Escuela Técnica Superior de Ingeniería

Universidad de Huelva

Detección Automática de Misoginia en MEMES mediante Técnicas de Deep Learning

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Autor: Pablo Cordón Hidalgo

Tutores: Victoria Pachón Álvarez

Jacinto Mata Vázquez

Huelva, Enero de 2023

Trabajo Fin de Grado

Grado en Ingeniería Informática

Título: Detección Automática de Misoginia en MEMES mediante Técnicas de *Deep Learning*

Autor: Pablo Cordón Hidalgo

Tutores: Victoria Pachón Álvarez

Jacinto Mata Vázquez

Departamento de Tecnologías de la Información

Escuela Técnica Superior de Ingeniería

Universidad de Huelva

Agradecimientos

A todas las mujeres de mi vida, que me han hecho ser la persona que soy hoy.

Sin vosotras nunca habría llegado hasta aquí.

Resumen

La misoginia es el odio o el desprecio hacia las mujeres. Es una forma de sexismo utilizada para mantener a las mujeres en un estatus social inferior al de los hombres, manteniendo así los roles sociales del patriarcado.

Una forma popular de comunicarse a través de las redes sociales son los MEMEs. Un meme es una imagen representada a través de un contenido pictórico con un texto superpuesto que se escribe a posteriori, con el objetivo fundamental de ser divertido y/o irónico.

Aunque la mayoría de los memes son creados con el objetivo de hacer bromas divertidas, poco después se normaliza su uso para ser utilizados como forma de difundir el odio contra las mujeres, dando lugar a mensajes sexistas y agresivos en entornos de Internet, que permiten expresar libremente el sexismo sin temor a represalias.

En este trabajo se estudian e implementan técnicas para la detección automática de misoginia en memes, o lo que es lo mismo, textos e imágenes.

Para ello, se han estudiado y comparado varias arquitecturas basadas en algoritmos de *Machine Learning* y *Deep Learning*, junto con otras aproximaciones de clasificación binaria mediante Transformers. Todo ello combinado con un algoritmo de detección de imágenes no seguras para ofrecer mejores resultados.

Para probar nuestros resultados, hemos participado en el 16º Taller Internacional de Evaluación Semántica, también conocido como SemEval-2022, en concreto en la tarea 5: *"Multimedia Automatic Misogyny Identification"* (MAMI). En esta memoria se describe toda la información sobre la tarea y las soluciones presentadas, analizando los resultados obtenidos.

Una vez finalizada la competición, se han propuesto nuevas técnicas para mejorar y perfeccionar los resultados obtenidos anteriormente en esta.

Como resultado del trabajo realizado en esta competición, se ha escrito un artículo científico publicado en las actas de congreso del 16º Taller Internacional de Evaluación Semántica.

Keywords: Aprendizaje Automático, Aprendizaje Profundo, Misoginia, python, meme, BERT Transformers

Abstract

Misogyny is hatred or contempt for women. It is a form of sexism used to keep women at a lower social status than men, thus maintaining the societal roles of patriarchy.

A popular way of communicating via social media platforms are MEMEs. A meme is an image portrayed through pictorial content with overlaid text which is written a posteriori, with the fundamental objective of being entertaining and/or ironic.

Even though most of memes are created with the goal of making amusing jokes, shortly after their standardization individuals began to use them to disseminate hate against women, leading to sexist and aggressive messages in internet environments that allow people to freely express sexism without the fear of retaliation.

In this paper we study and implement techniques for the automatic detection of misogyny in memes, or what is the same, text and images.

To do so, several architectures based on *Machine Learning* and *Deep Learning* algorithms have been studied and compared, along with various other approaches to binary classification using Transformers, all that combined with an unsafety image detection algorithm to provide better results.

To prove our results, we participated in the 16th International Workshop on Semantic Evaluation, also known as SemEval-2022, in particular in task 5: Multimedia Automatic Misogyny Identification (MAMI). All the information about the task and the solutions presented are described below, analyzing the results obtained.

After the competition finished, new techniques have been proposed to upgrade and refine the results obtained previously.

As a result of the work completed in the competition, a scientific article has been written in in the conference proceedings of this 16th International Workshop on Semantic Evaluation.

Keywords: Machine Learning, Deep Learning, Misogyny, python, meme, BERT Transformers

Índice

Índice de Tablas

Índice de figuras

Capítulo 1

Introducción

El Procesamiento del Lenguaje Natural (PLN) es un conjunto de técnicas computacionales para analizar y representar textos naturales en uno o varios niveles de análisis lingüístico con el fin de lograr un procesamiento del lenguaje similar al humano en una serie de tareas o aplicaciones. [1]

Esta serie de tareas son, por ejemplo, la traducción automática de un idioma a otro, el resumen o auto-rellenado de textos, y la detección de sentimientos y emociones, entre otras.[2]

En cuanto a los niveles lingüísticos el PLN abarca varios de ellos, como el sintáctico, fonético o el morfológico, sin embargo, el desarrollo de este trabajo se centrará en el nivel semántico, en el que se aprende el significado de las palabras y su contexto para conocer el sentido que se le da a una oración.

1.1 Motivación

La elección de este Trabajo de Fin de Grado surge por diversos motivos. El primero, el interés y curiosidad que suscita el campo del PLN, y sobre todo el *Machine Learning*, ya que como estudiante casi graduado sentía aún que sabía muy poco sobre las tendencias actuales de este. Además, que el tema a tratar sea la detección de misoginia, un tema tan en auge hoy en día, hace que el trabajo no solo sea gratificante en lo profesional y académico sino también en lo personal.

Las redes sociales tienen cada vez un mayor alcance e impacto en la sociedad, y son la vía principal de comunicación entre personas. Al poder además crear perfiles anónimos, estas se han convertido en el terreno perfecto para mandar mensajes de odio o desprecio hacia cualquier tipo de colectivo. En el caso de este trabajo, hacia las mujeres. El poder captar y censurar a tiempo estos mensajes (ya sean en formato textual o como un meme (texto e imagen superpuesta)) es esencial para que tanto Internet como el mundo real sean un lugar más seguro.

1.2 Objetivos

El principal objetivo de este trabajo es el estudio y desarrollo de técnicas para la detección automática de misoginia en memes, que poseen un formato común y han sido descargados de las principales páginas web de humor de internet.

Para evaluar el buen funcionamiento de las técnicas desarrolladas, se ha participado en el 16º Taller Internacional de Evaluación Semántica, también conocido como *SemEval-2022*, en concreto en la tarea 5: *"Multimedia Automatic Misogyny Identification"* (MAMI) [3].

SemEval consiste en una serie de seminarios internacionales de investigación en PLN cuya misión es avanzar en el análisis semántico de vanguardia y producir conjuntos de datos anotados de alta calidad sobre algunos de los problemas cada vez más difíciles en la semántica del lenguaje natural.

El trabajo realizado y presentado en el marco de la competición ha concluido con un artículo científico que ha sido aceptado y publicado en las actas de *SemEval-2022* [4].

1.3 Estructura de la memoria

La memoria se organiza de la siguiente manera:

- El **capítulo 1**, Introducción, presenta el tema de este trabajo, la motivación que llevó a su elección y realización, y también los objetivos que se pretenden conseguir y la forma en la que se estructura el documento.
- El **capítulo 2**, Marco Teórico, desarrolla de manera teórica todos los temas que abarca este trabajo, haciendo al lector capaz de comprender lo que se explica en este.
- El **capítulo 3**, Planteamiento, se realiza una descripción del problema, describiendo los datos proporcionados por la competición
- El **capítulo 4**, Experimentación y Resultados, describe la implementación y las soluciones propuestas, así como los resultados de la competición.
- El **capítulo 5**, Propuestas de Mejoras, indica las mejoras realizadas tras la finalización del periodo de evaluación de la competición para incrementar los resultados.
- El **capítulo 6**, Conclusiones y Trabajo Futuro, se expone en qué medida los resultados obtenidos son satisfactorios, así como posibles trabajos futuros que pudieran mejorar más aún los resultados.
- Para finalizar, en el **Anexo** se presenta el articulo científico derivado del trabajo realizado.

Capítulo 2

Marco Teórico

En este capítulo se describen teóricamente los conceptos que el lector necesita conocer para entender cómo hemos solucionado la tarea en la que hemos participado. Esto abarca desde algoritmos clásicos de Aprendizaje Automático (*Machine Learning*) como los árboles de decisión, hasta técnicas de Aprendizaje Profundo (*Deep Learning*) como los Transformers.

2.1 Aprendizaje Automático

El Aprendizaje Automático, o *Machine Learning* (ML) en inglés, es una rama de la Inteligencia Artificial en la que, a través de algoritmos, brinda a los ordenadores la capacidad de identificar patrones en grandes cantidades de datos y realizar predicciones.

Este aprendizaje permite que el ordenador pueda realizar ciertas tareas específicas de manera autónoma, es decir, sin la necesidad de una programación previa por parte de un humano. [5]

Los algoritmos de *Machine Learning* se dividen en tres categorías, siendo las dos primeras las más comunes:

- **Aprendizaje Supervisado:** El Aprendizaje Supervisado cuenta con un aprendizaje previo, en el que se entrena el algoritmo conociendo la salida esperada de un conjunto de datos de entrenamiento. A partir de estos, se genera un modelo capaz de predecir una salida dada una entrada diferente a la del conjunto de entrenamiento. Un ejemplo es un detector de clickbait, en el que a partir de un conjunto de titulares (entrada) y si son o no clickbait (salida), podrá predecir futuros titulares. [6]
- **Aprendizaje no Supervisado:** El Aprendizaje no Supervisado, en cambio, no cuenta con un conocimiento previo. Ante una gran cantidad de datos desordenados, encuentra patrones que permitan clasificarlos de alguna manera. Se usa, por ejemplo, para clasificar plantas y animales dadas sus características. [7]
- **Aprendizaje Reforzado:** En el Aprendizaje por Refuerzo no tenemos salidas, por lo que no es de tipo supervisado, ni intentamos clasificar grupos dadas unas muestras. El modelo aprenderá a resolver la tarea mediante un sistema de recompensas, a partir de las cuales, mediante prueba-error, conseguirá realizar las acciones necesarias para obtener una solución. Este aprendizaje es muy típico en videojuegos. [8]

En nuestro caso hemos usado Aprendizaje Supervisado para entrenar nuestros modelos.

2.1.1 algoritmos clásicos

Tres de los algoritmos más usados clásicamente en tareas de clasificación de textos son las Máquinas de Vector Soporte (SVM), los árboles de decisión y los bosques aleatorios.

SVM

Las máquinas de vectores de soporte (*Support Vector Machine,* SVM) son algoritmos de Aprendizaje Supervisado que se utilizan en muchos problemas de clasificación y regresión, como procesamiento de señales en medicina, procesamiento de lenguaje natural y reconocimiento de formas.

El objetivo del algoritmo SVM es encontrar un hiperplano que separe de la mejor manera dos clases diferentes. Para esto el hiperplano busca maximizar el margen entre las dos capas, indicado por los signos "+" y "-" en la figura 1. El margen se define como el ancho máximo de un área de hiperplano paralelo dentro de la cual no hay puntos de datos. El algoritmo puede encontrar este hiperplano solo en problemas que son linealmente separables. [9]

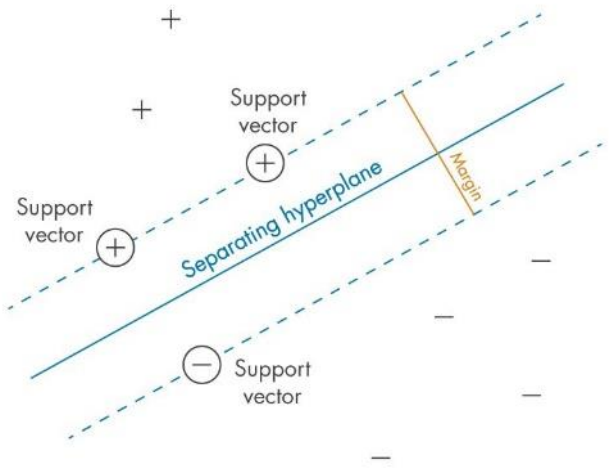

Figura 1: Definición de hiperplano y margen entre clases de la SVM

Árboles de decisión

En el campo del Aprendizaje Automático, un árbol de decisiones (*Decision Tree*) es una estructura de árbol similar a un diagrama de flujo donde los nodos internos representan características (o atributos), las ramas representan reglas de decisión y cada nodo "hoja" representa un resultado. El nodo superior de un árbol de decisiones en el Aprendizaje Automático se denomina nodo raíz.

Cada nodo en el árbol actúa como un caso de prueba de atributos, y cada borde que desciende de ese nodo corresponde a una de las posibles respuestas del caso de prueba. Este proceso es recursivo y se repite para cada sub-árbol derivado del nuevo nodo. [10]

Bosques Aleatorios

Un algoritmo de bosque aleatorio (*Random Forest*) [11] está formado por muchos árboles de decisión. El "bosque" generado por el algoritmo de bosque aleatorio se entrena mediante *bagging*. El *bagging* es un meta-algoritmo de conjunto que mejora la precisión de los algoritmos de aprendizaje automático. [12]

Este algoritmo establece el resultado basándose en las predicciones de los árboles de decisión. Predice tomando el promedio o la media de la salida de varios árboles. Al aumentar el número de árboles se incrementa la precisión del resultado.

2.1.2 Redes Neuronales

Las redes neuronales artificiales son una de las técnicas de clasificación más populares de las últimas décadas. Estas enseñan a las computadoras a procesar datos de una manera que está inspirada en la forma en que lo hace el cerebro humano.

Una red neuronal artificial consta de una serie de unidades de procesamiento llamadas nodos o neuronas (Figura 2), que reciben una señal de entrada y aplican una función, normalmente no lineal, para generar la salida. Dado que las conexiones entre las neuronas tienen pesos, la señal que pasa a través de ellos se multiplica, formando así la salida correcta. [13]

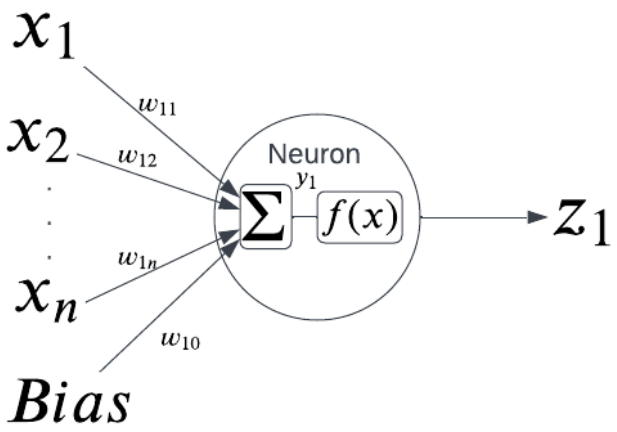

Figura 2: Esquema de la neurona de una red neuronal

Las redes neuronales tradicionales habitualmente están compuestas por una capa de entrada (*Input layer*), una capa interna u oculta (*hidden layer*) y una capa de salida (*output layer*). Sin embargo, cuando una red neuronal está formada por más de tres capas, esta se cataloga como profunda (*Deep Learning*). En la figura 3 podemos observar una representación de ambos modelos.

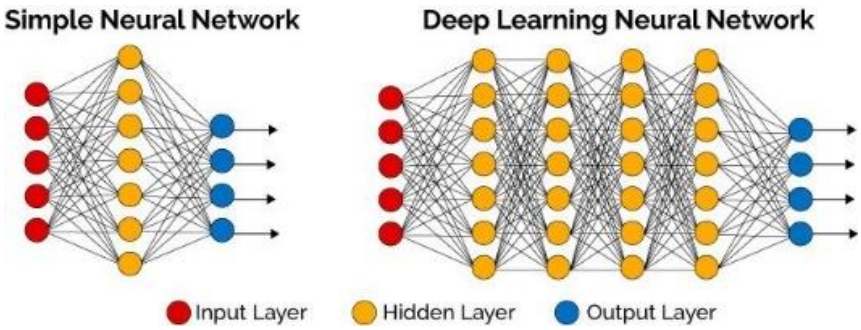

Figura 3: Estructura de red neuronal simple y compleja

Las redes neuronales profundas han demostrado ser mucho mejores que las redes neuronales simples. Esto se debe a que un mayor número de capas ocultas en la red permite un mayor nivel de abstracción a la hora de extraer características de los datos, pudiendo representar estructuras más complejas.

Redes Long-Short Term Memory (LSTM)

En el mundo de las redes neuronales existen muchas arquitecturas y modelos diferentes. Uno de los más importantes son las redes LSTM. Estas son un tipo especial de redes recurrentes. La característica principal de las redes recurrentes es que la información puede persistir introduciendo bucles en el diagrama de la red, por lo que, básicamente, pueden «recordar» estados previos y utilizar esta información para decidir cuál será el siguiente. Esta característica las hace muy adecuadas para manejar series cronológicas. Mientras las redes recurrentes estándar pueden modelar dependencias a corto plazo (es decir, relaciones cercanas en la serie cronológica), las LSTM pueden aprender dependencias largas, por lo que se podría decir que tienen una «memoria» a más largo plazo. [14]

Funciona de forma similar a como nuestro cerebro analiza un texto. Si por ejemplo deseamos comprar un ordenador y leemos alguna valoración hecha por un comprador, para tomar la decisión no nos enfocamos en la totalidad del texto: en lugar de ello nos enfocamos únicamente en las palabras que consideramos relevantes, y desechamos el resto de la información.

Un elemento clave para el funcionamiento de las redes LSTM son las celdas de estado, donde se podrán añadir datos o eliminar aquellos que no queremos que sigan en la memoria. Para añadir o eliminar datos de esta memoria se utilizan tres tipos de

compuertas: *forget gate* (permite eliminar elementos de la memoria), *input gate* (permite añadir nuevos elementos a la memoria) y *output gate* (permite crear el estado oculto actualizado). Podemos observar esta arquitectura en la figura 4

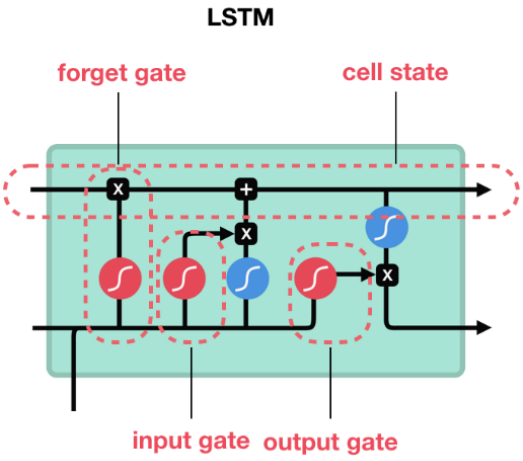

Figura 4: Elementos de una red LSTM

Cada una de estas compuertas está conformada por tres elementos: una red neuronal, una función sigmoidal y un elemento multiplicador. La función sigmoidal puede alcanzar valores entre 0 y 1 y permite anular los valores de entrada (si la salida es 0) o permitir el paso de estos (si la salida es 1), por lo que solo os daros relevantes pasan a través de la cadena de secuencia para obtener una mejor predicción.

Existe una mejora de este modelo llamado *Bidirectional LSTM*, o biLSTM [15]. Este consiste en dos LSTM: una que toma la entrada en dirección hacia delante y otra en dirección hacia atrás (figura 5). Las BiLSTM aumentan efectivamente la cantidad de información disponible para la red, mejorando el contexto disponible para el algoritmo (por ejemplo, saber qué palabras siguen y preceden inmediatamente a una palabra en una frase).

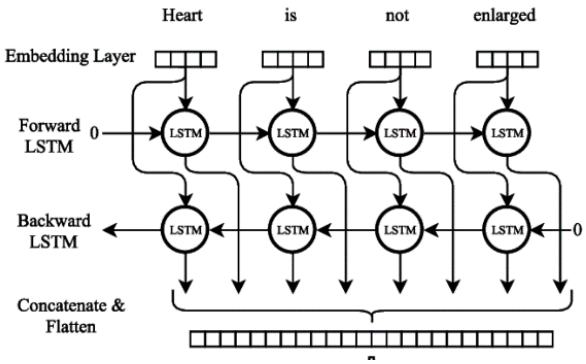

Figura 5: Arquitectura de un modelo biLSTM

2.1.3 Transformers

Hasta 2017, los modelos de *Deep Learning* Recurrentes (como el mencionado anteriormente LSTM) se convirtieron en el estándar para las tareas de PLN, pues eran capaces no sólo de tener en cuenta el significado general de cada palabra, sino también la posición de estas en la frase.

Sin embargo, a finales de este año, el ingeniero de Google Ashish Vaswani presentó en el paper "*Attention is All You Need*" [16] la arquitectura del Transformer, un modelo que tenía como principal innovación la sustitución de las capas recurrentes, como las LSTMs que se venían usando hasta ese momento en PLN, por las denominadas capas de atención.

Estas capas de atención codifican cada palabra de una frase en función del resto de la secuencia, permitiendo así introducir el contexto en la representación matemática del texto, motivo por el cual a los modelos basados en Transformer se les denomina también Embeddings Contextuales (figura 6).

Desde su irrupción en el paradigma del PLN, los modelos basados en la arquitectura de Transformer se han convertido en el nuevo estado del arte en tareas de análisis de texto, siendo así la familia de modelos que mejores resultados han obtenido en sus diferentes aplicaciones: generación de textos, resumen de textos, identificación de entidades, clasificación de documentos, preguntas y respuestas, etc.

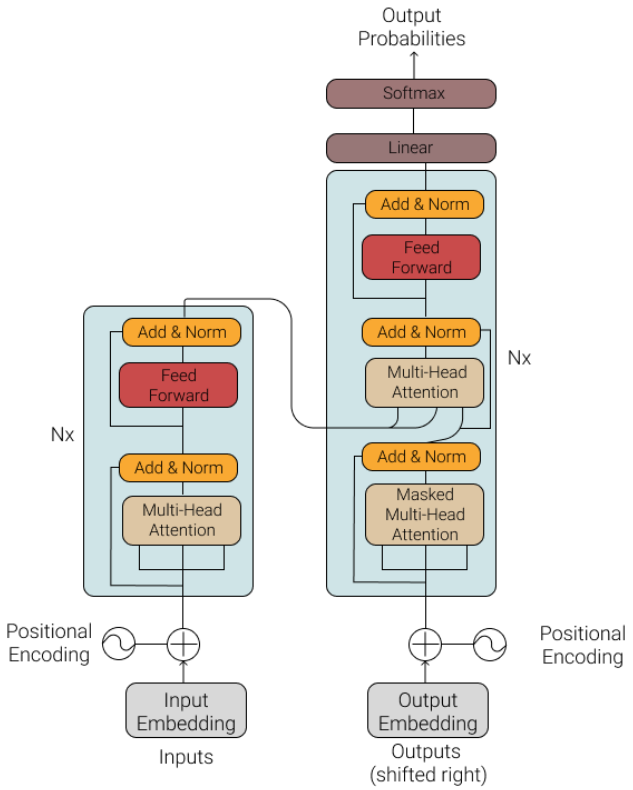

Figura 6: Arquitectura del modelo Transformers

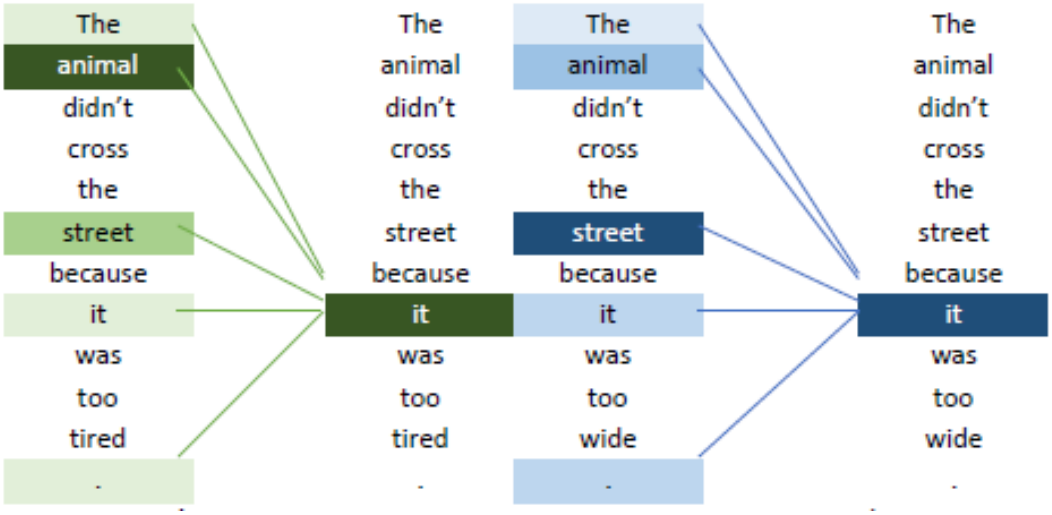

Figura 7: Ejemplo del mecanismo de atención

En la Figura 7 se aprecia dos ejemplos de cómo funcionan los mecanismos de atención en los Transformers sobre el pronombre "*it*", en donde en un ejemplo deduciría que se hace referencia al animal (izquierda) mientras que en el otro deduce que hace referencia a la calle (derecha). Ambas frases solo difieren en una palabra (*wide* y *tired*), pero su sentido cambia totalmente. El algoritmo es capaz de dar un mayor peso (en el ejemplo, un subrayado más oscuro) a la opción correcta en cada caso, analizando el contexto de la oración.

Entre los modelos de Transformers preentrenados más conocidos (y usados en este proyecto) se encuentran BERT y RoBERTa.

BERT (Bidirectional Encoder Representations from Transformers)

BERT [17] está descrito en un artículo publicado por investigadores de *Google AI Language*. A diferencia de los modelos direccionales, que leen la entrada de texto secuencialmente (de izquierda a derecha o de derecha a izquierda), el codificador Transformer lee toda la secuencia de palabras a la vez. Por eso se considera bidireccional, aunque sería más exacto decir que es no-direccional. Esta característica permite al modelo aprender el contexto de una palabra basándose en todo su entorno (izquierda y derecha de la palabra). Con este contexto, usa dos tareas de carácter no supervisado de manera simultánea.

- Modelado del lenguaje enmascarado (MLM): Para entrenar una representación bidireccional profunda, enmascara al azar el 15% de las palabras en una oración con un símbolo o token [MASK] y luego trata de predecirlas basándose en las palabras que rodean a la palabra enmascarada y su contexto. Técnicamente, la predicción de palabras de salida requiere (figura 8):
    - Agregar una capa de clasificación sobre la salida del codificador.

- Multiplicar los vectores de salida por la matriz de incrustación, transformándolos en la dimensión del vocabulario.
- Cálculo de la probabilidad de cada palabra en el vocabulario con *softmax*. La función *softmax* se emplea para "comprimir" un vector k-dimensional z de valores reales en un vector k-dimensional $\sigma(z)$ de valores reales en el rango [0,1]. Por ejemplo, si el vector de entrada fuera [2, -1, 4], la función *softmax* retornaría [0.118, 0.005, 0.875].

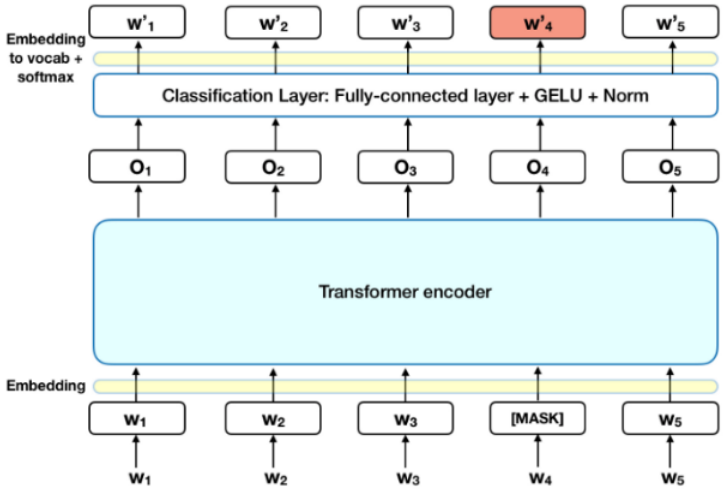

Figura 8: MLM BERT Transformers

- Predicción de la siguiente frase (NSP): En el proceso de entrenamiento de BERT, el modelo recibe pares de frases como entrada y aprende a predecir si la segunda frase del par es la siguiente del documento original. Durante el entrenamiento, el 50% de las entradas son un par en el que la segunda frase es la frase posterior del documento original, mientras que en el otro 50% se elige una frase aleatoria del Corpus como segunda frase. Se supone que la frase aleatoria estará desconectada de la primera frase.

Las entradas del *encoder* de BERT requiere que las frases tengan el siguiente formato:

[CLS] + Frase A + [SEP] + Frase B,

siendo:

- [CLS]: el token de clasificación
- [SEP]: el token de separación entre las 2 frases (aunque la frase B es opcional)

Podemos observar en la figura 9 la forma que toman las frases al introducirlas en el *encoder*:

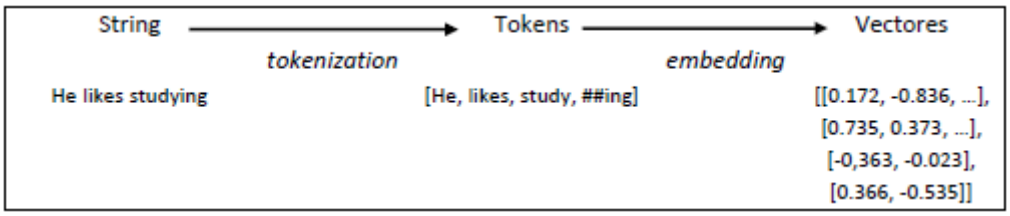

Figura 9: Entrada de las frases en BERT

Por lo tanto, el enconder necesita las siguientes entradas (Véase la Figura 10):

- Vectores de palabras (*embedding*)
- Un indicador que diga si pertenece a la frase A o a la frase B
- Un *embedding* posicional

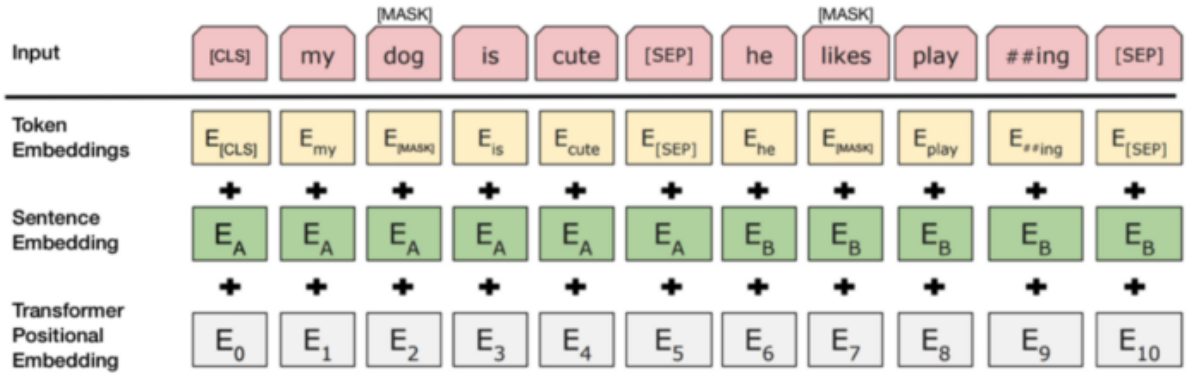

Figura 10: Entradas necesarias para el encoder

Existen distintas variaciones de este modelo, como pueden ser BERT-base-cased (distingue entre mayúsculas y minúsculas), BERT-large-uncased (Mayor Corpus de entrenamiento y solo minúsculas) o BERT-base-multilingual-cased (entrenado para varios idiomas), entre otros.

RoBERTa (A Robustly Optimized BERT Pretraining Approach)

RoBERTa [17] es una readaptación del modelo BERT, con el fin de mejorar sus resultados obtenidos en base a una serie de modificaciones centradas principalmente en el procedimiento de su entrenamiento. Estas modificaciones son las siguientes:

- Entrenamiento con lotes (*batch*) más grandes y secuencias más largas: Originalmente BERT se entrena para 1 millón de pasos con un tamaño de lote (batch) de 256 secuencias. En este trabajo, los autores entrenaron el modelo con 125 pasos de 2 mil secuencias y 31 mil pasos con 8 mil secuencias de tamaño de lote. Esto tiene dos ventajas, los lotes grandes puede mejorar la velocidad de

optimización y el rendimiento de la tarea final cuando la tasa de aprendizaje aumenta adecuadamente.

- Eliminación del objetivo de predicción de la siguiente frase (NSP): Como hemos definido anteriormente, en la predicción de la siguiente frase, el modelo se entrena para predecir si los segmentos de documentos observados proceden del mismo o de distintos documentos mediante una pérdida auxiliar de Predicción de la Siguiente Frase (NSP). Los autores experimentaron con la eliminación/adición pérdidas NSP de diferentes versiones y concluyeron que la eliminación de la pérdida NSP iguala o mejora ligeramente el rendimiento de la tarea posterior.

- Cambio dinámico del patrón de enmascaramiento: La implementación inicial de BERT realizó el enmascaramiento una vez durante el preprocesamiento de datos, lo que da como resultado una única máscara estática. Con el objetivo de evitar el uso de la misma máscara se duplican los datos de entrenamiento 10 veces. De esta forma, cada secuencia queda enmascarada de 10 maneras diferentes durante las 40 epochs de entrenamiento, por lo que cada secuencia de entrenamiento se vio con la misma máscara 4 veces durante el entrenamiento. Esta estrategia se compara con el enmascaramiento dinámico donde cada vez que alimentamos una secuencia al modelo se genera el patrón de enmascaramiento. Esto se vuelve crucial cuando se entrena previamente para más pasos o con conjuntos de datos más grandes.

2.2 Procesamiento del Lenguaje Natural

El Procesamiento del Lenguaje Natural (PLN) es el campo de conocimiento de la Inteligencia Artificial que se ocupa de la investigar la manera de comunicar las máquinas con las personas mediante el uso de lenguas naturales (idiomas), como el español o el inglés. [18]

Las lenguas humanas pueden expresarse por escrito (texto) u oralmente (voz). Naturalmente, el PLN está más avanzado en el tratamiento de textos, donde hay muchos más datos y son más fáciles de conseguir en formato electrónico.

Entre las diversas aplicaciones del PLN se encuentran la traducción automática de textos, los sistemas conversacionales o la clasificación de textos. Nuestro trabajo se centra en este último.

2.2.1 Word Embeddings

Word Embeddings [19] es una técnica del PLN que consiste, básicamente, en asignar un vector a cada palabra. Este vector guarda información semántica, lo que permite que pueda ser asociado o disociado a otros vectores (palabras) según distintos contextos gramaticales.

TF-IDF

La vectorización TF-IDF mide la importancia matemática de las palabras en el Corpus [20]. Es decir, las palabras comunes como "el" o "a" aparecerán con mucha frecuencia en casi todos los documentos. Sin embargo, otras palabras pueden aparecer con frecuencia sólo en uno o dos documentos y son las que probablemente sean más representativas del documento en el que están presentes.

El TF-IDF pretende poner esto en evidencia: ponderar las palabras comunes que aparecen en casi todos los documentos (por ejemplo, "el" o "a") y dar más importancia a las que sólo aparecen en unos pocos documentos.

En detalle, el TF IDF se compone de dos partes: TF, que es la frecuencia de términos de una palabra, es decir, el recuento de la palabra que aparece en un documento, e IDF, que es la frecuencia inversa del documento, es decir, el componente de peso que da mayor importancia a las palabras que aparecen sólo en unos pocos documentos.

Word2Vec

Word2Vec [21] es un método para construir estos *embeddings*. De hecho, su propio nombre traducido al español quiere decir “palabra a vector”. Puede obtenerse utilizando dos métodos (ambos con redes neuronales): *Skip-Gram* y *Common Bag of Words* (CBOW). Podemos observar ambos métodos en la figura 11

- CBOW: Este método toma el contexto de cada palabra como entrada e intenta predecir la palabra correspondiente al contexto.
- *Skip-Gram:* Este método hace lo contrario a CBOW, es decir, intenta predecir las palabras del contexto utilizando la palabra principal

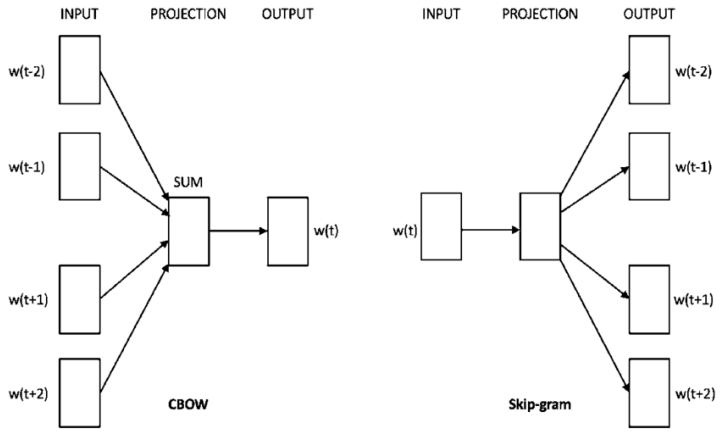

Figura 11: Arquitectura CBOW vs Skip Gram

GloVe

Global Vectors for Word Representation (GloVe) [22] es una extensión del método Word2Vec para el aprendizaje eficiente de vectores de palabras. GloVe nace para intentar solucionar el principal problema de Word2Vec, ya que este no aprende el contexto global de cada palabra. En el caso de GloVe, los *embeddings* aprendidos se generan a partir del contexto local y las estadísticas globales de cada palabra.

Para cada texto, GloVe construye una matriz explícita de coocurrencia de palabras o de contexto de palabras utilizando las estadísticas de todo el Corpus de texto. El resultado es un modelo de aprendizaje que puede dar lugar a incrustaciones de palabras generalmente mejores.

2.3 Transfer Learning

Transfer Learning es una técnica que consiste en usar un modelo que ya ha sido entrenado para una determinada tarea como base de un modelo para otra tarea similar [23]. De esta manera se consigue que el entrenamiento del modelo sea considerablemente más corto.

Una técnica muy utilizada en *Deep Learning* es el conocido como ajuste fino o *fine tuning*, en inglés. Esta técnica utiliza las propiedades del aprendizaje por transferencia, para acelerar el entrenamiento de grandes modelos en tareas concretas. Lo que se hace es entrenar modelos en tareas generales, como pueden ser, en el campo del PLN, el rellenado de mascara (Fill Mask) o predicción de la siguiente sentencia (Next Sentence Prediction). Una vez tenemos un modelo entrenado en una o más de las tareas mencionadas, se procede a re-entrenar el mismo modelo, con un Corpus más pequeño, sobre una tarea más concreta, por ejemplo, clasificar textos.

El ajuste fino se ha convertido en una tarea, no solo muy popular, sino necesaria. Entrenar un modelo de *Deep Learning* desde cero es una tarea que consume muchos recursos, se

necesitan cantidades masivas de datos a las que no todo el mundo tiene acceso, y el entrenamiento tarda tiempos poco asumibles. Entrenar un modelo muy grande para una tarea concreta puede llegar a tardar meses, un tiempo poco razonable. Se convierte en algo mucho más eficiente el entrenar modelos grandes en tareas generales, para luego solo tener que concretar mediante ajuste fino, convirtiendo así, una tarea de meses, en horas.

2.4 Tecnologías utilizadas

2.4.1 Python

Python[1] es el lenguaje de programación más popular para el campo del Aprendizaje Automático. Debido a su gran número de librerías y *frameworks* de fácil uso, la cantidad de información existente en Internet y su fácil curva de aprendizaje, ha sido el lenguaje elegido para este trabajo.

2.4.2 Google Colab

El entorno de desarrollo (IDE) utilizado para este trabajo ha sido Google Colab[2]. Colab es un entorno online basado en Jupyter, que permite combinar código ejecutable y texto enriquecido en un mismo documento. La elección de este entorno se debe a la posibilidad de acelerar la ejecución del código gracias a CPU y GPU alojadas que son proporcionadas gratuitamente por la propia empresa, además de poder enlazar con Google Drive todos los datos.

Al estar todo en la nube, también podemos continuar el mismo trabajo desde diferentes equipos, así como compartirlo todo con otros desarrolladores.

2.4.3 TensorFlow

TensorFlow[3] es una gran plataforma de código abierto para construir y entrenar redes neuronales desarrollada por Google. La arquitectura flexible de TensorFlow le permite implementar el cálculo a una o más CPU o GPU en equipos de escritorio o servidores con una sola API.

[1] https://www.python.org/
[2] https://colab.research.google.com/
[3] https://www.tensorflow.org/

2.4.4 HuggingFace

HuggingFace[4] es una comunidad de programadores que ha desarrollado una serie de APIs que permiten descargar y entrenar modelos de Aprendizaje Automático preentrenados de forma muy rápida y sencilla. En esta página se encuentran la gran mayoría de modelos preentrenados basados en Transformers que hemos usado para nuestro trabajo.

2.4.5 NudeNet

NudeNet[5] es librería de Python que, mediante modelos de Redes Neuronales identifica imágenes inseguras para los usuarios, es decir, violencia o desnudez, dándoles una puntuación en el rango de 0 (imagen segura) a 1 (imagen insegura). Para un trabajo de detección de memes misóginos, es muy útil a la hora de filtrar contenido que cosifique el cuerpo de la mujer.

[4] https://huggingface.co/
[5] https://github.com/notAI-tech/NudeNet

Capítulo 3

Planteamiento

Antes de comenzar a explicar los experimentos debemos tener muy claro el problema que estamos tratando y realizar una buena descripción de este. Así podremos determinar si se están cumpliendo los objetivos al finalizar el trabajo.

3.1 Descripción del problema a resolver

Nuestro problema consiste en abordar el problema planteado en la competición de SemEval 2022 en la tarea de detección de misoginia en memes[6]. Tal y como se cita en la página, "*La tarea propuesta, es decir, la Identificación Automática de Misoginia Multimedia (MAMI) consiste en la identificación de memes misóginos, aprovechando tanto el texto como las imágenes disponibles como fuente de información"*.

A lo largo de la competición, que comenzamos el 1 de octubre de 2021, encontramos varias fechas importantes:

- El *dataset* de entrenamiento se encontraba disponible desde el primer momento, para poder realizar modelos y estimar aproximadamente cuán bueno sería su funcionamiento en la competición
- Desde el 10 hasta el 30 de enero de 2022 comienza la fase de evaluación. Se nos facilita el *dataset* de test (sin etiquetar) para poder predecir sus clases y subir la predicción a la plataforma, que automáticamente nos devuelve qué porcentaje de acierto hemos obtenido.
- El 2 de febrero de 2022 se publican los datos de test etiquetados, para que cualquiera pueda tener acceso a ellos libremente.

3.2 Descripción del Dataset

Los datasets propuestos por la organización están compuestos por memes extraídos de las principales redes sociales de hoy en día, como Twitter y Reddit, y de páginas web dedicadas a la creación y difusión de memes en habla inglesa, como 9Gag, Knowyourmeme e imgur [3]. Podemos ver un ejemplo de meme del *dataset* en la figura 12

[6] https://competitions.codalab.org/competitions/34175

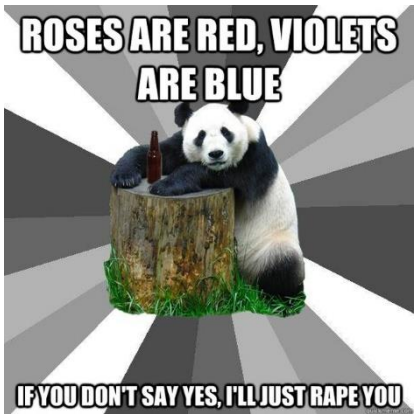

Figura 12: Ejemplo de meme del dataset de entrenamiento

Las imágenes fueron etiquetadas manualmente por voluntarios y cada fila del dataset sigue el siguiente formato (Tabla 1):

file_name	misogynous	shaming	stereotype	objetification	violence	Text Transcription
10.jpg	1	0	0	0	1	ROSES ARE RED, VIOLETS ARE BLUE IF YOU DON'T SAY YES, I'LL JUST RAPE YOU quickmeme.com

Tabla 1:Formato de cada instancia de los datasets

Donde *file_name* es el enlace .jpg a la imagen, *misogynous* nos indica si la imagen es o no misógina, *shaming* (humillación), *stereotype* (estereotipo), *objetification* (cosificación) y *violence* (violencia) es el tipo de misoginia, y *Text Transcription* es el texto extraído de la imagen. Para nuestro trabajo, que es de clasificación binaria, solo necesitaremos las columnas *file_name, misogynous* y *Text Transcription.*

El Corpus de entrenamiento se compone de 10.000 filas de las cuales 5.000 tienen valor 1 (misógino) y 5.000 tienen valor 0 (no misógino). La longitud media del texto transcrito de estos memes es de 18 palabras (figura 13), mientras que el Corpus de test, usado para evaluar los resultados de la competición, se compone de 1.000 filas con una longitud media de 13.4 palabras (figura 14), de las cuales, una vez dado el fichero etiquetado, pudimos conocer que 500 tienen valor 1 y 500 valor 0. En la tabla 2 podemos observar cómo se han repartido las filas de nuestros *datasets*.

	Entrenamiento	Test
Misógina (1)	5.000	500
No Misógina (0)	5.000	500
Total	10.000	1.000

Tabla 2: Proporción de memes de nuestros datasets

Podemos determinar, pues, que todos los Corpus dados están correctamente balanceados.

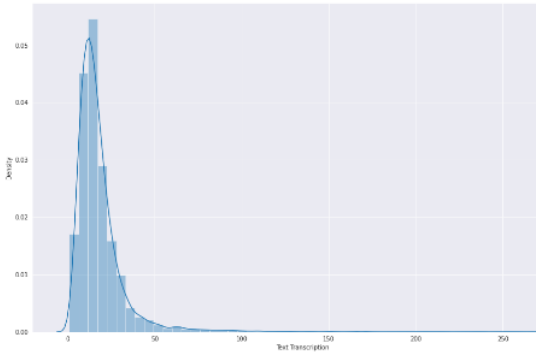

Figura 13: Distribución palabras Corpus entrenamiento

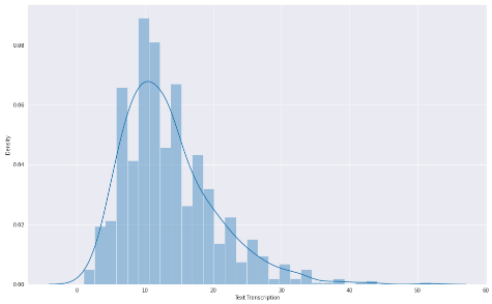

Figura 14: Distribución palabras Corpus test

Estudio de las palabras más frecuentes de cada clase

En las figuras 15 y 16 se muestran en formato de nube de palabras las 100 palabras más frecuentes por clase del *dataset* de entrenamiento. Así podemos observar qué términos son los más importantes a la hora de que un modelo pueda predecir si un texto pertenezca a la clase positiva o negativa.

En el formato de nube de palabras, cuanto más frecuentes son las palabras en el texto mayor tamaño tienen en la imagen.

Figura 15: Nube de palabras de la clase 1

Figura 16: Nube de palabras de la clase 0

Como podemos observar, aunque la nube de palabras de la clase 1 tiene más palabras ofensivas, ambas tienen muchas palabras en común. Esto se debe a que, dependiendo del contexto y de la imagen, una misma frase puede ser inocente o sin embargo misógina de forma sarcástica.

3.3 Métricas usadas

Para la evaluación de modelos de Aprendizaje Automático se usa un variado conjunto de métricas. Claramente cuanta más información tenemos del funcionamiento de un modelo, más fácil es determinar si satisface los objetivos propuestos. Para definir las métricas necesitamos introducir varios conceptos:

Matriz de confusión. La matriz de confusión (Tabla 3) es una matriz en las que se indica el número de predicciones de una clase, frente al número de instancias reales de la misma. Los elementos de la matriz se pueden definir de la siguiente forma:

- *True positive:* Predicciones positivas, cuya instancia real coincide.
- *True negative:* Predicciones negativas, cuya instancia real coincide.
- *False positive:* Predicciones positivas, cuya instancia real es negativa.
- *False negative:* Predicciones negativas, cuya instancia real es positiva.

Valor Real	Predicción	
	Negativo (0)	**Positivo (1)**
Negativo (0)	True Negative (TN)	False Positive (FP)
Positivo (1)	False Negative (FN)	True Positive (TP)

Tabla 3: Matriz de confusión

Para la competición MAMI de SemEval de 2022, los modelos se evaluaron usando la métrica *F1-Score*. Para poder calcular esta métrica introduciremos primero las métricas

Precision y *Recall*. Además, también evaluaremos individualmente nuestros modelos usando la *Accuracy*.

Precision: Mide la cantidad de predicciones positivas, que son a su vez instancias positivas, frente al total de predicciones de la clase.

$$\text{precision} = \frac{TP}{TP+FP}$$

Recall: Mide la cantidad de predicciones positivas, que son a su vez instancias positivas, frente al total de instancias positivas.

$$\text{recall} = \frac{TP}{TP+FN}$$

F1-Score: Es una medida que combina las dos métricas anteriores. En la competición, es la usada crear el ranking de modelos.

$$\text{f1} = 2 \cdot \frac{precision \cdot recall}{precision+recall}$$

Accuracy: Mide el porcentaje de elementos que han sido correctamente clasificados del total del modelo.

$$\text{acc} = \frac{TP+TN}{TP+TN+FP+FN}$$

3.4 Preprocesamiento del texto

De cara a la competición se han aplicado tres versiones de preprocesamiento del texto para limpiarlo y simplificarlo:

Preprocesamiento de texto v1: es el preprocesamiento más básico. Para esta versión, se aplicaron los siguientes filtros:

- Conversión de todo el texto en minúsculas.
- Eliminación de páginas web (palabras que empiecen por “http”).
- Eliminación de signos de puntuación.
- Eliminación de números.
- Eliminación de espacios en blanco múltiples.
- Eliminación de palabras con una longitud menor a 2 caracteres.
- Eliminación de *stopwords* en inglés.

Preprocesamiento de texto v2: Además de las características descritas en la v1.0, se añadieron las siguientes funcionalidades:

- Eliminación de emojis.
- Eliminación de usuarios (palabras que empiecen por @).
- Eliminación de hasgtags (palabras que empiecen por #).
- Eliminación de enlaces.

Preprocesamiento de texto v3: Este preprocesamiento es específico para los modelos basados en Transformers, ya que al ser modelos que necesitan entender el contexto de la frase, no hemos eliminado *stopwords* ni caracteres menores a cierta longitud, ya que esto empeoraría su rendimiento[24].

Capítulo 4

Experimentación y resultados

En este capítulo se explican los primeros experimentos que se realizaron para resolver el problema que nos plantea la competición, es decir, aquellos que realizamos durante su periodo de evaluación.

Para la solución del problema se han propuesto varios modelos, tanto de los considerados "clásicos", como las SVM, los árboles de decisión y los *random forest*, como de aquellos enfocados en el *Deep Learning*, como las redes LSTM y los Transformers.

4.1 Algoritmos clásicos

Si bien ya existen en la literatura algoritmos que superan a estos algoritmos clásicos, decidimos trabajar con ellos también de forma práctica como introducción al mundo de las tareas de clasificación y las redes neuronales.

Podemos observar en la figura 17 el flujo de trabajo seguido con los clasificadores clásicos.

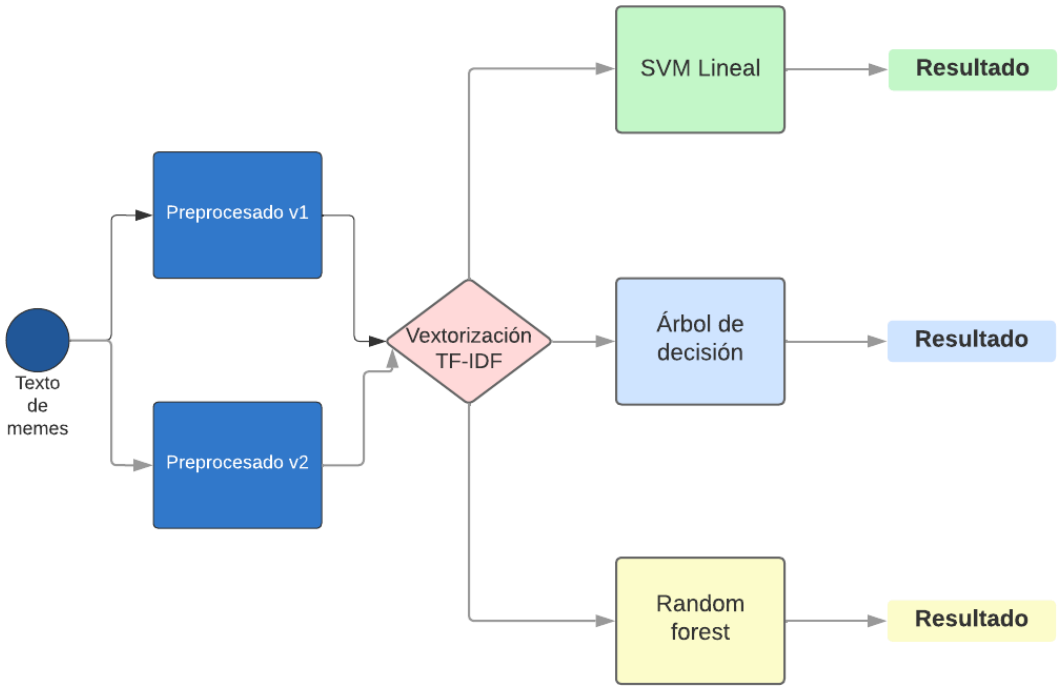

Figura 17: Flujo de trabajo seguido con los clasificadores clásicos

Primer modelo: SVM Lineal

El primer modelo se basó en el algoritmo clásico de Máquinas de Vector Soporte (SVM) lineal, al que se le proporciona una entrada (transcripción del texto escrito en el meme) y nos devuelve una salida (misógino o no).

Usamos una tolerancia de 0.0001, penalización de tipo l2 y un parámetro de regularización (C) de 1.0.

Se ha usado tanto el preprocesamiento de texto v1 como el preprocesamiento v2, y a ambos se le ha aplicado la vectorización TF-IDF. Ambos modelos fueron enviados a la competición, y podemos comprobar sus resultados en la Tabla 4.

	accuracy	**F1-score**	**Precision**	**Recall**
SVM – v1	0.63	0.628	0.64	0.63
SVM – v2	0.63	0.627	0.64	0.63

Tabla 4: Resultados modelo SVM Lineal

Segundo modelo: Árbol de Decisión

Al igual que en el modelo anterior, usaremos el preprocesamiento v1 y v2, y una vectorización TF-IDF para el texto. No se limitará el máximo número de hojas o la profundidad del árbol, y el mínimo número de muestras para dividir una hoja interna será 2.

Los resultados obtenidos son (Tabla 5):

	accuracy	**F1-score**	**Precision**	**Recall**
Árbol – v1	0.61	0.605	0.62	0.61
Árbol – v2	0.61	0.600	0.62	0.61

Tabla 5: Resultados del modelo Árbol de Decisión

Tercer modelo: Random Forest

Los *Random Forest* son conjuntos de árboles de decisión, que al ser combinados permiten obtener mejores resultados que uno solo.

En este caso hemos preprocesado el texto solo con nuestro preprocesamiento v1, pues en los anteriores modelos dio mejores resultados, y hemos usado vectorización TF-IDF. El número de estimadores (árboles) del modelo fue de 100 en el primero y 1000 en el segundo. Podemos observar los resultados en la Tabla 6:

	accuracy	**F1-score**	**Precision**	**Recall**
Random Forest (1000 trees)	0.63	0.615	0.65	0.63
Random Forest (100 trees)	0.63	0.615	0.65	0.63

Tabla 6: Resultados del modelo Random Forest

4.2 Recursos Externos

Tras comprobar que las redes neuronales convolucionales clásicas no tenían buenos resultados resolviendo esta tarea, para enriquecer la información de las imágenes hemos usado el módulo de Python *nudenet*, ya explicado anteriormente, para añadir a los *datasets* la columna "*unsafe*", que nos indica en el rango de 0 a 1 cuánto de insegura es la imagen para el espectador, siendo un 0 segura y un 1 insegura. (Tabla 7).

file_name	…	Text Transcription	Unsafe
10846.jpg	…	SANDWICH!!!!!! don't make me tell you twice woman	0.890

Tabla 7: Dataset tras añadir la columna unsafe

Esta columna nos aporta información extra sobre si el meme es o no misógino, y nos permite complementar los resultados obtenidos por los clasificadores textuales utilizados, haciendo que, si la columna *unsafe* supera cierto umbral, la imagen sea considerada insegura para el espectador y automáticamente se clasifique como misógina, independientemente de lo obtenido en la clasificación por texto.

Tras obtener esta columna, todos nuestros modelos devuelven dos resultados, el del clasificador de texto puro, y el del clasificador de texto apoyado por el filtrado de imagen usando la columna *unsafe*.

Hemos decidido que el umbral a partir del cual una imagen debe ser clasificada como misógina sea 0.45. En la figura 18 podemos observar cual ha sido el procedimiento a la hora de complementar la clasificación textual con el resultado obtenido en la columna unsafe.

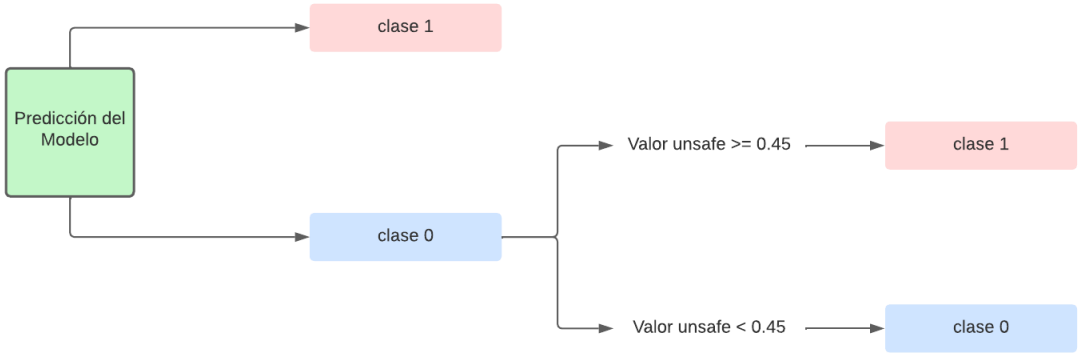

Figura 18: Complemento a la clasificación textual usando la columna unsafe

4.3 Deep Learning

En el caso de los algoritmos de *Deep Learning*, hemos usado los algoritmos predominantes en tareas de NLP hoy en día, es decir, la red neuronal LSTM y los Transformers.

El algoritmo BERT ha sido usado como clasificador, tras hacer *fine-tuning* con nuestro dataset, y como primera capa de *embedding* de dos redes neuronales diferentes: Una red biLSTM y una red Convolucional 1D.

4.3.1 Red Neuronal biLSTM

Para este modelo preprocesamos el texto con el preprocesamiento v2, y tras tokenizarlo usando el método GloVe, lo añadimos como primera capa de *Word Embedding* preentrenada del modelo. Podemos observar la arquitectura del modelo en la figura 19.

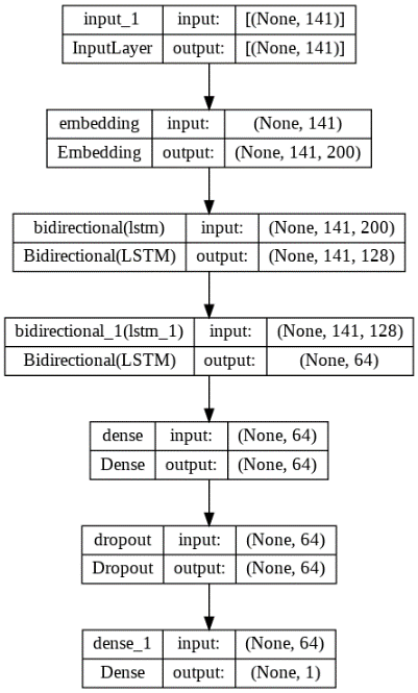

Figura 19: Arquitectura del modelo LSTM

Entrenaremos el modelo durante 10 épocas, con un *batch size* de 32 y *early stopping* con paciencia de 3 épocas teniendo en cuenta la *accuracy* del dataset de validación.

4.3.2 BERT Transformers

Como ya se comentó anteriormente, las representaciones codificadoras bidireccionales a partir de Transformers (BERT) son una técnica de Aprendizaje Automático basada en Transformers. Hemos elegido el modelo "*bert-base-uncased*" para clasificar nuestro

Corpus tras hacer fine-tuning con el modelo, y como primera capa de Embedding de las redes biLSTM y Conv1D. Podemos observar el flujo de trabajo seguido en la figura 20.

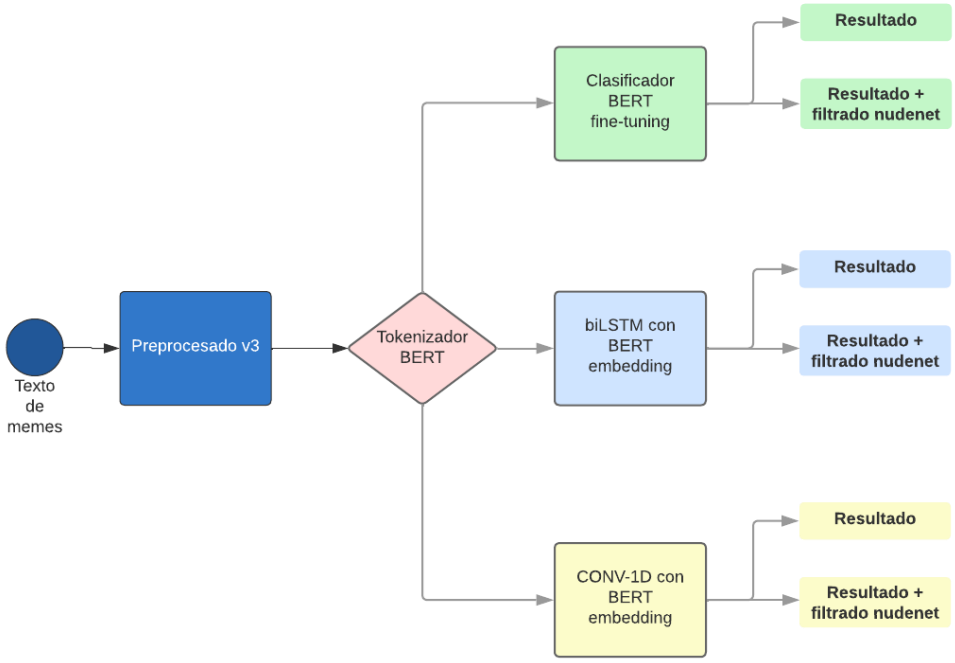

Figura 20: Flujo de trabajo usando BERT

Se ha decidido usar este modelo por estar pre-entrenado a partir de textos extraídos de BooksCorpus (800 millones de palabras) y de Wikipedia en inglés (2500 millones de palabras), y ser el principal modelo en la literatura de Transformers.

Clasificador BERT fine-tuning

Tras preprocesar el texto usando el preprocesado v3, específico para los modelos Transformer, tokenizaremos nuestro texto usando el propio tokenizador de BERT. Posteriormente, haremos *fine tuning* al modelo preentrenado "*bert-base-uncased*", entrenando con nuestros datos de *training* durante 4 *epochs*, un *batch size* de 32 y *learning rate* de 2e-5.

Clasificador biLSTM con BERT como embedding

BERT no solo puede ser usado como clasificador, si no también como tokenizador de texto, para poder luego añadir nuestro Corpus tokenizado con BERT como capa de *embedding* a distintos modelos. BERT ofrece una ventaja sobre modelos como Word2Vec, ya que mientras que cada palabra tiene una representación fija en Word2Vec, independientemente del contexto en el que aparezca, BERT produce representaciones de palabras que son informadas dinámicamente por las palabras que las rodean.

Este modelo biLSTM ha sido entrenado durante 8 *epochs*, con un *batch size* de 32 y optimizador Adam de 0,01.

Podemos observar la arquitectura del modelo en la figura 21

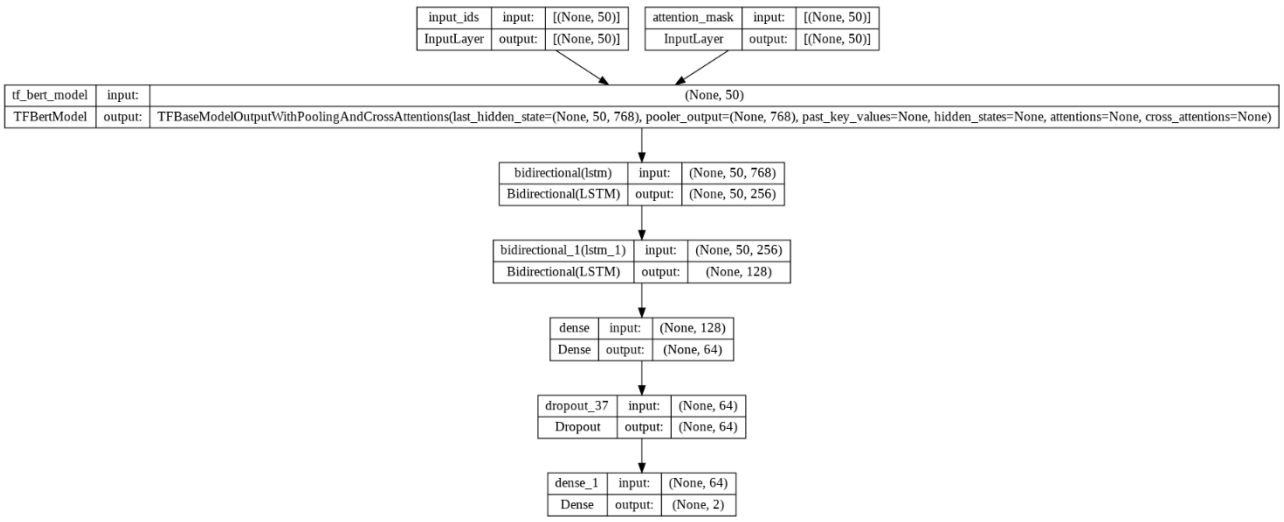

Figura 21: Arquitectura del modelo biLSTM con BERT como embedding

Clasificador Conv1D con BERT como embedding

Si bien las Redes Neuronales Convolucionales son conocidas por su uso en clasificación de imágenes, se ha demostrado que las redes convolucionales 1D son también muy buenas en muchos campos del PLN. [25]

Este modelo Convolucional con tres capas de Convolución 1D, al que se le ha añadido BERT como primera capa de embedding ha sido entrenado durante 8 *epochs*, con un *batch size* de 32 y optimizador Adam de 0,01.

Podemos observar la arquitectura del modelo en la figura 22.

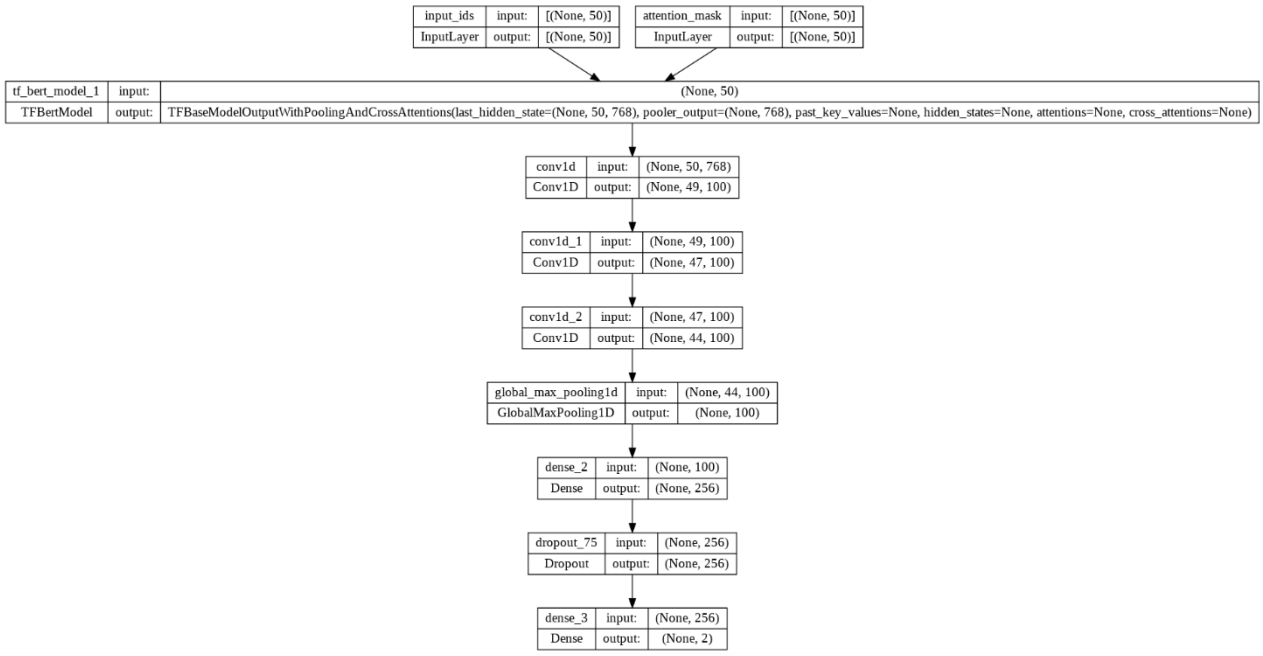

Figura 22: Arquitectura del modelo Conv1D con BERT como embedding

4.3.3 Resultados

En la Tabla 8 podemos ver los resultados obtenidos por estos modelos, ordenados de mayor a menor puntuación

Modelo	Accuracy	F1-Score	Precision	Recall
BiLSTM BERT embedding + nudenet	**0.670**	**0.665**	**0.66**	**0.65**
BiLSTM BERT embedding	0.662	0.652	0.65	0.64
Clasificador BERT Fine-tuning + nudenet	0.658	0.649	0.69	0.66
Clasificador BERT Fine-tuning	0.654	0.645	0.67	0.65
Conv1D BERT embedding + nudenet	0.641	0.639	0.69	0.64
Conv1D BERT embedding	0.635	0.630	0.68	0.62
biLSTM – gloVe embedding	0.62	0.605	0.63	0.62

Tabla 8: Resultados modelos BERT

De 83 participantes en la competición, se obtuvo el puesto 43, con un F1-Score de 0.665. Como referencia, los tres primeros puestos obtuvieron unos valores de F1-Score de 0.834, 0.811 y 0.794, respectivamente.

Al no ser los resultados esperados, se decidió continuar con el estudio de este problema en la fase post-evaluacion de la competición. En el siguiente capítulo, se detallan los diferentes estudios realizados para mejorar los resultados obtenidos.

Capítulo 5

Propuestas de mejoras

Una vez finalizada la competición, los organizadores aportaron a los equipos el conjunto de datos de test etiquetado, el cual estaba formado por los memes que proporcionaron durante la etapa de evaluación con sus respectivas etiquetas de misoginia.

Ante esta oportunidad, se decidió seguir trabajando para aportar nuevas propuestas e intentar mejorar los modelos entregados en la competición. El objetivo era mejorar los resultados que se obtuvieron en el ranking de la competición.

En este apartado, se describen las diferentes propuestas de mejora que se han llevado a cabo.

5.1. Pipeline de experimentación

Para los experimentos llevados a cabo durante el periodo de propuestas de mejoras hemos diseñado un pipeline, que podemos observar en la figura 23. Las propuestas de mejora se basan en la búsqueda de la optimización de los hiperparámetros a la hora de hacer fine-tuning, además de posteriormente construir *ensembles* para mejorar los resultados. Así, determinaremos mediante una serie de rankings qué hiperparámetros dan los mejores resultados para cada modelo y cuales dan mejores resultados en cada una de las clases (positiva, negativa y media de ambas) para formar un ensemble que intente combinar lo mejor de cada modelo.

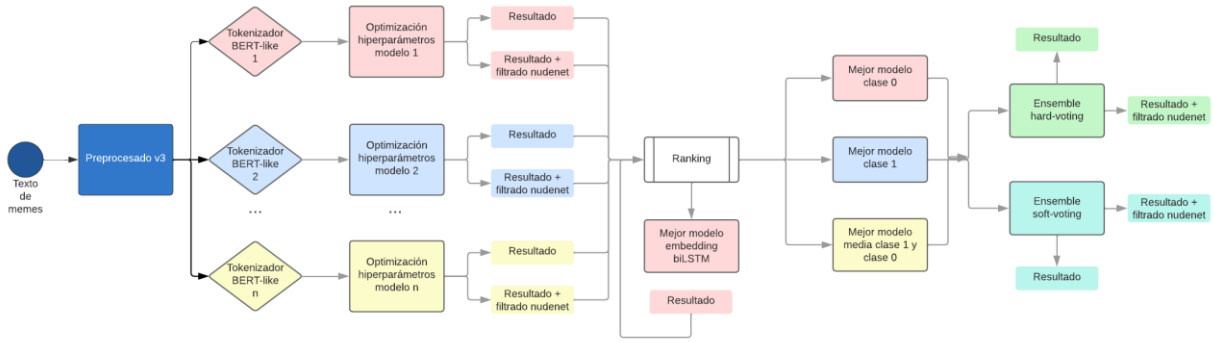

Figura 23: Pipeline de experimentación propuestas de mejoras

5.2. Optimización de hiperparámetros

Decidimos hacer optimización de hiperparámetros al hacer *fine-tuning* en los 5 modelos más populares basados en BERT para clasificación de textos, y además en un modelo pre-entrenado específicamente para detección de lenguaje sexista subido por la comunidad.

Los hiperparámetros que hemos seleccionado y sus valores son los más comunes usados en la literatura [26] y en las webs especializadas [27]. Estos son:

número de épocas

En el Aprendizaje Automático, una época se refiere a una iteración completa a través del conjunto de datos de entrenamiento. Por ejemplo, si está entrenando un modelo de redes neuronales en un conjunto de datos de 1000 imágenes y establece el número de épocas en 10, significa que el modelo verá todas las 1000 imágenes 10 veces durante el entrenamiento.

Un número mayor de épocas puede llevar a un mejor rendimiento del modelo en el conjunto de datos de entrenamiento, pero también puede aumentar el tiempo de entrenamiento y aumentar el riesgo de sobreajuste. Por lo tanto, es importante encontrar el equilibrio adecuado entre el número de épocas y el rendimiento del modelo.

batch size

El tamaño del lote o *batch size* se refiere a la cantidad de muestras de datos que se procesan en una sola actualización del modelo. Por ejemplo, si está entrenando un modelo en un conjunto de datos de 1000 imágenes y establece el tamaño de lote en 100, significa que el modelo procesará 100 imágenes a la vez y actualizará los pesos y sesgos del modelo después de cada lote de 100 imágenes.

Un tamaño de lote más grande puede llevar a una mayor estabilidad durante el entrenamiento, ya que cada actualización del modelo se basa en una mayor cantidad de datos. Sin embargo, un tamaño de lote más grande también puede requerir más memoria y puede ser más lento de procesar.

learning rate

El *learning rate* se refiere a la tasa a la que se actualizan los pesos y sesgos del modelo durante el entrenamiento. Por ejemplo, si un modelo está utilizando un learning rate de 0.1, significa que los pesos y sesgos del modelo se actualizarán en un 10% cada vez que se procesen un conjunto de datos de entrenamiento.

El *learning rate* es un hiperparámetro importante que puede afectar significativamente el rendimiento del modelo durante el entrenamiento. Un *learning rate* demasiado alto puede hacer que el modelo "salte" sobre el mínimo global y no converja, mientras que un *learning rate* demasiado bajo puede hacer que el entrenamiento sea muy lento.

Podemos observar los valores usados en la Tabla 9.

Hiperparámetro	Valores
número de épocas	[2, 3, 4]
batch size	[16, 32]
learning rate	[4e-5, 3e-5, 2e-5]

Tabla 9: Valores usados en la optimización de los hiperparámetros

Los modelos BERT preentrenados que hemos usado para optimizar hiperparámetros durante el fine-tuning son:

- **bert-base-uncased:** 12 capas de transformer, con un total de 768 capas ocultas. no diferencia texto en mayúscula o en minúscula.
- **bert-large-uncased:** 24 capas de transformer, con un total de 1024 capas ocultas. Versión ampliada de bert-base.
- **roberta-base:** misma arquitectura que bert-base, pero entrenado de forma diferente, consiguiendo así en la práctica mejores resultados.
- **distilbert-base-uncased:** modelo con la mitad de capas que bert-base, pero que teóricamente posee el mismo rendimiento que este. [28]
- **xlnet-base-cased:** arquitectura parecida a bert-base, pero entrenado con una mayor cantidad de datos durante más tiempo, y usando una metodología de enmascaramiento de palabras mejorada. [29]
- **roberta-large-mnli-misogyny-sexism:** este modelo, al igual que bert-large, es la versión “ampliada” de Roberta-base. Además, ya ha sido previamente entrenado en clasificación de textos sexistas y misóginos por la usuaria annahaz[7], y podemos usarlo directamente para clasificar nuestro Corpus de test o hacer fine-tuning usando nuestro Corpus de entrenamiento. Lo abreviaremos como roberta-misogyny de aquí en adelante.

Tras entrenar y comprobar los resultados de todos los clasificadores y sus combinaciones de hiperparámetros recopilamos la *accuracy* y el *f1-score* con y sin filtrado de nudenet y obtuvimos más de 100 filas de resultados diferentes. Podemos observar los 5 mejores resultados en la Tabla 10.

[7] https://huggingface.co/annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt

Modelo	Num epochs	Batch size	Learning rate	Acc nudenet	F1 Score nudenet	Acc	F1 Score
roberta-misogyny	**3**	**32**	**3e-05**	**0.692**	**0.680**	**0.681**	**0.673**
roberta-misogyny	4	32	4e-05	0.684	0.672	0.670	0.662
roberta-misogyny	3	16	3e-05	0.679	0.668	0.665	0.658
bert-base-uncased	3	32	2e-05	0.678	0.667	0.67	0.663
roberta-misogyny	4	16	4e-05	0.680	0.665	0.669	0.659

Tabla 10: Mejores resultados optimización de hiperparámetros

El mejor resultado, una F1-Score de 0.680, mejora en un 2.26% el resultado obtenido en la etapa de evaluación, y nos haría subir de la posición 43 a la 38.

Podemos observar todas las filas de esta macrotabla en el **Anexo 2**.

5.3. Ensemble

Un conjunto o *ensemble* es, como su propio nombre indica, un conjunto de modelos de clasificación. En este las predicciones de los distintos modelos se combinan para obtener una única predicción. Podemos ver una representación gráfica de un *ensemble* en la figura 24.

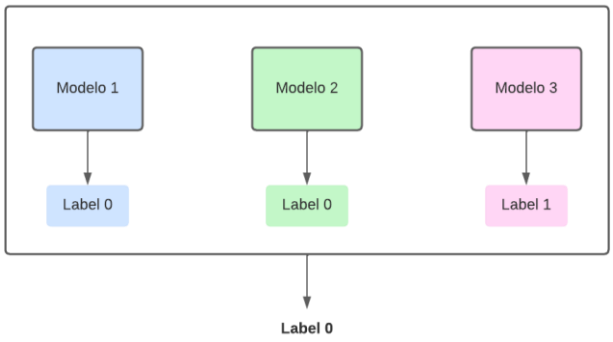

Figura 24: Ensemble de 3 modelos usando hard voting

En nuestro caso hemos realizado *ensembles* de tres modelos con dos técnicas diferentes; *hard voting* y *soft voting*.

En el *hard voting* cada modelo tendrá asociado un voto, en nuestro caso 1 o 0. La predicción final será lo que voten la mayoría de los modelos.

En el *soft voting* en lugar de tener en cuenta el voto final tomamos las probabilidades que los modelos dan a cada clase, dando más importancia a los resultados en los que un modelo esté muy seguro.

Primer *ensemble*

En el primer *ensemble* tomamos los tres mejores modelos de la etapa de evaluación. Estos son el clasificador de BERT, BERT como embedding en una red biLSTM, y BERT como embedding en una red Conv1D.

Podemos observar los resultados obtenidos en la Tabla 11:

Ensemble 1	accuracy	f1-score
Hard-voting	**0.674**	**0.667**
Soft-voting	0.651	0.642
Soft-voting + nudenet	0.659	0.645

Tabla 11: Resultados del primer ensemble

Como podemos comprobar, los resultados de este primer ensemble sólo mejoran en un 0.002 nuestro mejor resultado de la fase de evaluación. Esto probablemente se deba a que los modelos entrenados son casi siempre mejores detectando la clase 1 que la clase 0, por lo que decidimos realizar un segundo *ensemble* más preciso.

Segundo *ensemble*

Para este segundo ensemble decidimos escoger los tres modelos que mejor se complementaran entre sí, es decir el modelo que tiene mejor *f1-score* en la clase 1, el modelo que tiene mejor *f1-score* en la clase 0, y el modelo que tiene una mejor *f1-score* media. Estos tres modelos son el roberta-misogyny puro, sin ningún tipo de *fine-tuning* y el filtrado por nudenet, el mismo modelo anterior con *fine-tuning* (3 *epochs, batch size* de 32 y *learning rate* de 3e-05), y una red neuronal biLSTM con el modelo roberta-misogyny como capa de *embedding*.

Podemos observar las puntuaciones individuales de estos modelos en Tabla 12:

Modelo	F1-score clase 0	F1-score clase 1	F1-score media
roberta-misogyny puro + nudenet	0.72	0.65	0.685
roberta-misogyny fine-tuning	0.62	0.74	0.680
biLSTM roberta-misogyny embedding	0.68	0.70	0.690

Tabla 12: f1-scores de los modelos usados para el segundo ensemble

Una vez realizado el ensemble entre estos tres modelos, los resultados son los presentados en la Tabla 13:

Ensemble 2	accuracy	f1-score
Hard-voting	0.709	0.710
Soft-voting	0.674	0.669
Soft-voting + nudenet	**0.717**	**0.717**

Tabla 13: resultados del segundo ensemble

Este segundo ensemble mejora considerablemente los resultados obtenidos en la fase de evaluación, mejorando los resultados en un 7.25% y llegando así a la posición 22 del ranking de la competición.

Capítulo 6

Conclusiones y trabajo futuro

En este apartado se exponen las conclusiones a las que se han llegado con la realización de este trabajo y, seguidamente, se plantean futuras líneas de investigación para poder mejorar los resultados obtenidos.

6.1. Conclusiones

Con la realización de este trabajo se ha demostrado la importancia del procesamiento del lenguaje natural a la hora de identificar patrones de comportamiento, en este caso la misoginia u odio hacia las mujeres. El reconocimiento de actitudes de este tipo en Internet es importantísimo para censurarlas a tiempo, evitar que se expandan y así conseguir un mundo, tanto digital como físico, donde poder convivir en armonía entre géneros.

Identificar este tipo de actitudes es algo complejo, pues la misoginia puede no solo transmitirse en imágenes con un texto corto asociado, sino también en videos, textos largos e incluso audios. Además, el hecho de que esta puede expresarse no solo de forma directa si no también usando sarcasmo, ironía o incluso noticias falsas, nos indica que aún queda mucho margen de mejora para resolver esta tarea.

El ganador de la competición obtuvo un *f1-score* de 0.834 y nosotros, tras las propuestas de mejoras, hemos conseguido ascender desde la posición 43 hasta la 22. Esto es una mejora considerable teniendo en cuenta que hemos centrado nuestro trabajo en el procesamiento de textos, y no tanto en el de imágenes.

Fruto de este trabajo se ha publicado un paper científico en la competición *International Workshop on Semantic Evaluation* (SemEval), incluido en el Anexo de este documento. Además, se han hecho públicos tanto los datasets y modelos[8] como los cuadernos de Google colab[9] realizados, para que cualquiera pueda usarlos.

Por último, como resultados técnicos se han conseguido los siguientes hitos relevantes:

- Se ha probado la importancia de combinar la clasificación de texto e imagen, aunque en nuestro caso fuera en pequeña escala, ya que en la mayoría de casos da mejores resultados que usar simplemente uno de los dos.
- Se ha observado que la optimización de hiperparámetros, si bien consume muchos recursos y tiempo, mejora notablemente los resultados ante usar los parámetros por defecto.

[8] https://drive.google.com/drive/folders/1-8LAIlelK4JLcvvBNaggCdKkKcXjmwqo?usp=sharing
[9] https://drive.google.com/drive/folders/1GhKuOK4d1Fhs9CPl4_VpL-FrpeSm-Yjk?usp=sharing

- Se ha comprobado que los *ensembles*, cuando son usado correctamente, mejoran considerablemente los resultados de modelos individuales, y que incluso un ensemble de modelos poco potentes puede dar mejores resultados que un modelo de alto nivel computacional.
- Se ha demostrado que existen técnicas abiertas que no requieren excesivos recursos computacionales accesibles al público para poder realizar estudios y resolver problemas de clasificación de este tipo.

6.2. Opinión personal

La realización de este trabajo me ha abierto la puerta a una nueva forma de trabajar en el campo del *Deep Learning*, y especialmente el procesamiento del lenguaje natural, pues gracias a este he descubierto y aprendido una gran cantidad de herramientas y teoría que muy probablemente me servirán en el futuro.

Cuando, hace ya algo más de un año, se me ofreció la oportunidad de realizar un TFG en el campo del PLN, acepté aun no teniendo idea sobre el tema, e incluso con conocimientos muy básicos de Python, el lenguaje de programación con el que he programado todo el proyecto. Pero poco a poco, cada vez que iba avanzando, mi motivación aumentaba, llegando incluso a sentir un progreso exponencial no solo a nivel académico, sino también personal.

La detección de misoginia es una tarea tan compleja como necesaria, y el poder contribuir a censurar este tipo de actitudes logrando así un ambiente de paz para las mujeres internautas ha sido sin duda uno de mis mayores empujes a la hora de querer progresar. He disfrutado mucho de todo el desarrollo, tanto de la parte de programación como la de investigación y estoy seguro de que esta no será la última vez en la que me vea envuelto en un trabajo relacionado con este campo.

6.3. Trabajo futuro

A lo largo de este trabajo hemos podido observar que hay varios caminos por los que seguir expandiendo esta investigación del problema a resolver.

En primer lugar, hemos priorizado la clasificación textual a la visual, por lo que se podría hacer mayor hincapié en usar modelos de clasificación multimodales, como ERNIE-ViL, VisualBERT o el reciente modelo introducido por Google, MUM.

Asimismo, se podría probar a aumentar el tamaño de nuestros datos de *training* buscando memes en webs abiertamente misóginas, sabiendo que estos estarán etiquetados como 1, y en webs feministas sabiendo que estos tendrán la etiqueta 0.

Por último, los modelos de clasificación textual están en constante cambio, por lo que siempre sería muy interesante probar todos los nuevos modelos que se vayan publicando y así comprobar su efectividad en la resolución del problema.

Capítulo 7

Bibliografía

[1] E.D. Liddy, "Enhanced Text Retrieval Using Natural Language Processing," American Society for Information Science. Bulletin of the American Society for Information Science, vol. 24, pp. 14-16, Apr. 1998.

[2] R. Collobert and J. Weston, "A unified architecture for natural language processing," pp. 160-167, Jul 05, 2008.

[3] E. Fersini, F. Gasparini, G. Rizzi, A. Saibene, B. Chulvi, P. Rosso, A. Lees and J. Sorensen, "SemEval-2022 Task 5: Multimedia Automatic Misogyny Identification," in Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), 2022.

[4] P. Cordon, P. Gonzalez Diaz, J. Mata and V. Pach 'on, "I2C at SemEval-2022 Task 5: Identification of misogyny in internet memes," in Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pp. 689-694, 2022.

[5] J. Lilian and Sandoval, "ALGORITMOS DE APRENDIZAJE AUTOMÁTICO PARA ANÁLISIS Y PREDICCIÓN DE DATOS MACHINE LEARNING ALGORITHMS FOR DATA ANALYSIS AND PREDICTION A A Palabras clave Keyword," Derechos Reservados • Escuela Especializada En Ingeniería ITCA-FEPADE, .

[6] S. Rodríguez-Tapia and J. Camacho-Cañamón, "Los métodos de aprendizaje automático supervisado en la clasificación textual según el grado de especialización," Tonos Digital, vol. 2018, 2018.

[7] L. Quituisaca-Samaniego and H. Álvarez, "Aprendizaje no supervisado: agrupamiento o clustering," 2017.

[8] H. Díaz-Iza, L. Armesto and A. Sala, "Aprendizaje por Refuerzo para sistemas lineales discretos con dinámica desconocida: Simulación y Aplicación a un Sistema Electromecánico," 2017.

[9] S. Fransiska, Rianto and A.I. Gufroni, "Sentiment Analysis Provider by.U on Google Play Store Reviews with TF-IDF and Support Vector Machine (SVM) Method," Scientific Journal of Informatics, vol. 7, pp. 2407, -11. 2020.

[10] L. Rokach and O. Maimon, "Decision Trees," 2005, pp. 165-192.

[11] P. Calhoun, X. Su, K. Spoon, R. Levine and J. Fan, "Random Forest," 2021, pp. 1-20.

[12] N. Ponnaganti and R. Anitha, "A Novel Ensemble Bagging Classification Method for Breast Cancer Classification Using Machine Learning Techniques," Traitement Du Signal, vol. 39, pp. 229-237, 2022.

[13] T. Lei and A. Nandi, "Neural Networks," 2022, pp. 63-96.

[14] F. Salem, "Gated RNN: The Long Short-Term Memory (LSTM) RNN," 2022, pp. 71-82.

[15] Z. Huang, W. Xu and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging," 2015.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, "Attention Is All You Need," 2017.

[17] Z. Liu, W. Lin, Y. Shi and J. Zhao, "A Robustly Optimized BERT Pre-training Approach with Post-training," in Chinese Computational Linguistics, Cham: Springer International Publishing, 2021, pp. 471-484.

[18] S. Patil, "Natural language processing npl and its challenges," 2022.

[19] F. Almeida and G. Xexéo, "Word Embeddings: A Survey," 2019.

[20] S. Amin, M.I. Uddin, S. Hassan, A. Khan, N. Nasser, A. Alharbi and H. Alyami, "Recurrent Neural Networks With TF-IDF Embedding Technique for Detection and Classification in Tweets of Dengue Disease," Access, vol. 8, pp. 131522-131533, 2020.

[21] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," 2013.

[22] J. Pennington, R. Socher and C.D. Manning, "Glove: Global vectors for word representation," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

[23] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue and G. Zhang, "Transfer learning using computational intelligence: A survey," Knowledge-Based Syst., vol. 80, pp. 14-23, 2015.

[24] Y. Qiao, C. Xiong, Z. Liu and Z. Liu, "Understanding the Behaviors of BERT in Ranking," Cornell University Library, arXiv.org., 2019.

[25] R. Wang, Z. Li, J. Cao, T. Chen and L. Wang, "Convolutional Recurrent Neural Networks for Text Classification," in 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1-6, 2019.

[26] P. Izsak, M. Berchansky and O. Levy, "How to Train BERT with an Academic Budget," CoRR, vol. abs/2104.07705, 2021.

[27] Amog Kamsetty, "Hyperparameter Optimization for Transformers: A guide," Aug 26,. 2020.

[28] V. Sanh, L. Debut, J. Chaumond and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," CoRR, vol. abs/1910.01108, 2019.

[29] Z. Yang, Z. Dai, Y. Yang, J.G. Carbonell, R. Salakhutdinov and Q.V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding," CoRR, vol. abs/1906.08237, 2019.

Anexo I

Paper SemEval 2022 Task 5: Identification of Misogyny in Internet Memes

I2C at SemEval-2022 Task 5: Identification of Misogyny in Internet Memes

Pablo Cordón Hidalgo
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
pablo.cordonl13@alu.uhu.es

Pablo Gonzalez Díaz
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
pablo.gonzalez682@alu.uhu.es

Jacinto Mata Vázquez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
mata@uhu.es

Victoria Pachón Álvarez
Escuela Técnica Superior de Ingeniería
Universidad de Huelva (Spain)
vpachon@uhu.es

Abstract

In this paper we present our approach and system description on *Task 5 A in MAMI: Multimedia Automatic Misogyny Identification*. In our experiments we compared several architectures based on deep learning algorithms with various other approaches to binary classification using Transformers, combined with a nudity image detection algorithm to provide better results. With this approach, we achieved a test accuracy of 0.665.

1 Introduction

Misogyny is hatred or contempt for women. It is a form of sexism used to keep women at a lower social status than men, thus maintaining the societal roles of patriarchy. Misogyny has been widely practiced for thousands of years. It is reflected in art, literature, human societal structures, historical events, mythology, philosophy, and religion worldwide (Manne, 2017).

The Internet represents for many an extension of our offline interactions, and seemingly mundane everyday practices (e.g. participating in social media) form a significant part of our everyday experiences. Unfortunately, it is too common to see examples of harassment towards women and marginalized groups online within these experiences and practices (Drakett et al., 2018).

Women have a solid presence on the web, especially in picture-based web media like Twitter and Instagram: 78% of females utilize online media on numerous occasions each day, in contrast with 65% of men.

A popular way of communicating via social media platforms are MEMEs. A meme is an image portrayed through pictorial content with overlaid text which is written a posteriori, with the fundamental objective of being entertaining and/or ironic. Even though most of memes are created with the goal of making amusing jokes, shortly after their standardization individuals began to use them to disseminate hate against women, leading to sexist and aggressive messages in internet environments that allow people to freely express sexism without the fear of retaliation.

The detection of this disrespectful content is essential to eliminate it as soon as possible and stop spreading misogyny as a "joke".

In MAMI: Task 5, Track A (meme binary classification) (Fersini et al., 2022), participants must determine whether a meme (text + image) is misogynist or not.

Meme sentiment-related tasks analysis is challenging, as memes are used created for various purposes, they are always evolving and often use sarcasm and humour. While misogyny and hate speech detection in text has been widely explored by the NLP community (Badjatiya et al., 2017). Its detection in images and text and how they correlate has not been explored in depth.

In this field we used BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2021) for training and classifying text, and nudenet tool for image classification (https://github.com/notAI-tech/NudeNet).

Our results show that combining text and image classification results are slightly better than using only one of the two methods.

The rest of the paper is organized as follows. Section 2 contains a briefly description of the dataset and its structure, Section 3 features the analysis of some of the previous works related to our task. In section 4 we describe the different

models and algorithm used, and their configurations. Section 5 provides details about data and models setup, while Section 6 reports experimental results and the paper is concluded in Section 7.

2 Background

This paper is focused on subtask A: Binary classification. The corpus provided is composed of 5000 1-value rows (misogyny) and 5000 0-value rows (not misogyny), so it is already well-balanced. Each row has the following format:

file_name | misogynous | shaming | stereotype | objectification | violence | text transcription

Where "file_name" is the .jpg link to the image, and "text transcription" is the text extracted from the image.

For our binary classification task, we only need the fields "file_name", "misogynous" and "text transcription" to train the models presented. For models that also used nudity recognition, a column "unsafe" was added later.

The csv file used follows this structure:

- file_name: *"10.jpg"*
- misogynous: "*1*"
- text transcription: "*ROSES ARE RED, VIOLETS ARE BLUE IF YOU DON'T SAY YES, I'LL JUST RAPE YOU quickmeme.com*"

3 Related work

Sentiment analysis of text is a very active research area that still faces multiple challenges such as irony and humour detection (Farías et al., 2016). In this area, the focus of the NLP community has increased towards detection of offensive language, aggression, hate-speech detection (Wei et al., 2021) and specifically misogyny, taking into account it can be expressed in a direct, explicit manner or an indirect, sarcastic manner, and even if this message is genered or not-gendered (Samghabadi et al., 2020).

The analysis of misogyny in memes has already been done in a psychologic point of view (Drakett et al., 2018), but never in computational models. Multimodal analysis research has been extended during the last years, but the focus was mostly on Video and text or speech and text (Pozzi et al., 2016). The specific multi-modality of memes in sentiment analysis has only been addressed recently by investigating their correlation with other comments in online discussions (French, 2017).

The growing usage of memes as an alternative medium of communication on social media has also recently drawn the attention of the online abuse research community.

However, memes completely make sense only if one takes both text and image content into account. These modalities can also lead to totally different perceived sentiment when recombined. For example, a meme whose image is a scary clown and the text is "happy birthday" will have a very different sentiment from a meme with the same text but with an image of a funny clown.

Sabat et al. (2019) performed hate speech detection on memes and showed that images were more important than text for the prediction.

In the other hand, Bonheme and Grzes (2020), investigated the relationship of text and image in sentiment analysis of memes, and found that images and text were uncorrelated. Fusion-based strategies did not show significant improvements and using one modality only (text or image) tends to lead to better results.

4 System overview

We focus on exploring different training techniques for text using BERT and RoBERTa, given their superior performance on a wide range of NLP tasks, while for image we used the python module *nudenet*.

Each text encoder, image classifier and training method used in our model are detailed below.

4.1 Text Encoders

BERT (Devlin et al., 2018): pretrained model BERT-base uncased, released by the authors, was used as embedding layer, tokenizer and classifier. It consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.

RoBERTa (Liu et al., 2021): We use the RoBERTa-base model released by the authors. Like BERT, RoBERTa-base consists of 12 transformer layers, 12 self-attention heads per layer, and a hidden size of 768.

4.2 Image nudity classification

As memes are mostly done as "joke" and tend to be ironical and use a very refined and deep text-image relationship.

A similar approach to the one used by Messina et al. (2021) was applied, where one of the two modalities acts as the main one and the second intervenes to enrich the first (in our case, text will act as the main modality and image will be used just to enrich the results from the first one).

The goal of this task was to detect misogyny, and we decided to use a Not Safe For Work (NSFW) image classifier.

Nudenet is the classifier used for this task. It gives each image of our dataset an "unsafe" value from 0 (safe) to 1 (unsafe). The Neural Net for Nudity Classification is trained on 160,000 entirely auto-labelled (using classification heat maps and various other hybrid techniques) images.

A NSFW image classifier was used for two main reasons. First, because image-only classification using Convolutional Neural Networks did not reached good results using the training data. We only obtained an accuracy of 0.52. Second, because most of NSFW images are misogynous, as it could be demonstrated by using only the *Nudenet* classifier, obtaining a value of 0.83 for precision in the positive (misogynous) class.

4.3 Models

Based on Convolutional Neural Networks (Konda et al., 2019), BiLSTM Neural Networks (Zhou et al., 2016), and BERT Transformers (Devlin et al., 2018), several models have been developed: (1) BiLSTM Neural Network with RoBERTa as embedding, (2) BiLSTM Neural Network with BERT as embedding, (3) BiLSTM Neural Network with BERT as embedding and nude detection, (4) 1- Dimensional Convolutional Neural Network with BERT Tokenizer, and (5) BERT Transformer with nude detection.

In our models, RoBERTa and BERT were used with word-embedding strategies, as they have an advantage over models like Word2Vec. Each word, under Word2Vec, has a fixed representation regardless of the context within which the word appears. Nevertheless, BERT produces word representations that are dynamically informed by the words around them (Shi and Lin 2019a).

For example, given the two sentences *"The man was director of a bank in his hometown."* and *"The man went fishing by the bank of the river."*, Word2Vec would produce the same word embedding for the word "bank" in both sentences. However, BERT will create different word embedding for "bank" for each sentence.

4.3.1 BiLSTM Neural Network with RoBERTa as embedding

Long Short-Term Memory (LSTM), (Hochreiter and Schmidhuber, 1997) is a widely known recurrent neural network (RNN) architecture. We used Bidirectional LSTM (Schuster and Paliwal, 1997) models for our experiments. A Bidirectional LSTM (BiLSTM) layer processes the text both in the forward as well as backward direction and hence is known to provide better context understanding.

Introduced by Facebook, the Robustly optimized BERT approach RoBERTa, is a retraining of BERT with an improved training methodology, 1000% more data and compute power (Shi and Lin, 2019b).

The RoBERTa model used to extract the word embedding layer for the BiLSTM Neural Network was RoBERTa-base uncased.

4.3.2 BiLSTM Neural Network with BERT as embedding

For this approach, a BERT-based model was used. In particular, we implemented the BERT-base uncased model.

4.3.3 BiLSTM Neural Network with BERT as embedding + nude detection

Python's Nudenet module was used in every image of the given dataset to assign it a value of "unsafety" from 0 (safe) to 1 (unsafe). Then, if the text prediction is 1, the final prediction is set as 1, otherwise if the text prediction is 0 and the image unsafe value is greater than a threshold value, final prediction is set to 1.

The best values for this threshold were the ones with the best accuracy classifying using with nudity in images. These threshold values were 0.45 and 0.60.

4.3.4 1-Dimensional Convolutional Neural Network with BERT as embedding + nude detection

Convolutional neural networks (CNN) (Lecun et al., 1998) are originally designed to process and learn information from image features by applying convolution kernels and pooling techniques which are widely adopted for extracting stationary features; for instance, CNN has shown its adaptability in the field of text mining and NLP tasks. (Kim, 2014) reported series of experiments

with CNNs that achieve good results on sentence classification and sentiment analysis tasks.

To improve this model classification, BERT-base uncased model was used to create the word embedding layer. The nudity classification algorithm with a threshold of 0.45. was used to achieve better results.

4.3.5 BERT Transformer with nude detection

Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based machine learning technique for natural language processing (NLP) pre-training developed by Devlin et al. (2018). BERT model is pre-trained from unlabeled data extracted from the BooksCorpus with 800M words and English Wikipedia with 2,500M words.

It uses Transformer, (Vaswani et al., 2017) an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary.

Also, the nudity classification algorithm with a threshold of 0.45. was used to achieve better results.

5 Experimental setup

For each text transcription row in the corpus, a small preprocessing was applied. Every word was undercased, web pages' links, hashtags, usernames, emojis, punctuation symbols, numbers, and words with length less than 2 characters were removed, as well as English stop words using nltk.corpus module (Sarica, 2021).

For all the experiments, we split the training dataset in two parts: 80% for training and 20% for validation using a stratify approach.

The parameters used in the training phase were: batch size of 32 and 5 epochs.

6 Results

Table 1 shows a summarization of the training and test results obtained for each one of the models in the evaluation phase.

According to the official metrics (F1-score for the positive class), our results are all around 0.60 – 0.65, being the best model BiLSTM Neural Network with BERT as embedding + nude detection, with a 0.665 F1-score. With this result, we obtained the 43 place in the ranking.

As we expected, the model that obtained the best results during the training phase also obtained the best result in the evaluation phase.

7 Conclusions

In this paper, several approaches and systems descriptions on Task 5 (Subtask A) in SemEval 2022: Multimedia Automatic Misogyny Identification are detailed. The main aim was to develop various deep learning models and check how multi-modality of text and image could help achieve better classification results.

Six different models were developed and BiLSTM Neural Network with BERT as embedding + nude detection (0.45 threshold) was the definitive one. After training and analyzing each model, we achieved an F1-score of 0.665 in

Model	Training phase		Evaluation phase
	Accuracy	F1 - Score	F1 - Score
BiLSTM Neural Network with RoBERTa as embedding	0.793	0.799	0.607
BiLSTM Neural Network with BERT as embedding	0.810	0.832	0.652
BiLSTM Neural Network with BERT as embedding + nude detection threshold 0.45	**0.837**	**0.840**	**0.665**
BiLSTM Neural Network with BERT as embedding + nude detection threshold 0.6	0.835	0.838	0.663
1-D Convolutional Neural Network with BERT as embedding + nude detection	0.813	0.825	0.649
BERT Transformers + nude detection	0.806	0.812	0.639

Table 1: Results obtained with the different models

the evaluation phase for class "1". We can conclude that merging text and image classifiers improves the results in the task of misogyny detection in memes.

In future works, we intend to improve our image classifiers models. Also, we want to use other pretrained models based on transformers.

References

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. "Deep Learning for Hate Speech Detection in Tweets." Perth, Australia, International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/3041021.3054223.

Lisa Bonheme and Marek Grzes. 2020. *SESAM at SemEval-2020 Task 8: Investigating the Relationship between Image and Text in Sentiment Analysis of Memes* International Committee for Computational Linguistics. doi:10.18653/v1/2020.semeval-1.102.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." *CoRR* abs/1810.04805.

Jessica Drakett, Bridgette Rickett, Katy Day, and Kate Milnes. 2018. "Old Jokes, New Media – Online Sexism and Constructions of Gender in Internet Memes." *Feminism Psychology* 28 (1): 109-127. https://journals.sagepub.com/doi/full/10.1177/0959353517727560.

Delia Farías, Viviana Patti, and Paolo Rosso. 2016. "Irony Detection in Twitter." *ACM Transactions on Internet Technology* 16 (3): 1-24. http://dl.acm.org/citation.cfm?id=2930663

Elisabetta Fersini, Francesca Gasparini, Giulia Rizzi, Aurora Saibene, Berta Chulvi, Paolo Rosso, Alyssa Lees, and Jeffrey Sorensen. 2022. SemEval-2022 Task 5: Multimedia automatic misogyny identification. In Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). Association for Computational Linguistics.

Jean H French. 2017. "Image-Based Memes as Sentiment Predictors.". doi:10.23919/i-Society.2017.8354676.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9: 1735-1780. doi:10.1162/neco.1997.9.8.1735.

Yoon Kim. 2014. *Convolutional Neural Networks for Sentence Classification.*

Srinivas Konda, B. Rani, Varaprasad Mangu, G. Madhukar, and B. Ramana. 2019. "Convolution Neural Networks for Binary Classification." *Journal of Computational and Theoretical Nanoscience* 16: 4877-4882. doi:10.1166/jctn.2019.8399.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE* 86 (11): 2278-2324. doi:10.1109/5.726791.

Zhuang Liu, Wayne Lin, Ya Shi, and Jun Zhao. 2021. "A Robustly Optimized BERT Pre-Training Approach with Post-Training." In *Chinese Computational Linguistics*, 471-484. Cham: Springer International Publishing. https://library.biblioboard.com/viewer/822f5f9f-f7f4-11eb-926c-0a9b31268bf5.

Kate Manne. 2017. *Down Girl: The Logic of Misogyny*. New York: Oxford University Press. https://oxford.universitypressscholarship.com/10.1093/oso/9780190604981.001.0001/oso-9780190604981.

Nicola Messina, Fabrizio Falchi, Claudio Gennaro, and Giuseppe Amato. 2021. "AIMH at SemEval-2021 Task 6: Multimodal Classification using an Ensemble of Transformer Models. "Association for Computational Linguistics https://aclanthology.org/2021.semeval-1.140.

F. Pozzi, E. Fersini, E. Messina, and B. Liu. 2016. *Sentiment Analysis in Social Networks* Elsevier Science. https://books.google.es/books?id=aS2lCgAAQBAJ.

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i-Nieto. 2019. "Hate Speech in Pixels: Detection of Offensive Memes Towards Automatic Moderation." *arXiv Preprint arXiv:1910.02334.*

Niloofar Safi Samghabadi, Parth Patwa, Srinivas PYKL, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. "Aggression and Misogyny Detection using BERT: A Multi-

Task Approach." European Language Resources Association, may. https://aclanthology.org/2020.trac-1.20.

Serhad Sarica and Luo,Jianxi. 2021. "Stopwords in Technical Language Processing." *Plos One* 16 (8): 1-13. https://doi.org/10.1371/journal.pone.0254937.

Mike Schuster and Kuldip Paliwal. 1997. "Bidirectional Recurrent Neural Networks." *Signal Processing, IEEE Transactions On* 45: 2673. doi:10.1109/78.650093.

Peng Shi and Jimmy Lin. 2019a. *Simple BERT Models for Relation Extraction and Semantic Role Labeling* https://arxiv.org/abs/1904.05255.

———. 2019b. "Simple BERT Models for Relation Extraction and Semantic Role Labeling." http://arxiv.org/abs/1904.05255.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. *Attention is all You Need*.

Bencheng Wei, Jason Li, Ajay Gupta, Hafiza Umair, Atsu Vovor, and Natalie Durzynski. 2021. "Offensive Language and Hate Speech Detection with Deep Learning and Transfer Learning." . https://arxiv.org/abs/2108.03305.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. *Text Classification Improved by Integrating Bidirectional LSTM with Two-Dimensional Max Pooling*.

Anexo II

Macrotabla de resultados de la optimización de hiperparámetros.

index	name	accuracy_unsafe	f1_score_unsafe	accuracy	f1_score
93	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_3_lr_3e-05_bs_32	0.692	0.6802113097215762	0.681	0.6737570809542656
97	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_4_lr_4e-05_bs_32	0.684	0.6721651623612408	0.67	0.6626098051724983
92	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_3_lr_3e-05_bs_16	0.679	0.6681275258751974	0.665	0.6585921154142094
11	bert-base-uncased_epochs_3_lr_2e-05_bs_32	0.678	0.6674639270651316	0.67	0.6630122214234364
96	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_4_lr_4e-05_bs_16	0.68	0.6658186815711535	0.669	0.6595039043895192
98	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_4_lr_3e-05_bs_16	0.679	0.6649181868002818	0.671	0.6617922952378504
99	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_4_lr_3e-05_bs_32	0.675	0.6639920433315862	0.66	0.6532027743778049
31	bert-large-uncased_epochs_4_lr_4e-05_bs_32	0.674	0.6633330441715308	0.665	0.6583999453439913
1	bert-base-uncased_epochs_2_lr_4e-05_bs_32	0.674	0.6623175375283301	0.664	0.6562659846547315
46	roberta-base_epochs_3_lr_2e-05_bs_16	0.673	0.6616696550789386	0.662	0.6548428207306711
21	bert-large-uncased_epochs_2_lr_3e-05_bs_32	0.673	0.6608865276543407	0.661	0.6535127571696797
94	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-	0.675	0.660155241085872	0.666	0.6558177111671695

26	bert-large-uncased_epochs_3_lr_3e-05_bs_16	0.673	0.6600743262558798	0.663	0.6544833603492486
101	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_4_lr_2e-05_bs_32	0.675	0.6592487746061702	0.662	0.6516957675883333
34	bert-large-uncased_epochs_4_lr_2e-05_bs_16	0.672	0.6591726207547248	0.667	0.6581386109340208
29	bert-large-uncased_epochs_3_lr_2e-05_bs_32	0.671	0.6562715809731171	0.665	0.6549091999147059
43	roberta-base_epochs_3_lr_4e-05_bs_32	0.669	0.656193553681066	0.658	0.6499115566037736
32	bert-large-uncased_epochs_4_lr_3e-05_bs_16	0.669	0.656193553681066	0.663	0.6542593301248871
88	xlnet-base-cased_epochs_4_lr_2e-05_bs_16	0.668	0.6558358627324145	0.658	0.6501278772378516
2	bert-base-uncased_epochs_2_lr_3e-05_bs_16	0.667	0.6546642621067139	0.656	0.6480818414322251
16	bert-base-uncased_epochs_4_lr_2e-05_bs_16	0.668	0.6541666666666667	0.659	0.6492170132834625
61	distilbert-base-uncased_epochs_3_lr_4e-05_bs_32	0.668	0.6541666666666667	0.663	0.6533317697258851
24	bert-large-uncased_epochs_3_lr_4e-05_bs_16	0.668	0.6541666666666667	0.663	0.6535683996813241
74	xlnet-base-cased_epochs_2_lr_3e-05_bs_16	0.667	0.6541161733407703	0.654	0.6458169549266247
95	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_3_lr_2e-05_bs_32	0.67	0.6538500268528464	0.658	0.6473223308900264
67	distilbert-base-uncased_epochs_4_lr_4e-05_bs_32	0.667	0.6538371579302996	0.659	0.6503822726382604

38	roberta-base_epochs_2_lr_3e-05_bs_16	0.667	0.6535548085656233	0.658	0.6499115566037736
49	roberta-base_epochs_4_lr_4e-05_bs_32	0.667	0.6532691100261454	0.659	0.6503822726382604
62	distilbert-base-uncased_epochs_3_lr_3e-05_bs_16	0.668	0.652986821860994	0.662	0.6516957675883333
33	bert-large-uncased_epochs_4_lr_3e-05_bs_32	0.669	0.6526433213594974	0.666	0.6540308513327062
90	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_3_lr_4e-05_bs_16	0.665	0.6525901735908384	0.651	0.6435046860236471
30	bert-large-uncased_epochs_4_lr_4e-05_bs_16	0.667	0.6523917638769279	0.657	0.646915555171246
35	bert-large-uncased_epochs_4_lr_2e-05_bs_32	0.666	0.6523715752355339	0.656	0.6464920501816867
7	bert-base-uncased_epochs_3_lr_4e-05_bs_32	0.666	0.6517917088895283	0.662	0.6516957675883333
51	roberta-base_epochs_4_lr_3e-05_bs_32	0.665	0.6511866422184946	0.654	0.6453697174437101
40	roberta-base_epochs_2_lr_2e-05_bs_16	0.667	0.6511741185861343	0.657	0.6464176333484868
55	distilbert-base-uncased_epochs_2_lr_4e-05_bs_32	0.667	0.6511741185861343	0.658	0.6468095125971274
52	roberta-base_epochs_4_lr_2e-05_bs_16	0.666	0.6499700273736015	0.652	0.6413909086412426
60	distilbert-base-uncased_epochs_3_lr_4e-05_bs_16	0.666	0.6499700273736015	0.663	0.6515855957007523
22	bert-large-uncased_epochs_2_lr_2e-05_bs_16	0.665	0.6496984792731295	0.655	0.6443559285866705

39	roberta-base_epochs_2_lr_3e-05_bs_32	0.666	0.6493349963673627	0.654	0.6429397001519054
50	roberta-base_epochs_4_lr_3e-05_bs_16	0.665	0.6490790682473122	0.655	0.6441005802707931
13	bert-base-uncased_epochs_4_lr_4e-05_bs_32	0.661	0.6489865651937563	0.65	0.6421619145768922
14	bert-base-uncased_epochs_4_lr_3e-05_bs_16	0.663	0.6488116390283858	0.661	0.6502901353965185
3	bert-base-uncased_epochs_2_lr_3e-05_bs_32	0.662	0.6484966471987887	0.657	0.6476380965882533
8	bert-base-uncased_epochs_3_lr_3e-05_bs_16	0.664	0.6478740395135631	0.657	0.645115265113382
10	bert-base-uncased_epochs_3_lr_2e-05_bs_16	0.661	0.6475999896047194	0.659	0.6496926849463392
44	roberta-base_epochs_3_lr_3e-05_bs_16	0.665	0.6474689851631799	0.656	0.6442149768946586
5	bert-base-uncased_epochs_2_lr_2e-05_bs_32	0.661	0.647312552864103	0.651	0.6419480896545569
69	distilbert-base-uncased_epochs_4_lr_3e-05_bs_32	0.661	0.647312552864103	0.651	0.6421800972162841
89	xlnet-base-cased_epochs_4_lr_2e-05_bs_32	0.659	0.6463679080432116	0.645	0.637375826757578
25	bert-large-uncased_epochs_3_lr_4e-05_bs_32	0.648	0.6456171295792912	0.621	0.6207629768605378
66	distilbert-base-uncased_epochs_4_lr_4e-05_bs_16	0.66	0.6446250585323432	0.655	0.6443559285866705
87	xlnet-base-cased_epochs_4_lr_3e-05_bs_32	0.66	0.6443142588136834	0.644	0.6339120813126513
47	roberta-base_epochs_3_lr_2e-	0.658	0.6434513905395769	0.647	0.6378443428311134

78	xlnet-base-cased_epochs_3_lr_4e-05_bs_16	0.658	0.6431493013238953	0.647	0.6371206085682712
37	roberta-base_epochs_2_lr_4e-05_bs_32	0.658	0.6431493013238953	0.648	0.6382709350696328
91	annahaz/roberta-large-mnli-misogyny-sexism-4tweets-3e-05-0.05-singledt_epochs_3_lr_4e-05_bs_32	0.66	0.6427070197562001	0.652	0.6398052464228936
75	xlnet-base-cased_epochs_2_lr_3e-05_bs_32	0.653	0.6422942238248541	0.638	0.6317770318380633
79	xlnet-base-cased_epochs_3_lr_4e-05_bs_32	0.655	0.6422197310114603	0.643	0.6348942880898836
6	bert-base-uncased_epochs_3_lr_4e-05_bs_16	0.659	0.6421491618839534	0.654	0.6418753312135665
57	distilbert-base-uncased_epochs_2_lr_3e-05_bs_32	0.656	0.641963538565939	0.646	0.6364592361946936
59	distilbert-base-uncased_epochs_2_lr_2e-05_bs_32	0.657	0.6419530781074663	0.647	0.6366215480333814
63	distilbert-base-uncased_epochs_3_lr_3e-05_bs_32	0.655	0.6413628212791394	0.647	0.6378443428311134
70	distilbert-base-uncased_epochs_4_lr_2e-05_bs_16	0.66	0.6413562640292991	0.65	0.6366037610472457
36	roberta-base_epochs_2_lr_4e-05_bs_16	0.657	0.6413330698229356	0.645	0.6345627466058085
85	xlnet-base-cased_epochs_4_lr_4e-05_bs_32	0.657	0.6406988668920242	0.647	0.6358478401031593
27	bert-large-uncased_epochs_3_lr_3e-05_bs_32	0.655	0.6401718414127793	0.648	0.6377766069546891
53	roberta-base_epochs_4_lr_2e-05_bs_32	0.657	0.6397206416964012	0.651	0.6389073688762985
45	roberta-base_epochs_3_lr_3e-	0.655	0.6395553052757611	0.645	0.6345627466058085

9	bert-base-uncased_epochs_3_lr_3e-05_bs_32	0.653	0.6392837651706126	0.648	0.638752052545156
86	xlnet-base-cased_epochs_4_lr_3e-05_bs_16	0.652	0.6389666977902271	0.644	0.6358056265984655
68	distilbert-base-uncased_epochs_4_lr_3e-05_bs_16	0.656	0.638167444326171	0.65	0.6371763613142923
82	xlnet-base-cased_epochs_3_lr_2e-05_bs_16	0.652	0.6378003238981012	0.641	0.6316887225959482
4	bert-base-uncased_epochs_2_lr_2e-05_bs_16	0.654	0.6377175788644854	0.646	0.6344168639163248
17	bert-base-uncased_epochs_4_lr_2e-05_bs_32	0.652	0.6375000000000001	0.642	0.6321050987355925
64	distilbert-base-uncased_epochs_3_lr_2e-05_bs_16	0.655	0.6362566699456282	0.645	0.6315589002396425
84	xlnet-base-cased_epochs_4_lr_4e-05_bs_16	0.653	0.6355191331447556	0.644	0.6318038714374956
12	bert-base-uncased_epochs_4_lr_4e-05_bs_16	0.652	0.634969119226638	0.648	0.6353858073925527
42	roberta-base_epochs_3_lr_4e-05_bs_16	0.652	0.6339600890276382	0.646	0.6333141358436469
81	xlnet-base-cased_epochs_3_lr_3e-05_bs_32	0.65	0.6338529134846741	0.642	0.6310860496941517
71	distilbert-base-uncased_epochs_4_lr_2e-05_bs_32	0.652	0.6336163331311906	0.643	0.6300636974356264
23	bert-large-uncased_epochs_2_lr_2e-05_bs_32	0.655	0.632968961940477	0.65	0.6328712406015038
0	bert-base-uncased_epochs_2_lr_4e-05_bs_16	0.65	0.6328712406015038	0.64	0.6279454319966928
58	distilbert-base-uncased_epochs_2_lr_2e-05_bs_16	0.65	0.6328712406015038	0.639	0.6270503192800521

15	uncased_epochs_4_lr_3e-05_bs_32	0.649	0.6323186655367362	0.641	0.6291165224973372
65	distilbert-base-uncased_epochs_3_lr_2e-05_bs_32	0.65	0.630450849963045	0.643	0.628585785759542
76	xlnet-base-cased_epochs_2_lr_2e-05_bs_16	0.65	0.630450849963045	0.638	0.6244423695404087
83	xlnet-base-cased_epochs_3_lr_2e-05_bs_32	0.646	0.6296683753530704	0.628	0.6177181472894984
28	bert-large-uncased_epochs_3_lr_2e-05_bs_16	0.651	0.629098645944369	0.645	0.6284586055789818
54	distilbert-base-uncased_epochs_2_lr_4e-05_bs_16	0.649	0.6273564924372479	0.645	0.6287849556476448
72	xlnet-base-cased_epochs_2_lr_4e-05_bs_16	0.648	0.6272386858469305	0.644	0.6288558334271619
48	roberta-base_epochs_4_lr_4e-05_bs_16	0.644	0.6265776047261009	0.634	0.6220242152355223
77	xlnet-base-cased_epochs_2_lr_2e-05_bs_32	0.643	0.6263654146244972	0.63	0.6192538198103268
80	xlnet-base-cased_epochs_3_lr_3e-05_bs_16	0.643	0.6256978847212392	0.639	0.6267727004390848
41	roberta-base_epochs_2_lr_2e-05_bs_32	0.645	0.6234935765883003	0.638	0.6226005946646984
20	bert-large-uncased_epochs_2_lr_3e-05_bs_16	0.643	0.6228763023708088	0.637	0.6223435300580218
73	xlnet-base-cased_epochs_2_lr_4e-05_bs_32	0.638	0.6216302093785537	0.627	0.6160335337576524
56	distilbert-base-uncased_epochs_2_lr_3e-05_bs_16	0.638	0.6177805933903495	0.63	0.6155300904855127
19	bert-large-uncased_epochs_2_lr_4e-05_bs_32	0.563	0.5243666838997578	0.549	0.5171052871082102