UNMDP - Facultad de Ingeniería - Departamento de Informática

Desarrollo de una herramienta de análisis automático de texto para usuarios inexpertos del ámbito judicial.

Proyecto final para optar al grado de Ingeniero en Informática

Autores

Pablo Agustin Buendia
pabloagustinbuendia@gmail.com

Valentina Fernández
valen.fernandez.montenegro@gmail.com

Director

Ing. Bruno Constanzo

Co-Director

Ing. Martín Castellote

Mar del Plata, 1 de febrero de 2024

UNMDP - Facultad de Ingeniería - Departamento de Informática

Desarrollo de una herramienta de análisis automático de texto para usuarios inexpertos del ámbito judicial.

Proyecto final para optar al grado de Ingeniero en Informática

Autores

Pablo Agustin Buendia
pabloagustinbuendia@gmail.com

Valentina Fernández
valen.fernandez.montenegro@gmail.com

Director

Ing. Bruno Constanzo

Co-Director

Ing. Martín Castellote

Mar del Plata, 1 de febrero de 2024

Índice

Resumen

Con el crecimiento de las comunicaciones digitales, la evidencia judicial proveniente de estos medios ha aumentado exponencialmente, incrementando la complejidad y el tiempo requerido para su procesamiento y análisis. A veces, el volumen de datos supera los recursos disponibles, haciendo indispensable el uso de herramientas de automatización. Este proyecto tiene como objetivo agilizar y mejorar el análisis de evidencia textual, a partir del desarrollo de un sistema que integre modelos de procesamiento de lenguaje natural y los ponga a disposición de usuarios sin experiencia en el campo de la IA.

Introducción

En la actualidad, con el crecimiento de las redes sociales y las comunicaciones digitales en general, se generó un aumento exponencial de evidencia judicial encontrada en estos medios. Este fenómeno llevó al aumento de la complejidad y el tiempo que se debe dedicar al procesamiento y análisis de esta evidencia. En ocasiones este volumen llega a sobrepasar los recursos disponibles, por lo que el uso de herramientas de automatización del procesamiento se vuelven indispensables.

Específicamente, al tratar con evidencia en forma de texto, surge la necesidad de procesar de forma automática el lenguaje que usan las personas para comunicarse. Debido a la ambigüedad propia del lenguaje, ese procesamiento se vuelve una tarea compleja. La programación tradicional, que se basa en reglas y algoritmos definidos, a menudo no es capaz de procesar y analizar el lenguaje humano de forma exitosa. Es por eso que se opta por utilizar técnicas de inteligencia artificial, entrenando modelos específicamente para tareas de análisis de texto. En lugar de intentar definir todas las reglas y excepciones del lenguaje en un programa, se utilizan modelos de aprendizaje automático que pueden definir estas reglas a partir de datos.

Este tipo de herramientas pueden convertir el trabajo de varios días o semanas en una actividad que se puede completar en sólo unas horas. Su aplicación posibilita el uso de recursos de computación para pre-procesar información, de manera que una persona pueda volcarse a analizar sólo aquello que fue detectado de interés.

El objetivo de este proyecto es agilizar y mejorar el análisis de evidencia textual, y contribuir en el desarrollo de sistemas en el campo de la informática forense. En particular, se ha tomado la violencia de género como un caso concreto y urgente. Es importante destacar que todo el desarrollo realizado tiene la capacidad de ser extrapolado a otras investigaciones y delitos, ya sea mediante modelos genéricos o específicamente entrenados para distintos contextos. Se busca crear un sistema capaz de integrar modelos para facilitar su uso a usuarios inexpertos, sin conocimientos previos de inteligencia artificial.

Contexto

Violencia de género en Argentina

Hoy en día las interacciones cotidianas entre las personas están atravesadas por la tecnología, las redes sociales y las aplicaciones de mensajería. Muchas de estas interacciones terminan contenidas en los dispositivos electrónicos tales como celulares, notebooks, tablets, etc. Es por eso que varias de las investigaciones criminales actuales cuentan con un amplio volumen de evidencia digital a analizar. Los delitos relacionados con la violencia de género no son ajenos a esto.

La referente funcional de este proyecto es la Licenciada en Criminalística Lucía Algieri Perito, analista del Ministerio Público del Departamento Judicial de Necochea, quien se encarga del análisis de la evidencia digital en dispositivos celulares. En su experiencia con las formas de investigación con evidencia digital, la Licenciada afirma que es común encontrarse con dispositivos que contienen gran cantidad de mensajes los cuales, a fin de identificar cuestiones de violencia de género para poder clasificarse, requieren una lectura completa del contenido. Este contenido puede ser muy largo, llegando a comprender hasta varios años de conversaciones, ya que no es inesperado encontrarse con casos en que las víctimas denuncian años de hostigamiento.

Para el análisis de este contenido de evidencia digital, si bien se utilizan herramientas de visualización dentro de las cuales se pueden realizar búsquedas de palabras, nombres, etc. sigue siendo necesario realizar una lectura completa de todas estas conversaciones a fin de poder identificar patrones como palabras o insultos repetidos y que pueda ser utilizado para realizar una búsqueda por palabra. Este método de búsqueda es exhaustivo, lento, y mentalmente desgastante para el investigador. También podría darse el caso de que cambie el insulto o palabra buscada, ya que es muy común en contextos de violencia de género que las relaciones fluctúen entre periodos de violencia y periodos sin violencia. De esta forma también se pierden datos que podrían resultar de interés por lo que los investigadores se ven obligados a redoblar sus esfuerzos para no cometer errores.

La necesidad de encontrar mejores métodos para analizar y buscar patrones violentos en las conversaciones se pone en evidencia cuando se

considera la frecuencia de femicidios en Argentina. Desde enero hasta agosto de 2023 se registró un femicidio cada 28 horas, en el 72,3% de los casos perpetrados por personas conocidas por las víctimas. Muchas de las víctimas habían denunciado previamente a sus agresores y contaban con medidas de protección.[18] Es evidente que se necesita una acción rápida y efectiva para prevenir estos crímenes, y para eso es fundamental agilizar los procesos para que los investigadores puedan presentar sus resultados a la Justicia y ayudar a las víctimas lo más rápido posible.

En este contexto, la posibilidad de desarrollar herramientas tecnológicas que agilicen la identificación de patrones en las comunicaciones podría ser crucial para salvar vidas y brindar una respuesta más eficaz a esta crisis. Estas herramientas podrían lograr acotar el tiempo de proceso de recolección de pruebas y análisis de patrones de abuso de días o inclusos meses a pocas horas. Además, proporcionarían un respaldo crucial a las víctimas, fortaleciendo sus casos ante la ley.

Necesidad de datos

Para la creación de cualquier clase de herramientas que utilicen inteligencia artificial, es necesario contar con datos reales que puedan ser utilizados como ejemplos para el entrenamiento de los modelos (ver *Marco de referencia*).

Al lidiar con datos que en su mayoría deben ser provenientes de organismos públicos se debe tener en cuenta que la creación de datos es una tarea costosa. No sólo deben ser recopilados y almacenados, sino que adicionalmente requieren un proceso de anonimización previos a ser compartidos al público, acorde a las leyes de protección de datos vigentes en nuestro país.[21] Este proceso demanda tiempo y recursos para desvincular todos los datos personales o identificativos presentes.

En nuestro país, y gran parte del resto de Latinoamérica, la elaboración y disponibilidad de datos con perspectiva de género es limitada. Antes del año 2015, no existían estadísticas oficiales sobre femicidios o violencia de género. Hasta ese momento los datos se construían a partir del trabajo de organizaciones de la sociedad civil. La creación del registro oficial fue una respuesta a la movilización social “Ni una menos” de 2015, donde más de 300.000 personas marcharon exigiendo por la actuación del Estado para frenar la violencia contra las mujeres. Aún hoy en dia quedan muchos desafíos pendientes.[19]

Actualmente los datos existentes en todo el país se presentan de forma fraccionada, con baches temporales o variables faltantes, haciendo difícil su uso. Esto no solo dificulta la creación de herramientas tecnológicas, sino que hace más difícil tomar dimensión de la incidencia de la violencia de género en nuestro país y dificulta la elaboración de políticas públicas para mitigar el problema. [20] Entre las entidades que buscan dar una solución a estos problemas se destacan el Juzgado Penal, Contravencional y de Faltas N°10 de la Ciudad de Buenos Aires, que mantiene una iniciativa de apertura de una base de datos con perspectiva de género, y el Juzgado N°13 de la Ciudad de Buenos Aires que replicó la experiencia con su propia política de datos abiertos.

Trabajo previo

La temática para el desarrollo de este proyecto surgió a partir del desarrollo de las Prácticas Profesionales Supervisadas (PPS) realizadas en el Laboratorio de Investigación y Desarrollo de Tecnología en Informática Forense (Infolab), el demandante de este proyecto.

Como parte de estas prácticas profesionales se realizó un trabajo exploratorio de la librería de procesamiento de lenguaje natural (NLP) *SpaCy* sobre un dataset de casos de violencia de género anonimizado provisto por el Juzgado Penal, Contravencional y de Faltas N°10 de la Ciudad de Buenos Aires, Argentina. Este juzgado mantiene una singular política de datos abiertos que hizo posible esta exploración. Primero se realizó una limpieza y normalización del dataset. A partir de esto se extrajo un conjunto de oraciones denominadas “frases de agresión” compuesta por frases reales escritas por los agresores hacia sus víctimas.

Como parte de la exploración de la librería se utilizó una funcionalidad de reentrenamiento de distintos componentes del *pipeline* y de entrenamiento de modelos, además de las características de búsqueda avanzada en texto en las frases encontradas. En la exploración se distinguieron que algunas frases de agresión contenían amenazas de violencia de distintos tipos, por lo que decidió utilizar esta librería para encontrar amenazas de violencia física en las frases. Para esto, se consideraron varias alternativas: detectar palabras agresivas individuales con un modelo, utilizar un buscador de palabras avanzado o utilizar un clasificador de texto. Este último probó ser la mejor de las alternativas, por considerar todo el contexto dentro de cada frase. Se realizó un entrenamiento con solo 200 ejemplos de un modelo de clasificación de texto limitado con el objetivo de encontrar amenazas de violencia física.

Posteriormente, se extendió la idea del uso de la clasificación de texto para identificar violencia de género y detectar otros tipos de violencia en la escritura de un informe junto con el juez Pablo Casas. Este informe detallaba la posible aplicación de esta tecnología y la importancia de la existencia de políticas de datos abiertos.

Al finalizar las PPS se presentó la oportunidad de participar en el Hackatón “Soluciones tecnológicas para combatir la violencia de género” organizado por el entonces Ministerio de Desarrollo Productivo de la Nación. Este evento facilitó el contacto con diversos expertos en el dominio y abrió el espacio para discutir los diversos problemas que se enfrentan en el tratamiento de casos de violencia de género.

El Hackaton llevó posteriormente a una beca de investigación donde se siguió trabajando en temáticas de violencia de género y tecnología. La beca consistió en el desarrollo de un sistema web para contener registros de los datos de las víctimas de violencia, de forma que estos sean accesibles para la justicia y las mismas víctimas. El objetivo de este sistema era minimizar la revictimización y generar datos utilizables para la toma de decisiones y la generación de políticas públicas. Esta beca permitió contactar a otros actores del dominio como la fiscal general de Necochea encargada de casos de violencia de género. El desarrollo de un sistema cuyo usuario final son las mismas víctimas de violencia conllevó también entender con mayor profundidad la psicología de las víctimas y sus necesidades.

En resumen, a partir del conocimiento adquirido en el contacto con los diversos actores, es claro que la violencia de género es un problema urgente que abarca todos los ámbitos de nuestra sociedad y que requiere de que los procesos judiciales sean lo más eficaces y eficientes posibles. El presente proyecto surge como una propuesta de solución a la creciente complejidad del análisis de evidencia textual en el ámbito judicial y como continuación al trabajo realizado en las PPS mencionadas. Ante el desafío que representa el procesamiento de grandes volúmenes de datos en un contexto donde los recursos son limitados, se tomó la decisión de desarrollar un sistema que integre modelos de NLP para dar una solución a este problema. Desde la Universidad Nacional de Mar del Plata y el Infolab se remarca la importancia de abordar la problemática de la violencia de género y utilizar la tecnología como herramienta para generar un impacto positivo en la sociedad.

Objetivos del proyecto

Problemática a resolver

Las experiencias de uso de Inteligencia Artificial (IA) en ámbitos judiciales son limitadas. Usualmente, se orientan a proyectos con objetivos amplios y poco específicos. Es común que se desconozcan las capacidades y limitaciones de dichos sistemas. En algunos casos, incluso, sin aclarar los objetivos o el contexto del uso de la inteligencia artificial en estos proyectos.

Actualmente, existen dos métodos principales para el análisis de evidencia textual:

- De forma manual, lo que implica el esfuerzo de incorporar, analizar y clasificar los datos. Esto es un proceso tedioso y que puede demandar una gran cantidad de tiempo y recursos humanos.
- A partir del uso de herramientas informáticas complejas existentes, como Autopsy, que requieren del usuario contar con un extenso conocimiento previo.

Las herramientas informáticas proporcionan un mayor grado de automatización al proceso de análisis y diagnóstico, ya que permiten procesar altos volúmenes de datos en cortos periodos de tiempo y tratar casos reales de la forma más eficiente posible. Es necesario que los usuarios que no pueden acceder a este último tipo de herramientas fácilmente puedan aprovechar la potencialidad del análisis automático.

Objetivos

El objetivo de este proyecto es desarrollar un sistema que utilice técnicas de inteligencia artificial para analizar textos cargados a través de una interfaz web. Más precisamente, se buscan desarrollar modelos predictivos que puedan ser utilizados en el campo de la investigación judicial. Incluyendo, en particular, modelos que puedan ser utilizados para la identificación de casos de violencia de género.

Este sistema busca ofrecer una interfaz intuitiva para usuarios inexpertos en los campos de la inteligencia artificial y la informática, permitiéndoles utilizar tecnologías de procesamiento de lenguaje natural de manera

sencilla y efectiva. Se busca agilizar el análisis de grandes volúmenes de datos, reduciendo significativamente los periodos de tiempo necesarios. Además de proporcionar flexibilidad para utilizar distintos modelos de análisis de lenguaje según los requerimientos particulares.

Objetivos específicos

- Realizar el análisis del dominio necesario para comprender las necesidades y requerimientos específicos de los usuarios en el ámbito judicial.
- Generar tanto el producto final como la documentación e investigación de tecnologías utilizadas a raíz del desarrollo.
- Tener en cuenta la posibilidad de futura expansión del sistema a través de la incorporación de nuevos modelos o de la integración de tecnologías adicionales.
- Poner énfasis en la visualización de los datos resultantes del análisis para facilitar su comprensión y análisis.
- Disponibilizar el uso de la herramienta mediante una interfaz web para facilitar el acceso y priorizar la usabilidad.
- Permitir la carga de lotes de texto a carpetas que puedan ser procesadas para obtener un informe con los resultados deseados.
- Desarrollar al menos dos modelos de procesamiento de lenguaje: análisis de violencia y reconocimiento de entidades.
- Realizar pruebas para validar la efectividad y precisión de los modelos creados.

Alcance

El sistema permitirá cargar lotes de documentos a través de una interfaz web, a distintas carpetas, y aplicar los modelos de análisis sobre los archivos de una carpeta. Este análisis mostrará los resultados completos del mismo y algunos informes estadísticos sobre estos.

En un principio se implementará: un modelo de clasificación de violencia capaz de detectar si cada línea de un archivo es “violenta” o no, y qué tipo de violencia está presentando, y un modelo de detección de entidades nombradas en texto. Los modelos serán entrenados si es posible con datos de la región Argentina o latinoamericana.

La integración de tecnologías complementarias, como Tesseract [11] o Whisper [10] que permitirían manejar transcripciones desde imágenes o

audio a texto, se dejó fuera de los alcances del proyecto con el objetivo de acotar la complejidad y enfocar el esfuerzo.

Se entregarán al demandante (Laboratorio de Investigación y Desarrollo de Tecnología en Informática Forense) :

- Documento de requerimientos del sistema.
- Código del sistema realizado, en un repositorio, con su documentación de desarrollo correspondiente.
- Diagramas descriptivos del sistema.
- Manual de usuario en formato de video para su fácil visualización.
- Conjuntos de datos de entrenamiento y modelos creados para el sistema.
- Informe sobre el desarrollo de los modelos, comparaciones entre los mismos y resultados de los entrenamientos.

Marco de referencia

Esta sección presenta un marco de referencia breve que proporciona una base sobre los conceptos clave utilizados a lo largo de este proyecto.

Sistemas web

Un sistema web es un sistema que sigue la arquitectura cliente-servidor. En esta arquitectura el servidor es aquel que provee recursos o servicios y el cliente es aquel que solicita estos servicios. El cliente y el servidor interactúan y se comunican mediante el envío de solicitudes y respuestas utilizando el protocolo de comunicación HTTP. Este es un protocolo estándar de intercambio de datos, y es sin estado, es decir, que cada solicitud es procesada de manera independiente.

El servidor es el programa que se encarga de procesar las solicitudes que provienen del cliente y dar respuestas a estas solicitudes. En el servidor existe una capa de aplicación que se encarga de la lógica, el procesamiento de datos y manejo de base de datos. Esta parte del sistema es conocida como el backend, es la responsable de realizar operaciones, procesar los datos, conectarse con la base de datos y ejecutar algoritmos. Por otro lado, se denomina “frontend” a la parte del sistema web que interactúa directamente con el usuario, presentando la información de manera que el usuario pueda interactuar con el sistema. Esto incluye la presentación de los datos, la recopilación del input del usuario y la presentación de la interfaz. Las tecnologías utilizadas en el frontend incluyen HTML, CSS y JavaScript. [15]

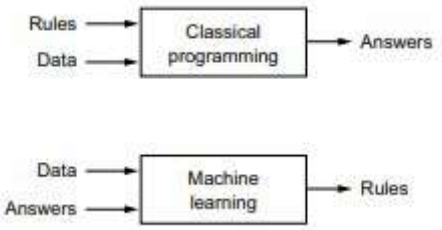
La arquitectura web de cliente-servidor, refiere a la estructura y el diseño del sistema. No significa necesariamente que el sistema esté accesible en internet. El sistema puede estar alojado en una red privada y ser accesible solo a usuarios dentro de esa red.

Inteligencia Artificial

La Inteligencia artificial puede describirse como el esfuerzo por automatizar tareas intelectuales normalmente realizadas por humanos. Incluye el *machine learning* y el *deep learning*, pero también puede incluir algoritmos simbólicos no enfocados en el aprendizaje.

En la programación tradicional se definen un conjunto de reglas o instrucciones que utilizan datos para obtener los resultados. En *machine*

learning en lugar de definir reglas, se empieza con un conjunto de datos y con los resultados esperados, y se encuentran las reglas. El objetivo es posteriormente poder aplicar esas reglas generadas a cualquier otro input para obtener un resultado.



Fuente: "Deep Learning with Python" (François Chollet) Capítulo 1, figura 1.2

Para lograr esto es necesario contar con datos de *input*, resultados esperados y una forma de evaluar si el algoritmo está haciendo un buen trabajo, midiendo la diferencia entre los *outputs* que está obteniendo y los que se debería obtener. El *machine learning* funciona convirtiendo los datos de entrada a una representación matemática significativa y adecuada para el algoritmo elegido y realizando transformaciones en los datos para llegar al *output* esperado.[1] La representación adecuada depende del algoritmo de entrenamiento, la arquitectura y las funciones de activación utilizadas (estas funciones son aquellas que deciden si una neurona en una red neuronal debe ser activada o no). Por ejemplo, en redes neuronales los datos son escalados entre 0 y 1, o -1 y 1, una función de activación ReLu no considera aquellos valores debajo de 0, mientras que una Sigmoidea prefiere que se dé un promedio centrado en 0.[16]

El deep learning es una rama dentro del machine learning que aprende usando varias capas que modifican la representación de los datos de forma cada vez más significativa. El entrenamiento y ajuste de los parámetros se guía usando una función denominada *loss function* para calcular la distancia entre las predicciones y el objetivo, aplicando el algoritmo de *backpropagation*. Este algoritmo fue descubierto simultáneamente por Werbos y Rummelhart-Hinton-Williams. El algoritmo funciona haciendo que el error calculado se propague desde la capa de salida a las capas de entrada, durante este proceso se ajustan los pesos en la red en función a cuánto contribuyen al error.[12][13]

Procesamiento de Lenguaje Natural

El procesamiento de lenguaje natural ("NLP" por sus siglas en inglés de "Natural Language Processing") es la rama de la inteligencia artificial que

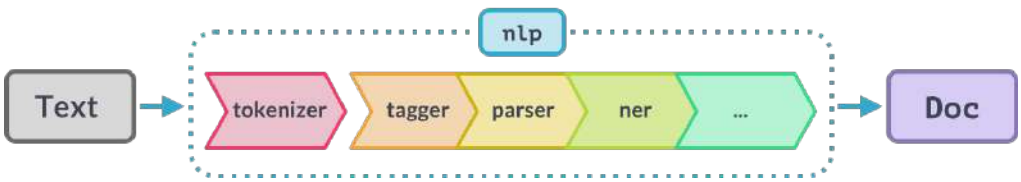
se encarga del procesamiento del lenguaje humano. Para procesar lenguaje se pueden utilizar diversas técnicas como la programación clásica, enfoques de inteligencia artificial clásicos basados en reglas, técnicas simbólicas, o el *machine learning*.

Se puede definir el lenguaje natural como aquel que las personas usan para comunicarse, y se distinguen de los lenguajes formales que utilizan un alfabeto fijo para formar cadenas utilizando un conjunto específico de reglas denominadas gramática formal. [14]

Los lenguajes formales son definidos por humanos con el propósito de describir o manipular objetos, números, conjuntos, etc. Debido a esto son precisos y sistemáticos, siguen una estructura lógica y coherente que permite su análisis a partir de reglas que determinan que cadenas son válidas y que cadenas no lo son.

Los lenguajes naturales, por otro lado, son el producto espontáneo de comunidades de personas que buscan comunicarse, están sujetos a evolución constante a raíz de cambios culturales, sociológicos, geográficos o políticos de las comunidades que los crearon. Si bien el lenguaje natural sigue un conjunto de reglas gramaticales y sintácticas, es mucho más ambiguo. La interpretación de los lenguajes naturales no sigue una línea de pasos definidos ni reglas establecidas, sino que es afectada por factores culturales, históricos, el uso de sarcasmo, metáforas, humor, etc. Debido a esta ambigüedad automatizar la "interpretación" o extracción de información de texto es una tarea compleja utilizando programación tradicional y es por eso que se opta por utilizar técnicas de inteligencia artificial.

Una librería utilizada para realizar procesamiento de lenguaje natural es SpaCy, es gratuita, de código abierto, y permite construir aplicaciones para procesar grandes volúmenes de texto. Una de sus principales ventajas es la gran variedad de modelos preentrenados que ofrece en una gran variedad de idiomas. La librería utiliza un *pipeline* de procesamiento para analizar texto. El *pipeline* tiene componentes entrenados para cada tarea de procesamiento.

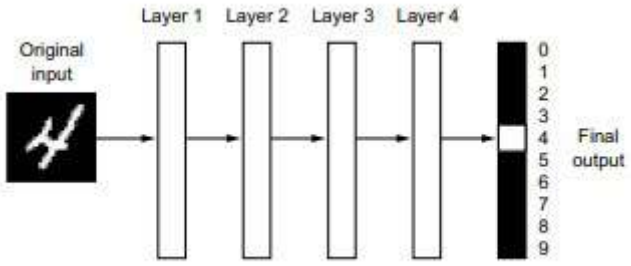


Fuente: "spaCy 101: Everything you need to know" [2]

SpaCy se destaca porque todas las etapas del *pipeline* de procesamiento son modelos de *machine learning* entrenados, incluyendo al *tokenizer*, el elemento encargado de dividir los textos en *tokens*. Los *tokens* son unidades atómicas que representan conceptos, en general palabras, pero también pueden ser signos de puntuación, espacios, emojis, siglas, entre otros. Los diferentes elementos del *pipeline* pueden ser reentrenados para lograr comportamientos específicos. SpaCy utiliza redes neuronales convolucionales para el entrenamiento de sus modelos, y permite también el entrenamiento utilizando *transformers*. [2]

Redes Neuronales

Las redes neuronales están formadas por capas de nodos, que imitan las neuronas biológicas de modo matemático, con una capa de entrada, varias capas ocultas y una capa de salida. Los nodos se conectan entre sí, y tienen un peso asociado y un umbral, si la salida del nodo supera el valor umbral este se activa y envía datos a la capa siguiente. [3] Una red neuronal se puede comparar a un proceso que destila información, pasándola por varios filtros refinando cada vez más, generando *features* o características descriptivas de los datos en cada capa o instancia de procesamiento.



Fuente: “Deep Learning with Python” (François Chollet) Capítulo 1, figura 1.5

Las redes neuronales convolucionales (CNN, de *Convolutional Neural Networks*, a veces también llamadas *convnets*) tienen capas convolucionales, capas de agrupación o *pooling*, y al menos una capa final totalmente conectada. Las capas convolucionales tienen un *input*, con un cierto número de dimensiones, y un *kernel* que se va aplicando a cada sección del *input*. A diferencia de las capas densamente conectadas que aprenden patrones globales, las capas convolucionales aprenden patrones locales. Esto implica que los patrones que aprendan son invariantes a la traslación, por lo que necesitan menos muestras de entrenamiento para aprender representaciones que tienen poder de generalización.

Además, pueden aprender usando múltiples capas, cada una enfocada en un patrón específico y tomando información de la capa anterior. Cada neurona aprende patrones basándose en el *kernel* que se ajusta durante el entrenamiento. A medida que avanzamos a través de las capas sucesivas, estas aprenden diferentes niveles de abstracción. Por ejemplo para el procesamiento de imágenes, las primeras capas pueden detectar líneas y contornos, las siguientes pueden reconocer formas más complejas como rasgos faciales, y las capas finales pueden llegar a identificar entidades completas.

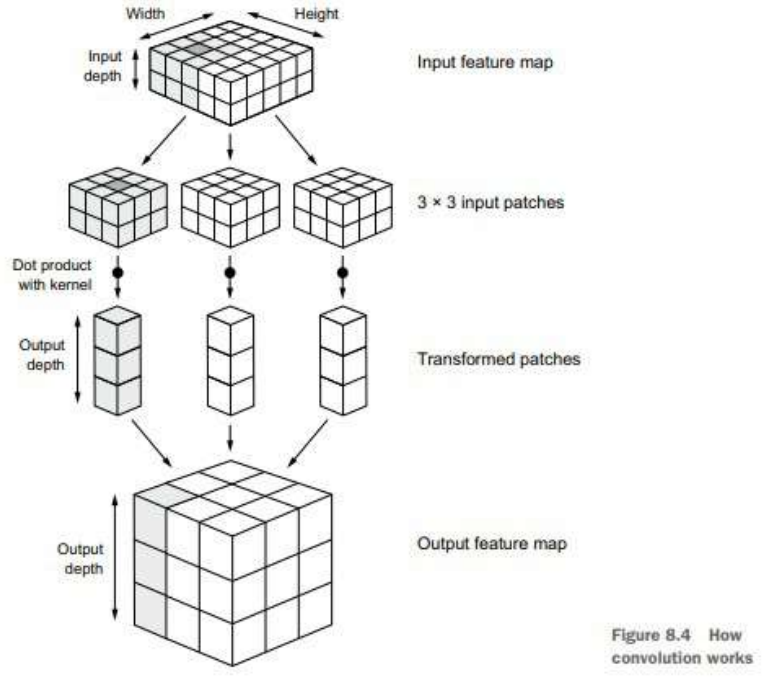
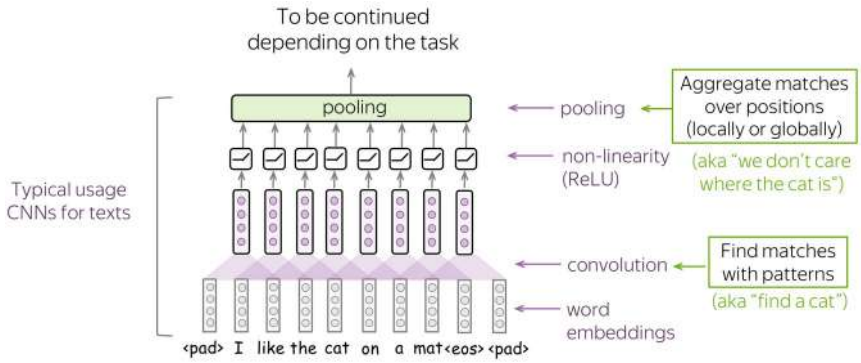


Figure 8.4 How convolution works

Fuente: "Deep Learning with Python" (François Chollet) Capítulo 8, figura 8.4
Convolución de dos/tres dimensiones para procesamiento de imágenes.

Originalmente, las CNN fueron desarrolladas para procesamiento de imágenes, si aplicamos el mismo concepto para el procesamiento de texto las convoluciones son de solo una dimensión.

Para buscar patrones en textos se emplea el uso de convoluciones y el mecanismo de *pooling*. Por un lado, la convolución aplica filtros a una ventana de texto y encuentra coincidencias con un patrón y luego el *pooling* resume los valores obtenidos en una región, conservando solo los patrones relevantes.



Fuente: "Convolutional Neural Networks for Text" [6]

La capa de *pooling* es utilizada para reducir el número de dimensiones del *input*, y, por lo tanto, el número de parámetros usados por la red, esto lo hace resumiendo los valores de cada dimensión, tomando el máximo (*max pooling*) o la media de cada *feature* (*mean pooling*). El *pooling* puede ser aplicado de forma local en cada *feature map* (la salida de una capa, que representa características específicas) de forma independiente, o puede ser aplicado de manera global, para así obtener un único vector que represente todo el texto. El *pooling* global es útil para la clasificación de texto, ya que logra producir un vector de tamaño fijo sin importar el tamaño del input. Usando *max pooling* y filtros que detectan n-gramas (subcadenas de texto) una red neuronal convolucional puede encontrar patrones en el texto que le permiten clasificarlo.

Transformers

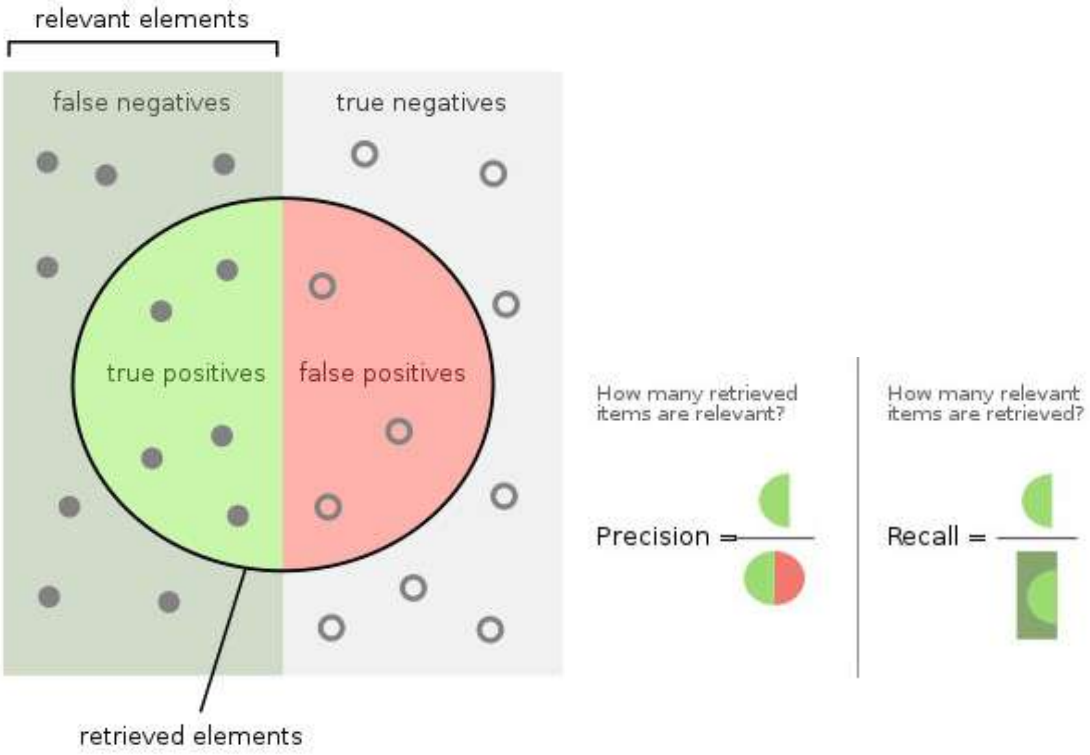
Los *transformers* son una arquitectura de red neuronal basada en mecanismos de atención que permiten que el modelo se centre en los datos de entrada más importantes para hacer predicciones, prescindiendo de la recurrencia y las convoluciones. Crean modelos de alta calidad, que son más fáciles de entrenar, a expensas de un mayor costo computacional en su entrenamiento e inferencia.

Implementación de modelos

El proceso de creación de un modelo consiste en un ciclo de entrenamiento iterativo de entrenamiento, testeo, validación, ajustes y reentrenamiento. Este ciclo continúa hasta lograr un nivel aceptable de precisión en los resultados obtenidos. El testeo y la validación son etapas donde el modelo es presentado con ejemplos que no utilizó para el entrenamiento y se comparan las predicciones con lo esperado.

En la etapa de *testing* esto se realiza con un conjunto reducido de datos provenientes del mismo *dataset* de donde se obtuvieron los datos de entrenamiento. Allí se pueden medir indicadores del modelo, como la precisión o el *recall*. La precisión mide cuántas predicciones positivas correctas hizo el modelo de todas las predicciones positivas que hizo, mientras que el *recall* mide de todas las instancias positivas que existen, cuantas pudo predecir correctamente. El *f-score* es una medida que combina la precisión y el *recall* para proporcionar una única métrica de rendimiento.

	Predicción positiva	**Predicción Negativa**
Positivo	Verdadero positivo (hit)	Falso negativo (miss)
Negativo	Falso positivo (false alarm)	Verdadero negativo (correct rejection)



[5] Representación gráfica de indicadores de precisión y recall.

Entonces, estos indicadores pueden calcularse de la siguiente manera:

$$Precisión = \frac{Verdaderos\ positivos}{Verdaderos\ positivos + Falsos\ positivos}$$

$$Recall = \frac{Verdaderos\ positivos}{Verdaderos\ positivos + Falsos\ negativos}$$

$$F1 = \frac{2}{recall\,\square^{-1} + precisión\,\square^{-1}} = 2\frac{precisión\,.\,recall}{precision + recall}$$

Por otro lado, en la etapa de validación se obtienen datos nuevos, que intentan cubrir situaciones típicas y también situaciones ambiguas o posiblemente problemáticas para el modelo. Esta etapa intenta identificar problemas como el *overfitting*, el *underfitting*, o la presencia de sesgos.

El *underfitting* ocurre cuando el modelo no se ajusta lo suficiente a los datos de entrenamiento y no logra capturar las relaciones y patrones presentes. Esta situación produce indicadores de precisión bajos tanto con los datos de entrenamiento como con los datos nuevos. Algunas posibles causas de este tipo de modelos son: datos “ruidosos” o con poca variabilidad, bajo número de datos, modelo demasiado simple, entre otros.

El *overfitting* ocurre cuando el modelo se ajusta demasiado a los datos de entrenamiento pero no da buenos resultados con datos nuevos. Ésta situación produce indicadores de precisión altos pero tiene resultados reales más bajos. Identificar el *overfitting*, y buscar sus causas (falta de datos, datos con similitudes, datos desbalanceados, etc.) es importante para poder realizar los cambios necesarios en el entrenamiento siguiente y lograr obtener un mejor modelo.

Gestión inicial

En esta sección se presentan la planificación y estimaciones iniciales que se realizaron durante la primera etapa del proyecto.

Planificación

Fecha de inicio prevista: ***24/10/2022***

Fecha de finalización prevista: ***31/07/2023***

Al inicio de la planificación del proyecto se plantearon cuatro etapas para la división de las tareas de desarrollo:

1. Análisis y diseño del sistema.
2. Creación del modelo detector de entidades.
3. Creación del modelo de clasificación de violencia.
4. Implementación del servidor web.

Tareas previstas por Etapa

Etapa 1:

- ***Análisis e Investigación de herramientas y tecnologías a utilizar:***

Esta tarea consistirá en un análisis y familiarización de las tecnologías, lenguajes de programación y herramientas que serán utilizadas a lo largo del proyecto. Principalmente, involucrará al lenguaje de programación Python, la librería SpaCy, librerías de visualización de datos, la herramienta de etiquetado Label Studio, el framework Django, entre otros.

- ***Relevamiento de requerimientos:***

Esta tarea consistirá en realizar un relevamiento de las necesidades de los usuarios, a partir de una o varias reuniones con la referente funcional, para la posterior creación del documento de especificación de requisitos de software (SRS).

- ***Diseño de arquitectura****:*

Esta tarea consistirá en el diseño de las interfaces web, lla estructura del servidor web y dónde estará alojado. Además, incluirá un diseño de las

API de los modelos de análisis y de las estructuras de datos que estas devolverán.

Etapa 2:

- ***Recopilación y etiquetado de datos para el modelo de detección de entidades:***

Esta tarea consistirá en la creación de un dataset etiquetado para poder llevar a cabo el entrenamiento y la prueba de un modelo de NLP capaz de detectar entidades en un texto.

- ***Implementación de modelo de detección de entidades****:*

Se utilizará la librería SpaCy de Python como herramienta para crear el modelo que será entrenado con un subconjunto del dataset previo. Este modelo recibirá un conjunto de textos y devolverá las entidades encontradas en los mismos

.

- ***Testeo y Entrenamiento de modelo de entidades****:*

Se entrenará el modelo y se compararán los resultados obtenidos con los esperados. Si es necesario se reentrenará el modelo hasta llegar a un nivel de precisión aceptable.

- ***Validación del modelo***:

Se validará el modelo avanzado con la referente funcional, y se harán las correcciones finales de acuerdo a las observaciones recibidas.

Etapa 3:

- ***Recopilación y etiquetado de datos para el modelo de detección de violencia de género****:*

Esta tarea implicará la creación de un dataset con datos etiquetados para poder llevar a cabo el entrenamiento y la prueba de un modelo de NLP con el objetivo de detectar este tipo de violencia. Se buscará clasificar partes de un texto como violentos, indicando el tipo de violencia, o no violentos.

- ***Implementación de modelo de detección de violencia de género:***

Implementación, usando SpaCy, de un modelo capaz de detectar comportamientos violentos en un texto. Encontrar y clasificar partes de un texto como violentos o no violentos. Se buscará que luego del

entrenamiento muestre como resultado un subconjunto de interés que se marcó con mayor posibilidad de contener evidencia relevante de una situación de violencia de género.

- ***Testeo y entrenamiento de modelo de detección de violencia de género:***

Se entrenará el modelo y se compararán los resultados obtenidos con los esperados. Si es necesario se lo reentrenará hasta llegar a un nivel de precisión aceptable.

- ***Validación del modelo***:

Se validará el modelo avanzado con la referente funcional, y se harán las correcciones finales de acuerdo a las observaciones recibidas.

Etapa 4:

- ***Implementación de servidor backend:***

Esta tarea implicará la creación del servidor en Django e implementación de toda la estructura de controladores, servicios y modelos que interactuarán con el módulo de SpaCy para ofrecer una serie de APIs que expongan los datos que después serán formateados por la interfaz web.

- ***Implementación de interfaz web; Carga de datos y uso de modelos:***

Esta tarea implicará crear el maquetado de las interfaces y estilos de las páginas web que servirán como interfaz para los usuarios. Además, se diseñarán las funcionalidades para cargar y llamar al servidor de backend para el procesado de datos.

- ***Implementación de interfaz web; Visualización de resultados:***

Esta tarea implicará crear el maquetado de las interfaces, estilos de las páginas web que visualizarán los resultados. Procesarán los datos devueltos por el servidor backend y mostrarán los resultados mediante gráficos, tablas y otras herramientas de visualización.

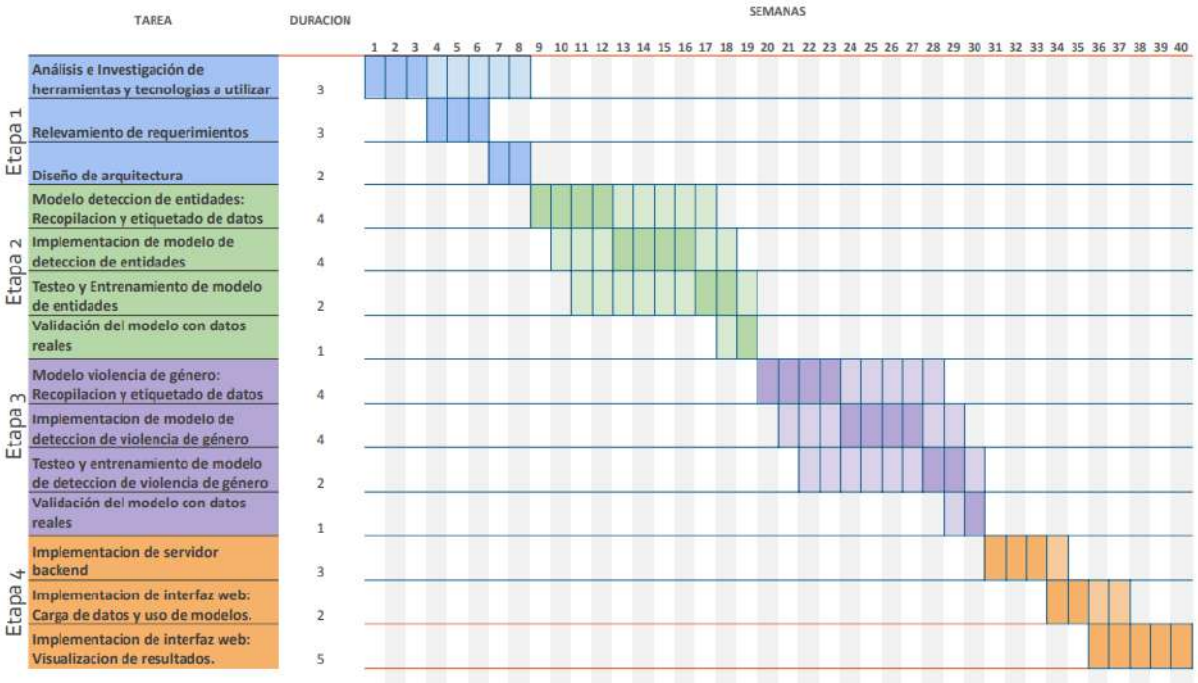
FODA

Fortalezas ● Poca inversión necesaria en recursos, herramientas y materiales ● Desarrollo propio y con herramientas de código abierto. ● Experiencia previa del equipo con Python y SpaCy. ● Contacto con actores del dominio, para la provisión de datos y asistencia técnica y respecto a la sensibilización a problemática .	**Debilidades** ● El equipo no dispone de experiencia en el desarrollo de aplicaciones que integren inteligencia artificial. ● El equipo tiene un número de personas reducido y debe etiquetar una gran cantidad de datos para poder entrenar los modelos.
Oportunidades ● Existencia de necesidad concreta de análisis de texto automático en el proceso legal. ● Expandir la capacidad del producto para analizar otros tipos de texto o incorporar nuevos módulos de análisis. ● Solución con impacto social.	**Amenazas** ● Cambios en las tecnologías que impacten en el proyecto. ● Falta de datos para la creación y entrenamiento de los modelos.

Estimaciones

Se prevé una carga horaria de 10 horas semanales por integrante, distribuidas en 5 días. Se tiene en cuenta que cada semana estas horas serán distribuidas a criterio de cada integrante. Llevando la estimación a unas 400 horas por integrante.

- Etapa 1: **160 horas**
- Etapa 2: **220 horas**
- Etapa 3: **220 horas**
- Etapa 4: **200 horas**



Debido a la retroalimentación entre tareas dada en cada etapa, se indica en el Gantt qué tarea tiene el foco en cada periodo de tiempo, destacando la misma con un color más intenso.

Análisis de Riesgo y planes de contingencia

Riesgo	Consecuencias	Probabilidad	Impacto	Peso
Un integrante abandona el proyecto.	Pérdida de recurso humano y necesidad de reorganización o redefinición de proyecto.	1	3	3
Un integrante se ausenta por un largo período.	Ausencia de recurso humano y demora en desarrollo del proyecto.	1	3	3
El Demandante pierde interés en el proyecto.	Cancelación del proyecto.	1	3	3
Dificultad de contacto con referente funcional.	Demoras en el proyecto, dificultad para la entrega y validación del producto en sus distintas etapas.	2	3	6
El Hardware limitado no permite el entrenamiento de ciertos modelos.	Limitación en el tipo de modelos a entrenar a modelos más simples.	3	3	9
El software o las librerías no se pueden utilizar o presentan fallos en otros equipos.	Demoras en el desarrollo conjunto del proyecto.	3	2	6
Falta de datos para el entrenamiento de los	Modelos entrenados con	2	3	6

modelos.	pocos datos, resultando en modelos de baja precisión.			
Cambian las tecnologías utilizadas para el desarrollo del proyecto.	Demoras en el proyecto por adaptación a cambios o elección de nueva tecnología.	2	2	4

Riesgo	Plan
Dificultad de contacto con referente funcional.	- Planear reuniones - Mantener canales
El Hardware limitado no permite el entrenamiento de ciertos modelos.	- División de tareas cuenta las capaci - Uso de hardware demandante.
Falta de datos para el entrenamiento de los modelos.	- Búsqueda de dato datasets preexiste selección y limpie
Cambian las tecnologías utilizadas para el desarrollo del proyecto.	- Uso de tecnología inicio, si es posibl - Elección de tecno
El software o las librerías no se pueden utilizar o presentan fallos en otros equipos.	- Utilización de *ven* proyecto y de nue - Documentar los p pasos a seguir y e

Metodología de trabajo

Se utilizó una metodología de trabajo que combina conceptos de las metodologías ágiles, Kanban y SCRUM. Se optaron por las metodologías ágiles debido a sus ventajas sobre los enfoques tradicionales, como la facilidad de adaptación a los cambios de requerimientos, y la periódica validación con el cliente.

Kanban es una metodología ágil que se originó en las líneas de producción de Toyota. Este sistema consiste en la gestión de tareas priorizando la optimización del flujo de trabajo y la flexibilidad. Se utiliza un tablero donde las tareas son divididas en columnas según su estado: “por hacer”, “en proceso”, “bloqueadas” y “terminadas”. Este enfoque visual permite identificar cuellos de botella y posibles mejoras en el proceso. Tradicionalmente el método también plantea una división de tareas más pequeñas o cortas que algunas de las tareas planteadas en este proyecto en el que existieron tareas que debido a su naturaleza y complejidad eran mejor resueltas como una tarea unificada más larga.

SCRUM es una metodología ágil de gestión de proyectos que se centra en la entrega iterativa de productos al cliente. Tradicionalmente estos productos se entregan en ciclos cortos llamados ‘sprints’, que duran un par de semanas y son regulares. SCRUM permite la retroalimentación regular con el cliente y la adaptabilidad del producto. En todo el desarrollo de este proyecto se tomó el concepto de la entrega regular de productos al cliente (mockup, modelos entrenados y sistema web) y la retroalimentación. Específicamente, a la hora de implementar los modelos predictivos, se incorporaron conceptos adaptados de SCRUM (planificación, implementación, revisión y retrospectiva) luego de cada ciclo de etiquetado, entrenamiento y testeo.

Se utilizó la herramienta Trello para la creación y manejo del tablero y el conteo de las horas dedicadas a cada tarea.

Análisis

El análisis del dominio y la creación de los requerimientos del sistema fueron realizados a partir del contacto con la referente funcional del proyecto. Además, se consultó material bibliográfico relevante para el análisis del dominio, principalmente leyes como la N°26.485 y artículos periodísticos anotados en la bibliografía de este proyecto.

Entregables esperados del análisis

- **Requerimientos funcionales**: Describen las características del sistema, definen qué funciones debe poder brindar el sistema.
- **Requerimientos no funcionales:** Describen qué requisitos debe cumplir el sistema en cuanto a rendimiento, escalabilidad, seguridad, mantenibilidad, entre otros.
- **Casos de uso**: Representan los procesos que realizan los actores o usuarios del sistema, como interactúan con el mismo para brindar los requerimientos funcionales.
- **Modelo del dominio**: Principales clases que forman parte del dominio del problema, representan entidades o conceptos clave.

Diseño

El diseño del sistema fue realizado a partir de los requerimientos funcionales y no funcionales definidos en la etapa de análisis, diagramando los diseños de componentes y entidades. Posteriormente se validó el diseño con la referente funcional y los directores del proyecto. Se dejó abierta la posibilidad de realizar cambios o mejoras si a la hora de la implementación o ante algún cambio de requerimiento fuera necesario.

Entregables del diseño

- **Diagrama de componentes**: Representa la división del sistema en componentes o partes modulares.
- **Diagrama entidad-relación**: Muestra cómo las entidades se relacionan entre sí, modela la base de datos utilizada por el sistema web para persistir los datos.

Implementación y testeo de los modelos

Para la creación de los modelos de análisis se siguieron una serie de pasos de modo cíclico buscando la continua mejora del modelo en cada iteración:

1. Recopilación de datos para el entrenamiento.
2. Limpieza de los datos recopilados. Esto incluye remoción de signos de puntuación extraños, espacios excesivos, etc.
3. Etiquetado del dataset generado.
4. Combinación de los nuevos datos con los datos utilizados en el entrenamiento anterior para crear un dataset nuevo.

5. División del dataset en un conjunto de entrenamiento y un conjunto de prueba.
6. Transformación de los dos conjuntos del dataset al formato adecuado para el entrenamiento por SpaCy.
7. Configuración de archivos para el entrenamiento y entrenamiento.
8. Documentación, composición del dataset y métricas obtenidas del entrenamiento.
9. Pruebas del modelo resultante con nuevo banco de datos reducido diseñado para encontrar posibles fallas en el modelo. El banco de prueba también es refinado de manera cíclica para revisar los errores más frecuentes.
10. Identificación y documentación de los resultados, de los errores más frecuentes del modelo, y de observaciones adicionales.
11. Definición los cambios a tomar para el próximo entrenamiento y repetición de los pasos desde el principio.

Implementación web

Para la implementación del sitio web se siguieron los siguientes pasos:

1. Diseño de prototipo de la Interfaz de Usuario (UI).
2. Diseño de la Base de Datos.
3. Implementación del backend y frontend, creando *‘models’*, *‘views’*, y *‘templates’* de cada una de las vistas del sitio, de acuerdo al formato MVT (Model-View-Template).
4. Implementación del Módulo de Procesamiento NLP.
5. Integración del Módulo de NLP.

Análisis del problema

En esta sección se presenta un resumen de los principales requerimientos y los casos de uso, del dominio, los requerimientos y del modelo de dominio.

Dominio

Análisis de evidencia digital

La informática forense se encarga de obtener, preservar, extraer y analizar pruebas digitales. En la actualidad las pruebas digitales aumentaron en tamaño y cantidad, debido al crecimiento del uso de la tecnología y la aparición de nuevos dispositivos. Las fuentes comunes de evidencia digital son los mensajes y otras comunicaciones que se realizan a través de diversos medios, como el correo electrónico, aplicaciones de mensajes y las redes sociales. El análisis de los grandes volúmenes de evidencia es una tarea tediosa que muchas veces requiere un largo tiempo y dedicación. La necesidad de analizar rápidamente grandes cantidades de datos para extraer información relevante es una prioridad clave para muchas investigaciones judiciales.

Actualmente, el análisis consiste en la lectura manual de las conversaciones implicadas. Cuando una persona identifica una situación de agresión en un mensaje, lo etiqueta manualmente. Las herramientas existentes utilizadas,como UFED[22], permiten solamente marcar un mensaje como ‘de interés’ y realizar búsquedas sencillas de palabras individuales en el texto. Se busca, al leer las conversaciones y marcar los mensajes violentos, identificar un patrón de agresión. Al identificar posibles patrones agresivos, se analiza su evolución en el tiempo, y se leen cada uno de los mensajes. Posteriormente a la lectura y el análisis se realiza un informe con todo lo recolectado e identificado.

Violencia de género

La violencia de género es una práctica estructural que viola los derechos humanos y las libertades fundamentales. Afecta principalmente a mujeres y personas en el colectivo LGBTI+ cuando sufren discriminación, agresión, hostigamiento o degradación por su identidad de género, expresión de género u orientación sexual. [7]

La ley Nº 26.485 en su artículo 5 considera distintos tipos de violencia dentro de la violencia de género: física, sexual, psicológica, económica y simbólica.

"1.- Física: La que se emplea contra el cuerpo de la mujer produciendo dolor, daño o riesgo de producirlo y cualquier otra forma de maltrato agresión que afecte su integridad física.

2.- Psicológica: La que causa daño emocional y disminución de la autoestima o perjudica y perturba el pleno desarrollo personal o que busca degradar o controlar sus acciones, comportamientos, creencias y decisiones, mediante amenaza, acoso, hostigamiento, restricción, humillación, deshonra, descrédito, manipulación aislamiento. Incluye también la culpabilización, vigilancia constante, exigencia de obediencia sumisión, coerción verbal, persecución, insulto, indiferencia, abandono, celos excesivos, chantaje, ridiculización, explotación y limitación del derecho de circulación o cualquier otro medio que cause perjuicio a su salud psicológica y a la autodeterminación.

3.- Sexual: Cualquier acción que implique la vulneración en todas sus formas, con o sin acceso genital, del derecho de la mujer de decidir voluntariamente acerca de su vida sexual o reproductiva a través de amenazas, coerción, uso de la fuerza o intimidación, incluyendo la violación dentro del matrimonio o de otras relaciones vinculares o de parentesco, exista o no convivencia, así como la prostitución forzada, explotación, esclavitud, acoso, abuso sexual y trata de mujeres.

4.- Económica y patrimonial: La que se dirige a ocasionar un menoscabo en los recursos económicos o patrimoniales de la mujer, a través de:

a) La perturbación de la posesión, tenencia o propiedad de sus bienes;

b) La pérdida, sustracción, destrucción, retención o distracción indebida de objetos, instrumentos de trabajo, documentos personales, bienes, valores y derechos patrimoniales;

c) La limitación de los recursos económicos destinados a satisfacer sus necesidades o privación de los medios indispensables para vivir una vida digna;

d) La limitación o control de sus ingresos, así como la percepción de un salario menor por igual tarea, dentro de un mismo lugar de trabajo.

5.- Simbólica: La que a través de patrones estereotipados, mensajes, valores, íconos o signos transmita y reproduzca dominación, desigualdad y discriminación en las relaciones sociales, naturalizando la subordinación de la mujer en la sociedad." [8]

Estos tipos de violencia pueden estar presentes o ser dejados en evidencia en un mensaje de texto. Un mensaje en sí mismo puede ser directamente un acto de violencia. Por ejemplo, puede ser una violencia psicológica, utilizando insultos o denigraciones que buscan dañar la autoestima y el bienestar emocional de la persona. También puede ser una violencia simbólica, transmitiendo estereotipos o patrones del desbalance de poder entre los géneros, reforzando así las estructuras de dominación existentes. Incluso puede llegar a ser una violencia sexual, enviando mensajes cohesivos o indeseados de naturaleza sexual, invadiendo la intimidad y el consentimiento de la persona. Además, un mensaje puede dejar en evidencia la amenaza o la posible presencia de cualquier tipo de violencia, como amenazas físicas que ponen en peligro la seguridad personal, o económicas que buscan controlar y limitar la autonomía financiera de la persona.

Análisis de requerimientos

Los siguientes requerimientos fueron ideados junto con la referente funcional del proyecto.

Requerimientos funcionales

1. **El sistema permitirá al usuario interactuar a través de una interfaz web.**

Nro.	**Descripción**
1.1	El sistema permitirá la carga de lotes de textos para su análisis, en los formatos .docx .txt, .xls y .zip. Estos archivos se originan de la exportación directa.
1.2	El sistema contará con aplicaciones para la clasificación según violencia de mensajes exportados directamente de WhatsApp y para la detección de entidades en cualquier tipo de texto.
1.3	El sistema mostrará al usuario las distintas aplicaciones de

	análisis que puede elegir aplicar, junto con una descripción breve de cada una de ellas, detallando su funcionamiento.
1.4	El sistema ejecutará el análisis solicitado por el usuario y mostrará los resultados de dicho análisis.
1.5	El sistema permitirá al usuario descargar un documento detallando los resultados del análisis. (resultados completos, estadísticas, entre otros.)

2. **El sistema contará con un sistema de registro de usuarios:**

Nro.	**Descripción**
2.1	El sistema permitirá solo a los usuarios de tipo administrador crear nuevos usuarios.
2.2	El sistema permitirá a los usuarios generar carpetas para guardar los archivos que desea procesar en cada análisis.
2.3	El sistema permitirá a los usuarios eliminar carpetas generadas.
2.4	El sistema podría incluir búsquedas o filtros de los resultados de distintos análisis.
2.5	El sistema permitirá a los usuarios modificar sus contraseñas.

3. **El sistema contará con una aplicación de detección de entidades:**

Nro.	**Descripción**
3.1	El sistema permitirá al usuario cargar un documento de texto o lote de textos, e indicar que quiere conocer las entidades presentes en ellos.
3.2	El sistema será capaz de verificar si un archivo que se subió al sistema ya existe en la misma carpeta.
3.3	El sistema permitirá realizar análisis múltiples veces sobre la misma carpeta o conjunto de archivos.
3.4	Según el modelo que se utilice se define el tipo de entidades a detectar. Por ejemplo, un modelo podrá detectar: organizaciones, personas, lugares, fechas.

3.5	El sistema mostrará cada entidad junto con el texto, o el extracto de texto, donde fue encontrada.
3.6	El sistema permitirá al usuario configurar sus preferencias de esquema de colores para la visualización de entidades.
3.7	El sistema incluirá esquemas de colores para personas con daltonismo.
3.8	El sistema será capaz de mostrar solo las frases que contengan entidades de una o varias categorías especificadas.
3.9	El sistema mostrará estadísticas de los resultados de las entidades encontradas. Algunas de estas podrían ser: ● Número de entidades de cada categoría. ● Porcentaje de entidades de cada categoría. ● Entidades repetidas.

4. **El sistema contará con una aplicación de detección de violencia de género en mensajes de WhatsApp:**

Nro.	**Descripción**
4.1	El sistema permitirá al usuario cargar un lote de textos, e indicar que quiere conocer cuáles de ellos pueden indicar la existencia de violencia de género.
4.2	El sistema proveerá una lista de los textos marcados como ‘relevantes’ o posibles indicadores de violencia.
4.3	El sistema clasificará cada texto en el lote cargado en una categoría según el tipo de violencia que puede detectar en él: ● Violencia sexual. ● Violencia física. ● Violencia económica. ● Violencia psicológica. ● Violencia simbólica. ● No violento.
4.4	El sistema realizará un análisis de la información adicional (fechas, archivos adjuntos, etc.) de los mensajes relevantes si fuera necesario.

4.5	El sistema mostrará como resultado los conjuntos de frases encontradas en cada categoría.
4.6	El sistema mostrará el score de violencia de cada frase clasificada.
4.7	El sistema permitirá descargar los resultados completos del análisis.
4.8	El sistema elaborará estadísticas referentes a los resultados. Presencia de textos violentos, número de mensajes identificados en cada categoría y que porcentaje representan en el total de frases y en el total de frases violentas.
4.9	El sistema elaborará un informe al finalizar el análisis, resumiendo los resultados obtenidos.
4.10	El sistema permitirá generar un *wordcloud* de visualización personalizado al usuario, permitiendo filtrar palabras específicas, preposiciones, y palabras provenientes de frases no violentas.

Requerimientos no funcionales

Tecnologías a aplicar:

- Python 3.10 en adelante.
- SpaCy v3 en adelante
- Label Studio
- Django 3.2 en adelante

Adecuación Funcional

- Los modelos entrenados para el sistema deben ser analizados en cuanto a su rendimiento y la calidad de sus resultados.

Usabilidad:

- El *frontend* del sistema debe correr en cualquier computadora con un navegador web moderno.
- El *backend* del sistema tiene que estar instalado en un servidor Linux.
- El sistema advertirá en su reporte al finalizar el análisis que los

resultados obtenidos por medio del análisis no garantizan un 100% de precisión y dará las recomendaciones apropiadas.

Extensibilidad:

- El sistema debe estar abierto para añadir nuevas tablas de estadística y procesar nuevos tipos de formatos de texto. Se busca que a futuro sea posible incorporar nuevos módulos de NLP sin que esto represente un gran cambio al sistema existente.

Mantenibilidad:

- El sistema deberá ser desarrollado y configurado empleando tecnologías que faciliten la mantenibilidad del sistema, como Django y Python.

Seguridad:

- El sistema deberá ser desarrollado y configurado utilizando tecnologías que faciliten la seguridad del sistema, como Django que provee características de seguridad por defecto como protección de *"cross site scripting"* (XSS), protección de "*Cross site request forgery"* (CSRF), protección contra inyección de SQL, entre otras.

Casos de uso abreviados

CU1: Registro de usuario al sistema.

Un administrador del sistema entra a la página de administración '/admin' e ingresa a la sección de usuarios donde llena el formulario de 'nuevo usuario' con el nombre de usuario, mail y contraseña del nuevo usuario.

CU2: Iniciar sesión

El usuario completa su información de inicio de sesión (nombre de usuario y contraseña.) y presiona el botón de inicio. El sistema lo redirige a la página principal donde se encuentran las aplicaciones disponibles.

CU3: Modificar contraseña

El usuario entra a ajustes, modifica contraseña, completa la información de inicio de sesión (contraseña anterior, confirmación de contraseña anterior y nueva contraseña) y presiona el botón de Cambiar contraseña. El sistema cambia la información de inicio de sesión del usuario y lo redirige a la página de login.

CU4: Crear carpeta y cargar archivos

El usuario entra a sus carpetas e indica que quiere crear una nueva carpeta junto con el nombre de la nueva carpeta. Dentro de esta carpeta presiona el botón de subir archivos y elige aquellos archivos que desea subir.

CU5: Eliminar carpeta

El usuario entra a sus carpetas e indica que quiere eliminar una carpeta junto con todo su contenido y los análisis que fueron realizados sobre esa carpeta. El sistema muestra un mensaje de confirmación, el usuario confirma la acción y el sistema borra la carpeta.

CU6: Procesar texto con aplicación de detección de entidades.

El usuario elige desde la sección aplicaciones la aplicación de detección de entidades, luego elige la carpeta que contiene los archivos que quiere analizar y que modelo se va a usar para el análisis. Luego de procesados los archivos se redirige al usuario a los resultados y el informe del mismo.

CU7: Procesar texto con aplicación de detección de violencia de género

El usuario elige desde la sección aplicaciones la aplicación de clasificación de texto, luego elige la carpeta que contiene los archivos que quiere analizar y el modelo de clasificación de violencia se va a usar para el análisis. Luego de procesados los archivos se redirige al usuario a los resultados y el informe del mismo

CU8: Filtrar resultados

Dentro de un análisis el usuario ingresa sus preferencias para la visualización de resultados relevantes en la sección de los resultados completos y el sistema muestra solo aquellos que coinciden con sus criterios de búsqueda.

CU9: Descargar resultados

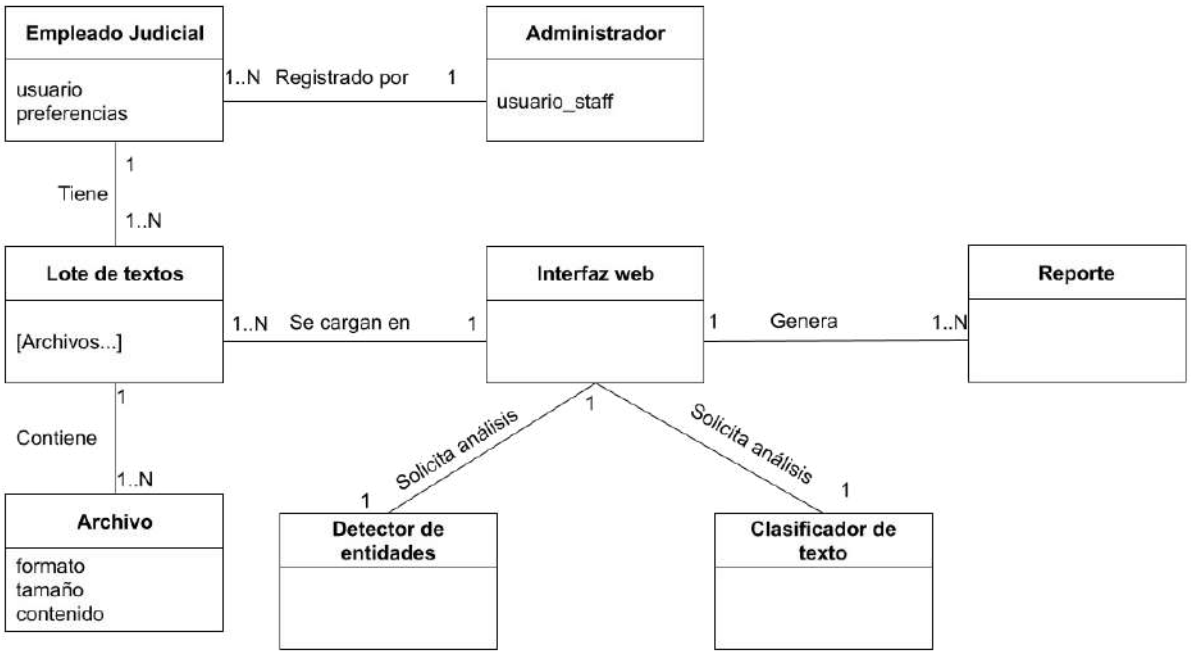
Dentro de un análisis el usuario indica si desea descargar los resultados completos o el informe abreviado de los resultados y el sistema realiza la descarga.

CU10: Borrar análisis

El usuario entra a la sección ‘Mis análisis’ del sistema e indica que quiere eliminar un análisis, o bien entra al análisis particular y presiona el botón

de “borrar”. El sistema muestra un mensaje de confirmación, el usuario confirma la acción y el sistema borra el análisis.

Modelo de Dominio



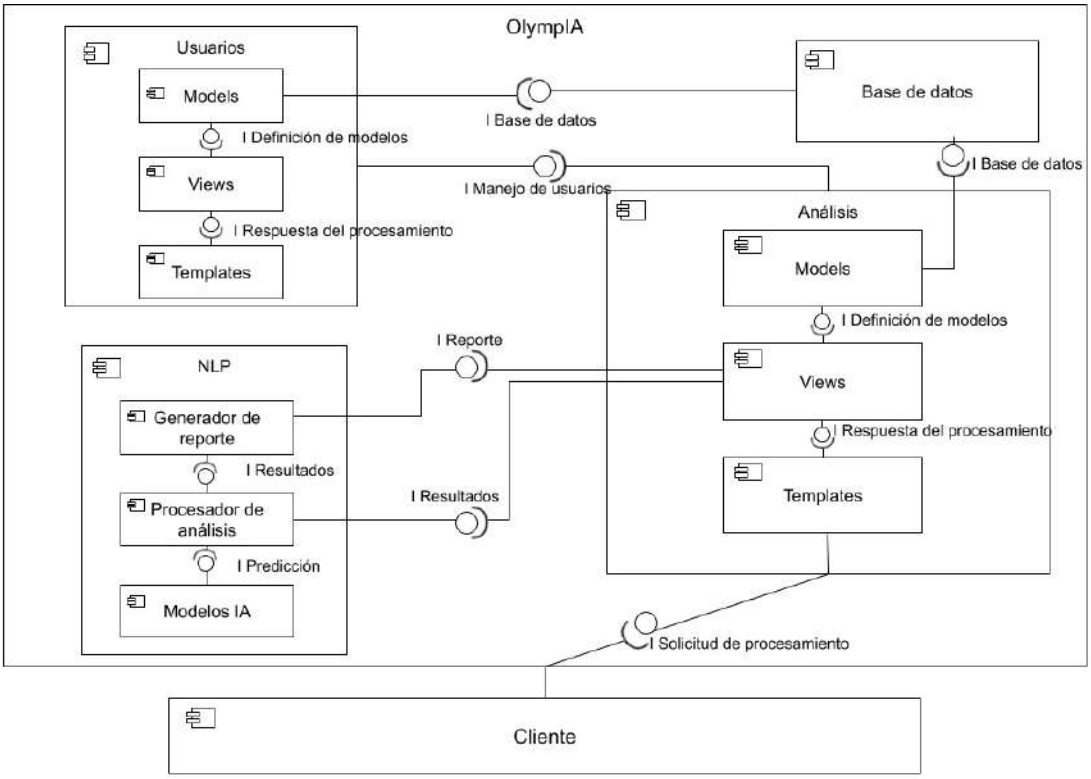
Diseño de la solución

Esta sección detalla las decisiones tomadas acerca del diseño del sistema.

Arquitectura

El sistema sigue una arquitectura cliente-servidor donde el cliente (el navegador web del usuario) interactúa con el servidor web que procesa sus solicitudes y mantiene actualizada una base de datos que utiliza para dar respuesta al cliente.

El servidor web sigue una arquitectura “Modelo-Vista-Template” (MVT)[9]. Los modelos actúan como una interfaz entre la base de datos y el servidor. Las vistas procesan las solicitudes realizadas por los clientes utilizando los modelos. El template se encarga de la presentación del sitio web en el navegador. El servidor contiene un componente para el manejo de usuarios (Componente *“Usuarios”*), que también sigue la lógica MVT. Se encarga de las cuestiones de seguridad y manejo de usuarios. Por último, tiene un componente de procesamiento de lenguaje natural (Componente *“NLP”*), que es el encargado de generar y persistir los resultados y reportes que serán luego presentados al usuario, utilizando los modelos entrenados.



Componentes

Base de Datos

Es el componente que almacena todos los datos del sistema. Es una base de datos relacional definida y creada y gestionada por el ORM de Django a partir de los modelos en las aplicaciones con las que interactúa. Este componente interactúa directamente con los componentes “Análisis” y “Usuarios”.

Análisis

El componente de análisis es el principal componente del sistema. Define la interacción con el cliente y los demás componentes. El cliente interactúa con los ‘Templates’, que corresponden a la interfaz gráfica que componen las páginas web de la aplicación. Desde esta interfaz se solicitan las peticiones, que son enviadas a las ‘Views’, encargadas de procesar y dar respuesta a lo solicitado. Si, por ejemplo, se solicita subir un lote de archivos a una carpeta, el template correspondiente llama a la vista que se encarga de realizar el procesamiento necesario. Esa vista pasa a los ‘Models’ los archivos que se deseen persistir en la Base de Datos.

Ante una solicitud de creación de un análisis (es decir, de procesamiento de archivos) ‘Análisis’ obtiene de la base de datos los datos necesarios y se los envía al módulo de procesamiento de lenguaje natural NLP. El componente Análisis requiere a su vez al componente Usuarios para administrar el control de acceso a los datos de los distintos usuarios.

Usuarios

El componente de manejo de usuarios sigue una estructura interna de MVT siguiendo con los lineamientos definidos por el framework de Django.

- Utiliza la persistencia de la base de datos para guardar tablas de usuarios, contraseñas, permisos y demás datos.
- Se encarga del login, logout y cambio de contraseña de usuarios utilizando sus templates y vistas.
- Interactúa de manera íntegra con el módulo Análisis, encargándose del control de acceso a todas las funcionalidades que este provee.

NLP

El componente de procesamiento de lenguaje natural (NLP) contiene:

- Los modelos entrenados para el procesamiento de texto:
    - Procesamiento de entidades: Consiste en un modelo entrenado para detectar en texto: dinero, fechas, horas, lugares, medidas, organizaciones, personas, tiempo, y entidades misceláneas.
    - Detección de violencia de género: En cuanto a la detección de violencia de género se optó por un diseño que implementa el uso de dos modelos. Un primer modelo binario que distingue entre frases violentas y frases no violentas. Este modelo es aplicado primero, debido a que es el encargado de realizar la distinción entre frases potencialmente violentas y las no violentas debe esperarse de él un nivel alto de precisión. Posteriormente las frases violentas detectadas se pasan por un segundo modelo, que clasifica el tipo de violencia presente: física, psicológica, sexual, simbólica o económica.
- El subcomponente encargado de efectuar el análisis requerido y generar los resultados completos. Este componente toma los archivos, hace las descompresiones necesarias, realiza un preprocesamiento para detectar información de forma manual (como la fecha de envío o el nombre del remitente) y luego carga los modelos para analizar las frases recopiladas.
- El componente generador del reporte que toma los resultados del análisis y crea un conjunto de estadísticas, gráficos y tablas para presentar la información de manera resumida al usuario.

Entidades del dominio

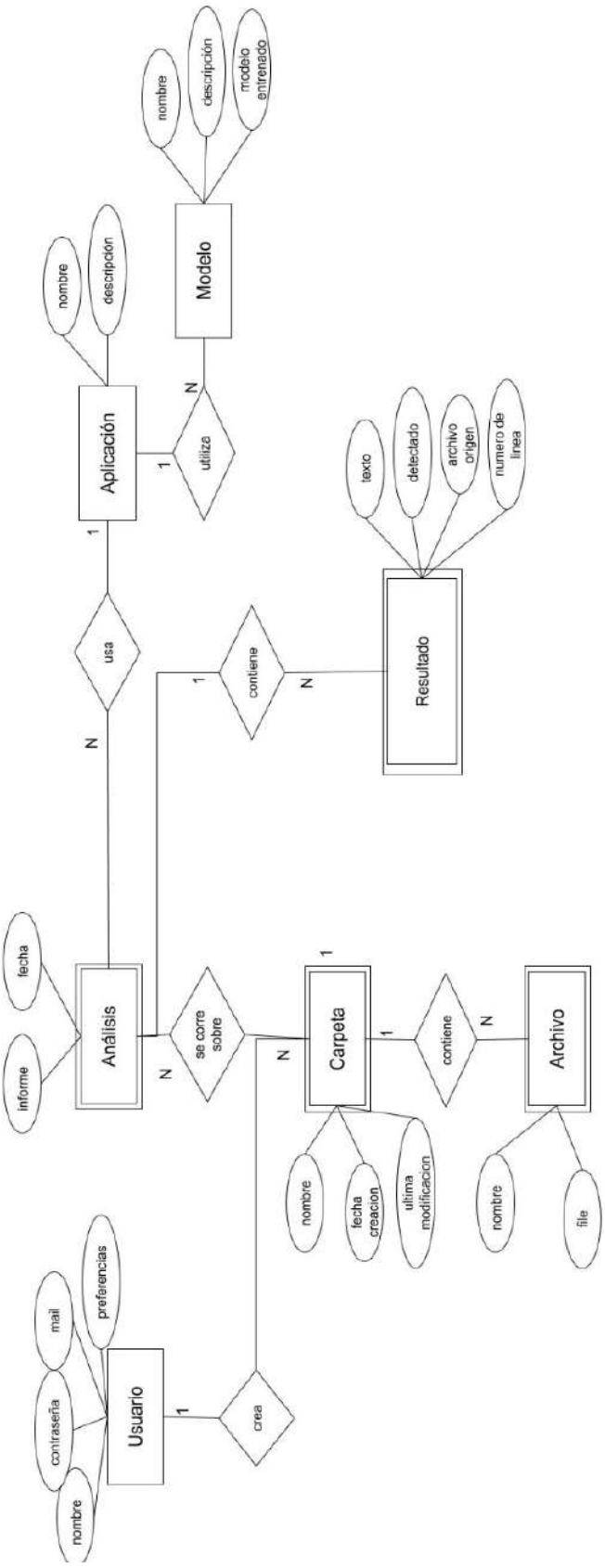


Diagrama entidad-relación.

Tecnologías

Implementación de Modelos

Entrenamiento

Para realizar los entrenamientos de los modelos se optó por el uso de la librería SpaCy de Python con la que el equipo tenía experiencia previa. SpaCy tiene la ventaja de estar disponible en una multitud de idiomas, incluyendo el español y también es de código abierto. La librería fue diseñada específicamente para producir algoritmos capaces de procesar y comprender grandes volúmenes de texto.

Etiquetado

A la hora de seleccionar una herramienta de etiquetado se tuvieron en cuenta:

- Facilidad de uso.
- Precio.
- Formatos de exportación disponibles.
- Conocimientos previos.

Se consideraron principalmente las herramientas Prodigy, Label Studio o el etiquetado sin el uso de herramientas.

El etiquetado manual sin el uso de herramientas se descartó, ya que las herramientas proporcionan eficiencia para afrontar la tarea repetitiva del etiquetado en el menor tiempo posible. Prodigy tiene la ventaja de, al ser un producto de spaCy, la exportación directa al formato necesario para el entrenamiento, pero es una herramienta paga. Por lo que se decidió utilizar Label Studio, una herramienta gratuita, que permite la exportación al formato estándar JSON que puede ser posteriormente modificado con facilidad para adaptarse al formato ‘spacy’. Además, el equipo tenía experiencia previa con el uso de esta herramienta.

Sistema Web

Backend

Para el desarrollo del backend del sistema web, se tuvieron en cuenta:

- Conocimientos previos del equipo.
- Facilidad de uso y rapidez del aprendizaje.
- Mantenibilidad.

Se optó por el uso del framework Django que facilita el desarrollo de la aplicación web, ya que viene con varios módulos y funcionalidades incorporadas. Contábamos con conocimientos previos del framework, además de tener disponible la amplia documentación. El uso de Django hace la aplicación más fácilmente escalable debido a su arquitectura MVT y la flexibilidad que conlleva. Adicionalmente, el framework incorpora manejo de usuarios y seguridad de forma automática, haciendo más sencilla la implementación.

Interfaz gráfica

En cuanto a la interfaz gráfica, fue necesario en primer lugar la creación de mockups y prototipos para la presentación y validación con el cliente. Para esta primera tarea se tuvieron en cuenta:

- Capacidad de maquetado de UX/UI.
- Conocimiento previo.
- Precio de herramienta.
- Facilidad de prototipado del diseño.

Se consideraron dos herramientas: Moqups y Figma. Moqups es una herramienta gratuita con la que el equipo tenía experiencia previa, pero no ofrece funcionalidades de prototipado y sus opciones para el diseño de UI son limitadas. Mientras que Figma ofrece un mayor rango para el diseño, además de opciones para la exportación y facilidades para la creación de mockups interactivos, si bien es un servicio usualmente pago, ofrece su plan de forma gratuita a estudiantes. Por lo tanto, se optó por el uso de Figma.

Para la implementación de la interfaz gráfica final se consideró el uso de HTML/CSS con Bootstrap, y, por otro lado, el uso de React. Debido a la naturaleza del framework Django, los conocimientos del equipo y la simplicidad de los requerimientos del diseño se optó por el uso simple de

Bootstrap en los templates de Django para generar contenido HTML dinámico.

Base de datos

En cuanto a la base de datos, Django permite el uso de los motores de bases de datos PostgreSQL, MySQL, Oracle y, por defecto, SQL Lite. Durante el proceso de desarrollo y de testing se optó por SQLite para facilitar modificaciones, creación de backups, testing y facilidad de uso inmediato. Cuando se haga el despliegue en un ambiente de producción se prevé una migración a PostgreSQL, para afianzar la seguridad del sistema y permitir mayor extensibilidad.

Integración del server web con modelos entrenados

Para la integración de los modelos entrenados en el sistema es necesario emplear el uso de un administrador de tareas para manejar el procesamiento en tiempo real de los lotes de texto solicitados. Para esta tarea se optó por Celery el sistema para gestionar colas de tareas distribuidas de forma asíncrona. Además, Celery requiere el uso de la base de datos no relacional Redis y de RabbitMQ para la transmisión de mensajes y el guardado de los avances del proceso asíncrono, respectivamente, son necesarios para posibilitar que los estados se actualicen mientras se procesan los archivos.

Seguridad

La seguridad dentro del sistema es manejada de las siguientes maneras:

- La autenticación y autorización son provistas por Django por defecto. Estas implementaciones proveen una solución predeterminada, ya desarrollada con los estándares de seguridad adecuados.
- Encriptación de datos mediante SSL/HTTPS para evitar que actores maliciosos o no autorizados obtengan las credenciales de los usuarios. Esta encriptación se realizará en la etapa de deploy del sistema, utilizando un dominio y un certificado SSL de Let's Encrypt. El proceso de implementación de la encriptación SSL/HTTPS es transparente para el sistema.
- Validación de input mediante el uso de Django Forms lo que protege contra SQL Injection.

- Prácticas de seguridad adicionales provistas por Django: CSRF, protección de cross-site scripting.
- Uso de herramientas, librerías y frameworks open-source (Django, SpaCy, Celery, etc.), que evita dependencias de software propietario y sujeto a restricciones.

Modelos de NLP

El modelo de detección de entidades nombradas fue diseñado como un modelo único entrenado como el elemento NER del *pipeline* de spaCy para detectar: dinero, fechas, horarios, lugares, medidas, organizaciones, personas, tiempo, y entidades misceláneas.

En cuanto al clasificador de violencia se decidió implementar a partir del entrenamiento de dos modelos independientes, que se aplican de manera consecutiva.

El primer modelo consiste en un clasificador binario, que distingue si una frase dada es violenta o no. Este es el modelo más refinado de los dos, que busca distinguir no solo entre frases claramente violentas o no violentas, sino también en frases ambiguas o complejas (ver sección de implementación y anexos de entrenamiento). Posteriormente el segundo modelo se encarga de clasificar aquellas frases marcadas como violentas en una categoría de ‘tipo de violencia’, este modelo simplemente clasifica la naturaleza o temática de la frase. Así, produciendo una clasificación final entre las categorías: no violento, violencia física, violencia sexual, violencia psicológica, violencia económica y violencia simbólica.

Interfaz de usuario

El diseño de la interfaz de usuario fue realizado definiendo:

- Posibles paletas de colores
- Pantallas principales del sistema
- Funcionalidades a mostrar

El diseño fue presentado a la referente funcional, modificado y por último los cambios fueron validados con la referente funcional una vez más.

Implementación de solución

En esta sección se detallan los pasos de la implementación del sistema, cada una de las iteraciones de entrenamiento de los modelos creados, y los pasos de implementación generales de la creación del servidor web.

Modelo de reconocimiento de entidades nombradas

El objetivo de este modelo es encontrar y clasificar a las distintas entidades presentes en una frase. Las posibles entidades que se busca identificar son: PERSONA, LUGAR, HORA, FECHA, DINERO, ORGANIZACIÓN, MEDIDA, TIEMPO y MISC.

Se realizaron siete iteraciones de entrenamientos para lograr un modelo con una precisión aceptable. Los datos recolectados y etiquetados para estos entrenamientos fueron sacados de noticias, un dataset de frases comunes en español creado por Google, y Tweets argentinos. Los tweets fueron extraídos utilizando la API de Twitter, y posteriormente sometidos a un trabajo de limpieza eliminando: menciones, links, hashtags y tweets realizados por bots o en inglés. Esta limpieza fue parcialmente automática, mediante el uso de scripts creados en Python para la limpieza, y parcialmente manual, mediante el uso de herramientas como Label Studio.

En la primera iteración de entrenamiento se utilizaron más de 2000 frases etiquetadas para el entrenamiento. Los modelos entrenados en esta iteración tuvieron dificultades para distinguir dígitos para fechas, medidas, y dinero. Muchas frases eran muy similares entre sí, lo que pudo causar overfitting en el modelo. Además, las frases más informales usadas para el entrenamiento, tienen un uso irregular de mayúsculas, minúsculas y puntuación en un texto, causando confusión en el modelo. Por último, se distinguió en ambos entrenamientos la necesidad de unificar el criterio de etiquetado en algunas categorías en las frases de entrenamiento.

Para las siguientes iteraciones, se acotó el dataset a menos ejemplos, pero buscando una representación equitativa de cada una de las entidades. Además, se unificó el criterio de etiquetado y se realizaron diferentes configuraciones de entrenamiento. Finalmente se llegó a modelos con errores aceptables, y que mantenían un buen funcionamiento con datos de entrada escritos correctamente y con datos

informales (usualmente conteniendo errores gramaticales o de puntuación). El siguiente cuadro muestra un resumen de los resultados de entrenamiento de los modelos:

	f-score	**% Etiquetas correctas encontradas**	**% Etiquetas faltantes**	**% Etiquetas erróneas**
Entrenamiento 1.1	54	28,95%	71,05%	1,52%
Entrenamiento 1.2	27	42,11%	57,89%	1,52%
Entrenamiento 2.1	29	52,63%	47,37%	2,28%
Entrenamiento 2.2	52	**79,49%**	20,51%	**12,9%**
Entrenamiento 3.1	74	68,42%	31,58%	0,76%
Entrenamiento 3.2	**79**	**71,05%**	**28,95%**	**0,38%**
Entrenamiento 4.1	56	52,63%	47,37%	0,38%
Entrenamiento 4.2	69	60,53%	39,47%	0,38%
Entrenamiento 4.3	61	57,89%	42,11%	0,76%
Entrenamiento 4.3.t	**79**	**84,62%**	**15,38%**	**2,56%**

(Ver “*Anexo de entrenamiento de modelo de entidades nombradas para ver los detalles de cada entrenamiento”*)

Se puede observar que los modelos que lograron los mejores resultados fueron los entrenamientos 2.2 y 4.3.t que fueron entrenados con transformers. Su desventaja es que estos modelos son considerablemente más pesados que los demás. Además, si bien en los entrenamientos de la tercera iteración lograron buenos resultados, los modelos entrenados en el entrenamiento 4 contienen más datos de entrenamiento con frases informales y con errores. Debido a esto este modelo puede ser más

adecuado para buscar entidades en textos que sean de esta naturaleza informal.

Modelo de clasificación de violencia binario

El objetivo de este modelo es clasificar una frase de un modelo en dos categorías posibles: “Violento” o “No Violento”. Al igual que el modelo de reconocimiento de entidades, esto se realizó con la librería SpaCy, con la diferencia de que el *pipeline* utilizado fue “textcat”.

Para la construcción de los datasets de entrenamiento, se utilizaron cuatro fuentes de datos:

1. Noticias: Un conjunto de oraciones provenientes de noticias obtenidas de internet. Las temáticas de estas oraciones son diversas y de naturaleza no violenta. Son oraciones que tienen gramática correcta, formal y de naturaleza descriptiva.

2. Frases de Google: Un conjunto de frases provenientes del *“Crowdsourced high-quality Argentinian Spanish speech data set”*, un dataset creado por Google. Este dataset contiene 2278 frases conjugadas en “español argentino” que consiste en transcripciones realizadas por hablantes de Buenos Aires. Son de naturaleza no violenta, con gramática correcta, más informal y corta.

3. Tweets: Frases extraídas mediante la API de Twitter. La API permite extraer una gran cantidad de datos con un término de búsqueda y localización geográfica específica. En este caso, estos tweets se buscaron utilizando palabras clave e insultos violentos en la región de Argentina. Se eliminaron menciones, links, *hashtags* y tweets realizados por bots o en inglés. Esta limpieza fue parcialmente automática y parcialmente manual. Estas frases hacen uso de una jerga muy marcada, son muy informales y suelen tener ortografía incorrecta.

4. Frases violentas del juzgado Nº10: extraídas del dataset del juzgado número 10 de Buenos Aires. Incluyen en su mayoría amenazas de violencia tipo física. Son frases que muchas veces tienen mala ortografía, son cortas e informales.

Se realizaron 7 iteraciones de entrenamientos. En los primeros entrenamientos se fue expandiendo el conjunto de datos, agregando conjuntos de frases acorde a los resultados de cada modelo, en ocasiones re etiquetando datos o eliminando repeticiones de frases similares. Se buscó en cada uno de ellos llegar a altos niveles de precisión y evitar el overfitting, ya que el modelo de clasificación binaria resulta indispensable para el buen funcionamiento del sistema. Este modelo es ejecutado previamente al modelo multicategoría, y es el encargado de identificar la presencia de violencia. En el entrenamiento número 6 se identificaron los principales errores encontrados en los entrenamientos anteriores, utilizando un conjunto de prueba compuesto por frases violentas y no violentas, ambiguas y no ambiguas. Los problemas identificados fueron:

1. Las frases del tipo “Te voy a” o “Sos una” tendían a ser clasificadas como violentas para cualquier combinación. Por ejemplo con la frase “Te voy a ayudar” es clasificada como “Violento”. Esto es porque muchos de los ejemplos de frases violentas comienzan de esa manera, por lo que el clasificador tiende a considerar como violentas todas estas frases.
2. Existen algunas frases violentas típicas “te voy a prender fuego”, que todavía no eran reconocidas cuando se las cambia ligeramente “te prendo fuego”.
3. Algunas frases no violentas que contienen alguna palabra similar a la de una frase violenta se les asigna un score “Violento” demasiado alto. Por ejemplo, la frase no violenta “Ese es mi perro” tiene un score de violento de 0.39, porque la palabra “perro” es muy parecida a “perra”, usada en la frase violenta “Sos una perra”.
4. Muchas frases violentas, pero que no incluyen insultos normalmente, por ejemplo del tipo de violencia económica (“Solo me haces perder plata”) o psicológica (“Guarda conmigo, porque sé donde vivís”) no eran detectadas correctamente como violentas por falta de ejemplos en el dataset de entrenamiento.

Cada uno de estos problemas fue resuelto en un proceso de varias etapas, que consistieron en: la normalización de las frases de entrenamiento, la modificación e incorporación de frases, la incorporación de frases violentas poco representadas y por último la eliminación de mayúsculas. Para el último entrenamiento se tuvieron en cuenta estas resoluciones y se agregaron más ejemplos de entrenamiento para mejorar los resultados.

A continuación se agrega un resumen del número de fallos para cada sección y el número total de fallos.

	F-score	Frases Violentas Correctas	Frases No Violentas Correctas	Errores totales	% Frases correctas encontradas
Entrenamiento 1	91	98	20	82	59%
Entrenamiento 2t	**94**	**84**	**90**	**26**	**87%**
Entrenamiento 3t	94	97	68	35	82,5%
Entrenamiento 4	98	100	27	73	63,5%
Entrenamiento 5	88	98	32	70	65%
Entrenamiento 6	84	96	68	36	82%
Entrenamiento 7	**79**	**96**	**82**	**22**	**89%**

(Ver detalles de los entrenamientos en “*Anexo de entrenamientos de modelos de clasificación de violencia binaria”*)

Modelo de clasificación de violencia multicategoria

El objetivo de este modelo es clasificar una frase violenta en una categoría que indique el tipo de violencia principal que presenta. Definiendo las categorías como:

Física: Amenazas o menciones de haber producido o tener la intención de producir dolor, daño, o afectar la integridad física.

Sexual: Amenazas o menciones de violencia sexual, es decir la vulneración del derecho de decidir voluntariamente acerca de la vida sexual o reproductiva a través de amenazas, coerción, uso de la fuerza o intimidación

Psicológica: Daños a las emociones y la autoestima. Se distingue que casi cualquier frase violenta detectada y clasificada en alguna de las otras categorías puede también ser referida como violencia psicológica

Económica: Se dirige a ocasionar un menoscabo en los recursos económicos o patrimoniales.

Simbólica: A través de patrones estereotipados, mensajes, valores, íconos o signos transmita y reproduzca dominación, desigualdad y discriminación en las relaciones sociales, naturalizando la subordinación de la mujer en la sociedad.

Para la implementación de este modelo se realizaron 3 iteraciones de entrenamiento. Los distintos datasets utilizados para los primeros dos entrenamientos se compusieron en base a las cuatro fuentes de datos usadas para el modelo binario (noticias, tweets, dataset de Google y frases violentas del juzgado Nº10), pero etiquetados en las categorías mencionadas. Esto resultó en un dataset de entrenamiento desbalanceado, es decir con más ejemplos para unas categorías que para otras. Esto supone un problema debido a que el desbalance hace que el modelo esté sesgado, tendiendo a elegir categorías más representadas. Para el tercer entrenamiento se buscaron y agregaron manualmente ejemplos de las categorías necesarias para obtener un dataset lo más balanceado posible.

Entrenamiento	**f-score**
Entrenamiento 1	31
Entrenamiento 2	56
Entrenamiento 3	74

(Ver detalles de los entrenamientos en “*Anexo de entrenamientos de modelos de clasificación de violencia multicategoria”*)

Sitio web

Para la implementación de la página web se optó por comenzar por el diseño de la UI y la creación de un prototipo para la presentación al cliente temprana, permitiendo la confirmación o cambios de los requerimientos previamente relevados.

Luego de esta etapa se pasó al diseño de la base de datos utilizada por Django, y la implementación del backend y frontend de las vistas. Además, se implementó el módulo de procesamiento NLP que hace uso de los modelos entrenados. Se integró el módulo de NLP al sistema utilizando

el manejador de tareas celery permitiendo así el procesamiento simultáneo de múltiples archivos de texto sin interrumpir el funcionamiento del sitio web.

El sistema fue documentado utilizando docstrings y durante todo el proceso se tomó nota del progreso y los errores o inconvenientes que surgieron.

El producto

El producto resultante del proyecto es una herramienta denominada OlympIA. Su principal característica es la de ofrecer el uso de modelos entrenados de NLP para el análisis de textos con especial atención a la detección de patrones relacionados con la violencia de género. OlympIA es planteado como una herramienta que puede ser utilizada al inicio de un proceso judicial, en el momento de recibir una denuncia y realizar una extracción o preservación de datos a analizar. Sin embargo, el sistema también tiene la potencialidad de ser aplicado en una etapa posterior, de investigación o pericia.

Inicialmente, los dos modelos utilizables en el producto son los de detección de entidades y de clasificación de violencia binaria y violencia multicategoría. Los modelos ofrecen distintos tipos de resultados, que incluyen gráficos, informes y herramientas visuales para poder identificar la información relevante. Además, la plataforma es extensible, y permite integrar distintos tipos de modelos.

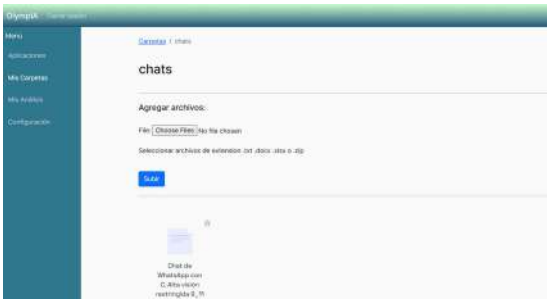
Características Principales

El sistema OlympIA se encuentra dividido cuatro secciones principales:

- Aplicaciones: Es la página donde empieza el flujo para realizar un nuevo análisis con cualquiera de los modelos cargados.
- Mis Carpetas: Es la página donde se cargan los archivos a ser analizados. La pantalla permite crear, modificar y borrar carpetas y sus archivos internos.
- Mis análisis: Es la página que permite visualizar y eliminar los resultados ya creados.
- Configuración: Es la página que permite configurar aspectos visuales como la paleta de colores o configuraciones de seguridad como la contraseña.

Interfaz de carga y análisis de textos

El sistema OlympIA ofrece una interfaz intuitiva que permite a los usuarios cargar lotes de textos de diferentes formatos (.docx, .txt, .zip) de manera sencilla a distintas carpetas. Esto permite que los usuarios ejecuten análisis de sus archivos de manera organizada.

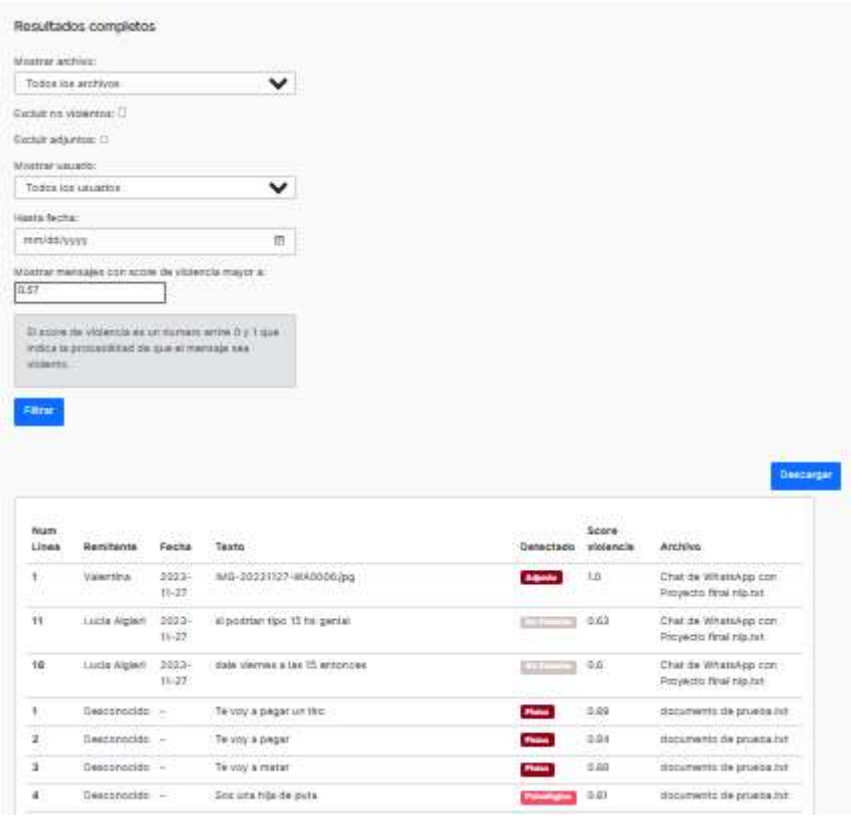


Clasificación de mensajes

Es una funcionalidad clave que permite la identificación de los potenciales indicadores de violencia de género y la clasificación de dichos mensajes según los distintos tipos de violencia:

- Sexual
- Física
- Económica
- Psicológica
- Simbólica
- No violenta

La clasificación de mensajes se puede utilizar con mensajes de WhatsApp. En este caso también es capaz de procesar información adicional como la fecha y hora de envío, y el usuario remitente. Las opciones de filtrado de los resultados, permiten facilitar la tarea de identificar patrones de agresión y su evolución en el tiempo.



Detección de entidades en texto

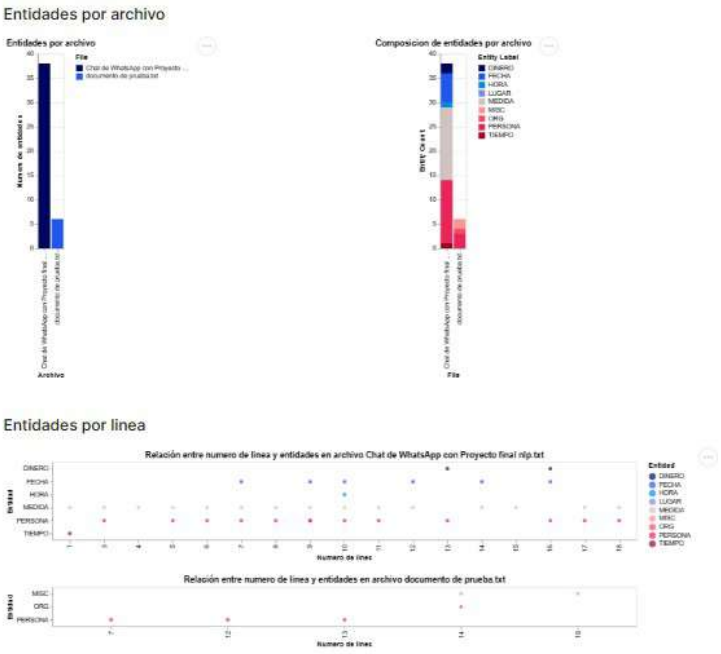
Es una funcionalidad clave que permite detectar distintos tipos de entidades presentes en las frases analizadas. Los modelos entrenados detectan las entidades:

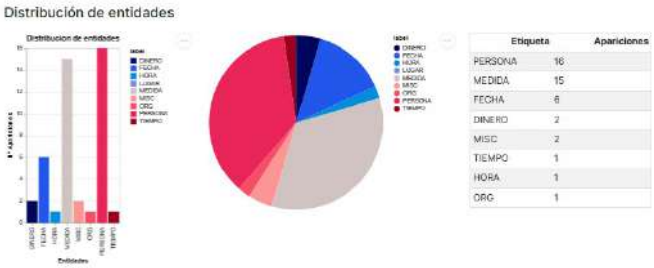
- PERSONA
- LUGAR
- HORA
- FECHA
- DINERO
- ORGANIZACIÓN
- MEDIDA
- TIEMPO
- MISC

Los resultados pueden ser filtrados para mostrar solo las líneas que contengan ciertos tipos de entidades específicas. Y en un futuro es posible incorporar al sistema nuevos modelos con diferentes conjuntos de entidades.

Visualización de resultados

La principal característica del producto es que al finalizar el análisis muestra los resultados completos. Estos resultados incluyen tablas, gráficos y otras herramientas visuales (como la nube de palabras) que permiten destacar con mayor facilidad la información relevante. Estos resultados además se pueden filtrar con distintos parámetros, como por ejemplo los tipos de entidades detectadas. Además, el producto ofrece la capacidad de descargar los documentos con los resultados obtenidos, y generar y descargar nuevas nubes de palabras descartando palabras específicas.





Memoria del proyecto

Esta sección detalla los acontecimientos ocurridos durante el desarrollo del proyecto, el cumplimiento de los objetivos y el análisis de los tiempos por etapa.

Objetivos

El objetivo principal del desarrollo de este proyecto fue usar las herramientas tecnológicas disponibles en el campo de la inteligencia artificial y el desarrollo web para contribuir a la lucha contra la violencia de género. Al inicio del proyecto, se encontró con un contexto donde los procesos de análisis de búsqueda de evidencia digital eran lentos y desgastantes para los investigadores y había un amplio margen de mejora. Los procesos tradicionales hacen más lenta la actuación de la Justicia para defender a las víctimas y desde la Universidad Nacional de Mar del Plata y el Infolab, se decidió proveer una solución para esta problemática.

El resultado final del desarrollo del proyecto es OlympIA. Por un lado, este sistema aprovecha la potencia de la librería de NLP de SpaCy para poder crear modelos de clasificación de violencia capaces de analizar miles de frases en minutos con precisión satisfactoria; y la facilidad de uso y la facilidad de uso de una aplicación web provistas por el uso del framework Django para poder utilizar estos modelos de un modo sencillo, visualmente rico y con mínima capacitación. La combinación final de estas cualidades dio como resultado un producto con una curva de aprendizaje sencilla, rapidez notablemente superior a las formas de búsqueda e identificación hecha por humanos y que incluyen tablas y gráficos varios, que facilitan la interpretación y la identificación de los datos necesarios para los usuarios. Además el sistema proporciona la flexibilidad necesaria para expansiones futuras con nuevos modelos.

Entidad	Apariciones
27/11/2023,	15
viernes	6
Castellote	5
Lucia Algieri	5
27/11/2023	2

El sistema implementado logra dismin
pocas horas, esto no solamente es
costos o en desgaste mental, psicológ
en analizar evidencia, sino que puede
las víctimas de violencia que se inte
velocidad y flexibilidad hace que Olyr
en la búsqueda de justicia y protección para las víctimas de violencia de género.

Objetivos específicos

Realizar el análisis del dominio necesario para comprender las necesidades y requerimientos específicos de los usuarios en el ámbito judicial.

El análisis del dominio se realizó durante las fases iniciales del proyecto a partir de varios puntos conjuntamente con el resto del equipo, durante varias reuniones:

- Comprensión del problema: realizado junto a la referente funcional y al director del proceso. Se identificaron las principales necesidades actuales y las posibilidades de mejora que el producto puede otorgar.
- Identificación de los requisitos y actores.
- Definición de alcances y límites: se identificaron los requisitos tecnológicos, los modelos a desarrollar y prioridades de desarrollo.
- Identificación requerimientos del proyecto, tanto funcionales como no funcionales.

Generar tanto el producto final como la documentación e investigación de tecnologías utilizadas a raíz del desarrollo.

A lo largo de la implementación de todos los modelos se documentaron los distintos entrenamientos, tomando nota de cada paso realizado y decisión tomada, los cambios respecto a entrenamientos anteriores, y los resultados de las pruebas.

Además, el sistema web implementado en Django fue documentado utilizando docstrings, definiendo cada una de las funciones y funcionalidades del sistema.

Tener en cuenta la posibilidad de futura expansión del sistema a través de la incorporación de nuevos modelos o de la integración de tecnologías adicionales.

Se usaron tecnologías ampliamente documentadas y reconocidas que facilitan la extensión futura del sistema. Django, con su arquitectura MVT, permite incorporar nuevas aplicaciones, modelos y la fácil migración del

motor de base de datos. SpaCy es una herramienta de código abierto, de uso amplio y posee documentación actualizada lo que facilita incorporar nuevos modelos rápidamente.

Además, se modularizó el componente de NLP para permitir incorporar nuevas aplicaciones o modelos. Esto permite integrar nuevas tecnologías en el futuro, como OCRs o herramientas de transcripción de audio como Whisper.

Poner énfasis en la visualización de los datos resultantes del análisis para facilitar su comprensión y análisis.

El sistema genera gráficos y tablas para visualizar los resultados de los análisis. Los gráficos de clasificación general incluyen gráficos de torta y de barras que muestran las entidades encontradas y las frases clasificadas como violentas. Los gráficos de composición de los archivos que muestran en qué archivo y en qué sección se encuentran las frases detectadas. Por último, los *'wordclouds'* (nubes de palabras) resaltan las palabras más frecuentes en los datos analizados y con la opción de excluir ciertos términos.

Todos los gráficos fueron creados utilizando Altair, una librería de visualización de datos de Python que permite su descarga en varios formatos y ofrece un nivel básico de interacción.

Disponibilizar el uso de la herramienta mediante una interfaz web para facilitar el acceso y priorizar la usabilidad.

Se creó una interfaz web usando Django, lo que permite su ejecución rápida de manera local y facilita el despliegue del proyecto en un servidor Linux.

Se usaron mock ups para visualizar el producto y hacer correcciones. Se mantuvieron canales de comunicación con la referente funcional y el codirector para dar actualizaciones del desarrollo y reuniones para validar el producto y asegurar que cumpla con las necesidades de los usuarios. Durante la última fase se priorizaron los cambios sugeridos por los referentes a nivel visual para mejor usabilidad por parte de los usuarios.

Permitir la carga de lotes de texto a carpetas que puedan ser procesadas para obtener un informe con los resultados deseados.

El sistema permite a los usuarios cargar lotes de texto para su procesamiento. Se soportan archivos de formatos: .txt, .doc, .docx, .xls y .zip que contengan archivos de cualquiera de los anteriores formatos. Los usuarios pueden administrar sus carpetas y organizar sus archivos de la manera más conveniente. Este aspecto es expandible en el futuro a nuevos formatos.

Desarrollar al menos dos modelos de procesamiento de lenguaje: análisis de violencia y reconocimiento de entidades.

Fueron desarrollados modelos de NER que capaces de identificar y clasificar entidades.

Además, se desarrollaron dos modelos de análisis de violencia: un Modelo Binario y un Modelo de Categorización de Violencia. El Modelo Binario clasifica los textos como violentos o no violentos y actúa de 'filtro' antes de aplicar el segundo modelo, por lo tanto, fue el modelo entrenado con mayor cantidad de datos. El Modelo de Categorización de Violencia categoriza los textos violentos como violencia física, psicológica, sexual, económica o simbólica. Ambos fueron entrenados en múltiples rondas hasta lograr resultados satisfactorios en sus clasificaciones.

Realizar pruebas para validar la efectividad y precisión de los modelos creados.

Durante el desarrollo se utilizaron dos formas de evaluación para validar la precisión de los modelos: el f-score producido por los modelos y la evaluación con frases reales.

El f-score es un parámetros que va de 0 a 100, generados al final del entrenamiento. Los primeros modelos tenían un score bajo, de entre 30 y 50. Conforme se expandieron, el f-score subió hasta cercanos a 99, un claro indicio de *overfitting*. Los últimos entrenamientos tuvieron un f-score cercano a 80, correspondientes con las pruebas con frases reales.

Se llevaron a cabo pruebas en conjuntos de datos de prueba no usados en los entrenamientos para evaluar la evolución de los modelos. Inicialmente, los conjuntos de pruebas solo incluían frases simples y posteriormente se agregaron frases más ambiguas.

División de Trabajo

Para la gestión de tareas, utilizamos Trello, una herramienta de gestión de proyectos para visualizar y dividir las tareas correspondientes a cada etapa, además de registrar los tiempos dedicados a cada una de ellas.

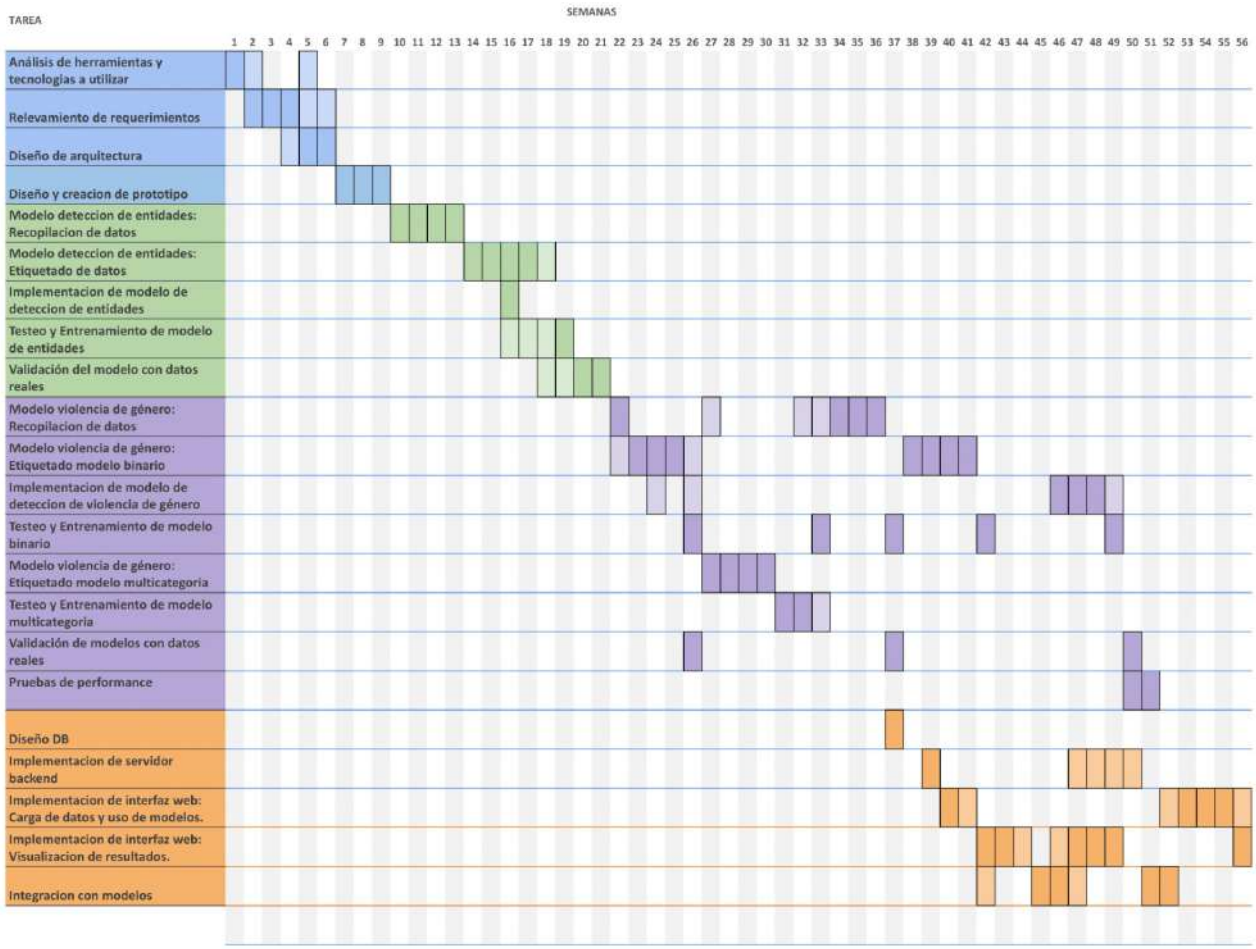
Durante las primeras etapas del proyecto, las tareas fueron realizadas en conjunto. Esto incluyó, el relevamiento de los requerimientos, el diseño del prototipo y su validación, la búsqueda de datos, la etiquetación de los mismos, la unión de los datos y el entrenamiento de los modelos.

En las dos últimas etapas del proyecto se decidió dividir las tareas entre los integrantes, dejando, por un lado, las tareas de implementación del server web, y por otro los últimos entrenamientos y pruebas de los modelos, aunque algunas tareas fueron realizadas en conjunto. Para este tipo de tareas, la división se dio de manera natural. Esto quiere decir que uno de los miembros que tenía más experiencia en un tipo de tarea (por ejemplo, el desarrollo con Django) llevaba a cabo la mayoría del trabajo. En momentos en los que se encontraba alguna dificultad, se revisaba en conjunto para poder resolver y seguir con el desarrollo. Esto se dio por ejemplo durante la integración de Django con Celery, tarea que demandó mucha investigación y cambios para que funcione exitosamente.

Las últimas tareas de refinamiento y de resolución de errores se realizaron en conjunto, al igual que la redacción del informe final. Aquí las tareas eran más parecidas entre sí. Estas tareas incluyeron además la elaboración de evaluaciones de rendimiento y precisión, y se dividieron basándonos en el conocimiento y quien había tenido más interacción previa con los modelos evaluados.

Análisis de tiempos

A lo largo del proyecto, hubo desviaciones en el tiempo estimado y real. La duración total del proyecto fue de aproximadamente 56 semanas, un aumento del 40% de las 40 semanas estimadas inicialmente. En términos de horas, se dedicaron 694 horas, lo que es un 13% menos de las 800 horas estimadas. Además, se había estimado una carga horaria constante de 2 horas diarias, esta constancia no se cumplió durante el desarrollo del proyecto debido a retrasos, o periodos con tareas más demandantes que implicaban un incremento horario.

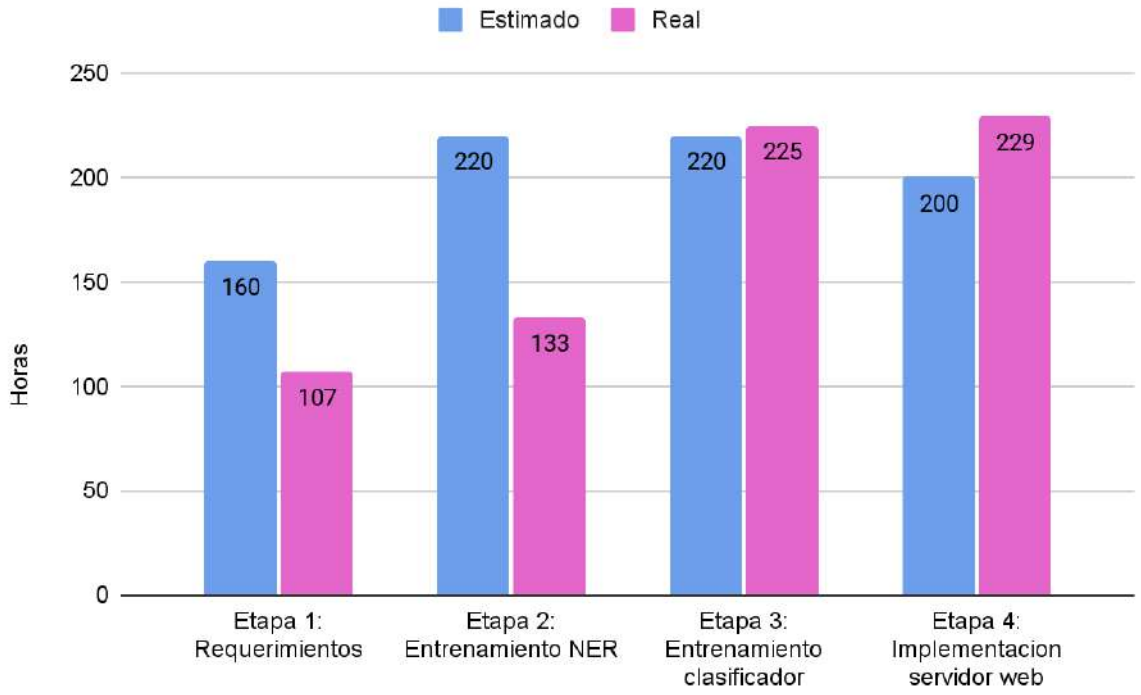


Se indica en el Gantt qué tareas tienen la mayor dedicación en cada periodo de tiempo, destacando las mismas con un color más intenso. Cada color indica una etapa diferente.

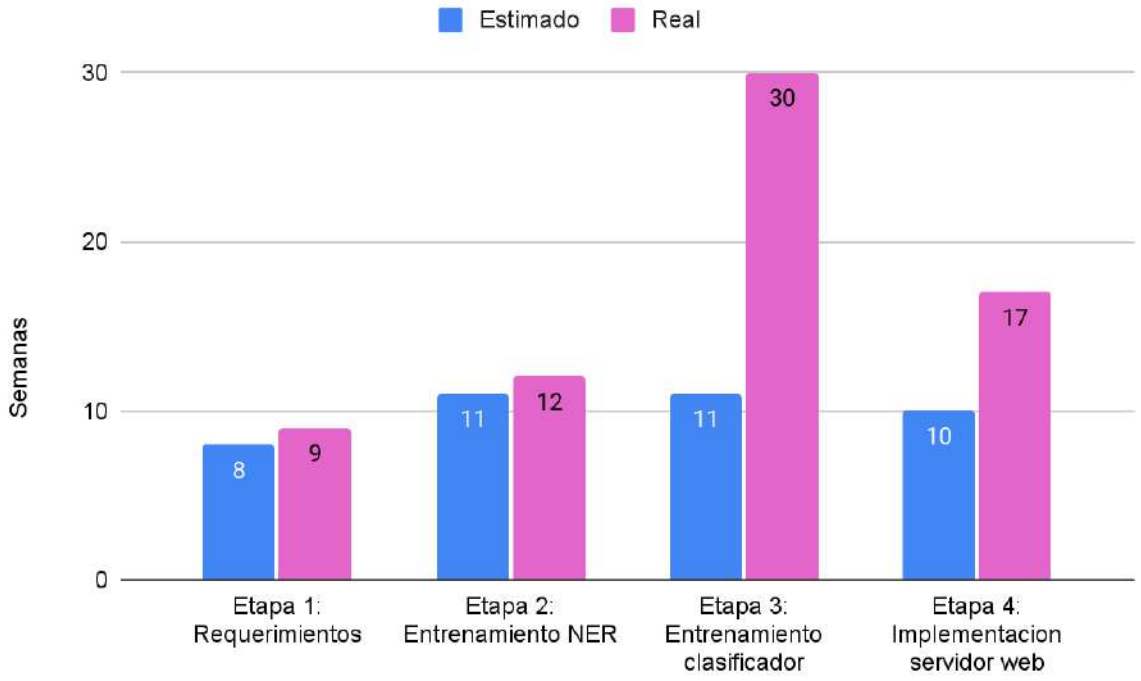
El presente informe fue redactado una vez finalizadas las etapas detalladas anteriormente. La redacción del informe tuvo una duración aproximada de dos meses de escritura, con dedicación irregular de horas semanales a lo largo de ese periodo, exceptuando el receso por las fiestas de fin de año. Además, la corrección y reentrega del informe necesitó de dos semanas adicionales.

Métricas y Análisis de las Etapas

Horas estimadas vs horas reales dedicadas por etapa



Semanas estimadas vs semanas reales dedicadas por etapa



- Etapa 1: Relevamiento de requerimientos

Durante la primera etapa del proyecto se realizó un relevamiento de los requerimientos del sistema y un análisis de las herramientas a utilizar, especialmente la librería SpaCy. Además, se definió la tarea adicional de

la creación de un prototipo del sistema web. Este prototipo permitió mostrar al cliente una versión preliminar del sistema, facilitando la validación de los requerimientos y la identificación de posibles cambios.

Esta etapa experimentó algunos retrasos debido a las vacaciones de Infolab en enero de 2023 y la ausencia de algunos integrantes del equipo por la preparación de exámenes finales. Pese a esto, la etapa fue completada aproximadamente en el número de semanas estimadas, ya que requirió un menor número de horas de dedicación.

- Etapa 2: Modelo de reconocimiento de entidades nombradas

Durante esta etapa, se trabajó en el desarrollo de un modelo de reconocimiento de entidades nombradas. Se dividió en dos la tarea de recopilación y etiquetado de datos para gestionarlas de forma más eficiente.

Para la tarea de extracción de datos para el etiquetado, se utilizaron algunos datasets preexistentes, y además se utilizó la API de Twitter para extraer datos de la región de Argentina. Debido al aviso de un próximo cambió de política de Twitter que haría su API un servicio pago, se dedicó tiempo extra a la extracción de tweets utilizando la API con la intención de usar estos datos también para el entrenamiento de los modelos de clasificación de violencia.

La clasificación de las frases extraídas demandó más tiempo del esperado, no solo por la cantidad de frases, sino que fue necesario redefinir los criterios de clasificación por resultados insatisfactorios y reetiquetar los datos. Muchas de las frases no llegaron a considerarse como válidas para usar en el modelo por su baja calidad o repetitividad. El descarte de este tipo de frases también requirió tiempo.

Durante el desarrollo de esta etapa salió al mercado la herramienta ChatGPT. Este chatbot tenía la ventaja de poder generar variaciones en los ejemplos existentes. Se utilizó para generar ejemplos conteniendo entidades de tipos poco representados en el dataset, a partir de ejemplos de datos ya recolectados. Por lo general solo 20 o 30 frases nuevas por categoría fueron utilizables e incorporadas en el dataset debido a las limitaciones de la herramienta. Actualmente, la herramienta no permite su uso para la creación de datos de entrenamiento de ningún tipo.

La etapa fue completada aproximadamente en el número de semanas estimadas, pero requirió un menor número de horas de dedicación.

- Etapa 3: Creación de modelos de clasificación de violencia

En esta etapa, se decidió utilizar dos modelos para la clasificación: uno binario (violento/no violento) y otro multicategoría (distinguiendo entre tipos de violencia). Esto llevó a la división de algunas tareas para distinguir entre la creación de estos modelos. Además, se añadió la tarea de pruebas de performance de los modelos, para poder comparar los distintos entrenamientos.

La tarea de extracción de datos se volvió más larga debido a la necesidad de realizar la búsqueda y extracción de forma manual, por el cambió de política del uso de la API de Twitter. Además, los conjuntos de datos preexistentes seleccionados requirieron trabajos de limpieza y normalización para poder ser utilizados para el entrenamiento. Durante esta etapa también se experimentó con diversas técnicas de expansión de frases, con el objetivo de mejorar y ampliar los datasets de entrenamiento. Los resultados se volcaron a un notebook disponible en el repositorio, pero finalmente no fueron utilizados para el entrenamiento.

En un principio, se creó un primer modelo binario, y posteriormente se creó el modelo multicategoría. Al hacer pruebas del modelo multicategoría junto con el modelo binario entrenado, se encontró que el modelo binario filtraba como ‘no violentas’ muchas frases que deberían ser consideradas violentas. En su mayoría se encontraban como violentas solo aquellas frases que aludían a violencia física. Por lo tanto, se identificó la necesidad de recolectar más datos y realizar más entrenamientos en el clasificador binario para lograr los resultados esperados. Debido a esto se volvió a la búsqueda de datos para el clasificador binario y a su reentrenamiento.

Dentro de esta búsqueda se encontró un dataset que recopilaba las publicaciones de la red social Reddit. El dataset consistía de archivos de varios gigabytes de tamaño. Realizamos un trabajo de automatización para la extracción de publicaciones en español, primero tomando la publicaciones pertenecientes a ‘subreddits’ de países latinoamericanos, y luego buscando todas aquellas frases que tenían una alta proporción de palabras presentes en un diccionario español. Estos procesos resultaron computacionalmente demandantes, y, por limitaciones de hardware, temporalmente demandantes. Finalmente al comenzar el trabajo de etiquetado de los datos extraídos, se encontró que un alto porcentaje de los datos consisten de frases poco útiles para el entrenamiento (frases

armadas repetidas, publicidades, mensajes de ‘bots’, etc.). Como un último intento, se utilizó uno de los modelos binarios entrenados con anterioridad sobre los datos extraídos, pero los resultados continuaron siendo desfavorables. Finalmente se descartó el uso del dataset, y se optó por la recolección manual de datos usando las funcionalidades de búsqueda avanzada de Twitter, testimonios en blogs y noticias, y otras fuentes.

La segunda mitad de esta etapa se realizó de manera paralela con la etapa 5. Se realizaron varios ciclos de entrenamiento y reentrenamiento (búsqueda de datos, etiquetado, entrenamiento y prueba). Debido a los distintos retrasos y la decisión de paralelizar parte de esta etapa con la siguiente, esta etapa llevó un poco más del número de horas estimadas y casi el triple de semanas estimadas.

- Etapa 4: Creación del servidor web.

Esta etapa consistió en la creación del sistema web que utiliza los modelos entrenados, para el análisis de datos y la creación de reportes de los resultados de esos análisis. Se modificó la planificación original definiendo como tareas propias el diseño de la base de datos y la integración de los modelos de spacy al sistema.

Uno de los principales desafíos de esta etapa fue la integración de Celery, que requirió la atención sincrónica de los dos integrantes del equipo, ya que uno tenía conocimientos sobre Celery y el otro estaba familiarizado con todo el sistema de Django. Esto incrementó el tiempo necesario para completar las tareas de integración. La etapa tomó más semanas y horas de las inicialmente estimadas, debido al paralelismo con la etapa anterior y los retrasos en la integración de los modelos. Otro desafío de esta etapa fue la integración de gráficos generados por Altair en la parte de visualización de datos.

Además, la etapa también incluyó tareas de refinamiento del sitio a nivel visual y de funcionamiento. Esto incluyó la corrección de *layouts*, funcionamiento de los botones y reorganización de la ventana de resultados de los análisis.

Conclusión

Existen distintas legislaciones del país que protegen a la mujer de las situaciones de violencia de género, como la Ley 26.485 *“Ley de Protección Integral para Prevenir, Sancionar y Erradicar la Violencia contra las Mujeres en los ámbitos en que desarrollen sus relaciones interpersonales”*, la *“Convencion Interamericana para Prevenir, Sancionar y Erradicar la Violencia contra La Mujer”* - conocida como "*Convención de Belem do Pará*", la recientemente promulgada 26.485 *“Ley Olimpia”* entre otras. Es crucial el reconocimiento de la desigualdad y la violencia sufrida por las mujeres y el colectivo LGBTQ+ por el Poder Legislativo dando lugar a normas como las anteriormente mencionadas. También son necesarias las políticas de gobierno para la prevención y la aplicación de la Justicia de forma eficiente y eficaz desde el Poder Judicial para que los casos de violencia no queden impunes.

Hoy en día la Justicia, en su búsqueda por recolectar la evidencia necesaria para poder llevar a cabo los juicios, debe recurrir a métodos manuales al lidiar con grandes cantidades de texto. Como única alternativa existen herramientas que requieren de entrenamiento avanzado de sus usuarios. En ese contexto, se dió la oportunidad de contribuir con una herramienta tecnológica simple, rápida y de fácil uso, que posibilitará a los investigadores y fiscales el análisis de grandes cantidades de datos en muy poco tiempo.

El objetivo principal de este proyecto fue el de ofrecer una herramienta para la lucha contra la violencia de género, específicamente buscando agilizar los procesos judiciales que protegen a las víctimas y llevan a sus agresores a la justicia. A lo largo del desarrollo del proyecto se enfrentaron muchos desafíos, muchas veces desafíos no técnicos, por ejemplo la falta de datos de entrenamiento (resultante de las limitadas políticas de datos abiertos presentes en la región) los tiempos extendidos para el etiquetado, las incompatibilidades con el software o la integración con otras herramientas. Uno de los aspectos más inesperados fue el nivel de violencia encontrada en las redes. En un principio se consideró que encontrar ejemplos de violencia sexual podría ser problemático debido a las políticas de restricción de contenido de las redes, pero este tipo de violencia fue encontrado fácilmente, y en abundancia. Se lograron extraer miles de ejemplos de expresiones violentas como datos de entrenamiento.

El desarrollo incluyó la creación exitosa de modelos de procesamiento de lenguaje, para el análisis de violencia y reconocimiento de entidades. Las pruebas continuas validaron la precisión de los modelos, y fueron un aspecto clave para la incorporación de los conceptos teóricos de desarrollo de modelos de inteligencia artificial. Finalmente, se creó un sistema basado en inteligencia artificial para el análisis de textos a través de una interfaz web. Se buscó priorizar la usabilidad del sistema a partir de la creación de prototipos y la comunicación con el referente.

El proyecto se dividió en varias etapas, que incluyeron el relevamiento de requerimientos, el diseño de la solución, la creación de los modelos y la implementación del sistema web, para lograr crear el producto OlympIA. Durante el desarrollo se extrajeron datos de varias fuentes del lenguaje en español. Se priorizaron los ejemplos de la región de Argentina, para poder generar modelos especializados para su uso en la región. Se ejecutaron varios ciclos de entrenamiento y validación hasta alcanzar una precisión aceptable. Los modelos de clasificación de violencia resultantes del proyecto lograron distinguir una gran proporción de frases ambiguas o complejas.

El proceso de entrenamiento mediante SpaCy, una vez clasificadas las frases, compiladas en el formato correcto, fue relativamente sencillo. El tamaño de la gran mayoría de estos modelos no excedió los 30 MB, con la excepción de los casos que incorporan vectores preexistentes o fueron entrenados con *transformers*. Esto permitió modificar, refinar y reentrenar los modelos varias veces, hasta que se llegó a resultados satisfactorios donde los modelos distinguían y clasificaban correctamente casos de test desafiantes. Esto es significativo, por ejemplo en uno u otro contexto, una frase con el mismo verbo podría tener significados distintos, y esto podría tipificar o no un delito. Por ejemplo no es lo mismo la frase "Te voy a matar" que "Me quiero matar" ya que la primera se encuadraria dentro del delito de amenaza (149bis del Código Penal) y la segunda frase, si bien utiliza el mismo verbo y en el mismo tiempo verbal no tipifica delito. Los modelos fueron entrenados teniendo en cuenta estas consideraciones, que si bien a una persona le pueden parecer triviales a la hora de identificar violencia, computacionalmente son complejas de distinguir.

La performance de los modelos fue de un promedio de 10000 bytes por segundo. Una persona puede leer en promedio alrededor de 170 a 300 palabras por minuto [17], mientras que estos modelos llegan a procesar alrededor de 72.000 palabras por minuto. Esto es un aumento considerable de eficiencia. En un caso judicial donde se debe realizar una pericia sobre lo que puede ser un millón de mensajes, el trabajo de

lectura tomaría mínimamente 55 horas, casi 7 días laborales completos, considerando un operario que lee sin pausa a 300 palabras por minuto y no contando el tiempo de etiquetado y organización de la información. El sistema permitiria que este trabajo fuera realizado en menos de 15 minutos. Si bien es necesario un posterior chequeo de los resultados, las herramientas de filtrado y visualización de datos harían de esto un trabajo de análisis más sencillo, en el que el esfuerzo puede ser volcado a la interpretación.

Durante el desarrollo del sitio web, se logró incorporar y llevar a cabo la propuesta visual realizada durante la etapa de los *mockups* con una alta fidelidad. Se logró llevar a cabo la integración con los modelos utilizando las herramientas para el funcionamiento asíncrono que supusieron un desafío adicional al previsto originalmente.

A pesar de la planificación inicial, la ejecución del proyecto reveló desafíos imprevistos, especialmente en la fase de creación de modelos de clasificación de violencia. En ocasión, fue necesario ajustar la planificación en respuesta a cambios en las circunstancias, como modificaciones en las políticas de plataformas externas, falta de datos para los entrenamientos y otras eventualidades.

En cuanto al trabajo futuro, se considera la expansión del sistema mediante la incorporación de nuevos modelos de procesamiento de lenguaje natural y la integración de tecnologías adicionales, como OCR desde imágenes o la transcripción de audio a texto. Se sugiere la incorporación del manejo de nuevos formatos de archivos de texto y la incorporación de ‘traductores de formato’ que sean utilizados sobre extracciones de dispositivos, especialmente para la incorporación del formato utilizado por la herramienta UFED.

A nivel académico, el proyecto permitió poner en práctica gran parte de los conocimientos aprendidos durante el transcurso de la carrera y desarrollar competencias esenciales, no solo en lo que respecta a la informática sino también en la gestión de proyectos. En la planificación inicial se definieron las tareas y los tiempos estimados. Además, se hizo uso del diagrama FODA y se realizó un análisis de riesgo. Para el análisis se utilizaron los conceptos aprendidos para el relevamiento de requerimientos y el diseño del sistema. En el diseño se utilizaron ampliamente las técnicas de diseño de casos de uso y de diagrama de entidades. Durante el desarrollo se usaron los principios, las buenas prácticas y las técnicas estudiadas en los años anteriores.

El proyecto permitió cumplir a nivel personal el logro de desarrollar un producto completo. Los diversos desafíos enfrentados durante las etapas del proyecto se resolvieron exitosamente aplicando los mismos principios que se usaron durante el transcurso de nuestra carrera universitaria: investigación, trabajo en equipo y consulta permanente con los referentes. El desarrollo de este proyecto, es considerado como un logro y un proceso de aprendizaje de gran valor, tanto en aspectos técnicos como no técnicos que se van a utilizar en nuestra futura carrera profesional.

Como conclusión final, se resalta que este trabajo brinda una solución tecnológica que logra agilizar los procesos judiciales del estado, permitiendo minimizar el error y el agotamiento humano y llegar a resultados de manera rápida y efectiva. Logrando así un producto de alto impacto social, en una temática tan crucial como lo es la violencia de género.

Bibliografía

[1] François Chollet. (2021). Deep Learning with Python, Second Edition. Shelter Island, Ny Manning Publications.

[2] spaCy. (2016). spaCy 101: Everything you need to know · spaCy Usage Documentation. SpaCy 101: Everything You Need to Know. https://spacy.io/usage/spacy-101

[3] IBM. (n.d.). ¿Qué son las redes neuronales? ibm.com https://www.ibm.com/es-es/topics/neural-networks

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. ArXiv. https://arxiv.org/abs/1706.03762

[5] F-score. (2023, December 20). Wikipedia. https://en.wikipedia.org/wiki/F-score

[6] Lena Voita. (2023, December 20) Convolutional Models for Text. Github.io. https://lena-voita.github.io/nlp_course/models/convolutional.html

[7] Dirección Nacional de Promoción y Fortalecimiento para el Acceso a la Justicia del Ministerio de Justicia y Derechos Humanos de la Nación y Programa Víctimas contra las Violencias. (2022). Guía de Información (2da ed.). Ediciones SAIJ.

[8] Argentina. (2009). Ley 26.485 Protección Integral a las Mujeres: Ley para Prevenir, Sancionar y Erradicar la Violencia. Boletín Nacional.

[9] Django documentation, Django documentation, Docs.djangoproject.com. https://docs.djangoproject.com/

[10] OpenAI. (2023). Introducing Whisper. Openai.com. https://openai.com/research/whisper

[11] tesseract-ocr. (2019, October 20). tesseract-ocr/tesseract. GitHub. https://github.com/tesseract-ocr/tesseract

[12] Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. Proceedings of the IEEE, 78(10), 1550–1560. https://doi.org/10.1109/5.58337

[13] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533–536. https://doi.org/10.1038/323533a0

[14] Hopcroft, J. E., Rajeev Motwani, & Ullman, J. D. (2020). Introduction to automata theory, languages, and computation. Pearson Education ; Dorling Kindersley.

[15] Maarten Van Steen, & Tanenbaum, A. S. (2017). Distributed systems. Chapter 2. Published By Maarten Van Steen.

[16] Tiwari, S. (2023, February 17). Activation functions in Neural Networks - GeeksforGeeks. GeeksforGeeks. https://www.geeksforgeeks.org/activation-functions-neural-networks/

[17] Claudia Gabriela D'Angelo (2013). VELOCIDAD DE LECTURA Y ACCESO A LA EDUCACIÓN SUPERIOR. X Jornadas de Sociología. Facultad de Ciencias Sociales, Universidad de Buenos Aires, Buenos Aires.

[18] AHORA QUE SI NOS VEN - 206 FEMICIDIOS EN 2023. (n.d.). Recuperado el 24 de febrero de 2024. Ahoraquesinosven.com.ar. https://ahoraquesinosven.com.ar/reports/206-femicidios-en-2023

[19] Barria, J. (2021, 17 de marzo). A casi seis años del primer Ni Una Menos: El día que cambió la historia. Recuperado el 24 de febrero de 2024, de https://diariocronica.com.ar/709662-a-casi-seis-anos-del-primer-ni-una-menos-el-dia-que-cambio-la-historia.html

[20] Sabina Bercovich y Szulmajster, Ivana Feldfeber, Mailén García, Yasmín Belén Quiroga. (Marzo 2021) Datos con perspectiva de género y justicia abierta: la experiencia del Juzgado 10.

[21] Argentina. (2000) Ley 25.326 Protección de los datos personales.

[22] Cellebrite UFED - Cellebrite. (n.d.). Cellebrite.com. https://cellebrite.com/es/cellebrite-ufed-es/

Apéndices

Glosario

Inteligencia artificial: Campo de la ciencia relacionado con la creación de computadoras y máquinas que pueden razonar, aprender y actuar de una manera que normalmente requeriría inteligencia humana o que involucre datos cuya escala exceda lo que los humanos pueden analizar.

API: Conjunto de definiciones y protocolos que se usa para diseñar e integrar el software de las aplicaciones.

Bot: Aplicación de software automatizada que realiza tareas repetitivas en una red.

Celery: Gestor de tareas distribuido y asíncrono desarrollado en Python. Es una herramienta especializada para aplicaciones de alta disponibilidad y con alta carga.

CSS: De las siglas en inglés *Cascading Style Sheets* (Hojas de Estilo en Cascada), es un lenguaje declarativo que controla el aspecto de las páginas web en el navegador.

Dataset: Conjunto organizado de datos que se utiliza para realizar análisis o alimentar modelos de aprendizaje automático.

Django: Framework de aplicaciones web gratuito y de código abierto (open source) escrito en Python.

Framework: es una herramienta de desarrollo que, por lo general, se define como una aplicación o conjunto de módulos que permiten el desarrollo ágil de aplicaciones mediante la aportación de librerías y/o funcionalidades ya creadas. Django es un ejemplo de esto.

HTML: Lenguaje de marcado de hipertexto o *HyperText Markup Language* por sus siglas en inglés. Es un lenguaje descriptivo que especifica la estructura de las páginas web.

Javascript: Lenguaje de programación que se usa con mayor frecuencia para scripts dinámicos de lado del cliente en páginas web, pero también

se usa a menudo en el lado del servidor — usando un entorno de ejecución como Node.js.

JQuery: Biblioteca de JavaScript minificada de código abierto creada para simplificar las operaciones de JavaScript.

Kanban: Sistema de control de inventario que se basa en la utilización de tarjetas o señales visuales para indicar el estado de los diferentes elementos que conforman el proceso. Estas tarjetas se utilizan para controlar la producción, el movimiento de materiales y la asignación de tareas.

Metodologías ágiles: Conjunto de técnicas aplicadas en ciclos de trabajo cortos, con el objetivo de que el proceso de entrega de un proyecto sea más eficiente. Kanban es un ejemplo de una metodología ágil, pero también lo es

NLP: Procesamiento de lenguaje natural, es una rama dentro del *machine learning* que brinda a las computadoras la capacidad de analizar, manipular y corregir el lenguaje humano.

Python: Lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning.

RabbitMQ: Agente de mensajería de código abierto, distribuido y escalable que sirve de intermediario para una comunicación entre productores y consumidores.

Redis: Base de datos en memoria de código abierto, basado en el almacenamiento de tabla de hashes.

Overfitting: Fenómeno que ocurre cuando un modelo de aprendizaje automático, como una red neuronal, se ajusta demasiado bien a los datos de entrenamiento y no generaliza bien a datos nuevos. Esto significa que el modelo ha “memorizado” los datos de entrenamiento y no es capaz de generalizar para hacer predicciones precisas en datos desconocidos.

Scrapping: También conocido como *web scraping*, es una técnica que consiste en extrapolar información de sitios web de manera automática y masiva. Esta técnica se utiliza para recopilar miles o incluso millones de datos a través de la extracción de información de las páginas web.

Scrum: Metodología ágil que consiste en abordar cualquier proyecto dividiéndolo en sprints o partes más pequeñas. Dentro de este entorno de trabajo hay que seguir una serie de fases para abordar cada tarea, y participan unos roles específicos que garantizan el cumplimiento de esta filosofía de trabajo.

SpaCy: Librería de NLP de código abierto escrita en Python versátil y potente diseñada para facilitar tareas de procesamiento avanzado del lenguaje natural.

Tesseract: Motor de reconocimiento óptico de caracteres para varios sistemas operativos, es software libre, liberado bajo la licencia Apache.

Whisper: Sistema de reconocimiento automático de voz (ASR) entrenado en 680,000 horas de datos supervisados multilingües y multitarea recopilados de la web.

Anexo: Entrenamientos de modelos de reconocimiento de entidades nombradas

Entrenamiento 1

Dataset

Se creó un dataset propio a partir de un dataset de noticias ya existente, un dataset de frases comunes en español creado por Google, y Tweets argentinos. Los tweets fueron extraídos utilizando la API de Twitter, y posteriormente sometidos a un trabajo de limpieza eliminando: menciones, links, hashtags y tweets realizados por bots o en inglés.

Resultados Entrenamiento 1.1

Se realizó un entrenamiento utilizando 2000 frases etiquetadas excluyendo los tweets. Corriendo con un batch de 1000, 1800 frases de entrenamiento y 200 frases de test este modelo logró, en 5 loops de entrenamiento, un **f-score de 54**.

Resultados Entrenamiento 1.2

Para el siguiente entrenamiento se utilizaron 2500 frases, incluyendo los tweets. Estas frases se distinguen por el uso de la jerga, la ortografía incorrecta y la informalidad en la redacción. Corriendo con un batch de 1000, 1800 frases de entrenamiento y 200 frases de test este modelo corriendo 5 loops de entrenamiento logró un **f-score de 28**.

Testing y observaciones

Se observaron dificultades para distinguir dígitos para fechas, medidas, y dinero. Muchas frases eran muy similares entre sí y eso puede causar *overfitting* en el modelo.
En cuanto a las frases más informales usadas para el entrenamiento, se observó que el resultado del f-score baja significativamente debido a que el NER no puede valerse del atributo ‘shape’ para la distinción de las entidades. El atributo ‘shape’ distingue el uso de mayúsculas, minúsculas y puntuación en un texto.
Se distinguió la necesidad de unificar el criterio de etiquetado en algunas categorías de las frases de entrenamiento.

Entrenamiento 2

Dataset

Una parte reducida del dataset fue reetiquetado con un criterio unificado y más riguroso.

Resultados Entrenamiento 2.1

Se entrenó un modelo con 500 frases, combinando 200 frases comunes con 300 tweets, etiquetadas siguiendo este nuevo criterio. Con un *batch* de 150 se que logró un **fscore de 29**.

Testing y observaciones 2.1

El objetivo de este modelo fue la prueba del criterio unificado de etiquetado. Por el número reducido de frases de entrenamiento no se esperaban buenos resultados. El modelo fue probado con un conjunto representativo de frases conentidades de todas las categorías, y algunas con ortografía incorrecta o jerga.
En la frase de prueba: “**Juan Martinez** nació en **Hawaii** el **10 de Marzo de 1961**. Durante la **década del 80**, empezó a aparecer en varias películas. En **1981**, viajó a **Argentina** y fundó dos empresas. **Un mes** después, en **diciembre**, chocó con su auto en la **playa**. Llamó a su amigo **Luis** para decirle que perdió la memoria. Cuando **Luis** le preguntó cómo recordaba su nombre, **Juan** se dio cuenta de que estaba bien. Le pagó **2000 pesos** por ayudarlo." *(entidades buscadas resaltadas en negrita).*
El modelo identificó:

Juan Martinez	PERSONA
Hawaii	MISC
10 de Marzo	HORA
1961	FECHA

Durante	PERSONA
80, empezó a aparecer en varias películas. En 1981	FECHA
Argentina	PERSONA
diciembre	LUGAR
playa	LUGAR
Luis	PERSONA
Juan	MISC
2000 pesos	DINERO

Resultados Entrenamiento 2.2

Para el mismo dataset, se realizó un nuevo entrenamiento con el uso de vectores. El *pipeline preentrenado* de spacy utilizado fue el de "es_core_news_lg". Esto quiere decir que el componente NER reentrenado puede utilizar la información de componentes anteriores en el *pipeline* entrenados previamente. La velocidad de entrenamiento bajó notoriamente, pero el resultado fue mucho mejor para el mismo dataset. **El f-score fue de 52**.

Testing y observaciones 2.2

Para la misma frase de prueba hay resultados significativamente mejores. Sin embargo, un gran problema es que el vector aumenta el tamaño del modelo. En el entrenamiento 2.2 sin vectores, el tamaño del modelo es de 4,7 MB. En este el tamaño se incrementó a 646,7 MB.

Juan Martinez	PERSONA
Hawaii	LUGAR
10 de Marzo de 1961	FECHA
80	TIEMPO
Argentina	MISC
dos empresas	MEDIDA
mes	TIEMPO
playa	LUGAR
Luis	PERSONA
Luis	PERSONA
Juan	PERSONA
2000 pesos	DINERO

Entrenamiento 3

Dataset

Se agregaron 450 frases nuevas de tipos poco representadas generadas con ChatGPT para crear variaciones de frases existentes. El dataset contiene alrededor de 650 frases.

Resultados Entrenamiento 3.1

Con el pipeline de entrenamiento vacía y un batch size de 200 se llegó a un **f-score de 74.4**.

Testing y observaciones 3.1

Hay una mejora con respecto al entrenamiento anterior sin vectores, y los resultados son parecidos a los del entrenamiento 2.2 con el dataset “es_core_news_lg”.

Juan Martinez	PERSONA
Hawaii	LUGAR
10 de Marzo	FECHA
1981	LUGAR
Argentina	LUGAR
Luis	PERSONA
Luis	PERSONA
recordaba	FECHA
Juan	PERSONA
2000 pesos	DINERO

Si bien no tiene un número significativo de errores, no reconoció todas las entidades presentes. Al tener un mayor número de frases de entrenamiento, el modelo es menos propenso al *overfitting*, y puede tener mejores resultados con frases nuevas que el anterior.

Resultados Entrenamiento 3.2

Se entrenó al modelo con el dataset anterior, modificando ciertos atributos en el archivo de configuración del entrenamiento (principalmente incluyendo más atributos del token en el modelo) y agregando un *pipeline* preentrenado es_core_news_md. El entrenamiento se volvió mucho más lento, pero el **f-score aumentó considerablemente llegando a 79**.

Testing y observaciones 3.2

El modelo tiene mejores resultados comparados al entrenamiento anterior.

Juan Martinez	PERSONA
Hawaii	MISC
10 de Marzo	FECHA
Argentina	LUGAR
dos empresas	MEDIDA
diciembre	FECHA
playa	LUGAR
Luis	PERSONA
Luis	PERSONA
Juan	PERSONA
2000 pesos	DINERO

El conjunto de prueba el modelo identificó la mayoría de las entidades en los ejemplos de prueba simples con buena ortografía. Tuvo algunos errores en aquellos con mala ortografía.

Entrenamiento 4

Dataset

Se agregaron alrededor de 300 tweets para mejorar la detección en frases más informales.

Resultados Entrenamiento 4.1

Con el pipeline de entrenamiento vacía y un batch size de 200 se llegó a un **f-score de 56.5**.

Testing y observaciones 4.1

Juan Martinez	PERSONA
10 de Marzo	FECHA
Argentina	LUGAR
diciembre	LUGAR
playa	LUGAR
Juan	PERSONA
2000 pesos	DINERO

El modelo no detecta varias de las entidades presentes y comete algunos errores. La inclusión de los tweets confunde al modelo, haciendo que haya menos peso al atributo ‘Shape’ que normalmente juega un rol clave en la detección de entidades.

Resultados Entrenamiento 4.2

Se entrenó con el mismo dataset pero modificando atributos clave. Se incluyó en el modelo más característico de cada uno de los tokens y se

agregó un *pipeline* preentrenado es_core_news_md. El entrenamiento fue más lento pero el **fscore aumentó a 69**.

Testing y observaciones 4.2

Juan Martinez	PERSONA
Hawaii	LUGAR
10 de Marzo de 1961	FECHA
Argentina	LUGAR
dos empresas	MEDIDA
playa	LUGAR
Luis	PERSONA
Luis	PERSONA
Juan	PERSONA
2000 pesos	DINERO

Aunque el f-score es menor que en iteraciones previas, la performance del modelo es mejor en datos reales. Como el dataset tiene frases más diversas evita el *overfitting*.

Resultados Entrenamiento 4.3

Se agregaron al dataset anterior algunas noticias etiquetadas y se llegó a un **fscore de 61**.

Testing y observaciones 4.3

Hay un ligero empeoramiento con respecto al entrenamiento 4.2. “Hawaii” ya no es reconocido como LUGAR y la fecha no se reconoce en su totalidad.

Juan Martinez	PERSONA
Hawaii	MISC
10 de Marzo	FECHA
80,	MEDIDA
Argentina	LUGAR
dos empresas	MEDIDA
playa	LUGAR
Luis	PERSONA
Luis	PERSONA
Juan	PERSONA
2000 pesos	DINERO

Resultados Entrenamiento 4.3 con transformers

Se utilizó el último dataset con un modelo con *transformers*. Se llegó a un **fscore de 77**.

Testing y observaciones 4.3 con transformers

Hay una mejora considerable con respecto al entrenamiento 4.3 tradicional. “Hawaii” se reconoce como LUGAR correctamente. Además, se reconoce una nueva medida de tiempo (“un mes”). Como desventaja, aumenta significativamente el tamaño del modelo, a 440,9 MB (el modelo anterior pesaba 29,3 MB) y el tiempo de procesamiento, pasando de 0.5 segundos (modelo convencional) a 7 segundos (transformer) para un archivo de 200 frases.

Juan Martinez	PERSONA
Hawaii	LUGAR
10 de Marzo	FECHA
Argentina	LUGAR
dos empresas	MEDIDA
Un mes	TIEMPO
diciembre	FECHA
playa	LUGAR
Luis	PERSONA
Luis	PERSONA
Juan	PERSONA
2000 pesos	DINERO

El modelo no detecta varias de las entidades presentes. Los nuevos los tweets confunden al modelo y reduce el peso del atributo ‘Shape’ que normalmente es clave en la detección de entidades. El crecimiento del dataset y los cambios en la configuración aumentaron el tiempo para entrenar el modelo. Se observó SpaCy siempre usaba un sólo *thread* del procesador y una cantidad mínima de RAM por lo que había un desperdicio de recursos. En cambio el uso de los *transformers* permite usar casi todos los *cores* al mismo tiempo y aumentar el uso de la RAM a varios GB. Utilizando esa solución el **f-score llegó a 79**.

Comparación entre modelos

Se utilizó un dataset de prueba común con frases representativas de las distintas categorías.

	f-score	**% Etiquetas correctas encontradas**	**% Etiquetas faltantes**	**% Etiquetas erróneas**
Entrenamiento 1.1	54	28,95%	71,05%	1,52%
Entrenamiento 1.2	27	42,11%	57,89%	1,52%

Entrenamiento 2.1	29	52,63%	47,37%	2,28%
Entrenamiento 2.2	52	**79,49%**	20,51%	**12,9%**
Entrenamiento 3.1	74	68,42%	31,58%	0,76%
Entrenamiento 3.2	**79**	**71,05%**	**28,95%**	**0,38%**
Entrenamiento 4.1	56	52,63%	47,37%	0,38%
Entrenamiento 4.2	69	60,53%	39,47%	0,38%
Entrenamiento 4.3	61	57,89%	42,11%	0,76%
Entrenamiento 4.3.t	**79**	**84,62%**	**15,38%**	**2,56%**

Los mejores resultados fueron los entrenamientos 2.2 y 4.3 con *transformers*, aunque son considerablemente más pesados que los demás. Si bien en los entrenamientos sin el uso de *transformers* el entrenamiento 3.2 logró muy buenos resultados, los modelos entrenados en el entrenamiento 4 contienen más datos de entrenamiento con frases informales y con errores, por lo que puede ser más adecuado para textos que de esta naturaleza informal.

Anexo: Entrenamientos de modelos de Clasificación de Violencia Binaria

Entrenamiento 1

Dataset Entrenamiento 1

Para la construcción de este dataset, se utilizaron cuatro fuentes de datos:

1. Noticias: Un conjunto de oraciones provenientes de noticias obtenidas de internet. Las temáticas de estas oraciones son diversas y de naturaleza no violenta. Son oraciones que tienen gramática correcta, formal y de naturaleza descriptiva.
2. Frases de Google: Un conjunto de frases provenientes del *“Crowdsourced high-quality Argentinian Spanish speech data set”*, un dataset creado por Google. Este dataset contiene 2278 frases conjugadas en “español argentino” que consiste en transcripciones

realizadas por hablantes de Buenos Aires. Son de naturaleza no violenta, con gramática correcta, más informal y corta.

3. Tweets: Frases extraídas mediante la API de Twitter. En este caso, estos tweets se buscaron utilizando palabras clave e insultos violentos en la región de Argentina. Se eliminaron menciones, links, *hashtags* y tweets realizados por bots o en inglés. Tienen una jerga muy marcada, son muy informales y contener ortografía incorrecta.
4. Frases violentas del juzgado Nº10: extraídas del dataset del juzgado número 10 de Buenos Aires. Incluyen en su mayoría amenazas de violencia tipo física. Son frases que muchas veces tienen mala ortografía, son cortas e informales.

Para este entrenamiento se etiquetaron 4599 frases, 1823 violentas y 2776 no violentas.

Resultados Entrenamiento 1

Parámetro	Valor
Vector	es_core_news_sm
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Score Violento	90
Score No Violento	93
Score Total	91

Pruebas y observaciones Entrenamiento 1

El score obtenido por el modelo fue alto, y eso indicaría que el modelo tiene un alto grado de confiabilidad. Sin embargo, las pruebas sobre un primer conjunto reducido de frases de ejemplo muestran que el score no es tan acertado. Hubo dos problemas principales:

- La tendencia del modelo a solo clasificar como “Violento” frases que contienen el mensaje “te voy a matar” porque es una frase muy común en las frases de amenazas provistas por eldel juzgado Nº 10. Del dataset inicial:
  - 19 frases se repiten con exactamente el texto “te voy a matar”
  - 228 frases contienen “te voy a matar” lo que representa un 12,5% del total de 1823 frases violentas.
- La desproporción de los tipos de frases: Hay muchas más frases no violentas.

En el siguiente set de prueba, las frases resaltadas en verde se consideran correctamente clasificadas mientras que las de rojo se consideran incorrectamente clasificadas. Mientras más fuerte es el color, más significativo es el error.

Score Violento	Score No Violento	Frase
0.41	0.59	Hola
0.36	0.64	Hola, me llamo Pablo
0.33	0.67	Soy de mar del plata
0.73	0.27	Te voy a matar
0.57	0.43	Te voy a prender fuego
0.55	0.45	Te voy a pegar un tiro
0.70	0.30	Sos una mierda
0.49	0.51	Sos una genia
0.42	0.58	Vas a aparecer muerta en una zanja
0.39	0.61	Si no me das al nene voy con un fierro a tu casa
0.34	0.66	No servis para nada
0.49	0.51	Sos una estupida
0.44	0.56	Voy a matarte lentamente
0.30	0.07	Vas a morir esta noche
0.43	0.57	Me la vas a pagar cuando salga de la carcel
0.34	0.66	Me las vas a pagar una por una

El alto score obtenido es el resultado de *overfitting*. Al existir una desproporción de frases similares entre sí en el dataset de entrenamiento, el modelo aprendió demasiado fielmente las más comunes y perdió precisión para clasificar otras frases.

Entrenamiento 2

Dataset Entrenamiento 2

Se compuso a partir de las frases incluidas en el entrenamiento 1, pero removiendo:

- Las repeticiones de la frase exacta “te voy a matar” (dejando las variaciones).
- Algunas frases no violentas para emparejar la proporción de frases de cada categoría.

En total se etiquetaron 3525 frases: 1521 frases violentas y 2004 frases no violentas.

Resultados Entrenamiento 2

Parámetro	Valor
Vectores	es_core_news_sm

Pipeline	textcat
Batch size	800
Proporción datos para entrenamiento	70%
Score Violento	89
Score No Violento	95
Score Total	92

Pruebas y observaciones Entrenamiento 2

Si bien el modelo es correcto para la mayoría de las frases simples, clasifica frases con la estructura de frases violentas típicas como violentas aunque no lo sean. Además, falla al identificar frases claramente violentas pero sin la estructura “sos ...” o “te voy a...”.

Score Violento	Score No Violento	Frase
0.30	0.70	Hola
0.24	0.76	Hola, me llamo Pablo
0.26	0.74	Soy de mar del plata
0.84	0.16	Te voy a matar
0.75	0.25	Te voy a prender fuego
0.69	0.31	Te voy a pegar un tiro
0.78	0.22	Sos una mierda
0.43	0.57	Sos una genia
0.39	0.61	Vas a aparecer muerta en una zanja
0.49	0.51	Si no me das al nene voy con un fierro a tu casa
0.34	0.66	No servis para nada
0.46	0.54	Sos una estupida
0.39	0.61	Voy a matarte lentamente
0.32	0.68	Vas a morir esta noche
0.54	0.46	Me la vas a pagar cuando salga de la carcel
0.48	0.52	Me las vas a pagar una por una

Resultados Entrenamiento 2 con transformers

Parámetro	Valor
Vector	Ninguno
Pipeline	transformer, textcat
Batch size	500
Proporción datos para entrenamiento	70%
Score Violento	91
Score No Violento	96
Score Total	94

Pruebas y observaciones Entrenamiento 2 con transformers

Score Violento	Score No Violento	Frase
0	1	Hola
0	1	Hola, me llamo Pablo
0	1	Soy de mar del plata
1	0	Te voy a matar
1	0	Te voy a prender fuego
1	0	Te voy a pegar un tiro
1	0	Sos una mierda
1	0	Sos una genia
1	0	Vas a aparecer muerta en una zanja
1	0	Si no me das al nene voy con un fierro a tu casa
0.52	0.48	No servis para nada
1	0	Sos una estupida
1	0	Voy a matarte lentamente
1	0	Vas a morir esta noche
1	0	Me la vas a pagar cuando salga de la carcel
1	0	Me las vas a pagar una por una

El score obtenido es alto y superior al obtenido con el entrenamiento tradicional, algo común de los modelos entrenados con transformers. Varias frases ahora son correctamente detectadas como “Violentas”. También hay una diferencia en los score comparado al modelo anterior, prefiriendo en este caso valores ‘0’ o ‘1’ a valores intermedios de acuerdo a las características y ambigüedades de cada frase. El modelo tiende a clasificar un gran porcentaje de las frases como violentas, llevando a un alto número de falsos positivos.

Entrenamiento 3

Dataset Entrenamiento 3 con transformers

Se utilizó el dataset del entrenamiento 2 y se reetiquetaron varias frases ambiguas que estaban siendo consideradas en la categoría “Violento”. Esto hizo que frases que siguen la estructura “sos …” o “te voy a…”. ya no sean siempre consideradas como violentas.

Resultados Entrenamiento 3 con transformers

Parámetros	Valor
Vectores	Ninguno
Pipeline	transformer, textcat

Batch size	500
Proporción datos para entrenamiento	70%
Score Violento	92
Score No Violento	96
Score Total	94

Pruebas y observaciones Entrenamiento 3 con transformers

Score Violento	Score No Violento	Frase
0	1	Hola
0	1	Hola, me llamo Pablo
0	1	Soy de mar del plata
1	0	Te voy a matar
1	0	Te voy a prender fuego
1	0	Te voy a pegar un tiro
1	0	Sos una mierda
0	1	Sos una genia
1	0	Vas a aparecer muerta en una zanja
1	0	Si no me das al nene voy con un fierro a tu casa
0	1	No servis para nada
1	0	Sos una estupida
1	0	Voy a matarte lentamente
0	1	Vas a morir esta noche
1	0	Me la vas a pagar cuando salga de la carcel
0	1	Me las vas a pagar una por una

La frase “Sos una genia” se clasificó correctamente, pero otras dos (“Vas a morir esta noche” y “Me las vas a pagar una por una”) no se clasificaron como “Violento”. Hay una leve tendencia a clasificar a las frases como “No Violento”.

Entrenamiento 4

Dataset Entrenamiento 4

Se agregaron 700 frases violentas para incrementar la variedad y reducir el overfitting. En total se etiquetaron 4396 frases, 1390 frases violentas y 3006 frases no violentas.

Resultados Entrenamiento 4

Este entrenamiento se hizo con tres diferentes configuraciones en el número de épocas.

Parámetro	Valor

Vectores	es_core_news_md
Pipeline	textcat
Batch size	100
Proporción datos para entrenamiento	70%
Épocas	8
Score Violento	98
Score No Violento	99
Score Total	99

Parámetro	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	8
Score Violento	98
Score No Violento	99
Score Total	99

Parámetro	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	6
Score Violento	97
Score No Violento	98
Score Total	98

Pruebas y observaciones Entrenamiento 4

Entrenamiento 4 con Batch Size=100 y Batch Size=1000 y Épocas=8

Score Violento	Score No Violento	Frase
0.33	0.67	Hola
0.22	0.78	Hola, me llamo Pablo
0.13	0.87	Soy de mar del plata
0.90	0.10	Te voy a matar
0.70	0.30	Te voy a prender fuego
0.74	0.26	Te voy a pegar un tiro
0.73	0.27	Sos una mierda
0.33	0.67	Sos una genia
0.35	0.65	Vas a aparecer muerta en una zanja

0.34	0.66	Si no me das al nene voy con un fierro a tu casa
0.28	0.72	No servis para nada
0.33	0.67	Sos una estupida
0.30	0.70	Voy a matarte lentamente
0.06	0.94	Vas a morir esta noche
0.13	0.87	Me la vas a pagar cuando salga de la carcel
0.07	0.93	Me las vas a pagar una por una

Entrenamiento 4 con Batch Size=1000 y Épocas=6

Score Violento	Score No Violento	Frase
0.35	0.65	Hola
0.25	0.75	Hola, me llamo Pablo
0.14	0.86	Soy de mar del plata
0.87	0.13	Te voy a matar
0.68	0.32	Te voy a prender fuego
0.70	0.30	Te voy a pegar un tiro
0.72	0.28	Sos una mierda
0.34	0.66	Sos una genia
0.34	0.66	Vas a aparecer muerta en una zanja
0.35	0.65	Si no me das al nene voy con un fierro a tu
0.30	0.70	No servis para nada
0.34	0.66	Sos una estupida
0.32	0.68	Voy a matarte lentamente
0.09	0.91	Vas a morir esta noche
0.14	0.86	Me la vas a pagar cuando salga de la carcel
0.09	0.91	Me las vas a pagar una por una

Pese al score de 99 (lo que indica overfitting), el modelo es mucho menos exacto en la clasificación de frases violentas. “Te voy a matar” se clasifica correctamente, pero “Voy a matarte lentamente” se clasifica como “No Violento”. Las frases no violentas son demasiado homogéneas y fácilmente diferenciables de las violentas. Para solucinar esto se evaluó:

- Frenar el entrenamiento mediante uso de épocas.
- Agregar frases que sean más ambiguas a los datos de entrenamiento.
- Aumentar el número de ejemplos para diversificar las clasificaciones.

Resultados con transformers Entrenamiento 4 con transformers

Se hizo un entrenamiento con transformers utilizando el mismo dataset.

Parámetro	Valor
Vectores	Ninguno
Pipeline	transformer, textcat
Batch size	500
Proporción datos para entrenamiento	70%
Score Violento	99
Score No Violento	99
Score Total	99

Pruebas y observaciones Entrenamiento 4 con transformers

Score Violento	Score No Violento	Frase
0	1	Hola
0	1	Hola, me llamo Pablo
0	1	Soy de mar del plata
0.98	0.02	Te voy a matar
0.79	0.21	Te voy a prender fuego
0.98	0.02	Te voy a pegar un tiro
0.98	0.02	Sos una mierda
0.99	0.01	Sos una genia
0	1	Vas a aparecer muerta en una zanja
0.98	0.02	Si no me das al nene voy con un fierro a tu
0	1	No servis para nada
0.98	0.02	Sos una estupida
0.98	0.02	Voy a matarte lentamente
0	1	Vas a morir esta noche
0.98	0.02	Me la vas a pagar cuando salga de la carcel
0	1	Me las vas a pagar una por una

No hay una mejora de la detección de las frases violentas con respecto al modelo anterior.

Entrenamiento 5

Dataset Entrenamiento 5

Se agregó un conjunto de comentarios de la página Reddit, extraídos de un dataset de frases del año 2013 y 2014. Se realizaron tres combinaciones con las frases del entrenamiento 4:

- Un dataset compuesto solo por las frases de Reddit.
- Frases de Reddit y frases no violentas del entrenamiento 4 para balancear.
- Frases de Reddit y todas las frases del entrenamiento 4.

En tota 6610 frases:, 3680 frases violentas y 2930 frases no violentas.

Resultados Entrenamiento 5

Sólo Frases de Reddit

Parámetros	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	50
Score Violento	43
Score No Violento	84
Score Total	63

Frases de Reddit + Frases del entrenamiento 5 parcial

Parámetros	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	50
Score Violento	82
Score No Violento	80
Score Total	81

Frases de Reddit + Frases del entrenamiento 5 completo

Parámetros	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	50
Score Violento	87
Score No Violento	90
Score Total	88

Pruebas y observaciones Frases de Reddit + Frases del entrenamiento 5 completo

Score Violento	Score No Violento	Frase

0.35	0.65	Hola
0.31	0.69	Hola, me llamo Pablo
0.09	0.91	Soy de mar del plata
0.9	0.1	Te voy a matar
0.7	0.3	Te voy a prender fuego
0.9	0.1	Te voy a pegar un tiro
0.77	0.23	Sos una mierda
0.5	0.5	Sos una genia
0.46	0.54	Vas a aparecer muerta en una zanja
0.4	0.6	Si no me das al nene voy con un fierro a tu casa
0.18	0.82	No servis para nada
0.52	0.48	Sos una estupida
0.5	0.5	Voy a matarte lentamente
0.1	0.9	Vas a morir esta noche
0.21	0.79	Me la vas a pagar cuando salga de la carcel
0.12	0.88	Me las vas a pagar una por una

El score de frases violentas mejoró pero no suficiente clasificarlo como “Violento”.

Entrenamiento 6

Datasets y normalización

Para identificar los problemas de los últimos entrenamientos se utilizó un conjunto de prueba especial:

Frases No Violentas Simples	Frases Violentas Simples	Frases No Violentas Ambiguas	Frases Violentas Ambiguas
Hola, soy Pablo.	Te voy a matar.	Hola, te voy a buscar.	Hola, si te metés conmigo te voy a buscar.
Vivo en Mar del Plata.	Te voy a violar	Se donde vivís.	Guarda conmigo, porque yo se dónde vivís.
Sos una genia	Sos una puta.	Ese es mi perro.	Sos una perra.
Te voy a ayudar.	Te voy a pegar un tiro.	Te voy a dar una mano con el parcial.	Te voy a dejar en la calle.
Voy a sacar a pasear al perro.	Voy a pegarte un balazo en la cabeza.	Me voy para casa.	Voy a ir a tu casa con un fierro.
Ayer perdí plata en el casino.	Hija de puta me hiciste perder plata.	Solo necesito un poco más de plata	Solo me haces perder plata
¿Tenes fuego?	Te voy a prender fuego.	Mandale un fueguito.	Te prendo fuego.
Andá a trabajar	Andá a trabajar negra de mierda	Que noche negra para el fútbol	Que negra boluda que sos.
¿Cogemos?	Vamos a coger quieras o no.	No quiero coger hoy.	Que incogible que sos.

¿Rompiste el celular?	Rompes todo y yo tengo que pagarlo	Te rompí el plato sin querer.	Si yo quiero te puedo romper la casa.
Tomé mucho frio en la garganta anoche.	Mira este cuchillo lo voy a meter en tu garganta.	Ese cuchillo viene con el kit.	Ese cuchillo te lo puedo meter bien adentro tuyo.

Los resultados de correr este dataset de prueba en el modelo 5.4 fueron los siguientes:

Clasificación	Score Violento	Score No Violento	Frase
No Violento	0.32	0.68	Hola, soy Pablo.
Violento	0.91	0.09	Te voy a matar.
Violento	0.77	0.23	Hola, te voy a buscar.
Violento	0.87	0.13	Hola, si te metés conmigo te voy a buscar.
No Violento	0.18	0.82	Vivo en Mar del Plata.
Violento	0.8	0.2	Te voy a violar
No Violento	0.17	0.83	Se donde vivís.
No Violento	0.25	0.75	Guarda conmigo, porque yo se dónde vivís.
Violento	0.5	0.5	Sos una genia
Violento	0.86	0.14	Sos una puta.
No Violento	0.39	0.61	Ese es mi perro.
Violento	0.75	0.25	Sos una perra.
Violento	0.53	0.47	Te voy a ayudar.
Violento	0.91	0.09	Te voy a pegar un tiro.
No Violento	0.47	0.53	Te voy a dar una mano con el parcial.
Violento	0.64	0.36	Te voy a dejar en la calle.
No Violento	0.43	0.57	Voy a sacar a pasear al perro.
No Violento	0.21	0.79	Voy a pegarte un balazo en la cabeza.
No Violento	0.11	0.89	Me voy para casa.
No Violento	0.17	0.83	Voy a ir a tu casa con un fierro.
No Violento	0.11	0.89	Ayer perdí plata en el casino.
Violento	0.53	0.47	Hija de puta me hiciste perder plata.
No Violento	0.07	0.93	Solo necesito un poco más de plata

No Violento	0.13	0.87	Solo me haces perder plata
No Violento	0.21	0.79	¿Tenes fuego?
Violento	0.74	0.26	Te voy a prender fuego.
No Violento	0.4	0.6	Mandale un fueguito.
No Violento	0.46	0.54	Te prendo fuego.
No Violento	0.18	0.82	Andá a trabajar
Violento	0.8	0.2	Andá a trabajar negra de mierda
No Violento	0.28	0.72	Que noche negra para el fútbol
Violento	0.63	0.37	Que negra boluda que sos.
No Violento	0.22	0.78	¿Cogemos?
No Violento	0.37	0.63	Vamos a coger quieras o no.
No Violento	0.21	0.79	No quiero coger hoy.
No Violento	0.38	0.62	Que incogible que sos.
No Violento	0.17	0.83	¿Rompiste el celular?
No Violento	0.19	0.81	Rompes todo y yo tengo que pagarlo
No Violento	0.2	0.8	Te rompí el plato sin querer.
No Violento	0.17	0.83	Si yo quiero te puedo romper la casa.
No Violento	0.32	0.68	Tomé mucho frio en la garganta anoche.
Violento	0.75	0.25	Mira este cuchillo lo voy a meter en tu garganta
No Violento	0.34	0.66	Ese cuchillo viene con el kit.
Violento	0.69	0.31	Ese cuchillo te lo puedo meter bien adentro tuyo.

Los problemas principales que se identifican a raíz de estas pruebas son:

1. Las frases del tipo “Te voy a” o “Sos una” tienden a ser clasificadas como violentas. Por ejemplo “Te voy a ayudar” es clasificada como “Violento”. Esto es porque muchos ejemplos de frases violentas comienzan así.

2. Algunas frases violentas comunes como “te voy a prender fuego”, no fueron reconocidas cuando se las cambia ligeramente “te prendo fuego”.
3. Algunas frases no violentas que contienen una palabra común a la de una frase violenta tienen un score de violencia alto. “Ese es mi perro” tiene un score de 0.39 porque “perro” es muy parecida a “perra”, usada en la frase violenta “Sos una perra”.
4. Frases violentas que no incluyen insultos típicos, como los del tipo de violencia económica (“Solo me haces perder plata”) o psicológica (“Guarda conmigo, porque sé donde vivís”) no se clasifican cómo violentas por falta de ejemplos en el dataset.

Por tal motivo,se modificó el dataset del entrenamiento 5.4 en las siguientes etapas:

Etapa 1: Limpieza de signos y normalización de frases.

- Se hizo una limpieza de las frases.
- Se removieron algunos signos de puntuación.
- Se eliminaron todos los espacios excesivos y se dejó uno solo.
- Se eliminaron todos los signos de pregunta excesivos y se dejó uno solo.
- Se eliminaron todas las tildes de acentuación de las letras.

Se volvió a entrenar el modelo y los puntos 1 y 3 mejoraron. En cambio, se los puntos 4 y 2 empeoraron. Es decir, esta normalización mejoró los casos de frases no violentas clasificadas como violentas, pero aumentó las instancias de frases violentas no reconocidas como tal.

Etapa 2: Modificación e incorporación de frases

- Se agregaron variaciones de las frases para evitar el *overfitting*. Por ejemplo se agregaron más ejemplos no violentos que empiezan con “te voy a”.
- Se reemplazaron expresiones de otros países por expresiones argentinas.
- Se descompusieron o se eliminaron frases largas y complejas que contenían componentes violentos y no violentos para reducir la ambigüedad. Se hizo un corte para remover todas las frases con una longitud mayor a 200 caracteres.
- Se removieron frases duplicadas y se mejoró el script balancear el número de frases violentas y no violentas para los datasets de test y train.

Al reentrenar se obtuvo los puntos 1, 2 y 3 mejoraron. El punto 4 no mejoró. Es decir, no detecta frases de violencia económica o psicológica sin un insulto explícito.

Etapa 3: Incorporación de frases de violencia económica y psicológica

Se agregaron alrededor de 300 frases de violencia, económica, psicológica y frases parecidas no violentas para evitar el *overfitting*. El dataset final se compuso de 4072 frases. Se notó una mejora del punto 4 y los errores totales pasaron de 12 a 3.

Etapa 4: Manejo de mayúsculas

Un problema adicional fue el efecto de la presencia de mayúsculas en los resultados.

Clasificación	**Score Violento**	**Score No Violento**	**Frase**
Violento	0.79	0.21	sos una hija de puta
No violento	0.46	0.54	SOS UNA HIJA DE PUTA
No violento	0.49	0.51	sOS unA hIJa dE Puta

Para resolver este problema se consideraron dos posibilidades:

- Agregar mayúsculas de manera aleatoria en todas las frases para intentar que la presencia de estas no se considere como un factor que aumenta la posibilidad de violencia.
- Clasificar las frases pasándolas completamente a minúscula.

Durante esta etapa se agregaron 100 nuevas frases de violencia económica, psicológica y física. Además, se agregaron frases no violentas provenientes del sitio de noticias Telám y frases sintácticamente parecidas a frases violentas.
El entrenamiento utilizando un script para agregar mayúsculas de forma aleatoria logró un score de 69. Mientras que el entrenamiento donde todas las frases estaban en minúscula dio un score de 83. Además, las pruebas demostraron un considerable peor desempeño de los casos ambiguos para el modelo entrenado con las mayúsculas aleatorias, casi la mitad de las frases fueron mal clasificadas:

Clasificación	Score Violento	Score No Violento	Frase
No Violento	0.34	0.66	Hola, soy Pablo.
Violento	0.84	0.16	Te voy a matar.
No Violento	0.61	0.39	Hola, te voy a buscar.
No Violento	0.68	0.32	Hola, si te metés conmigo te voy a

			buscar.
No Violento	0.27	0.73	Vivo en Mar del Plata.
Violento	0.71	0.29	Te voy a violar
No Violento	0.51	0.49	Se donde vivís.
No Violento	0.4	0.6	Guarda conmigo, porque yo se dónde vivís.
No Violento	0.47	0.53	Sos una genia
No Violento	0.6	0.4	Sos una puta.
No Violento	0.58	0.42	Ese es mi perro.
No Violento	0.5	0.5	Sos una perra.
Violento	0.71	0.29	Te voy a ayudar.
Violento	0.7	0.3	Te voy a pegar un tiro.
No Violento	0.49	0.51	Te voy a dar una mano con el parcial.
Violento	0.71	0.29	Te voy a dejar en la calle.
Violento	0.75	0.25	Voy a sacar a pasear al perro.
No Violento	0.38	0.62	Voy a pegarte un balazo en la cabeza.
No Violento	0.51	0.49	Me voy para casa.
Violento	0.72	0.28	Voy a ir a tu casa con un fierro.
No Violento	0.34	0.66	Ayer perdí plata en el casino.
No Violento	0.62	0.38	Hija de puta me hiciste perder plata.
No Violento	0.41	0.59	Solo necesito un poco más de plata
No Violento	0.56	0.44	Solo me haces perder plata
No Violento	0.36	0.64	¿Tenes fuego?
Violento	0.71	0.29	Te voy a prender fuego.
No Violento	0.47	0.53	Mandale un fueguito.
No Violento	0.59	0.41	Te prendo fuego.
No Violento	0.5	0.5	Andá a trabajar
Violento	0.77	0.23	Andá a trabajar negra de mierda
No Violento	0.43	0.57	Que noche negra para el fútbol
No Violento	0.64	0.36	Que negra boluda que sos.
No Violento	0.28	0.72	¿Cogemos?
No Violento	0.63	0.37	Vamos a coger quieras o no.
No Violento	0.52	0.48	No quiero coger hoy.
No Violento	0.54	0.46	Que incogible que sos.
No Violento	0.18	0.82	¿Rompiste el celular?
No Violento	0.36	0.64	Rompes todo y yo tengo que pagarlo
No Violento	0.47	0.53	Te rompí el plato sin querer.
Violento	0.78	0.22	Si yo quiero te puedo romper la casa.
No Violento	0.44	0.56	Tomé mucho frio en la garganta anoche.
Violento	0.83	0.17	Mira este cuchillo lo voy a meter en tu garganta
No Violento	0.34	0.66	Ese cuchillo viene con el kit.
Violento	0.73	0.27	Ese cuchillo te lo puedo meter bien

			adentro tuyo.

Solo con minúsculas solo hay un error, en una frase considerada ambigua:

Clasificación	Score Violento	Score No Violento	Frase
No Violento	0.44	0.56	hola, soy pablo.
Violento	0.88	0.12	te voy a matar.
No Violento	0.68	0.32	hola, te voy a buscar.
Violento	0.91	0.09	hola, si te metés conmigo te voy a buscar.
No Violento	0.37	0.63	vivo en mar del plata.
Violento	0.83	0.17	te voy a violar
No Violento	0.34	0.66	se donde vivís.
Violento	0.84	0.16	guarda conmigo, porque yo se dónde vivís.
No Violento	0.47	0.53	sos una genia
Violento	0.83	0.17	sos una puta.
No Violento	0.4	0.6	ese es mi perro.
Violento	0.77	0.23	sos una perra.
No Violento	0.55	0.45	te voy a ayudar.
Violento	0.89	0.11	te voy a pegar un tiro.
No Violento	0.44	0.56	te voy a dar una mano con el parcial.
Violento	0.8	0.2	te voy a dejar en la calle.
No Violento	0.58	0.42	voy a sacar a pasear al perro.
Violento	0.79	0.21	voy a pegarte un balazo en la cabeza.
No Violento	0.52	0.48	me voy para casa.
Violento	0.79	0.21	voy a ir a tu casa con un fierro.
No Violento	0.45	0.55	ayer perdí plata en el casino.
Violento	0.85	0.15	hija de puta me hiciste perder plata.
No Violento	0.4	0.6	solo necesito un poco más de plata
Violento	0.8	0.2	solo me haces perder plata
No Violento	0.45	0.55	¿tenes fuego?
Violento	0.87	0.13	te voy a prender fuego.
No Violento	0.44	0.56	mandale un fueguito.
Violento	0.73	0.27	te prendo fuego.
No Violento	0.32	0.68	andá a trabajar
Violento	0.81	0.19	andá a trabajar negra de mierda
No Violento	0.48	0.52	que noche negra para el fútbol
Violento	0.83	0.17	que negra boluda que sos.
No Violento	0.3	0.7	¿cogemos?
Violento	0.8	0.2	vamos a coger quieras o no.
No Violento	0.37	0.63	no quiero coger hoy.
Violento	0.74	0.26	que incogible que sos.
No Violento	0.18	0.82	¿rompiste el celular?

No Violento	0.39	0.61	rompes todo y yo tengo que pagarlo
No Violento	0.7	0.3	te rompí el plato sin querer.
Violento	0.72	0.28	si yo quiero te puedo romper la casa.
No Violento	0.2	0.8	tomé mucho frio en la garganta anoche.
Violento	0.86	0.14	mira este cuchillo lo voy a meter en tu garganta
No Violento	0.39	0.61	ese cuchillo viene con el kit.
Violento	0.93	0.07	ese cuchillo te lo puedo meter bien adentro tuyo.

Los mejores resultados se encontraron cuando las frases del dataset estaban todas en minúsculas y las frases de prueba también se pasaban a minúscula.

Resultados Entrenamiento 6

Entrenamiento 6 con normalización incluyendo mayúsculas

Parámetros	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	50
Score Violento	69
Score No Violento	70
Score Total	70

Entrenamiento 6 con normalización sin mayúsculas

Parámetros	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	70%
Épocas	50
Score Violento	83
Score No Violento	84
Score Total	84

Pruebas y observaciones Entrenamiento 6 sin mayúsculas

Score	Score No	Frase

Violento	Violento	
0.42	0.58	hola
0.22	0.78	hola, me llamo pablo
0.52	0.48	soy de mar del plata
0.88	0.12	te voy a matar
0.87	0.13	te voy a prender fuego
0.89	0.11	te voy a pegar un tiro
0.82	0.18	sos una mierda
0.47	0.53	sos una genia
0.81	0.19	vas a aparecer muerta en una zanja
0.89	0.11	si no me das al nene voy con un fierro a tu casa
0.51	0.49	no servis para nada
0.72	0.28	sos una estupida
0.68	0.32	voy a matarte lentamente
0.40	0.60	vas a morir esta noche
0.87	0.13	me la vas a pagar cuando salga de la carcel
0.68	0.32	me las vas a pagar una por una

El score se redujo a 84, lo que indica una reducción del *overfitting*. Hay una mejora general en la clasificación de las frases tanto violentas como no violentas. Hay una mayor uniformidad de los scores. Si bien hay mejoras en la clasificación de las frases violentas, también hay frases no violentas con un score más ambiguo, por ejemplo “sos una genia”.

Entrenamiento 7

Dataset Entrenamiento 7

Se incluyeron todas las frases usadas durante el entrenamiento anterior. Se agregó un nuevo conjunto de frases violentas del juzgado Nº10 y 100 frases no violentas similares a tipos de frases violentas, como frases que comienzan con “te voy…” para reducir el overfitting.
Se etiquetaron 4396 frases, 2203 frases violentas y 2193 frases no violentas.

Resultados Entrenamiento 7

Parámetro	Valor
Vectores	es_core_news_md
Pipeline	textcat
Batch size	1000
Proporción datos para entrenamiento	60%
Épocas	50
Score Violento	79

Score No Violento	79
Score Total	79

Pruebas y observaciones entrenamiento 7

Score Violento	Score No Violento	Frase
0.45	0.55	hola
0.11	0.89	hola, me llamo pablo
0.53	0.47	soy de mar del plata
0.91	0.09	te voy a matar
0.94	0.06	te voy a prender fuego
0.95	0.05	te voy a pegar un tiro
0.84	0.16	sos una mierda
0.55	0.45	sos una genia
0.79	0.21	vas a aparecer muerta en una zanja
0.77	0.23	si no me das al nene voy con un fierro a tu casa
0.8	0.2	no servis para nada
0.74	0.26	sos una estupida
0.9	0.1	voy a matarte lentamente
0.56	0.44	vas a morir esta noche
0.95	0.05	me la vas a pagar cuando salga de la carcel
0.83	0.17	me las vas a pagar una por una

El score obtenido fue de 79. Este score es menor al del anterior entrenamiento. Las pruebas muestran resultados correctos para la mayoría de las frases. Los errores se dan en las frases no violentas, mientras que la mayoría de las frases violentas se clasifican correctamente.

Comparación entre modelos

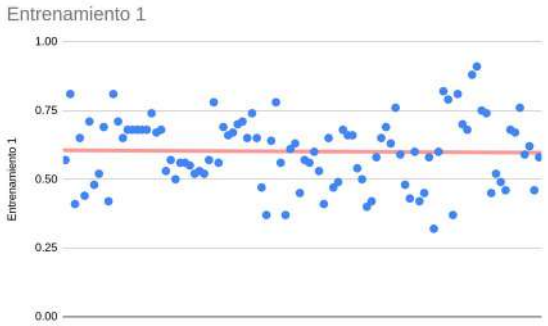
Para evaluar de manera uniforme y completa el rendimiento de los modelos se armó un dataset de prueba extendido, compuesto por 100 frases violentas y 100 frases no violentas. Para que una frase sea considerada violenta, el score mínimo debe ser de 70.
Se agrega un resumen del número de fallos para cada sección y el número total de fallos. A menos puntaje, mayor confiabilidad del modelo. Los modelos resaltados en amarillo fueron entrenados con transformers.

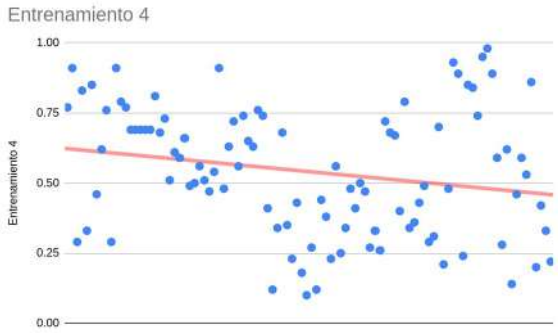
Errores	Entrenamiento 1	Entrenamiento 2	Entrenamiento 3	Entrenamiento 4	Entrenamiento 5	Entrenamiento 6	Entrenamiento 7
No Violento	2	16	3	0	2	4	4
Violento	80	10	32	73	68	32	18

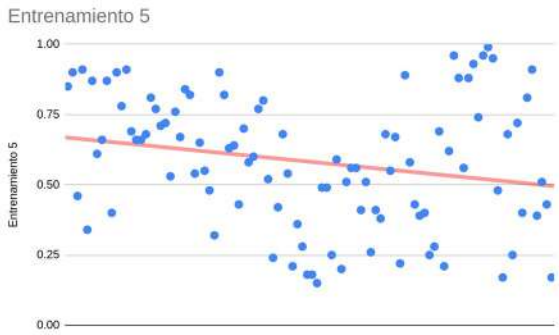
Totales	82	26	35	73	70	36	22

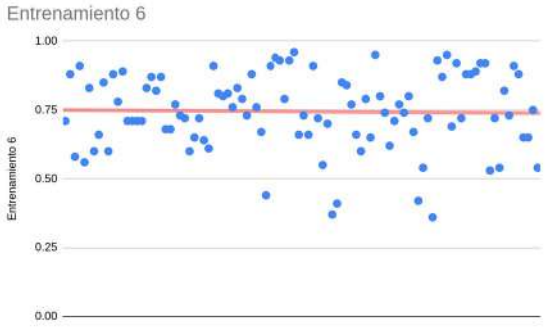
Se adjuntan gráficos de los entrenamientos sin transformers. Cada punto representa el puntaje obtenido por la frase. El escenario ideal es con los puntos de los gráficos de la izquierda arriba y los de la derecha abajo. La línea roja muestra una tendencia general.

Violentas

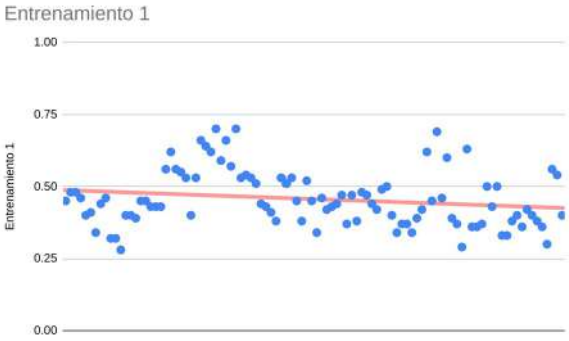


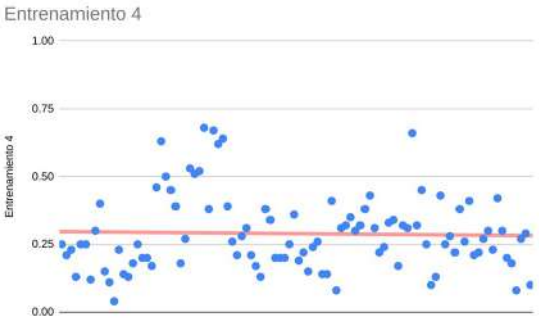


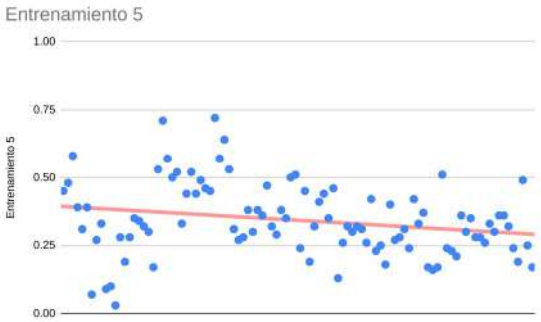


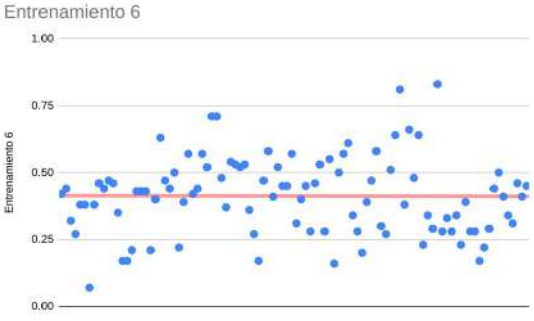


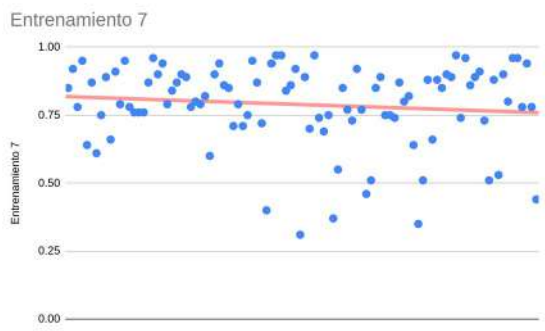
No violentas

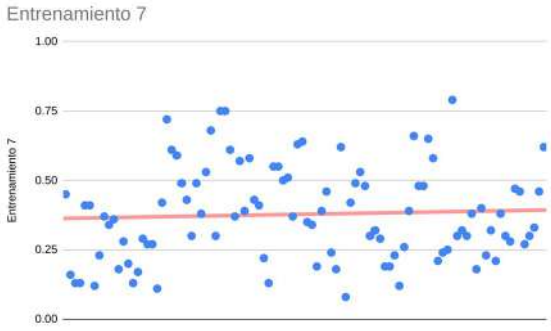












Se concluye que los modelos más confiables son el del entrenamiento 2 con transformers y el del entrenamiento 7.

Medición de performance

Para el procesamiento de 200 frases (5.718 bytes):
Entrenamiento 1 - 0.25 segundos : **22872 bytes/segundo**
Entrenamiento 2 - 6.94 segundos : **824 bytes/segundo**
Entrenamiento 3 - 6.93 segundos : **825 bytes/segundo**
Entrenamiento 4 - 0.55 segundos : **10396 bytes/segundo**
Entrenamiento 5 - 0.54 segundos : **10589 bytes/segundo**
Entrenamiento 6 - 0.54 segundos : **10589 bytes/segundo**
Entrenamiento 7 - 0.53 segundos : **10788 bytes/segundo**

Anexo: Entrenamientos de modelos de Clasificación de Violencia Multicategoría

Entrenamiento 1

Dataset Entrenamiento 1

Se creó un dataset con las cuatro fuentes usadas para el modelo binario (noticias, tweets, dataset de Google y frases del juzgado Nº10). Se incluyó una categoría 'Indefinido' para frases sin una categoría específica. Se tomó un subconjunto del dataset del Entrenamiento 1 de clasificación de violencia binaria. Se etiquetaron 1117 frases: 29 frases de categoría Sexual, 426 frases de categoría Física, 16 frases de categoría Económica, 244 frases de categoría Psicológica, 144 frases de categoría Simbólica y 258 frases de categoría Indefindo.

Resultados Entrenamiento 1

Parámetro	Valor
Vectores	es_core_news_sm
Pipeline	textcat
Batch size	800

Proporción datos para entrenamiento	70%
Score Sexual	**24**
Score Física	**80**
Score Económica	**0**
Score Psicológica	**44**
Score Simbólica	**22**
Score Indefinido	**18**
Score Total	**31**

Resultados Entrenamiento 1 con transformers

Parámetro	Valor
Vectores	Ninguno
Pipeline	transformer, textcat
Batch size	500
Proporción datos para entrenamiento	70%
Score Sexual	**52**
Score Física	**87**
Score Económica	**22**
Score Psicológica	**37**
Score Simbólica	**40**
Score Indefinido	**34**
Score Total	**46**

Pruebas y observaciones

Excepto para la categoría ‘Física’, el modelo de reconocimiento multicategoría tiene problemas para detectar correctamente el tipo de violencia. Esto se atribuye principalmente a la falta de ejemplos, especialmente en las categorías ‘Económica’ y ‘Sexual’. En la categoría ‘Física’, el resultado es adecuado. Incluso el modelo logra reconocer frases que no contienen un insulto o una mención al cuerpo de la mujer.
El entrenamiento con transformers aumenta el score del modelo y de cada categoría. Pero no produce un cambio significativo en los resultados de las pruebas. Solo mejoran levemente los resultados de las categorías ‘Sexual’ y ‘Psicológica’.

Entrenamiento 2

Dataset

Se seleccionó un número más limitado de datos del dataset del entrenamiento 1 con datos más representativos y sin frases indefinidas. Se agregaron frases simples, y versiones modificadas de estas frases más parecidas a las expresiones habituales en Argentina. En total, 901 frases: 30 frases de categoría Sexual, 247 frases de categoría Física, 164 frases

de categoría Económica, 294 frases de categoría Psicológica y 166 frases de categoría Simbólica.

Resultados Entrenamiento 2

Parámetro	Valor
Vectores	es_core_news_sm
Pipeline	textcat
Batch size	200
Proporción datos para entrenamiento	70%
Sexual	**33**
Física	**86**
Económica	**76**
Psicológica	**73**
Simbólica	**68**
Score Total	**56**

Resultados Entrenamiento 2 con transformers

Parámetro	Valor
Vectores	Ninguno
Pipeline	transformer, textcat
Batch size	200
Proporción datos para entrenamiento	70%
Sexual	**89**
Física	**94**
Económica	**94**
Psicológica	**83**
Simbólica	**80**
Score Total	**88**

Pruebas y observaciones Entrenamiento 2

Hubo mejores resultados. El incremento de ejemplos para las categorías “Económica” y el decremento en las categorías “Psicológica” y “Simbólica” tuvieron un efecto positivo en los scores. Hubo muchos menos errores en las categorías “Psicológica” y “Económica”. Los entrenamientos con transformers mejoraron los resultados comparados al tradicional.

Entrenamiento 3

Dataset Entrenamiento 3

Para balancear el dataset, tener un número similar de frases en cada categoría y así lograr que el modelo no esté sesgado a favor de alguna, se realizaron búsquedas manuales de datos con las opciones de búsqueda

avanzada de twitter y transcripciones de conversaciones reales en artículos, blogs y noticias. Se armó un dataset que contiene 1397 frases: 199 frases de categoría Sexual, 305 frases de categoría Física, 408 frases de categoría Económica, 255 frases de categoría Psicológica y 230 frases de categoría Simbólica.

Resultados Entrenamiento 3

Parámetro	Valor
Vectores	es_core_news_sm
Pipeline	textcat
Batch size	200
Proporción datos para entrenamiento	70%
Sexual	68
Física	84
Económica	81
Psicológica	70
Simbólica	68
Score Total	74

Resultados con transformers Entrenamiento 3

Parámetro	Valor
Vectores	Ninguno
Pipeline	textcat
Batch size	200
Proporción datos para entrenamiento	70%
Sexual	**84**
Física	**89**
Económica	**93**
Psicológica	**81**
Simbólica	**77**
Score Total	**85**

Pruebas y observaciones

Se observó una mejora para con respecto al entrenamiento tradicional anterior, en especial en las categoría “Sexual”. Para el entrenamiento con transformers, no se observaron mejoras considerables y el score continuó en el rango de 85-90.