grupos: orden-M, o marginal, condicional, o lexicográfico, orden-P, y reducido [Barnett, 1976], [Astola *et al.*, 1990], [Serra, 1992], [Pitas y Tsakalides, 1991], [Flores *et al.*, 2004], [Lukac *et al.*, 2005], [Angulo, 2007], [Aptoula y Lefèvre, 2007], [Koschan y Abidi, 2008].

Los órdenes marginal y reducido permiten dotar de un orden parcial al espacio $L_1 \times L_2 \times L_3$. En el orden-M, dos vectores se ordenan comparando marginalmente cada una de sus componentes usando la relación de orden usual para escalares $<$. En el orden reducido, dos vectores se ordenan en base a la comparación de dos escalares calculados a partir de las componentes de cada vector. El orden-P se basa en el cálculo de envolventes convexas para los elementos del espacio $L_1 \times L_2 \times L_3$. Dos vectores que pertenecen a diferentes envolventes convexas se ordenan en función de la envolvente más externa. El problema en este caso es que sólo se pueden ordenar vectores que pertenecen a diferentes envolventes. En el orden lexicográfico, dos vectores se ordenan mediante la comparación marginal de sólo una parte, o todas, sus componentes en una secuencia predefinida llamada cascada. Para este tipo de orden, si se usan todas las componentes se puede definir una relación de orden total.

Mediana Marginal: Para el procesamiento marginal, o componente a componente, el vector mediana **z** de un conjunto de *n* vectores $(X_{1,1},X_{1,2},X_{1,3}),\ldots,(X_{n,1},X_{n,2},X_{n,3})$ se define como

$$\mathbf{z} = (z_1, z_2, z_3) = \begin{cases} z_1 = median(X_{1,1}, X_{2,1},..., X_{n,1}) \\ z_2 = median(X_{1,2}, X_{2,2},..., X_{n,2}) \\ z_3 = median(X_{1,3}, X_{2,3},..., X_{n,3}) \end{cases}. \tag{6.4}$$

En función de esta definición, el vector **z** no siempre está contenido dentro de los vectores que forman la observación de ventana **X**. Por ejemplo, sea la observación $\mathbf{X} = (\underline{X}_1,\underline{X}_2,\underline{X}_3)$, con $\underline{X}_1 = (255,0,0)$ (rojo puro), $\underline{X}_2 = (0,255,0)$ (verde puro), y $\underline{X}_3 = (0,0,255)$ (azul puro). El vector mediana de **X**, calculado de manera marginal, es $\mathbf{z} = (0,0,0)$, el cual corresponde al color negro que no está contenido en la observación **X**.

Mediana Usando una Relación de Orden Total: Para este caso se define una relación de orden total basada en la combinación de un orden reducido y una cascada lexicográfica [Angulo, 2007], [Benalcázar *et al.*, 2012]. Cada par $\underline{X}_i = (X_{i,1},X_{i,2},X_{i,3})$ y $\underline{X}_j = (X_{j,1},X_{j,2},X_{j,3})$, con $i,j = 1,\ldots,n$, que forman la configuración de ventana **X** se ordenan en función del siguiente criterio:

$$\underline{X}_i <_{\underline{X}_0} \underline{X}_j \Leftrightarrow \begin{cases} d(\underline{X}_i,\underline{X}_0) > d(\underline{X}_j,\underline{X}_0) \text{ ó} \\ d(\underline{X}_i,\underline{X}_0) = d(\underline{X}_j,\underline{X}_0) \text{ y} \begin{cases} X_{i,1} < X_{j,1} \text{ ó} \\ X_{i,1} = X_{j,1} \text{ y } X_{i,2} < X_{j,2} \text{ ó} \\ X_{i,1} = X_{j,1} \text{ y } X_{i,2} = X_{j,2} \text{ y } X_{i,3} < X_{j,3} \end{cases} \end{cases}, \tag{6.5}$$

donde $\underline{X}_0 = (X_{0,1},X_{0,2},X_{0,3}) \in (L_1 \times L_2 \times L_3)$ es un vector de referencia y $d(\underline{X}_i,\underline{X}_0)$ es la distancia euclidiana entre $\underline{X}_i$ y $\underline{X}_0$ para el orden reducido. En función de lo anterior, el menor de dos vectores es aquel que tiene la mayor distancia euclidiana con respecto al vector $\underline{X}_0$. Los empates se resuelven aplicando la cascada lexicográfica (1→2→3) que analiza, en este caso, primero al canal de rojo, luego al canal de verde y finalmente al canal de azul. En función de la aplicación en cuestión, es posible definir otra cascada lexicográfica.

En base al criterio de orden total definido en la ecuación 6.5 se pueden ordenar en una secuencia ascendente, o descendente, los vectores $(X_{1,1},X_{1,2},X_{1,3}),\ldots,(X_{n,1},X_{n,2},X_{n,3})$ que conforman una configuración de ventana RGB **X**. El vector mediana **z** de la configuración de ventana **X** corresponde al vector que se encuentra en la posición central de dicha secuencia ordenada. La ventaja en este caso, y a diferencia de la mediana marginal, es que el vector **z** pertenece siempre a la configuración de ventana **X**. En general, para definir el vector mediana de una configuración de ventana en color se pueden también utilizar otras relaciones de orden total diferentes a la considerada en esta tesis.

Ejemplo 6.1. En la Figura 6.4 se ilustra el procedimiento para obtener una configuración de aperture a partir de la configuración de ventana en color $\mathbf{U} = (\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3)$, donde $\mathbf{u}_1 = (7,3)$, $\mathbf{u}_2 = (3,5)$, y $\mathbf{u}_3 = (5,1)$ (Figura 6.4-a). Para facilitar la visualización de este procedimiento, se asume que la imagen a procesar contiene solamente dos canales que se denotan con R (rojo) y G (verde). Los puntos correspondientes a los vectores que forman la configuración de ventana **U** se colorean únicamente para facilitar su distinción. Se asume que la restricción de rango para el aperture es $K = K_R \times K_G$, donde $K_R = K_G = \{-2,-1,0,1,2\}$. El criterio de orden para encontrar el vector mediana de **U** está dado por la ecuación 6.5, donde el vector de referencia para el cálculo de la distancia euclidiana es $\mathbf{u}_0 = (8,0)$ y la cascada lexicográfica es (R→G).

El primer paso para obtener la configuración de aperture en color, **Z**, a partir de la configuración de ventana RG, **U**, consiste en encontrar el vector mediana de sus vectores. Las distancias euclidianas de $\mathbf{u}_1$, $\mathbf{u}_2$, y $\mathbf{u}_3$ con respecto al vector $\mathbf{u}_0$ son 3.16, 7.07, y 3.16, respectivamente. El empate de distancia euclidiana de $\mathbf{u}_1$ y $\mathbf{u}_3$, con respecto al vector $\mathbf{u}_0$, se rompe aplicando la cascada lexicográfica (R→G), según la cual $\mathbf{u}_3 < \mathbf{u}_1$. En función de esto, se obtiene la secuencia ordenada $\mathbf{u}_2$, $\mathbf{u}_3$, $\mathbf{u}_1$, donde los vectores están ordenados de manera ascendente. Por lo tanto, el vector mediana de entre los vectores de la configuración **U** es $\mathbf{z} = \mathbf{u}_3$. El segundo paso consiste en calcular la traslación de rango de **U** por el vector mediana $\mathbf{z} = \mathbf{u}_3$ (Figura 6.4-b). El tercero, y último paso, consiste en limitar, o truncar, el rango de cada componente de los vectores $\mathbf{u}_1 - \mathbf{z}$, $\mathbf{u}_2 - \mathbf{z}$, y $\mathbf{u}_3 - \mathbf{z}$, resultantes de la traslación

de rango, al espacio $K_R \times K_G$ obteniendo así $\mathbf{Z} = ((\mathbf{u}_1 - \mathbf{z})^*, (\mathbf{u}_2 - \mathbf{z})^*, (\mathbf{u}_3 - \mathbf{z})^*)$, que es la configuración de aperture en color correspondiente a la configuración **U** (Figura 6.4-c).

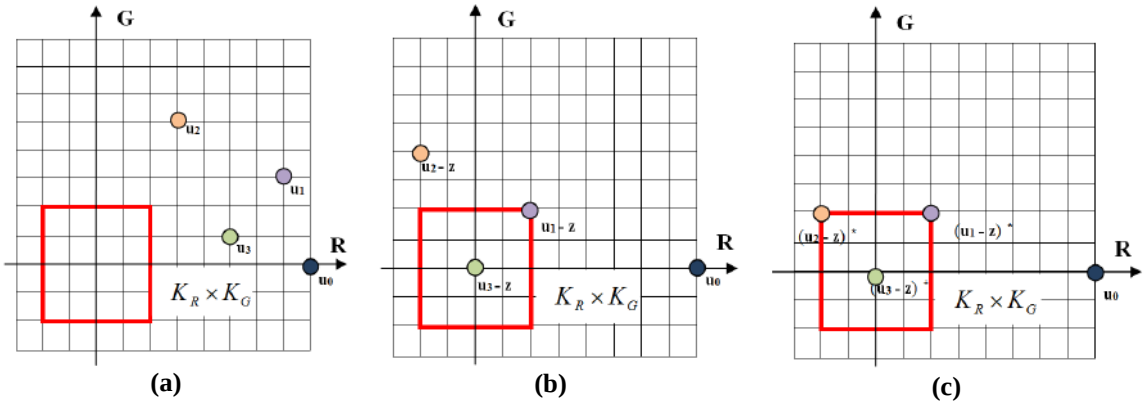


Figura 6.4. Ilustración del procedimiento para obtener la configuración de aperture a partir de una configuración de ventana en color $\mathbf{U} = (\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3)$, con $\mathbf{u}_1 = (7,3)$, $\mathbf{u}_2 = (3,5)$, y $\mathbf{u}_3 = (5,1)$. El vector mediana de los vectores $\mathbf{u}_1$, $\mathbf{u}_2$, y $\mathbf{u}_3$ se obtiene usando la relación de orden total definida en la ecuación 6.5 con el vector de referencia $\mathbf{u}_0 = (8,0)$ y la cascada lexicográfica (R→G). (**a**) Puntos correspondientes a los vectores que forman la configuración de ventana **U** en el espacio RG. (**b**) traslación de rango de **U** por $\mathbf{z} = (5,1)$. (**c**) Limitación, o truncamiento, del rango de la traslación de **U** por **z** al espacio $K_R \times K_G$.

6.4. Experimentos

En esta sección se presenta una aplicación del diseño automático de W-operadores y el preprocesamiento aperture para procesar imágenes médicas color en el espacio RGB. El problema considerado consiste en la segmentación automática de los vasos sanguíneos de las imágenes oculares color de la base de datos DRIVE.

6.4.1. Imágenes y Protocolo

El protocolo utilizado para este experimento es similar al utilizado en el capítulo anterior. La diferencia, en este caso, es que se procesan directamente las imágenes oculares color sin aplicar ningún preprocesamiento a nivel de imagen. Las ventanas utilizadas para todos los modelos testeados son $W = \{w_1,\ldots,(0,0),\ldots,w_n\}$, con $n = 15\times15$ píxeles. Esto resulta en configuraciones de ventana en color **X** conteniendo 225 vectores $\underline{X}_i = (X_{i,1},X_{i,2},X_{i,3})$, donde las componentes X_i, $X_{i,2}$, y $X_{i,3}$ contienen las intensidades de los canales de rojo, verde, y azul del *i*-ésimo píxel observado a través del punto w_i de la ventana *W*. Esto implica a su vez que la dimensión de **X** es 675.

6.4.2. Preprocesamiento de las Configuraciones de Ventana

En este experimento, para la restricción de rango del preprocesamiento aperture, se usó el producto cartesiano $K_R \times K_G \times K_B$, con $K_R = K_G = K_B = \{-10,-9,\ldots,0,\ldots,9,10\}$. Los métodos testeados para definir el vector **z** son: (1) vector correspondiente al píxel *observado*, (2) vector mediana de la configuración **X**, calculado de manera *marginal*, y (3) vector mediana de la configuración **X**, calculado usando la relación de orden *total* definida en la ecuación 6.5, con la cascada lexicográfica (G→R→B). Esta cascada se ha definido teniendo en cuenta que, en las imágenes oculares RGB, el canal que tiene el mayor contraste entre los vasos sanguíneos y el fondo, y el menor nivel de ruido es el verde. El vector de referencia

para el cálculo de las distancias euclidianas para el orden reducido es $\underline{X}_0$ = (255,255,255), correspondiente al blanco. Para fines de comparación, también se diseñaron clasificadores en base al uso de configuraciones de *ventana* en color sin preprocesamiento (4), y usando configuraciones aperture en niveles de *gris* aplicando la transformación de color a niveles de gris definida en el capítulo anterior (5).

6.4.3. Resultados, Comparaciones, y Discusión

El modelo de clasificación utilizado en este experimento para el diseño de W-operadores es la regresión logística debido al bajo costo computacional que implica el ajuste de sus parámetros. Usando cada par, de los 20 pares de imágenes de entrenamiento disponibles en la base de datos DRIVE, se diseñó un clasificador lineal de base para cada modelo testeado para el preprocesamiento de las configuraciones de ventana en color. Los 20 clasificadores de base fueron combinados en un ensamble, usando las definiciones presentadas en la Sección 5.6 del capítulo anterior. Los resultados de cada modelo se evalúan en función de la curva ROC y el valor de área (AUC) bajo dicha curva. En la Figura 6.5 se presentan los resultados obtenidos en el experimento en cuestión. Se puede notar que los valores de AUC de los tres modelos de segmentación con preprocesamiento aperture en color (total, marginal, y observado) son mayores que el valor de AUC del modelo de segmentación sin preprocesamiento de las configuraciones de ventana en color. Esto evidencia que el preprocesamiento aperture en color contribuye a disminuir el costo de diseño, o estimación, de los clasificadores con respecto al diseño basado en las configuraciones de ventana en color, sin aumentar significativamente el costo de restricción.

Comparando entre sí los valores de AUC del preprocesamiento aperture en color, se tiene que la traslación de rango utilizando el vector mediana marginal mejora los resultados con respecto al vector mediana basado en la relación de orden total, y el vector correspondiente al píxel observado en 3.6% y 2%, respectivamente. Si bien esta diferencia no es muy significativa, lo sorprendente es que los falsos colores que se crean en el preprocesamiento aperture con el vector mediana marginal no afectan la performance del ensamble de clasificadores lineales evaluado mediante la curva ROC y el valor de área bajo dicha curva (AUC). Adicionalmente, para la segmentación de los vasos sanguíneos oculares, el uso de la información de color, en el modelo RGB, para el diseño del ensamble de clasificadores lineales no mejora significativamente su desempeño en comparación con el desempeño del ensamble basado en niveles de gris. Esto se evidencia fácilmente comparando las curvas ROC y los valores de AUC del preprocesamiento aperture en color, basado en el vector mediana marginal (el mejor resultado en color), con el resultado del preprocesamiento aperture en niveles de gris (regresión logística-aperture). Esto se debe, en gran parte, a que tanto los vasos sanguíneos oculares como el fondo retiniano tienen el mismo color rojizo.

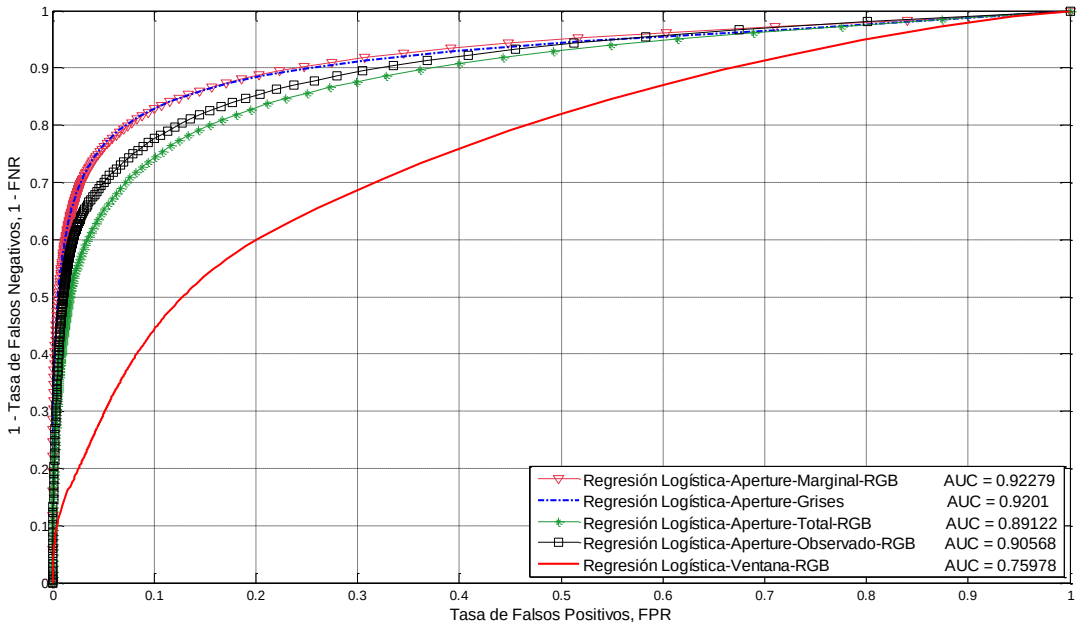


Figura 6.5. Curvas ROC y valores de AUC para los ensambles testeados en la segmentación de los vasos sanguíneos de las imágenes oculares color RGB de la base de datos DRIVE.

Con respecto al uso de la información de color para el diseño de los clasificadores lineales, la disminución del costo de diseño usando las configuraciones de aperture ocurre porque, en este caso, el espacio de búsqueda del clasificador óptimo es mucho menor que para las configuraciones de ventana. Esto sucede gracias a la reducción del rango de cada variable que forma las configuraciones de aperture. Adicionalmente, para el caso de las configuraciones de ventana, el costo de diseño de clasificadores lineales es elevado porque el nivel de correlación que existe entre sus variables es alto, tal como se evidencia en la Figura 6.6-a. En esta figura se muestra a modo de ejemplo, y utilizando un mapa de colores, la matriz de correlación entre las 675 variables que forman las configuraciones de ventana en color utilizadas para el diseño de un clasificador lineal de base. Se puede observar que la correlación para todas las variables de las configuraciones de ventana es positiva y varía en el rango entre 0.8 y 1, aproximadamente.

Las Figuras 6.6-b, -c, y -d contienen los mapas de color de las matrices de covarianza de las configuraciones de aperture en color calculadas utilizando la traslación de rango en base al vector mediana con la relación de orden total, el vector mediana marginal, y el vector perteneciente al píxel observado, o píxel a procesar, respectivamente. Comparando estos tres mapas de color se puede notar que el aperture basado en el vector mediana marginal (Figuras 6.4-c) es el que más disminuye la correlación entre las 675 variables utilizadas para el diseño de los clasificadores. Los apertures basados tanto en el vector mediana con la relación de orden total (Figuras 6.4-b) como en el vector RGB perteneciente al píxel observado (Figuras 6.6-d) disminuyen la correlación entre las variables usadas para el diseño de los clasificadores en menor proporción que el aperture basado en la mediana marginal. Las líneas vertical y horizontal que aparecen en la Figura 6.6-d corresponden a las correlaciones entre las tres variables del píxel observado y las demás variables de las configuraciones de aperture en color. Estas correlaciones tienen el valor indeterminado 0/0 debido a que la media y la varianza de estas tres variables son siempre 0 para el aperture en color basado en el vector del píxel observado.

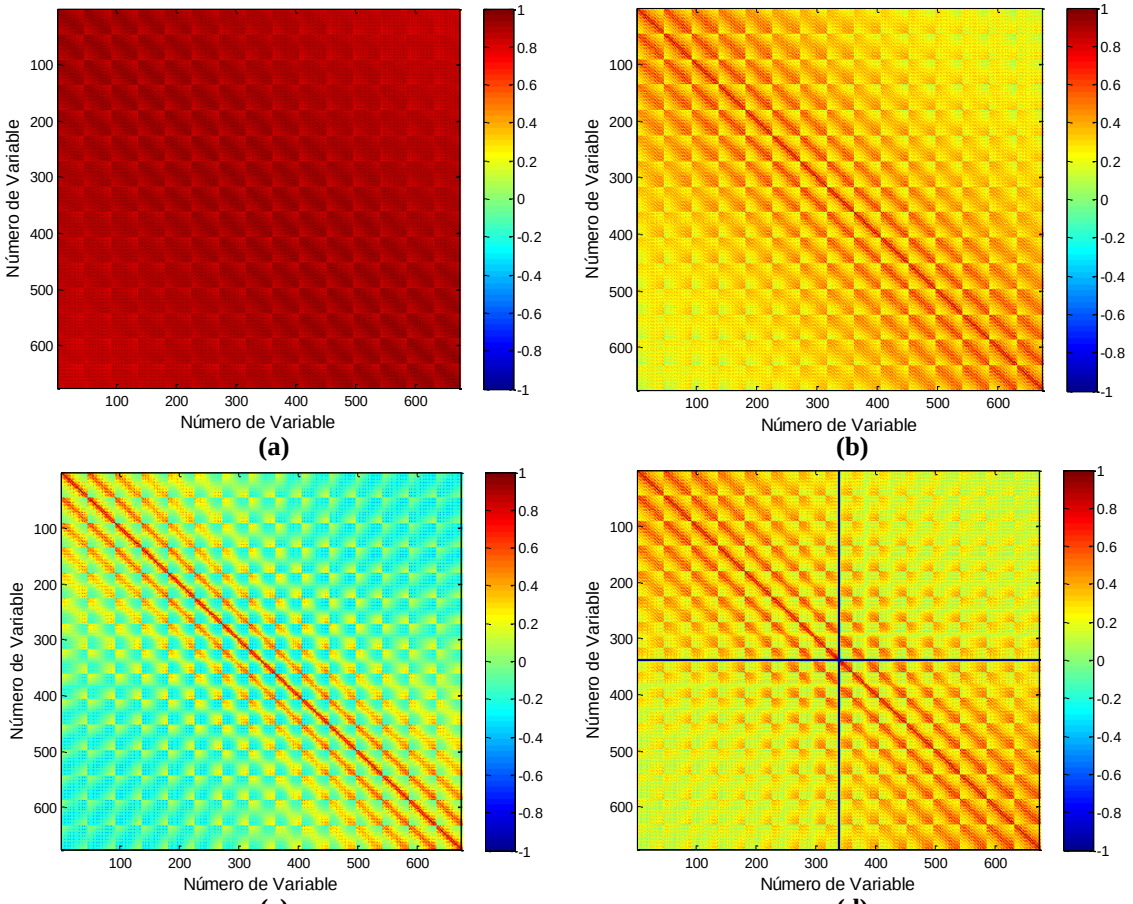


Figura 6.6. Matrices de correlación de un conjunto de ejemplos de entrenamiento para un clasificador lineal usado en la segmentación de los vasos sanguíneos en imágenes oculares RGB de la base de datos DRIVE. Para facilitar la visualización, las matrices de correlación son representadas mediante mapas de color. El color de cada píxel corresponde al nivel de correlación entre las variables de sus coordenadas. Nivel de correlación entre las 675 variables de las configuraciones de: (**a**) ventana en color, y las configuraciones de aperture en color con traslación de rango usando el vector mediana basado en una (**b**) relación de orden total y (**c**) el vector mediana marginal, y (**d**) usando el vector del píxel observado. Todas las matrices de correlación se han calculado en base a un conjunto de 270399 configuraciones.

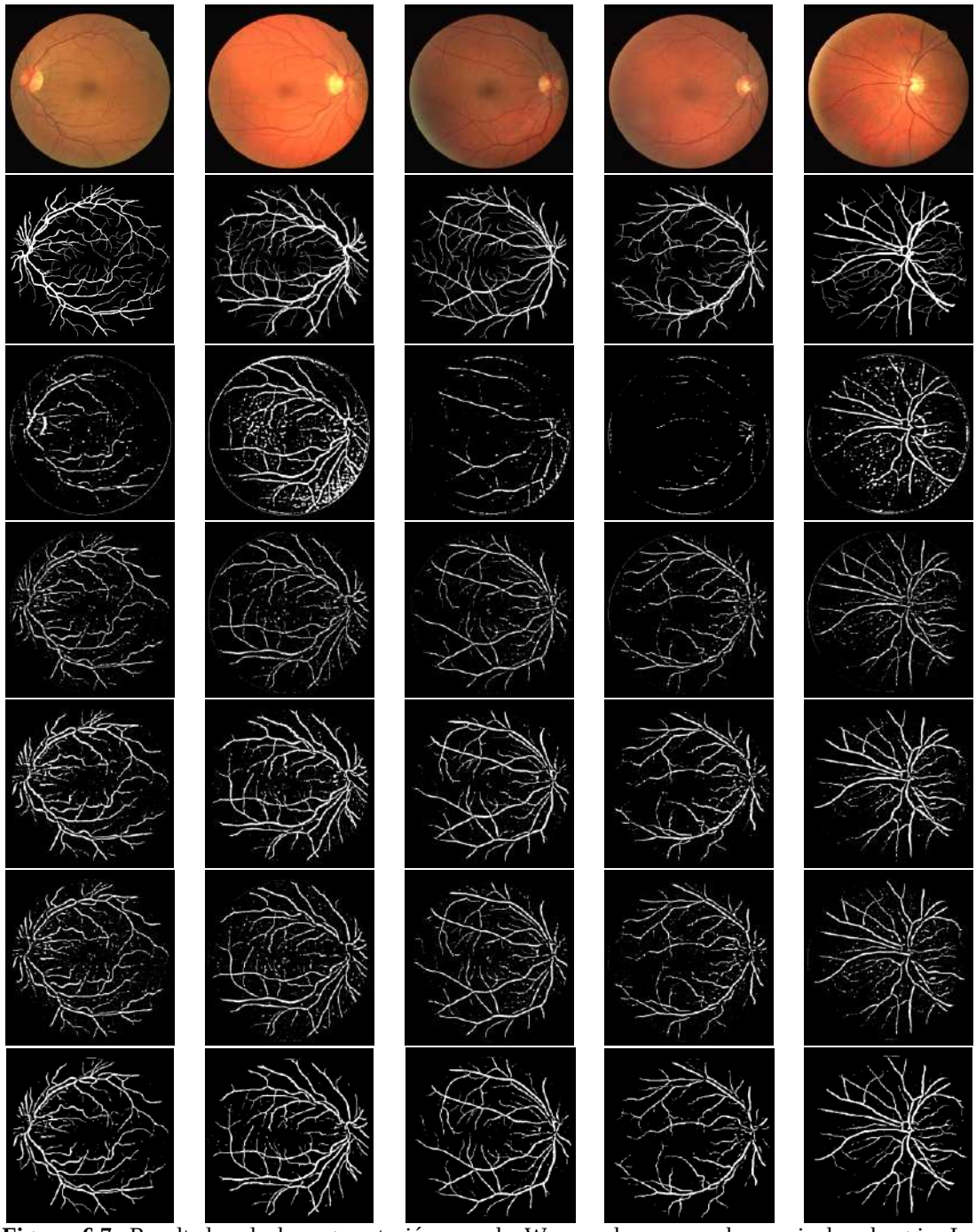


Figura 6.7. Resultados de la segmentación usando W-operadores en color y niveles de gris. La primera fila muestra las imágenes originales color RGB. La segunda fila muestra las imágenes ideales segmentadas manualmente. La tercera fila muestra los resultados del ensamble de clasificadores basado en las configuraciones de ventana en color. La cuarta fila contiene los resultados del ensamble basado en el preprocesamiento aperture en color con el vector mediana calculado usando la relación de orden total. La quinta fila muestra los resultados del ensamble con el aperture en color basado en el uso del vector mediana marginal. La sexta fila muestra los resultados del ensamble basado en el aperture en color con el vector correspondiente al píxel observado. La séptima fila muestra los resultados del ensamble basado en el método regresión logística-aperture en niveles de gris (Capítulo 5, Sección 5.9.1). Todas las imágenes segmentadas fueron obtenidas umbralando la probabilidad de clase de cada píxel en un valor de 0.3.

En la Figura 6.7 se presentan 5 imágenes oculares color RGB acompañadas de sus respectivas imágenes resultantes de la segmentación automática de los vasos sanguíneos usando W-operadores en color. La primera y segunda filas muestran las imágenes

originales y de gold estándar, respectivamente. La tercera fila muestra los resultados obtenidos al diseñar un ensamble de clasificadores lineales sin preprocesamiento de las configuraciones de ventana en color. La cuarta, quinta, sexta y séptima filas muestran los resultados de diseñar ensambles con el preprocesamiento aperture en color basado en el vector mediana usando la relación de orden total, el vector mediana marginal, y el vector correspondiente al píxel observado, y el método regresión logística-aperture en niveles de gris (Capítulo 5, Sección 5.9.1), respectivamente.

Las imágenes de la tercera, cuarta, quinta, sexta y séptima filas de la Figura 6.7 corroboran los resultados numéricos presentados en la Figura 6.5. En las imágenes de la tercera fila, se puede observar como el W-operador basado en las configuraciones de ventana es muy sensible a las variaciones del color tanto dentro de una imagen como entre imágenes. La variación del color dentro de una imagen hace que, en algunos casos, se reconozcan regiones de la retina como partes del árbol arterial ocular (falsos positivos). La variación del color entre imágenes hace que, en algunos casos, no se reconozcan ni siquiera los vasos sanguíneos gruesos (falsos negativos). Las imágenes de la cuarta fila muestran que el W-operador basado en el aperture con el vector mediana calculado usando la relación de orden total es muy sensible a la variación del color del árbol arterial. Por esta razón, en todas las imágenes, existen regiones del árbol arterial principal que no son detectadas. Las imágenes de la quinta fila evidencian que el W-operador basado en el aperture con el vector mediana marginal es capaz de segmentar, en todos los casos, el árbol arterial principal (vasos sanguíneos gruesos). Sin embargo, los vasos sanguíneos delgados son clasificados como parte del fondo. Las imágenes de la sexta fila muestran que el W-operador basado en el aperture usando el vector correspondiente al píxel observado es sensible a las variaciones de color dentro de una imagen. Por esta razón, en todas las imágenes, se detectan erróneamente algunas regiones de la retina como vasos sanguíneos. Estos falsos positivos aparecen como artefactos de ruido en las imágenes segmentadas.

En la Tabla 6.1 se resumen los valores de AUC y los datos del costo computacional de todos los métodos de diseño de W-operadores en color y en niveles de gris testeados en este experimento. El número de épocas para el ajuste de los parámetros de todos los clasificadores lineales de base es 500. Respecto a los valores de AUC, el mejor método es el ensamble basado en el aperture en color usando el vector mediana marginal seguido muy de cerca por el ensamble basado en el aperture en niveles de gris. Respecto al costo computacional se puede notar, sin ninguna sorpresa, que tanto el escaneo de las imágenes de entrenamiento, ajuste de parámetros, y testeo de los clasificadores es más costoso para las imágenes color que para las imágenes en niveles de gris. De hecho, el procesamiento en niveles de gris presenta el menor tiempo de escaneo, entrenamiento de los clasificadores, y testeo en comparación con los demás métodos evaluados. Esto se debe, principalmente, al incremento de la dimensión de las configuraciones de ventana en color, en un factor de tres, con respecto a las configuraciones de ventana en niveles de gris.

De los tres tipos de preprocesamiento aperture testeados para la información en color, el que presenta un mayor costo computacional es aquel basado en el cálculo del vector mediana usando la relación de orden total. Esto es debido a que para el criterio de orden total, definido en la ecuación 6.5, se requiere del cálculo de las distancias euclidianas entre cada vector de una configuración de ventana y el vector de referencia. Adicionalmente, para resolver los casos donde existen empates en los valores de distancia, se deben aplicar una serie de comparaciones usando la cascada lexicográfica. Todos los experimentos fueron

realizados usando un computador con procesador Intel core i-5, con velocidad de CPU de 2.30 GHz, y 8 GB de memoria RAM. Los algoritmos fueron implementados en Matlab®.

Tabla 6.1. Valores de AUC y tiempos promedios de escaneo de las imágenes de entrenamiento, ajuste de parámetros de los clasificadores (entrenamiento), y aplicación de los ensambles de 20 clasificadores de base para la segmentación de vasos sanguíneos oculares usando imágenes color RGB con 565×584 píxeles.

Método	**Valor de área bajo la curva (AUC)**	**Tiempo de escaneo por cada par de imágenes de entrenamiento [h]**	**Tiempo de entrenamiento por cada clasificador lineal de base [h]**	**Tiempo de aplicación del ensamble por cada imagen de testeo [h]**
Aperture-RGB-Mediana-Marginal	0.92279	0.97	0.48	0.06
Aperture-Grises	*0.9201*	*0.40*	*0.26*	*0.02*
Aperture-RGB-Mediana-Total	0.89122	1.10	0.73	0.39
Aperture-RGB-Observado	0.90568	0.81	0.71	0.03
Ventana-RGB	0.75978	0.73	1.12	0.03

6.5. Anotaciones Finales

Usando el paradigma de reconocimiento de patrones, en este capítulo se ha propuesto un nuevo método para el diseño automático de W-operadores para la clasificación de los píxeles de imágenes color. Las definiciones presentadas son válidas para cualquier modelo, o espacio de representación, de las imágenes color. Sin embargo, se han realizado experimentos usando únicamente el modelo RGB. Para este modelo, se han analizado las principales problemáticas que existen para el diseño automático de clasificadores. En concreto, se han analizado y propuesto soluciones para prevenir el sobreajuste de parámetros y los efectos negativos de la maldición de la dimensión para un escenario práctico con una cantidad fija y limitada de ejemplos de entrenamiento. Las soluciones propuestas consisten en la definición de restricciones tanto en la complejidad de los modelos de clasificación como en el tamaño del espacio de las configuraciones de ventana en color. Para este último caso, se han extendido las definiciones del preprocesamiento aperture para el caso donde las configuraciones de ventana están compuestas por vectores.

En base al método propuesto en este capítulo, se ha aplicado el diseño automático de W-operadores en color a la segmentación de los vasos sanguíneos en imágenes oculares RGB de la base de datos DRIVE. Los resultados obtenidos usando W-operadores en color, basados en ensambles de clasificadores lineales y preprocesamiento aperture, evidencian que el color no es un factor importante para la segmentación de los vasos sanguíneos oculares. Esto debido a que tanto los vasos sanguíneos (objeto de interés) como la retina (fondo) presentan un color rojizo muy similar entre ellos.

Trabajos futuros deben incluir la aplicación de la propuesta realizada en este capítulo para la segmentación de imágenes multicanal e imágenes en color representadas en otros espacios diferentes al RGB. Adicionalmente, se deben extender las definiciones presentadas en este capítulo para el caso donde la imagen ideal, o imagen resultante del procesamiento, sea una imagen en niveles de gris, o una imagen color.

6.6. Referencias

[1] J. Angulo, "Morphological colour operators in totally ordered lattices based on distances: Application to image filtering, enhancement and analysis," *Computer Vision and Image Understanding*, vol. 107, pp. 56-73, 2007.

[2] E. Aptoula and S. Lefèvre, "A comparative study on multivariate mathematical morphology," *Pattern Recognition*, vol. 40, pp. 2914-29, 2007.
[3] J. Astola, P. Haavisto and Y. Neuvo, "Vector median filters," *Proceedings of the IEEE*, vol. 78, pp. 678-89, 1990.
[4] V. Barnett, "The ordering of multivariate data," *Journal Of The Royal Statistical Society*, vol. 139, pp. 318–54, 1976.
[5] M.E. Benalcázar, J. Padín, M. Brun, J.I. Pastore, V.L. Ballarin, L. Peirone and G. Pereyra, "Measuring Leaf Area in Soy Plants by HSI Color Model Filtering and Mathematical Morphology," *Proc. 8th Argentinean Bioengineering Society Conference (SABI 2011) and 7th Clinical Engineering Meeting*, Mar del Plata Argentina, (2011).
[6] M.E. Benalcázar, J.I. Pastore, M. Brun and V.L. Ballarin, "Filtros Aperture para Clasificación de Imágenes Color," *Proc. Torneo Regional de Inteligencia Computacional - TRICV*, Córdoba - Argentina, 88, 2012.
[7] F.C. Flores, A.M. Polidorio and R.A. Lotufo, "Color image gradients for morphological segmentation: the weighted gradient improved by automatic imposition of weights," *Proc. Computer Graphics and Image Processing, 2004. Proceedings. 17th Brazilian Symposium on*, 146-53 (2004).
[8] R.C. Gonzalez and R.E. Woods, *Digital image processing*, Prentice Hall, Upper Saddle River, N. J., 2002.
[9] R. Hirata, E.R. Dougherty and J. Barrera, "Aperture filters," *Signal Processing*, vol. 80, pp. 697-721, 2000.
[10] A. Koschan and M. Abidi, *Digital Color Image Processing*, Wiley, New Jersey, 2008.
[11] R. Lukac, B. Smolka, K. Martin, K.N. Plataniotis and A.N. Venetsanopoulos, "Vector filtering for color imaging," *IEEE Signal Processing Magazine*, vol. 22, pp. 74–86, 2005.
[12] I. Pitas and P. Tsakalides, "Multivariate ordering in color image processing," *IEEE Transactions on Circuits Systems Video Technology*, vol. 1, pp. 247–56, 1991.
[13] J. Serra, "Anamorphoses and function lattices (multivalued morphology)," in *Mathematical Morphology in Image Processing*, E.R. Dougherty, Ed. New York: Marcel-Dekker, 1992, pp. 483–523.
[14] V. Vapnik and A. Chervonenkis, "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities," *Theory of Probability & Its Applications*, vol. 16, pp. 264-80, 1971.
[15] V.N. Vapnik, *The Nature of Statistical Learning*, Springer, New York, 2000.

CAPÍTULO VII

Discusión, Conclusiones, y Trabajo Futuro

7.1. Discusión

El Procesamiento Digital de Imágenes (PDI) es una subdisciplina aplicada del procesamiento digital de señales que se encarga de la manipulación y análisis de imágenes y video. Dentro de PDI, la morfología matemática es una técnica no lineal de procesamiento y análisis de imágenes basada en la caracterización de formas y estructuras. El origen de esta técnica se remonta a inicios de los años 60 del siglo pasado, siendo Georges Matheron y Jean Serra sus inventores. La morfología matemática se basa en dos operaciones fundamentales que son la erosión y la dilatación. Estas operaciones están definidas en función de un patrón de imagen de pequeño tamaño, relativo al tamaño de la imagen a procesar o analizar, denominado elemento estructurante. En base a este patrón se procesa o analiza una imagen, píxel a píxel, aplicando las definiciones de erosión y dilatación. Es usual que para un problema de PDI se requiera aplicar una secuencia de erosiones y dilataciones con elementos estructurantes de diferentes formas y tamaños. El diseño de un operador morfológico consiste en encontrar dicha secuencia de operaciones y elementos estructurantes.

Existen fundamentalmente dos enfoques para diseñar un operador morfológico: heurístico y automático. El enfoque heurístico aparece con el nacimiento de la morfología matemática. En este enfoque el diseñador, en base a su experiencia y conocimiento, es el que define una secuencia de erosiones y dilataciones con sus respectivos elementos estructurantes. El diseño automático surge a finales de los años 80 del siglo pasado. En este enfoque se diseña un operador morfológico en base a un conjunto de imágenes de entrenamiento y un modelo de aprendizaje computacional supervisado. El conjunto de entrenamiento contiene tanto las imágenes con el problema a resolver (imágenes observadas) como la respuesta deseada del procesamiento (imágenes ideales o de gold estándar).

Bajo el enfoque heurístico, el desempeño de un operador está altamente condicionado en la experiencia del diseñador de naturaleza subjetiva. Además, para un problema determinado, encontrar heurísticamente una secuencia adecuada de erosiones y dilataciones puede tomar un largo periodo de tiempo. Bajo el enfoque automático, el desempeño de un operador depende de la cantidad y calidad de las imágenes de entrenamiento. Adicionalmente, el desempeño también está condicionado a la similitud entre las distribuciones probabilísticas del modelo de aprendizaje y el problema en cuestión.

Varios métodos han sido propuestos en la literatura científica para el diseño automático de operadores morfológicos invariantes ante traslaciones y localmente definidos por medio de una ventana. Dentro del lenguaje morfológico, este tipo de operadores se denominan W-operadores. Los métodos propuestos se basan en el uso de la regla plug-in, *k*NN, representación morfológica, y en base a la definición de restricciones. Las restricciones han sido aplicadas tanto al espacio de configuraciones como a la clase o familia de operadores utilizados para el diseño.

La mayoría de estas propuestas para el diseño automático de W-operadores han sido aplicadas al procesamiento de conjuntos reducidos de imágenes binarias, en buena parte sintéticas, usando ventanas de pequeño y mediano tamaño. Para imágenes en escala de grises son pocos los métodos propuestos. Además, debido a su alto costo computacional, estos métodos son aplicables para resolver únicamente problemas que requieren el uso de ventanas de pequeño y mediano tamaño y cuando las imágenes a procesar contienen pocos niveles de gris. Para el caso de imágenes multicanal, y en particular de imágenes color, los métodos propuestos son muy escasos y limitados en cuanto a su aplicabilidad práctica. En estas condiciones, se concluyó que la aplicación de los métodos propuestos en la literatura científica resulta inviable desde el punto de vista práctico para resolver problemas reales de PDI.

Esta limitación práctica es aún más crítica si se tiene en cuenta que los problemas reales de PDI, y en particular de imágenes médicas, involucran grandes tamaños de imágenes o volúmenes (stack de imágenes). Además, en el campo médico, el procesamiento usualmente involucra a imágenes en escala de grises y color. Estas imágenes contienen un amplio rango de niveles de gris o vectores de color y se caracterizan por tener una gran variación estadística tanto dentro de una imagen como entre imágenes. Además, la generación de imágenes médicas es un proceso continuo, por lo tanto, los conjuntos de imágenes a procesar son grandes.

En esta tesis se propuso un nuevo enfoque para el diseño automático de W-operadores usando el paradigma de reconocimiento de patrones. La propuesta consistió en definir y representar a un W-operador mediante un clasificador para cuando el objetivo del operador es segmentar o clasificar. La entrada del clasificador es un vector formado por un número finito de variables aleatorias observadas en las imágenes a procesar a través de una ventana. La salida del clasificador es también una variable aleatoria conteniendo una etiqueta para el vector de entrada. Por lo tanto, en esta tesis se ha reducido el problema del diseño automático de un W-operador para clasificación a un problema de reconocimiento de patrones. Este problema consistió en el diseño estadístico de un clasificador. En este contexto, la principal ventaja del reconocimiento de patrones es su amplia y bien sustentada teoría matemática que permitió estudiar las propiedades de generalización dado un modelo de clasificación y el número de ejemplos de entrenamiento disponibles.

Bajo el enfoque propuesto, el primer paso del diseño consistió en seleccionar el modelo de clasificación a utilizar. Para esto se tuvo en cuenta que, según los teoremas de no free lunch, todos los modelos de clasificación tienen en promedio la misma performance calculada en función del costo 0-1. Por lo tanto, no existe ninguna razón por la que *a priori* se deba tener preferencia por uno u otro modelo. Sin embargo, para un problema determinado, aspectos prácticos como el número de observaciones disponibles, número de variables de las configuraciones de ventana, número de clases, recursos computacionales disponibles, y conocimiento a priori del problema en cuestión son los que influyeron en la selección de un modelo de clasificación.

En esta tesis se propuso el uso de la familia de clasificadores paramétricos que modelan la distribución condicional *a posteriori* de cada clase. Estos modelos particionan el espacio de configuraciones de ventana mediante la implementación de una frontera de decisión continua que es seleccionada dada una clase de funciones. En este contexto, el objetivo del diseño de un clasificador fue encontrar la frontera de decisión que mejor aproxime a la frontera de decisión óptima. Si la frontera de decisión óptima no está contendida dentro de

la clase de funciones seleccionada, entonces existe un costo de restricción. Sin embargo, la teoría de Vapnik-Chervonekis garantiza, probabilísticamente que, si la dimensión VC de la clase de funciones seleccionada es finita y mucho menor que el número de ejemplos de entrenamiento disponibles, entonces es posible lograr una buena aproximación. Esto significa que, en las condiciones establecidas, un clasificador con buena performance durante el entrenamiento tendrá también una buena performance al ser aplicado sobre las imágenes de testeo.

Se planteó el uso de la familia de modelos de clasificación paramétricos debido a que su velocidad de convergencia a una buena aproximación (costo de diseño) es, usualmente, más rápida que la de los modelos no parámetricos. Adicionalmente, los modelos paramétricos son menos propensos a la maldición de la dimensión que los modelos no paramétricos. Para un problema determinado, bajo el enfoque propuesto, se debe seleccionar un modelo de clasificación paramétrico cuya complejidad, o dimensión VC, no sea más grande que o igual al número de ejemplos de entrenamiento disponibles. Esto para evitar el sobreajuste de parámetros, u overfitting, que resulta en un bajo error de entrenamiento, pero una mala capacidad de generalización de la regla de clasificación o elevado error de testeo.

El modelo de clasificación paramétrico propuesto en esta tesis son las redes neuronales tipo feed-forward. Estas redes neuronales son aproximadores universales, lo que implica que el costo de restricción de este modelo tiende a cero cuando en la capa oculta contienen un número suficientemente grande de neuronas. En un contexto práctico, con un número fijo y limitado de configuraciones de ventana, este tipo de redes permitieron implementar fronteras de decisión complejas con un costo computacional usualmente menor que el de otros clasificadores paramétricos como por ejemplo SVMs. Una gran ventaja de las redes neuronales fue que permitieron estimar la probabilidad condicional de clase dada la configuración de ventana observada en cada píxel de las imágenes a procesar. En función de esta probabilidad, el diseñador puede seleccionar el mejor umbral para la clasificación.

Se evidenció que la representación computacional y aplicación, o testeo, de un W-operador mediante redes neuronales tiene un bajo costo computacional. Esto debido a que, para la etapa de testeo, sólo se requiere el almacenamiento en memoria de los parámetros ajustados en la etapa de entrenamiento. Este conjunto de parámetros es, usualmente, mucho más pequeño que el tamaño de las tablas de búsqueda, el conjunto de elementos estructurantes de la base, y la colección de intervalos maximales. Estas formas de definir y representar a la función característica de un W-operador son las utilizadas por los métodos de diseño propuestos en la literatura científica previo a la realización de esta tesis.

Para reducir el costo de estimación o diseño de un clasificador, dada una arquitectura de red neuronal, en esta tesis se propuso aplicar en algunos casos un preprocesamiento de las configuraciones de ventana. Las propuestas realizadas consistieron en el uso de una transformación de color a niveles de gris cuando las imágenes a procesar son imágenes color. También se propuso el uso de la transformada wavelet discreta 2D de Haar, y aperture para imágenes en escala de grises e imágenes color. Estos tipos de preprocesamiento permitieron reducir el tamaño del espacio de configuraciones de ventana. Esta reducción implicó una restricción en el espacio de búsqueda del operador óptimo. Sin embargo, con estos preprocesamientos la mejora del costo de estimación excedió, usualmente, al costo de restricción, por lo cual el uso del espacio búsqueda restringido representó un beneficio para el diseño de los clasificadores.

Se evidenció que para las configuraciones de ventana, es usual que exista una alta correlación entre sus variables. Esto debido a que los niveles de gris o vectores de los píxeles más cercanos al píxel a procesar, usualmente, son similares o tienen poca variación. Esta correlación es detrimental para el diseño de clasificadores debido a que la cantidad de información efectiva disponible para el ajuste de parámetros de los clasificadores es más baja que la que se dispondría si las variables fueran independientes. Adicionalmente, para el caso de imágenes color representadas en el modelo RGB, también se evidenció que la correlación se presenta no sólo entre los vectores de píxeles cercanos, sino también entre las componentes del vector de un píxel. Esta correlación obliga al aumento de la cantidad de ejemplos de entrenamiento para reducir el costo de diseño y por ende también disminuir el error de clasificación. El problema en este caso radica en que el número de ejemplos de entrenamiento es fijo. En esta tesis se ha evidenciado, empíricamente, que el preprocesamiento aperture reduce el nivel de correlación entre las varibles de las configuraciones de ventana.

En el diseño de clasificadores para la segmentación de imágenes en escala de grises y color se enfrentaron dos problemas prácticos importantes. El primer problema ocurre cuando existe un desbalance entre las frecuencias de clase de los píxeles pertenecientes al objeto a segmentar y el fondo de la imagen. Cuando este desbalance se da a favor del fondo, el clasificador diseñado predice casi siempre la etiqueta correspondiente al fondo. Esto hace que, en la imagen resultante del procesamiento, la mayor parte o incluso todos los objetos de interés aparezcan como parte del fondo. A pesar de la incapacidad del operador diseñado para segmentar el objeto de interés, es usual que su nivel de error sea bajo, lo cual aparenta un buen desempeño. Este bajo nivel de error empírico se debe a la supremacía de los píxeles del fondo. Sin embargo, para este caso, la tasa de falsos negativos es muy alta comparada con la tasa de falsos positivos, la cual en este caso puede ser incluso nula. Para la solución de este problema, en esta tesis se propuso el uso de un balanceo artificial de las frecuencias de clase de las configuraciones de ventana. Esta solución permitió que el ajuste de parámetros de un clasificador minimice el promedio entre su tasa de falsos positivos y falsos negativos. La principal ventaja de esta propuesta es que no aumenta la carga computacional del diseño de un clasificador. Esto debido a que el balanceo artificial consiste solamente en una multiplicación de las frecuencias de clase por constantes.

El segundo problema ocurre cuando el número total de configuraciones de ventana es más grande que los recursos computacionales (memoria RAM) disponibles para el diseño. En estas condiciones diseñar un único clasificador usando, simultáneamente, todo el conjunto de entrenamiento resulta intratable desde el punto de vista práctico. Para resolver este problema se propuso el diseño de un ensamble de clasificadores. Bajo esta propuesta, se planteó el diseño de un clasificador por cada par de imágenes de entrenamiento disponible. La dimensión VC de los clasificadores de base debe estar acorde con la cantidad de configuraciones de ventana que se pueden obtener a partir de cada imagen de entrenamiento. Esto para disminuir el riego de overffitting o sobreajuste de parámetros. La predicción del ensamble se obtuvo mediante una suma ponderada de las predicciones de cada clasificador de base.

Para el esamble de clasificadores propuesto en esta tesis, dada una configuración de ventana, el peso de la probabilidad condicional estimada con cada clasificador de base se calculó usando la entropía de Shannon. De esta manera se asigna un peso alto a las predicciones con elevada entropía y un peso bajo, o incluso nulo, para aquellas predicciones que no informativas o son ambiguas. Con esta propuesta se logró reducir el

costo computacional de diseño de clasificadores para la segmentación de imágenes en escala de grises y color. En contraste con esta ventaja, el uso del ensamble puede acarrear un incremento del costo de restricción. Sin embargo, se presentó evidencia de que el ensamble permite implementar fronteras más complejas que las fronteras de decisión de los clasificadores de base. En función de esto es esperable que el costo de restricción del ensamble no sea tan alto. Además, se demostró matemáticamente que el ensamble propuesto en esta tesis permite disminuir el error promedio de aproximación de los clasificadores de base.

Se evidenció que el mayor costo computacional del diseño automático de W-operadores usando redes neuronales, especialmente para imágenes en escala de grises y color, radica en el escaneo de las imágenes de entrenamiento para obtener las configuraciones de ventana. Otra etapa crítica, desde el punto de vista computacional, consistió en el ajuste de los parámetros de la red. Una vez entrenadas las redes del ensamble, se evidenció que su aplicación en la etapa de testeo es rápida debido a que para esto se pueden utilizar las operaciones de convolución o correlación de imágenes y transformaciones píxel a píxel. Las operaciones de convolución o correlación se pueden aplicar siempre y cuando no se utilice un preprocesamiento no lineal para las configuraciones de ventana. Para los casos donde se requiera de un preprocesamiento no lineal, por ejemplo, el aperture o wavelets, se evidenció un leve incremento en el costo computacional. Sin embargo, este incremento de tiempo no es significativo en comparación con los tiempos sin preprocesamiento. Además en este caso, el preprocesamiento de las configuraciones de ventana mejoró la calidad de los resultados.

En el contexto de PDI, se mostró que las redes neuronales tipo feed-forward de tres capas (entrada, oculta, y salida) pueden ser interpretadas como un banco de filtros espaciales conectado a un clasificador. El banco de filtros está formado por las neuronas de la capa oculta. El clasificador lo componen las neuronas de la capa de salida. Dada una configuración de ventana, la capa oculta retorna la respuesta de cada una de sus neuronas, o filtro, para dicha configuración. De esta manera se obtiene un vector que pertenece a un nuevo espacio de características, cuya dimensión es igual al número de neuronas de la capa oculta. En base a este nuevo vector de características, los clasificadores de la capa de salida asignan una etiqueta a la configuración de ventana de la capa de entrada. Para la regresión logística, una red neuonal tipo feed-forward con una sóla neurona, la clasificación de una configuración de ventana se realiza en función de la respuesta de un único filtro.

Usando esta interpretación de banco de filtros para las redes neuronales aplicadas a PDI, en esta tesis se mostró cómo calcular la configuración de ventana que maximiza la respuesta de cada neurona, o filtro, que forma la capa oculta. Los valores para las variables de la configuración de ventana que maximiza la respuesta de una determinada neurona de la capa oculta son función de los pesos o parámetros de dicha neurona. Para un problema particular, esta información puede ser útil para visualizar y tratar de comprender el conjunto de características que busca una red neuronal en las imágenes a procesar. Sin embargo, se evidenció que no siempre es posible lograr esta comprensión debido a que los patrones que maximizan las respuestas de las neuronas de la capa oculta pueden resultar demasiado abstractos para la comprensión humana.

En base a los avances teóricos y prácticos conseguidos en esta tesis, se aplicó el diseño automático de W-operadores a problemas reales que involucraron el uso de imágenes binarias, en escala de grises, y color. Para el caso de las imágenes binarias, los

W-operadores fueron utilizados para el filtrado de ruido de imágenes oculares segmentadas con morfología matemática difusa, detección de bordes en imágenes contaminadas con ruido artificial tipo sal y pimienta con densidad 0.1, identificación de texturas en escaneados de mapas, y reconocimiento del carácter "a" en escaneados de texto. Las ventanas utilizadas para estos experimentos fueron de mediano y gran tamaño que van entre 5×5 a 11×11 píxeles. En todos los casos testeados, el diseño de W-operadores usando redes neuronales permitió obtener, a nivel general, mejores resultados en performance y costo computacional que aquellos basados en *k*NN, multi-resolución piramidal, y SVMs. Para la detección de bordes, los resultados obtenidos fueron también superiores a los del gradiente morfológico por erosión. Para el reconocimiento de la letra "a" los resultados de los W-operadores con redes neuronales también superaron a los resultados de las redes neuronales convolucionales.

Para el caso de imágenes en escala de grises, el diseño automático de W-operadores fue aplicado a problemas reales de segmentación de imágenes médicas. Los problemas considerados consistieron en la segmentación tanto de vasos sanguíneos como de exudados en imágenes de fondo ocular, y la segmentación de la glándula prostática en volúmenes de resonancias magnéticas T2W. Las ventanas utilizadas en estos experimentos son ventanas de gran tamaño con 15×15, 16×16, y ventanas ralas de 37×37, con una cantidad efectiva de 19×19 píxeles. Se utilizaron imágenes y volúmenes provenientes de bases de datos públicas, lo cual permitió realizar una comparación de los resultados obtenidos en esta tesis con los resultados de otros métodos propuestos en la literatura científica. En todos los casos testeados, los resultados obtenidos en esta tesis fueron comparables a los mejores resultados publicados en la literatura científica por cada base de datos utilizada. Esta es la primera vez que se aplicaron los W-operadores a problemas reales de PDI que involucran en general gran cantidad de imágenes a procesar, y en particular, imágenes médicas.

Para el caso de imágenes en color, se aplicó el diseño de W-operadores a la segmentación de los vasos sanguíneos en imágenes oculares en el espacio de color RGB. Para este problema de segmentación se utilizaron ventanas de gran tamaño formadas por 15×15 píxeles. Las redes neuronales que se usaron para este caso fueron menos complejas que aquellas que se utilizaron para cuando las imágenes oculares fueron preprocesadas aplicando una transformación de color a nivel de gris. Esto para disminuir tanto el costo de estimación como el costo computacional de diseño de los clasificadores debido al incremento de la dimensión de las configuraciones de ventana.

En la segmentación de los vasos sanguíneos en imágenes oculares de la base de datos DRIVE, los W-operadores en escala de grises diseñados en base a redes neuronales tipo feed-forward fueron capaces de segmentar por completo los vasos sanguíneos del árbol arterial principal (vasos sanguíneos gruesos). También se evidenció que estos W-operadores son robustos a las variaciones de intensidad tanto dentro de una imagen como entre imágenes oculares. En contraste con estas ventajas, se observó también que los vasos sanguíneos delgados no fueron segmentados. Para este problema de segmentación también se evidenció que la incorporación de la información de color en el espacio RGB no contribuye a la mejora de la calidad de la segmentación. Esto para cuando se utilizaron W-operadores en color diseñados en base a regresión logística y preprocesamiento aperture.

La segmentación de exudados en imágenes oculares fue evaluada mediante la clasificación de las imágenes procesadas como imágenes con o sin edema macular diabético (DME). Los resultados obtenidos en esta tesis son comparables con los dos mejores métodos para la clasificación de DME sobre la base de datos utilizada: HEI-MED. Sin embargo, los valores de área bajo lo curva ROC obtenidos en esta tesis, y en los trabajos de la literatura científica usados para la comparación, evidencian que este problema requiere de mayor investigación para mejorar la calidad de los resultados.

En la segmentación de la glándula prostática se evidenció que los W-operadores en escala de grises basados en redes neuronales tienen un desempeño que es comparable con los mejores métodos aplicados sobre las resonancias magnéticas del *NCI-ISBI 2013 Challenge*. En particular, se observó que los W-operadores son capaces de segmentar la glándula central de cada uno de los cortes analizados. Sin embargo, también se observó que con frecuencia fallan, a nivel de falsos positivos y falsos negativos, en la segmentación de la zona periférica. Estos resultados, y los obtenidos en los trabajos usados para la comparación, evidencian que este problema de segmentación todavía no ha sido resuelto y por lo tanto requiere de nueva investigación.

7.2. Conclusiones

El diseño automático de operadores morfológicos permite reducir la subjetividad del diseño heurístico y al mismo tiempo acelerar la resolución de un problema. Sin embargo, esto no implica para nada que el diseño heurístico no tenga validez práctica. Esto porque no siempre es posible, o porque puede resultar un proceso demasiado complicado, el obtener las imágenes ideales requeridas para el diseño automático.

Los métodos propuestos en la literatura científica para el diseño automático de operadores de ventana o W-operadores involucran, usualmente, un alto costo computacional de diseño y testeo. Adicionalmente, la mayoría de ellos son vulnerables a la maldición de la dimensión. Esto implica que, para obtener una buena estimación, el número de ejemplos de entrenamiento debe aumentar de manera exponencial cuando se incrementa el tamaño de la ventana. Por esta razón, su aplicación a problemas que involucran imágenes reales binarias, en escala de grises, y color conteniendo un gran número de píxeles, y ventanas de mediano y gran tamaño resulta impráctico.

Las **principales contribuciones** de esta tesis son:

- Se realizó una exhaustiva revisión bibliográfica y un completo análisis teórico de los métodos propuestos en la literatura científica para el diseño automático de operadores de ventana o W-operadores.
- Basado en un sólido análisis teórico, se propuso un nuevo paradigma para el diseño automático de W-operadores usando reconocimiento de patrones.
- Se formuló y analizó teóricamente la definición y representación computacional de un W-operador usando redes neuronales tipo feed-forward de tres capas y regresión logística. Este es un método general para problemas de clasificación y segmentación de imágenes binarias, en escala de grises, y color RGB usando ventanas de pequeño, mediano, y gran tamaño.
- Se propuso y analizó teóricamente el diseño de un ensamble de clasificadores para escenarios prácticos con una limitada capacidad computacional.

- Se propuso el uso de un balanceo artificial de las frecuencias de clase para conseguir un equilibrio entre la tasa de falsos positivos y la tasa de falsos negativos durante el entrenamiento de un clasificador.
- Se propuso realizar el preprocesamiento de las configuraciones de ventana usando aperture, una transformación de color a niveles de gris, y la transformada wavelet 2D de Haar.
- En base a los avances teóricos propuestos en esta tesis, se hizo posible el uso de los W-operadores para la segmentación de imágenes médicas.
- En particular, se consideraron los problemas de segmentación de vasos sanguíneos y exudados en imágenes oculares, y la segmentación de la glándula prostática en resonancias magnéticas T2W.
- Para estos problemas se realizaron comparaciones de los resultados obtenidos en esta tesis con los mejores métodos propuestos para cada caso. Adicionalmente, se presentó un detalle del costo computacional del uso del paradigma propuesto.

7.3. Trabajo Futuro

Las definiciones presentadas en esta tesis para W-operadores para segmentación y clasificación de imágenes en escala de grises y color RGB son válidas para el caso general donde las imágenes ideales contienen más de dos o tres clases. En esta tesis el testeo se ha realizado considerando sólo dos y tres clases y sobre tres problemas diferentes. En función de lo anterior, un trabajo futuro consiste en el testeo de los modelos propuestos para dar solución a otros problemas de segmentación y clasificación de imágenes en escala de grises y color.

Los W-operadores en color sólo han sido testeados en imágenes RGB. Por lo tanto, queda pendiente su aplicación y análisis para otros modelos de representación del color. De la misma manera, también se puede aplicar el diseño automático de W-operadores para el procesamiento de imágenes espectrales y video.

Otro trabajo futuro consiste en la aplicación de reconocimiento de patrones para definir operadores de ventana para la solución de problemas de regresión. Para este caso, las imágenes de entrada y salida del procesamiento son imágenes en escala de grises. Para este problema también se puede utilizar a las redes neuronales tipo feed-forward.

El tamaño y forma de la ventana ha sido definido heurísticamente tomando en cuenta las características de los objetos a reconocer o segmentar. Bajo este procedimiento, es posible que no todos los píxeles dentro de la ventana seleccionada contribuyan significativamente a la clasificación. Sin embargo, estos píxeles obligan a aumentar el número de ejemplos de entrenamiento para evitar el overfitting y disminuir el costo de estimación. Por este motivo, otro trabajo futuro consiste en incorporar, como parte del diseño de los clasificadores, un procedimiento de selección de caraterísticas. En base a este procedimiento, el objetivo es determinar de manera automática la ventana a ser usada para un problema determinado.

Con el continuo incremento de la capacidad de procesamiento y almacenamiento de los computadores, en un futuro se puede considerar el diseño de W-operadores usando redes neuronales tipo feed-forward con más de tres capas. En este caso, el entrenamiento de los clasificadores se podría realizar usando simultáneamente todo el conjunto de imágenes de entrenamiento disponibles para un problema. Esto permitiría mejorar el desempeño de los operadores debido a la reducción del costo de restricción y estimación que se consigue

aumentando la dimensión VC de las redes neuronales y el número de configuraciones de ventana, respectivamente.

Una línea general de investigación futura, aún no explorada dentro de PDI, es el diseño automático de W-operadores para el caso de imágenes observadas e ideales en color. Un problema aún más general es aquel donde las imágenes de entrada y salida del procesamiento son imágenes multicanal.