UNIVERSIDAD AUTONOMA DE MADRID

Escuela Politécnica Superior

20|21

Trabajo fin de grado

Detección de mensajes de odio en Twitter: un estudio basado en perfiles dentro de la red social

Gloria del Valle Cano

Escuela Politécnica Superior
Universidad Autónoma de Madrid
C\Francisco Tomás y Valiente nº 11

Campus Internacional excelencia UAM CSIC+

www.uam.es

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Detección de mensajes de odio en Twitter: un estudio basado en perfiles dentro de la red social

Autora: Gloria del Valle Cano
Tutora: Lara Quijano-Sánchez

junio 2021

Gloria del Valle Cano
Detección de mensajes de odio en Twitter:
un estudio basado en perfiles dentro de la red social

Gloria del Valle Cano
C\ Francisco Tomás y Valiente Nº 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

A mi familia

El esfuerzo solo libera plenamente su recompensa
después de que una persona se niega a abandonar.

Napoleon Hill

Agradecimientos

En primer lugar me gustaría agradecer la oportunidad que me ha servido mi tutora, Lara Quijano-Sánchez, ya que sin ella no habría tenido la ocasión de investigar este problema. De igual manera, me gustaría agradecerle su paciencia y toda la confianza que ha depositado en mí, ya que además de ser un referente profesional, ha sido toda una gran coach de principio a fin.

También me gustaría agradecer la colaboración prestada por la Oficina Nacional de lucha contra los Delitos de Odio de la Secretaría de Estado de Seguridad (Ministerio del Interior). Gracias a ella, ha sido posible aumentar la base de datos de tweets para entrenar los modelos. Además, agradezco su confianza en mi trabajo, el cual pretende aportar soluciones a esta problemática social detectando de forma temprana el discurso de odio online.

De manera especial les agradezco enormemente a mis padres, a mi hermano y mis abuelos, que no hayan dudado ni solo un momento de mi valía y que me hayan apoyado en todos mi buenos y no tan buenos momentos durante estos años de trabajo y estudio. A pesar de no entender nada del campo, siempre han hecho el esfuerzo de escucharme y brindarme toda su ayuda, recursos y comprensión.

Me gustaría destacar a mis amigos y compañeros de la carrera, porque siempre nos hemos ayudado y animado a continuar. En especial a Laura, Óscar y Jimena, que también empezaron conmigo y desde entonces se han vuelto imprescindibles en mi vida. A Michael y Leyre por ser los mejores compañeros. Y a Dani, por creer en mí desde siempre.

Resumen

Las redes sociales son el vivo reflejo de la sociedad actual. Siendo una fuente prácticamente inagotable de datos, podemos utilizar los recursos que nos ofrecen para medir fenómenos que ocurren dentro de ellas, como la generación y viralización de contenido ofensivo y de odio por parte de los usuarios en Twitter. En un mundo tan polarizado como en el que vivimos hoy en día, este creciente suceso se trata de un tema preocupante para la sociedad actual.

Este trabajo propone ahondar en los esfuerzos para la detección de odio y prevención del mismo en la red social Twitter, estudiando de manera novedosa el análisis relativo a perfiles de los usuarios y su entorno, así como el propio texto de sus tweets.

En primer lugar, se realiza una mejora del algoritmo de *HaterNet*, un sistema inteligente que se diseñó en colaboración con la Oficina Nacional de Lucha Contra los Delitos de Odio de la Secretaría de Estado de Seguridad de España (Ministerio del Interior), que es capaz de identificar y monitorear la evolución del discurso de odio en la red social bajo el enfoque de una red neuronal LTSM + MLP. Para ello se construye un modelo basado en BERT, que ha sido probado tanto con el dataset público en español de *HaterNet*, como con otros de la literatura actual en español y en inglés, obteniendo resultados que reflejan una visible mejora, principalmente en español.

Paralelamente, se ha creado una base de datos de perfiles de usuarios en Twitter en forma de grafo relacional que ha servido para extrapolar características textuales y de centralidad. En cuanto a esta parte, se ha probado por separado con distintos modelos de Machine Learning clásicos y otros de Deep Learning, obteniendo evidencias de que que estas características son considerables a la hora de detectar *haters*.

Tras esto se construye un modelo final con el objetivo de obtener un sistema inteligente que es capaz de analizar características más allá de las que sus textos puedan poseer.

Por último, se ha comprobado que la aportación de las características de los usuarios es información muy reveladora para la comprensión de la viralidad del odio en la red, por lo que este trabajo ha permitido abrir el campo de estudio y romper las fronteras textuales dando pie a futuras investigaciones en modelos combinados desde una perspectiva diacrónica y dinámica.

Palabras clave

PLN, Discurso de odio, Twitter, Deep Learning, Análisis de la red social, BETO, Modelado de temas

Abstract

Social networks have become an online reflection of our social interactions. Being an endless and inexhaustible source of data, we can use the resources they offer to analyse phenomena that occur within them, such as the generation and viralization of offensive and hateful content by users on Twitter. In a world as polarized as the one we live in today, the escalating nature of this behaviour poses a matter of concern for present-day society.

This project comprises both an in-depth study of the efforts and techniques used so far for the detection and prevention of hateful content on the popular social network Twitter, as well as a proposal for a novel approach for feature analysis based on user profiles, related social environment and generated tweets.

First of all, the entering point is an improvement on the performance of the HaterNet algorithm, an intelligent system that was designed in collaboration with the Spanish National Office Against Hate Crimes of the Spanish State Secretariat for Security (Ministry of Interior), which is able to identify and monitor the evolution of hate speech in the social network under the approach of a neural network LTSM + MLP. For this purpose, a model based on BERT has been built and tested with the HaterNet's Spanish public dataset and several others from the current literature in both Spanish and English, obtaining results that reflect an improvement, particularly remarkable in the Spanish case.

At the same time, a user database has been created in the form of a relational graph that has been used to extrapolate textual and centrality characteristics. Regarding this part, it has been tested separately with different classical Machine Learning models and other Deep Learning models, evidentiating the potential importance of this features when detecting haters.

After this, a final model is developed as a proposal of an intelligent system that is able to analyze features beyond those that intrinsically lie in the text.

Finally, it has been found that the contribution of user characteristics plays a significant part on gaining a deeper understanding of hate virality in the network, so that this work has allowed to open the field of study and break the textual boundaries giving rise to future research in combined models from a diachronic and dynamic perspective.

Keywords

NLP, Hate Speech, Twitter, Deep Learning, Social Network Analysis, BERT, Topic Modeling

Índice

Listas

Lista de cuadros

Lista de figuras

Lista de tablas

1 Introducción

En este Trabajo de Fin de Grado se presenta una nueva manera de abordar la detección del discurso de odio en las redes sociales. Para afrontar este problema, se crean diferentes modelos basados en técnicas Deep Learning que servirán para enfrentarse al discurso de odio (*hate speech*) en la red social *Twitter*. En el primero de ellos, se mejora el algoritmo base de *HaterNet* [1], una red que combina un Long Short-Term Memory (LSTM) con un MultiLayer Perceptron (MLP) que se desarrolló en colaboración con la Oficina Nacional de Lucha Contra los Delitos de Odio (Secretaría de Estado de Seguridad, Ministerio del Interior de España). Para ello se construye un modelo basado en Bidirectional Encoder Representations from Transformers (BERT) [2] que analiza únicamente el texto de los tweets para su clasificación en odio o no odio. A continuación, tras una recopilación de perfiles de usuarios en la red social y construcción de sus características, se analiza la importancia de las mismas a la hora de detectar si es un perfil de *hater* o no *hater*. Por último, se unifica el texto y las características numéricas y categóricas de los usuarios para la entrada de un modelo BERT + MLP que presenta una mejoría en la clasificación odio o no odio. Además, se exponen los experimentos realizados, así como el análisis y las conclusiones de los mismos.

1.1. Motivación del proyecto

La motivación de este trabajo es múltiple. La lucha contra distintas conductas discriminatorias basadas en prejuicios como misoginia, xenofobia, cyberbullying y racismo, entre otros, es sin lugar a dudas una de las primeras motivaciones a abordar en este proyecto. Aquellos mensajes que tienen como objetivo promover y alimentar un dogma contra determinadas personas o un grupo de individuos, es uno de los problemas más graves de la era digital, un fenómeno que se alimenta del odio de otros y se propaga extensiblemente entre los usuarios como si de una enfermedad se tratase. Este suceso se puede encontrar en todo tipo de red social, especialmente en Twitter, la red social donde los usuarios expresan libremente sus opiniones sin ningún tipo de censura o filtro, encuentran la facilidad para transmitir mensajes de carácter ofensivo a través de la creación de numerosas cuentas de forma anónima. Y lo que es peor, las personas afectadas por este acoso también suelen sufrirlo en la vida real. Es por ello que este fenómeno supone un peligro y una violación de los derechos humanos para aquellas

personas que se pueden encontrar en un estado de vulnerabilidad provocado por esta discriminación individual y colectiva.

Enfrentarse al problema de la detección del discurso de odio en español es otra de las motivaciones del proyecto. Se han realizado numerosos estudios sobre este tema en inglés [3] y desde la perspectiva plurilingüe [4], sin embargo, el español es uno de los idiomas más olvidados en este problema [1, 4, 5], y a su vez uno de los más hablados en todo el mundo. Se trata de la segunda lengua materna tras el chino mandarín, y la cuarta lengua en hablantes tras el chino mandarín, inglés e hindi, con 586 millones de hablantes en todo el mundo. [1]

Es por ello que afrontar el reto en español supone un desafío en el campo de la investigación. Pocos son los datasets y estudios existentes en el idioma, pero la necesidad de trabajar en ello es cada vez mayor dada la relevancia del mismo a nivel sociocultural. En los últimos años, el número de delitos de odio en España continúa en una tendencia alcista, según el *Informe 2019 sobre la evolución de los delitos de odio en España* [6], se registran 1.706 hechos, un 6,8 % más que en 2018, principalmente de racismo y xenofobia. Asimismo, son la discriminación, las amenazas e injurias los hechos delictivos que más se repiten, siendo Internet (54,9 %) y las redes sociales (17,2 %) los medios más empleados para la ejecución de los mismos.

Finalmente, introducirse en el mundo del Natural Language Processing (NLP) se trata de una oportunidad idónea para comenzar una carrera profesional en la Ciencia de Datos, la cual está cambiando el mundo a pasos agigantados.

1.2. Objetivos

Tras una amplia revisión del estado del arte sobre la detección del discurso de odio se ha detectado que más de 32 trabajos, tanto estudios como modelos, son en Twitter [3], dentro de los cuales se observan 10 para la detección de odio en inglés, 3 en español y ninguno que combine perfiles y relaciones entre usuarios (ver Capítulo 2), se ha detectado una oportunidad científica para la construcción de un modelo multimodal en español. Es por ello que el objetivo principal de este proyecto es la creación de un modelo de buen rendimiento que sirva como base sólida para la detección de odio en español, mejorando los algoritmos creados anteriormente [1, 4, 5].

Es preciso señalar que durante dicha revisión se observa la existencia de dos estudios [7, 8] de los que se pueden extraer conclusiones relevantes a la hora de detectar usuarios *haters*. Por consiguiente, el segundo objetivo es investigar cómo afectan las características textuales y numéricas de los perfiles de los usuarios de la red social en la detección de odio y la relación con la difusión dentro de ella. Hasta ahora no se han presentado estudios que engloben ambos temas en un modelo único, por lo que la construcción del mismo puede ofrecer una enfoque diferente que anime a la construcción de

[1] Ver más en Wikipedia: Idioma español

modelos multimodales para la detección de odio en más idiomas.

Por último, se pretende motivar a los investigadores a seguir trabajando en proyectos NLP en el idioma español, ya que no solo se necesita de un modelo eficiente y automático, sino de una suficiente cantidad de datos etiquetados para poder entrenarlos.

En definitiva, la principal aportación de este trabajo es la creación de una metodología que extrae características determinadas de los perfiles de los usuarios dentro de la red social Twitter, con el objetivo de modelar atributos junto con el propio texto del tweet, lo que resultará clave en la detección de odio dentro de la red social. Como se verá a lo largo de esta memoria el algoritmo presentado en este trabajo que combina los dos tipos de atributos mejora en un 18 % el mejor algoritmo base que utiliza únicamente información textual.

1.3. Estructura del trabajo

- **Capítulo 1. Introducción.** Presentación del proyecto, intereses y motivaciones, objetivos y estructura del mismo.
- **Capítulo 2. Estado del Arte.** Elaboración de un estado del arte sobre la detección del discurso de odio.
- **Capítulo 3. Diseño e implementación.** Definición del alcance del proyecto. Explicación de los modelos creados para la clasificación de odio. Justificación de los diseños e ilustración de los mismos.
- **Capítulo 4. Pruebas y resultados.** Detalle de los experimentos realizados para probar cada uno de los modelos. Explicación de los resultados y comparativa con el estado del arte actual.
- **Capítulo 5. Conclusiones y trabajo futuro.** Conclusiones obtenidas tras la realización del proyecto y propuestas de modificación de cara a una ampliación y mejora del trabajo en un futuro.

2

Estado del Arte

En este capítulo se expone el estado del arte del problema de la detección del discurso de odio. Para ello se ha realizado un amplio estudio bajo un Protocolo de Revisión Bibliográfica, el cual se puede consultar en el Anexo A.

2.1. El problema del Hate Speech Detection

El desafío de contrarrestar el discurso del odio se remonta años atrás. Sin embargo, se trata de un dilema actual único, debido a un mayor uso de la tecnología y la escala de la información que se genera a diario [9]. Tanto es así que todos los estudios encontrados datan desde 2015 [3], siendo a partir de entonces cuando el crecimiento del mismo ha sido más notable. El campo del NLP , en particular, el Sentiment Analysis, es testigo de un creciente interés en tareas relacionadas con cuestiones sociales y éticas, alentado por el compromiso mundial de luchar contra la violencia, el extremismo y las ya conocidas noticias falsas (fake news), entre otras cuestiones que ocurren en el escenario online, y fuera de él. Estos sucesos son un ejemplo claro y presente de cómo una herramienta potente como es una red social puede verse gravemente afectada por el contenido de odio y por la dificultad de la distinción entre la libertad de expresión y el discurso de odio [10].

Existe un problema principal con el discurso del odio que dificulta su clasificación: la subjetividad. En muchos casos lo que es y no es discurso de odio está abierto a interpretación, además del dominio y del contexto, por lo que la magnitud y el alcance del problema depende de cada proyecto. Esto implica la necesidad definir a priori qué se entiende como discurso de odio, lo que a niveles prácticos genera una variedad de etiquetados en los datasets y cantidad de los mismos.

Tras una revisión terminológica, se detallarán los datasets y los modelos utilizados hasta la fecha tras una recogida previa de la información.

2.1.1. Definiciones

Si bien no existe un consenso sobre qué es discurso de odio debido a la dificultad de diferenciarlo de la libertad de expresión, se han definido diversos alcances a lo largo de los últimos años. Por ejemplo, en la Figura 2.1 se ilustran los conceptos asociados, según la investigación realizada en Poletto *et al.* [3].

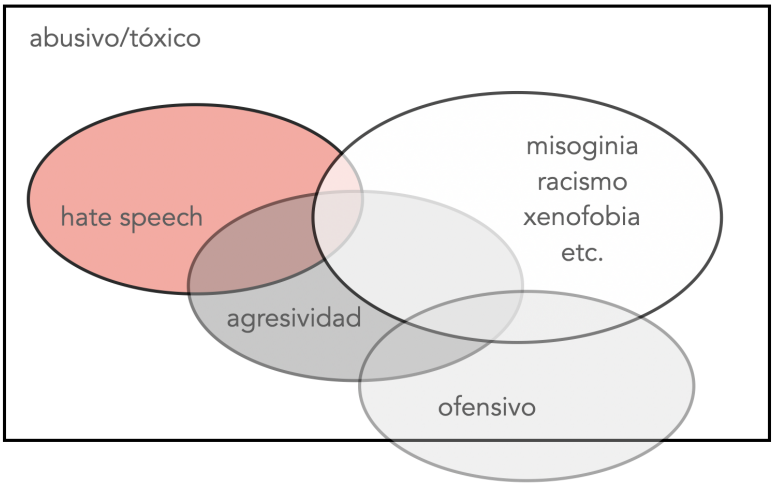


Figura 2.1: Se ilustra la diferencia entre conceptos relacionados con el discurso del odio según la explicación realizada en Poletto *et al.* [3]. Se observa la cercanía entre conceptos como el discurso de odio y la agresividad, y esta última con el contenido ofensivo, si bien el discurso de odio va más allá del propio contenido ofensivo. Conceptos derivados como la misoginia o el racismo engloban todos los términos asociados. En general, todo se considera abusivo o tóxico, aunque es cierto que algunos comentarios ofensivos están considerados menos perjudiciales.

De esta manera se determina la diferencia entre contenido ofensivo y discurso de odio, pese a que este primero puede estar fuertemente conectado e influir en la generación de contenido de odio, se basa en la mera opinión y no infringe ninguna ley que constituya una ofensa criminal. No obstante, es también preocupante en términos de tolerancia, ética social y respeto por los derechos humanos, además de presentar una línea muy fina entre los mismos que corresponde a la agresividad.

Bajo estas definiciones se pueden presentar diferentes enfoques en el campo de la investigación, algunos de forma lingüística, otros en la intención del agresor, y otros en la intención de la víctima [11–13].

2.1.2. Datasets

Como se ha comentado previamente, la tarea de la detección de odio se encuentra en una etapa muy reciente de desarrollo, pero está comenzando a ser cada vez más popular en NLP [3].

En cuanto al tipo de clasificación de los mensajes (e.g. tweets) se pueden destacar tres grupos. El

primero de ellos es la clasificación binaria, la cual se comprende entre valores excluyentes, es decir, *hate* o *non-hate* [14]. El segundo se basa en tres o más valores mutuamente o no excluyentes teniendo en cuenta, por ejemplo, la división de odio fuerte, ofensivo, agresivo, sexista o racista [12,15]. La última se basa en la anotación combinada, esto es, una división primera sobre lenguaje abusivo o no abusivo a la que se le suceden clases más concretas como discurso de odio, despectivo o profano [16].

También es destacable el origen del etiquetado, existiendo opciones principales como anotaciones por expertos [1] (ya sean jueces o desarrolladores expertos en el tema), etiquetados por voluntarios no expertos [3] o en una plataforma crowdsourcing [14,17], e incluso utilizando un clasificador automático para asignar estas etiquetas [8]. Esta estrategia cobra relevancia ya que es importante que los datos estén debidamente etiquetados. Y aún así, siempre estarán compuestos por una serie de juicios humanos, ya sea por su trasfondo cultural o su percepción individual, por lo que hay que sumar un sesgo de base implícito. Eso no significa que los investigadores no puedan usarlos, pero es necesario ser conscientes sobre lo que realmente representan en la fase de resultados.

Los diferentes datasets encontrados durante el estudio bibliográfico se pueden contemplar desde diferentes escenarios, no obstante se agruparán primeramente por idiomas dado el interés que nos ocupa en el presente estudio. A continuación se precisa en cada uno de ellos la estrategia de clasificación. Es relevante destacar que aunque existen algunos datasets recogidos de otras redes sociales como Facebook [18,19], son en mayor medida los que se recogen de Twitter. Existen en total al menos 32 datasets públicos relacionados de diferentes lenguas, en su mayoría inglés, que provengan de Twitter [3]. Estos últimos serán los comentados en este trabajo.

Datasets en inglés

La gran mayoría de los datasets existentes, como se podría esperar, son en inglés. En este estudio se recogen los más utilizados o citados y los que tienen un número considerable de tweets. Los datasets seleccionados se recogen en la Tabla 2.1.

Se refleja una amplia variabilidad de aproximaciones y etiquetados. Después de todo, la elección del dataset dependerá del problema y de los objetivos de cada proyecto.

Datasets en español

La variedad de datasets es menor. Tan solo existen dos datasets en español, si bien recientemente se publicó la tarea Profiling Hate Speech Speaders on Twitter, la cual ofrecía en ficheros agrupados 200 tweets por cada 200 autores, siendo estos últimos el objetivo a etiquetar. Dado que los tweets no estaban etiquetados, se tuvo que descartar el dataset por contextualizar un problema diferente al presente trabajo.

Nombre	Clases	Etiquetas	Total	Año	Puntuación	Etiquetado
Waseem and Hovy [12]	4	racist, sexist, both, normal	16907	2016	0.85	expertos y crowdsourcing
Davidson *et al.* [17]	3	hateful, offensive (but not hateful), neither	24802	2017	0.92	crowdsourcing
Founta *et al.* [11]	7	offensive, abusive, hateful speech, aggressive, cyberbullying, spam, normal	80000	2018	–	crowdsourcing
Basile *et al.* [14] [HatEval (en)]	2	hate, non-hate	13000	2019	0.83	crowdsourcing

Tabla 2.1: Se muestran los diferentes datasets más relevantes encontrados en inglés, etiquetados con diferentes clases y estrategias.

Los datasets en español provienen de *HaterNet* [1] (Tabla 2.2) y *HatEval* en español (es) [14] (Tabla 2.3). Ambos, publicados en 2019 con apenas tres meses de diferencia (primero HatEval y después HaterNet), presentan una clasificación binaria en *hate* o *non-hate* con un número similar de tweets.

El dataset de *HaterNet* proporciona un conjunto de datos completo y etiquetado. Contiene 6000 tweets etiquetados procedentes de 2 millones de tweets descargados entre febrero y diciembre del 2017 sin etiquetar. El dataset fue filtrado aplicando seis diccionarios de discurso de odio y uno de insultos. Tras ello, fue etiquetado a mano por cuatro expertos procedentes de diferentes escenarios, utilizando el voto por mayoría [1]. Según la Kappa de Fleiss, este dataset tiene una puntuación de 0.588, lo que quiere decir que se encuentra entre un valor de acuerdo moderado bastante fiable. Los detalles del dataset se encuentran presentados en la Tabla 2.2.

Clase	Descargados	Seleccionados	Etiquetados
0	–	–	4433
1	–	–	1567
Total	2 millones	8710	**6000**

Tabla 2.2: Distribución del dataset de HaterNet, donde se refleja que un 73.89 % del corpus pertenece al etiquetado *non-hate* (0) y un 26.11 % de *hate* (1).

Lo bueno de este dataset es que no solo contiene el texto del tweet y la etiqueta asociada, sino también el identificador del tweet, lo que permite su búsqueda a día de hoy a través de la API de Twitter.

Este factor es muy destacable de cara a la posibilidad de descargar información necesaria adicional para el presente estudio, lo que es determinante a la hora de seleccionar un dataset de referencia.

El segundo dataset fue propuesto por los organizadores de SemEval 2019: task 5: HatEval [14], tarea que consistía en la detección de discurso de odio contra dos objetivos: mujeres e inmigrantes. Los datos fueron extraídos principalmente desde julio hasta noviembre de 2017 y desde julio hasta septiembre de 2018 utilizando la plataforma crowdsourcing Figure Eight (F8). Este dataset reportó suficiente fiabilidad, con una puntuación de 0.89, lo que indica un valor de acuerdo casi perfecto. Posteriormente se reforzó el etiquetado por dos expertos, hablantes del castellano, y utilizando finalmente voto por mayoría, aunque no consta puntuación final sobre su fiabilidad.

Clase	Train	Test	Total
0	2921	940	3861
1	2079	660	2739
Total	5000	1600	**6600**

Tabla 2.3: Distribución del dataset de HatEval (es), donde se refleja que un 58.5 % del corpus pertenece al etiquetado *non-hate* (0) y un 41.5 % de *hate* (1).

Este dataset, ver en Tabla 2.3, contiene el identificador indexado propio (no referente al original de Twitter), el texto en sí del tweet y la etiqueta asociada al mismo. En la tarea, además, se dividieron los datos en tres *sets* (Stratified Sampling): entrenamiento, prueba y validación, juntando el entrenamiento y validación en *Train* y apartando los datos de prueba en *Test*.

2.1.3. Estudios anteriores

Estudios en inglés

Los primeros estudios realizados sobre la detección de odio en redes sociales constan desde 2015 [20]. Estos utilizaron estrategias híbridas basadas en recursos léxicos y modelos Machine Learning. Estos fueron principalmente Bag of Words (BOW) , TF-IDF , Global Vectors for Word Representation (GloVe) , Part-Of-Speech (POS) , Logistic Regression (LR) , Support Vector Machine (SVM) , Naive Bayes (NB) y k – Nearest Neighbor (kNN) entre otros [12,17,21–23]. Fue en 2017 cuando se introdujo el primer método basado en redes neuronales [23] y se probó la eficacia de los mismos por encima de los anteriores, si bien en algunos casos los modelos tradicionales ofrecían buenos resultados. En las Tablas 2.4 y 2.5 se reflejan una selección de los estudios encontrados en la literatura que proponen nuevos modelos para la tarea de la detección del discurso de odio.

Además de la creación de modelos para combatir este problema, se han realizado estudios que reflejan conclusiones reveladoras para la toma de decisiones en la creación de modelos futuros. Por una parte, los organismos gubernamentales y diferentes tipos de empresas necesitan estudios de

Dataset	Paper	Fecha	Estrategias	Modelos	Validación	Mejor F1-score
951,736 comentarios (Yahoo)	Djuric *et al.* [20]	May. 2015	BOW, TF, TF-IDF, **paragraph2vec embeddings**	**LR**	–	–
Recolección de tweets (propio)	Zia *et al.* [21]	Nov. 2016	unigrams, TF-IDF, retweets, favs, autenticidad de la página	**SVM**, NB, kNN	–	0.971
Waseem [24]	Waseem [24]	Ene. 2016	char n-grams, skip n-grams, word n-grams, tweet length, género del autor, POS, clusters	**LR**	–	0.912
Waseem and Hovy [12]	Waseem and Hovy [12]	Jun. 2016	género del author, longitud de los tweets, longitud de la descripción, localización, char n-grams, word n-grams	**LR**	–	0.7393
6502 comentarios de Facebook [18]	Del Vigna *et al.* [18]	Ene.2017	POS, sentiment analysis, word2vec, CBOW, n-grams, word polarity	SVM, **LSTM**	–	0.731
Waseem and Hovy [12]	Badjatiya *et al.* [23]	Abr. 2017	char n-grams, **random embeddings**, GloVe	LR, RF SVM, **GBDT**, DNN, CNN, **LSTM**	–	0.930

Continúa en la siguiente página

Tabla 2.4: Resumen de los estudios encontrados en la literatura en inglés, con sus datasets correspondientes, puntuación, modelos y estrategias utilizadas.

Empieza en la anterior página

Dataset	Paper	Fecha	Estrategias	Modelos	Validación	Mejor F1-score
Davidson *et. al* [17]	Davidson *et al.* [17]	May. 2017	n-grams, TF-IDF, POS, readability sentiment, URLs	**LR**, NB, DT, RF, SVM	–	0.900
Waseem [24], Waseem and Hovy [12]	Park and Fung [25]	Jun. 2017	word embeddings, random embeddings, char n-grams	CharCNN, WordCNN, **HybridCNN**	–	0.8270
6655 tweets de Waseem [24]	Gambäck and Sikdar [26]	Ago. 2017	word embeddings, random embeddings, char n-grams	**CNN**	–	0.7829
WZ, WZ-S.amt, WZ-S.exp, WS.gb, WZ.pj, DT, RM, [27]	Zhang *et al.* [28]	Oct. 2018	n-grams, POS, TF-IDF, menciones, hashtags, faltas ortográficas y erratas, emojis, **word embeddings**	**CNN+sCNN**, CNN+GRU	–	0.820–0.940
5143 comentarios de Twitter y Facebook [19]	Salminen *et al.* [19]	Mar. 2019	n-grams, TFIDF, word2vec, doc2vec	LR, DT, RF, Adaboost, **SVM**	–	0.96
Waseem and Hovy [12], Davidson [17]	Mozafari *et al.* [29]	Oct. 2019	–	BERT+LSTM, BERT, BERT+NLL, **BERT+CNN**	80–10–10	0.880, 0.920
Davidson [17]	Kovács *et al.* [30]	Abr. 2021	–	CNN-LSTM, **RoBERTa+FastText**	5 k-Fold	0.798

Tabla 2.5: Resumen de los estudios encontrados en la literatura en inglés, con sus datasets correspondientes, puntuación, modelos y estrategias utilizadas.

conocimiento para diseñar mecanismos de detección de odio con el fin de evitar y controlar estos incidentes de manera proactiva. Desde este punto de vista, una técnica importante para el análisis de datos es crear y mantener un entorno para recopilar datos, lo que permite a los analistas observar la evolución del discurso de odio y monitorearlo. Una de estas investigaciones es la realizada por Olteanu *et al.* [31], que además sugiere que el contenido generado por el usuario está determinado por muchos factores, como las posibilidades y normas de la plataforma, así como elementos exógenos, en particular eventos que ocurren en la actualidad, con el impacto de los medios de comunicación. Entre otros hallazgos, observan que la violencia extremista tiende a conducir a un aumento del discurso de odio en línea, particularmente en mensajes que abogan directamente por la violencia. En el estudio de Oliveira *et al.* [32] recolectan, procesan, buscan, analizan y visualizan una gran cantidad de tweets en varios idiomas, en tiempo real, con capacidades de análisis de sentimiento, a un bajo costo de implementación y operación.

Otra alternativa respaldada por varias organizaciones es contrarrestar ese discurso de odio con más discurso, como la de Mathew *et al.* [7]. En esta investigación analizaron el discurso de odio y los las respuestas de usuarios que contrarrestan estas opiniones, también conocido como contra discurso (counter speech). Si bien el enfoque de clasificación es sobre el odio o sobre counter speech, realizan varios análisis léxicos, lingüísticos y psicolingüísticos en estas cuentas de usuario, encontrando que los tweets de odio de cuentas verificadas tienen mucha más viralidad en comparación con un tweet de una cuenta no verificada. Además, las cuentas de odio parecen usar más palabras sobre emociones negativas. Mientras tanto, los usuarios que contrarrestan el odio usan más palabras relacionadas con el gobierno o las leyes. Como continuación del trabajo, Ribeiro *et al.* [8] muestran que los usuarios de odio o suspendidos difieren de los normales o activos en términos de sus patrones de actividad, uso de palabras y estructura de red.

Si bien los enfoques supervisados logran un rendimiento casi perfecto, esto es solo dentro de conjuntos de datos específicos. Por este motivo existen diversos estudios que se centran en la investigación de errores y sesgos tanto en los conjuntos de datos como en las estrategias tomadas para la detección del odio. En la investigación de Arango *et al.* [33], se analizan problemas metodológicos, así como un sesgo importante en el conjunto de datos. Las dificultades que se encuentran están relacionados principalmente con problemas de muestreo y sobreajuste de datos. Además, Badjatiya *et al.* [34] encuentran problemas con la interpretación de las palabras, por lo tanto, un sesgo intrínseco en ellas. Para proponer una manera de reducirlo, por ejemplo, en modelos basados en redes neuronales como Convolutional Neural Network (CNN) , utilizan técnicas de reemplazo en el texto, como Named Entity Recognition (NER) tags, POS tags o Centroid Embedding. MacAvaney *et al.* [35] encuentran dificultades como las sutilezas en el lenguaje, las diferentes definiciones sobre lo que constituye el discurso de odio y las limitaciones de la disponibilidad de datos para la capacitación y prueba de los mismos. Además, proponen un enfoque de multi-view SVM que reduce los problemas de interpretabilidad en las redes neuronales. También se menciona la necesidad de automatización de los modelos y relación

con el mundo real. Taieb *et al.* [36] discuten otras circunstancias que complican la tarea como el proceso de construcción y anotación de los conjuntos de datos, así como las métricas de evaluación de los modelos. En el artículo de Muneer *et al.* [37] se discute la necesidad de la detección del ciberacoso sin la implicación de las víctimas. Además se defiende la utilidad de la LR , el Stochastic Gradient Descent (SGD) y las SVM .

Hasta entrados en el año 2021, ya habiendo comenzado la realización de este trabajo, no aparecen estudios que incluyan más información en los modelos que la de sus características textuales. En el estudio de Vijayaraghavan *et al.* [38] se muestra que el contexto social y cultural mejora el rendimiento de manera significativa en comparación con los modelos basados en puramente texto, si bien no se destacan más que la procedencia geográfica de los tweets o la relación entre usuarios. Otro estudio interesante es el propuesto por Perifanos *et al.* [39], que ofrece las ideas de incluir información visual procedente de las imágenes que comparten en un entorno de aprendizaje multimodal, lo que puede mejorar la precisión de los modelos.

Tras la realización de este estudio se puede concluir en que no existe ningún tipo de publicación que incluya atributos de los usuarios basado en perfiles con características textuales en un modelo único y multimodal, por lo que se ofrece, de esta forma, la principal propuesta de este trabajo.

Estudios en español

A continuación se reflejan en la Tabla 2.6 las publicaciones realizadas para el español.

Dataset	Paper	Fecha	Mejor modelo	Validación	F1-score
HaterNet [1] (Tabla 2.2)	Pereira *et. al* [1]	Oct. 2019	LSTM+MLP	LOOCV	0.611
HaterNet [1] (Tabla 2.2) *HatEval (es)* [14] (Tabla 2.3)	Aluru *et. al* [4]	Abr. 2020	mBERT	70-20-10	0.733 0.734
HaterNet [1] (Tabla 2.2) *HatEval (es)* [14] (Tabla 2.3)	Plaza-del-Arco *et. al* [5]	Mar. 2021	BETO	10 k-Fold	0.772 0.776

Tabla 2.6: Resumen de las principales características de los modelos realizados para el idioma español en la literatura. Se indican los datasets utilizados para cada uno de los estudios, así como los modelos utilizados, con su validación y puntuación correspondientes.

En el estudio realizado por Pereira *et. al* [1] se contrarresta el odio desde la perspectiva de la víctima. Se trata del primer sistema inteligente que monitoriza y visualiza el odio en la red social. Tras el estudio de varias aproximaciones, se utiliza el mejor modelo basado en LSTM + MLP que realiza un preprocesamiento del texto de entrada enriquecido por la técnica TF-IDF. En este preprocesamiento se dividen los datos de entrada en palabras, emojis y embeddings del tweet, siendo estos últimos obtenidos por la técnica word2vec. Además, introducen el dataset comentado previamente (ver en Tabla 2.2). Esta publicación no solo ofrece un detector del discurso de odio en sí, sino también una metodología

diferente en la cual se combina la clasificación textual con técnicas de monitorización e identificación de las regiones, emisores y receptores del odio. Para cuando realizaron el estudio, no se había contemplado la posibilidad de incluir transformadores para la detección del odio, ni tampoco existía una primera versión de BERT en español hasta entrados en 2020 [40] (BETO), pero se había probado en diferentes estudios el buen funcionamiento de modelos basados Deep Learning [23], muchos de ellos combinados con preprocesamientos basados en word2vec embeddings, n-grams y diferentes recursos léxicos.

Referente a la publicación de Aluru *et. al* [4], se prueba en diferentes idiomas la eficacia de cuatro modelos: MUSE + CNN - Gated Recurrent Unit (GRU) , Translation + BERT , LASER + LR y Multilingual BERT (mBERT) , con una amplia optimización de hiperparámetros. Tras ello se comprueba la eficacia de utilizar mBERT en numerosos idiomas, a pesar de no ser el más apropiado en cuestión para cada uno de ellos. Los resultados de realizar una traducción previa a BERT no son del todo lejanos a los resultados con mBERT, pero BERT está entrenado para el inglés y la precisión depende mucho de la calidad de la traducción. Por otro lado, se observa también que la utilización de transformadores es mucho más útil para datasets con suficiente información, mientras que para los corpus más pequeños los resultados con LASER + LR pueden ser más prometedores.

Es en este año cuando se ha realizado la primera publicación oficial con BETO, según Plaza-del-Arco *et. al* [5], se alcanza la mejor puntuación hasta la fecha, demostrando que la utilización de un BERT en español presenta una mejor adecuación para el idioma. También se contemplan las posibilidades con modelos Machine Learning tradicionales y otros en Deep Learning, además de la comparativa con XLM y mBERT . Se comprueba que las redes neuronales CNN , LSTM y Bidirectional Long Short-Term Memory (BiLSTM) son mejores entendiendo el contexto que sus modelos previos, pero los transformadores son los más adecuados hasta el momento.

Es por ello que en la propuesta de este proyecto se busca a modo de algoritmo base, una solución basada en BERT equiparable a los demás algoritmos que se apoyan meramente en la clasificación textual. De esta manera podemos realizar una amplia comparativa con el estado del arte actual, reportando fielmente la mejora que supone la opción multimodal que se propone.

2.1.4. Otros idiomas

Aunque, como se ha comentado previamente, el inglés es el idioma sobre el que más se trabaja, existen diferentes estudios basados en otros idiomas que aportan ideas de valor para la causa.

Por ejemplo, Battistelli *et al.* [41] enfatizan la importancia del contexto para la detección del odio en francés, o en [42] se crea la plataforma *Contro l'odio* para monitorear el discurso de odio contra los inmigrantes en la esfera italiana de Twitter, explorando la robustez temporal de AlBERTo. En este último estudio, además, se afirma que el modelo es muy sensible a la distancia temporal del conjunto

de datos, pero con una ventana de tiempo adecuada, el rendimiento aumenta, ya que el discurso del odio es muy sensible a determinados eventos sociales. En el estudio de Garland *et al.* [43] también se utiliza la estrategia del counter speech, esta vez en alemán, si bien se desconoce su eficacia real para frenar la propagación del odio y es difícil de cuantificar. Se hace una apreciación en la falta de grandes conjuntos de datos etiquetados para entrenar a los modelos. Combinando una variedad de *embeddings* de párrafos con funciones de regresión logística regularizadas en un corpus de millones de tweets relevantes, se logran puntuaciones macro F1 en conjuntos de pruebas equilibrados de muestra que van desde 0.76 a 0.97. Previamente a este estudio, analizaron que el *bullying* es más propenso a ser viral y efectivo [44].

Sreelakshmi *et al.* [45] resaltan la presencia del inglés en otros idiomas, como el artículo de donde se habla de tweets mixtos en hindú e inglés. Se observa que una combinación de FastText con un clasificador SVM + Radial Basis Function (RBF) dieron el mejor resultado (F1 de 0.858) para datos de código mixto.

Finalmente, Aluru *et al.* [4], como se ha mencionado previamente en la Subsección 2.1.3, reflejan aportaciones interesantes para el problema monolingüe y plurilingüe, aportando resultados interesantes también para el árabe, inglés, alemán, indonesio, italiano, polaco, portugués y francés.

3

Diseño e implementación

En este capítulo se detalla el alcance del proyecto y se introduce el diseño de los modelos creados para la detección del discurso de odio en la red social Twitter.

3.1. Definición del proyecto

Para entender el marco social y jurídico de este proyecto se debe perfilar lo que se ha de entender por discurso de odio. En este trabajo se entenderá el concepto en el marco de un delito de odio según lo define la Organización para la Seguridad y la Cooperación en Europa (OSCE) [2]:

> *Un delito de odio es toda infracción penal, incluidas las cometidas contra las personas o la propiedad, dónde el bien jurídico protegido, se elige por su, real o percibida, conexión, simpatía, filiación, apoyo o pertenencia a un grupo. Un grupo se basa en una característica común de sus miembros, como su "raza", real o percibida, el origen nacional o étnico, el lenguaje, el color, la religión, la edad, la discapacidad, la orientación sexual, u otro factor similar.*

De esta manera se define el alcance del proyecto como un problema de clasificación binaria, en *hate* o *non-hate*.

En base al estudio realizado en el anterior Capítulo 2, se plantean diversas hipótesis que servirán para enfrentarse a la tarea y contrastar en la fase de resultados.

H-1.– Los modelos basados en transformadores como BERT presentan una buena aproximación para el problema de la detección del discurso de odio, ya que se requiere para el mismo de una comprensión contextual de los tweets.

(Dadas las evidencias que los investigadores han aportado de que el aprendizaje por transferencia ofrece resultados más competitivos en cuestiones de comprensión del contexto, y el éxito de BERT [2] en diversas tareas de NLP , en concreto en la detección de discurso de odio, como se ha visto en el Capítulo 2 en numerosos trabajos como Badjatiya *et al.* [34] o en concreto en español, Plaza-del-Arco *et al.* [5]).

H-1.1.– En concreto, BETO se trata del mejor modelo que clasifica en el idioma en español.

[2] Ver más en BOE-A-2019-777 o en Hate Crime Data 2019 (OSCE)

H-2.– El contexto de la red social es útil para la clasificación de odio en Twitter.

H-3.– La creación de modelos multimodales para la clasificación textual, en concreto en el problema de la detección del odio, supone una mejora de modelos basados en solamente texto.

De cara a la consecución de los objetivos presentados en la Sección 1.2 y la validación de las hipótesis expuestas anteriormente, se ha diseñado el siguiente esquema del proyecto:

1.– Elaboración de un modelo base al que llamaremos HaterBERT. Se procede a probar diferentes modos de emplear BERT así como diferentes configuraciones de hiperparámetros con el fin de comprobar la Hipótesis 1.

2.– Extracción de las características de los usuarios de la red social. Se realizará un estudio en base a atributos extras que se pueden obtener a través de la API de Twitter que permita modelar al usuario así como a todo su entorno. Asimismo, se realiza un enfoque general en víctimas y agresores.

3.– Codificación, automatización, manipulación y limpieza de los atributos extraídos de los usuarios.

4.– Análisis y estudio de la relevancia de las características sobre un conjunto de usuarios fuertemente conectado con el fin de identificar el comportamiento *hater* o *no hater*. De esta manera se comprobarán la validez y utilidad de las dichas características, según la Hipótesis 2.

5.– Creación y validación de SocialHaterBERT, una arquitectura multimodal que completa a HaterBERT con las nuevas características, comprobando así la Hipótesis 3.

Si se desea ver más sobre la organización del proyecto, ésta se puede encontrar en el Anexo F, donde se detalla un diagrama de Gantt que refleja la organización y distribución del mismo a lo largo de las semanas.

Además, es oportuno comentar ciertas limitaciones que surgen de cara al enfrentamiento del problema:

L-1.– Como se ha comentado previamente en la revisión del estado del arte (ver 2.1.3), los datasets en español son escasos. Si bien se pensó en conseguir un dataset específico nuevo, no se podía proceder a su etiquetado ya que se requiere de una realización a través de expertos.

L-2.– A no ser que se posea de una clave de empresa para la API de Twitter, no se pueden realizar consultas por texto de tweets con antigüedad mayor a 14 días, por lo que para su búsqueda es imprescindible poseer el *id* del tweet. Es por esto que el trabajo final de SocialHaterBERT se va realizar sobre el dataset de HaterNet (ver 2.2) en español.

L-3.– La API de Twitter tiene una limitación de 900 *requests* por cada 15 minutos. Esto puede dificultar la tarea y se requiere de una organización prudente de la misma. Es conveniente mencionar que los modelos pueden tardar entre 5 y 10 minutos a priori en entrenar por cada configuración de hiperparámetros. Los modelos multimodales pueden tardar mucho más, entre 1 y 4 horas.

De esta manera se representa el alcance y la organización del proyecto, cuyo diseño comienza a describirse en la siguiente sección.

3.2. HaterBERT

En esta sección se explica el diseño de HaterBERT, modelo base que sirve para la clasificación textual de odio o no odio. Es preciso recalcar que este modelo está basado en BERT, el cual se encuentra explicado en el Anexo B si se desea comprender mejor su funcionamiento. A continuación se detallan las modificaciones oportunas realizadas sobre la base de dicho transformador.

Es destacable remarcar que ha sido necesaria la utilización y el aprendizaje de las librerías de Tensorflow (Keras) y Pytorch a tales efectos. Además, se ha hecho uso de la librería de HuggingFace (transformers), la cual pone a disposición diversas herramientas para NLP y transformadores pre-entrenados, lo que ha sido imprescindible para añadir a HaterBERT los modelos pre-entrenados de BERT [2] y BETO [40], en su defecto en español.

De la misma forma se ha necesitado entender cómo realizar un ajuste de BERT (*Fine-Tuning*) adecuado al problema de la clasificación textual. Para poder abordar el problema, se ha elegido como referencia la implementación de BERT Classifier de DE-LIMIT [4], cuyo estudio se vió en la Sección 2.1.3, ya que resulta ser intuitiva de ajustar y proporciona buenos resultados para el problema plurilingüe, si bien se realizan los cambios oportunos en el ajuste de hiperparámetros. La optimización de hiperparámetros se detallará en el Capítulo 4.

Además, según se ha revisado en la Sección 2.1.3, Plaza-del-Arco *et al.* [5] también realizaron un estudio en español con BETO. En su publicación no se detalla en concreto temas sobre la implementación ni el Fine-Tuning, si bien se conoce que eligen BETO sin sensibilidad en mayúsculas. En esta propuesta se ha partido de la idea de que la sensibilidad de las mayúsculas puede aportar más información sobre la polaridad del tweet, por ende, al problema de la detección de odio, ya que los usuarios pueden utilizarlas para expresar más énfasis en sus ideas. También se encuentran diferencias en la optimización de hiperparámetros (ver Capítulo 4).

3.2.1. Preprocesamiento

Con el objetivo de poder aprovechar los recursos de *pre-trained* BERT es necesario utilizar el tokenizer, `BertTokenizer`, proporcionado por la librería transformers, ya que dicho transformador tiene un vocabulario fijo específico y una forma particular de transformar las palabras en tokens y en máscaras (ver Anexo B), si bien se realizan las siguientes modificaciones para cada entrada de texto en el `encode`:

- Tokenizar la oración.
- Añadir el token `[CLS]` al principio de la oración.
- Añadir el token `[SEP]` al final de la oración.
- Asignar los tokens a sus IDs correspondientes.

Después de ello se realiza un padding para garantizar que todas las secuencias tengan la misma longitud, rellenando 0 al final de cada secuencia hasta que cada secuencia tenga la misma longitud que la secuencia más larga. A continuación, para cada oración se crea la máscara de atención para los identificadores correspondientes. Se decide que:

- Si el ID es 0, entonces es padding y se activa en la máscara como un 0.
- Si el ID mayor o igual que 0, entonces es un token y se activa a 1.

Tras ello, se convierten las salidas en tensores para poder ser tratados. De esta manera son aptas para generar el DataLoader de Pytorch, lo que ayuda a ahorrar memoria durante el entrenamiento y aumenta la velocidad del mismo.

3.2.2. Clasificador

Afortunadamente, la librería de transformers pone a disposición una serie de interfaces diseñadas para tareas NLP , adaptando las entradas y las salidas dependiendo de la necesidad del problema. En este caso, es necesario utilizar `BertForSequenceClassification`, clase que contiene una capa de entrada adaptada para secuencias de texto u oraciones. De esta manera se puede utilizar BERT para el análisis de sentimiento, en concreto para el problema de la detección de odio.

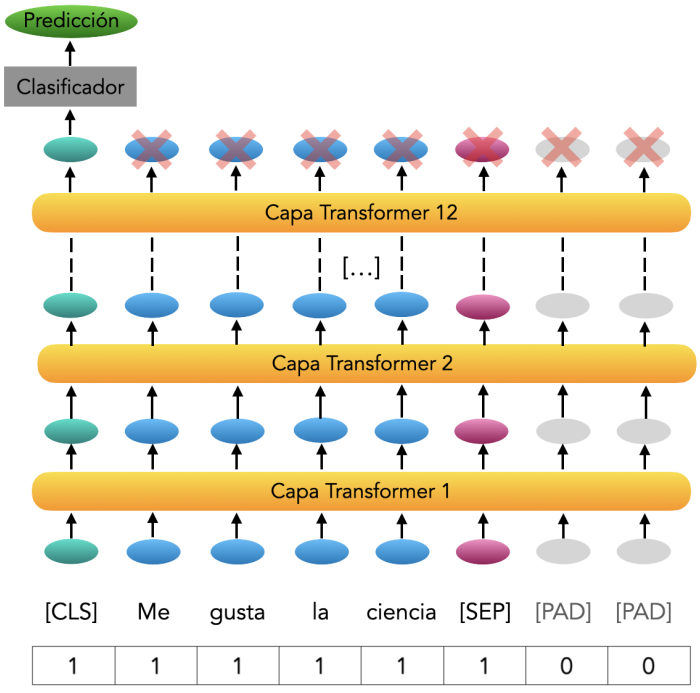


Figura 3.1: Ilustración del formato del preprocesamiento de una frase para el Fine-Tuning con BertForSequenceClassification. Se observa que la secuencia de dígitos corresponde a la máscara de atención y que a la frase se le ha realizado un padding para tener el mismo tamaño que todas las demás oraciones del dataset.

Análogamente, se puede indicar en el *path* qué modelo BERT utilizar de la librería de transformers. En el caso de utilizar un dataset en inglés se hará uso de BERT `bert-base-cased`, mientras que en español será BETO [40] `dccuchile/bert-base-spanish-wwm-cased` [3].

Para poder ajustar el clasificador es imprescindible utilizar un optimizador. Se utilizará AdamW [46], una muy buena opción para BERT que además utiliza el propio BERT pre-entrenado. Se hablará de los hiperparámetros probados referentes a AdamW en el Capítulo 4.

La fase de entrenamiento y validación se detalla en la Figura 3.2, junto con el preprocesamiento explicado anteriormente.

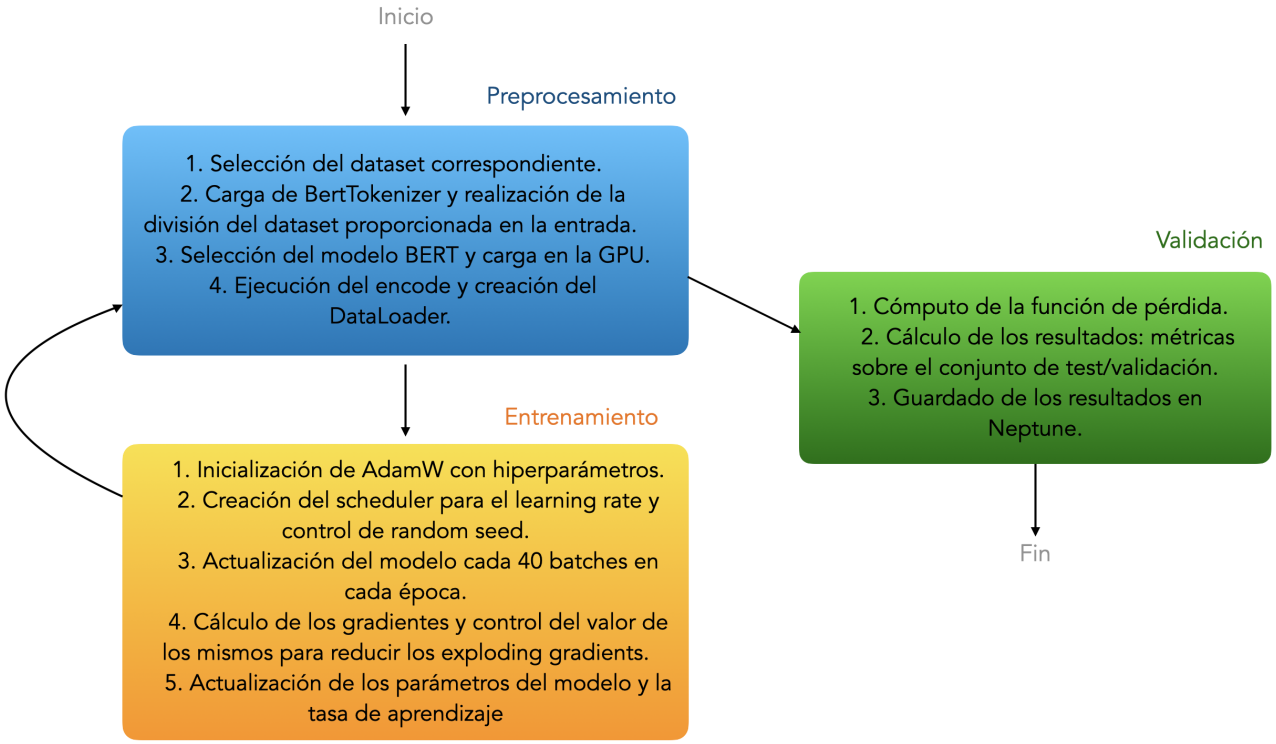


Figura 3.2: Diagrama de flujo que muestra las fases de preprocesamiento, entrenamiento y validación del modelo. Se resume el preprocesamiento explicado anteriormente y se detallan las fases de entrenamiento y validación del modelo.

A modo aclarativo es preciso destacar que también se ha hecho uso de Neptune AI con el fin poder controlar, etiquetar, monitorear y guardar los experimentos. Estos mismos, además de las métricas de validación se podrán ver en detalle en el Capítulo 4.

[3] BETO es un modelo BERT formado con un gran corpus en español y tiene un tamaño similar a BERT. Fue entrenado con la estrategia de enmascarado.

3.2.3. Utilización de una CNN

A lo largo del estudio realizado (ver Capítulo 2) se encontró que una buena propuesta es combinar la salida de BERT con una CNN. Para ello se introdujo una CNN similar a la de Mozafari *et al.* [29], si bien se tuvo que realizar las modificaciones oportunas, es decir, cambiar la dimensión de entrada de la capa linear por la de la dimensión de los tensores y cambiar el tamaño de la salida a un problema binario o de 3 clases, como en el caso del dataset de Davidson [17].

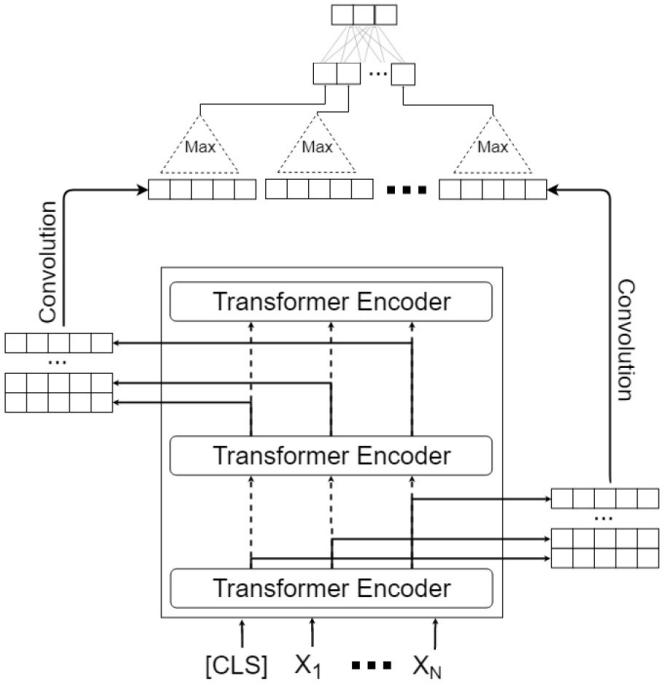


Figura 3.3: Detalle de la combinación de un transformador con una CNN. Fuente [29].

La CNN en cuestión recibe las salidas del `encode` de BERT en vez de la final del mismo. Estas salidas se concatenan obteniendo una matriz a la cual se le realiza una operación de convolución 2D $*$ 768 (tamaño de la inscrustación final de la arquitectura BERT). El máximo valor obtenido es operado por cada salida de BERT, aplicando max pooling en la salida de la convolución. Esta salida se concatena y se genera un vector de salida al que se le aplica función softmax con el fin de ofrecer la clasificación correspondiente. Este procedimiento se puede ver ilustrado en la Figura 3.3.

Los resultados de HaterBERT + CNN, así como la optimización de hiperparámetros, se pueden encontrar en el Anexo G.

3.3. SocialHaterBERT

Para conseguir que HaterBERT se alimente de características de la red social, es necesario primero conseguir toda la información relativa a los mismos. Dada la Limitación 2 (ver Sección 3.1), el dataset de HaterNet es el único que dispone del identificador de los tweets para tener la posibilidad de realizar su búsqueda a través de la API de Twitter y, por ende, recopilar todos los campos asociados para el análisis futuro.

Con el objetivo de perder la menos información posible, ya que los tweets se suspenden o se eliminan fácilmente a lo largo del tiempo, al igual que los usuarios, se realiza primero una extracción previa de tweets disponibles. Se trata de un problema común [8] que dificulta la realización de la tarea. Consecuentemente, es necesario reentrenar el modelo base de HaterBERT con los datos disponibles finalmente (ver en Capítulo 4).

De los 6000 tweets del dataset de HaterNet se extraen finalmente 4154 tweets. Los campos recogidos se pueden ver en la Tabla C.1. También se buscan en Twitter los 2 millones de tweets, obteniendo finalmente 1.5 millones aproximadamente con los mismos campos.

Tras ello se procede a ampliar el estudio de la red social utilizando Neo4j, una base de datos que, a diferencia de las bases de datos tradicionales, que organizan los datos en filas, columnas y tablas, Neo4j tiene una estructura flexible basada en grafos y definida por las relaciones almacenadas entre los registros de datos. Cada registro de datos o nodo almacena punteros directos a todos los nodos a los que está conectado y se pueden realizar consultas con conexiones complejas de mayor rendimiento y profundidad.

De esta manera, se extraen los usuarios existentes tras el filtrado del dataset, guardándose en la base de datos 200 tweets por cada usuario, lo que sirve para su posterior análisis. De aquí en adelante, a esta base de datos se la llamará, por abreviar, **SocialGraph** [4], la cual se recoge a través de la implementación de un *Crawler* con Py2neo, un cliente de Neo4j que ofrece las herramientas necesarias para trabajar con la base de datos en Python. En la implementación realizada se distinguen tres tipos de nodos:

- User: nodo que recoge toda la información relativa al usuario.
- Tweet: nodo que recoge toda la información relativa a los tweets.
- Multimedia: nodo que recoge la url referente al dominio o multimedia que comparten.

Los atributos que almacenan cada uno de estos nodos son todos los que la API de Twitter deja obtener. Si se desea ver en detalle los atributos y las relaciones entre los nodos se debe consultar el Anexo C, en las tablas C.2, C.3 y C.4.

Tras la búsqueda con el *Crawler* implementado, se extraen finalmente 3339 tweets disponibles

[4] También se entenderá como SocialGraph cualquier base de datos que se elabore con el mismo procedimiento y guarde las mismas características.

procedentes del dataset de HaterNet, a lo que además se añadieron 52 tweets nuevos gracias a la colaboración de la Oficina Nacional de Lucha Contra los Delitos de Odio, por lo que finalmente se obtienen 3391 tweets. Estos tweets son los que servirán para el entrenamiento de SocialHaterBERT y para entrenar HaterBERT de nuevo. Por abreviar, en este trabajo se realizará referencia al mismo con el nombre de **ACHaterNet**.

A continuación se detallarán los siguientes pasos realizados para la elaboración de un análisis a través de los atributos del usuario extraídos por la API, de nuevos atributos incluidos y de las características encontradas en los tweets guardados en dicha base de datos.

3.3.1. Cálculo de las medidas de centralidad de SocialGraph

Como se ha comentado previamente, Neo4j permite realizar consultas con flexibilidad, por lo que para guardar la información que se considere se puede hacer uso de NetworkX. Esto ha sido útil puesto que dicha librería permite, entre otras operaciones, realizar cálculos de medidas de centralidad en grafos, o en otras palabras, en la red de usuarios. La razón por la cual se obtienen estas medidas es por la relevancia de diversos estudios que demuestran la importancia de las mismas en las redes sociales, véase [47–49]. Entre otros motivos, estar estratégicamente o adecuadamente conectado o posicionado dentro de una red social puede afectar mucho más a la capacidad de un nodo o usuario para influir en otros que la cantidad de seguidores que éste posea. Estas medidas de centralidad que se han seleccionado se encuentran descritas en la Tabla C.5 del Anexo C.

3.3.2. Ampliación de las características de SocialGraph

Con el fin de obtener más allá de las características que la API nos ofrece se codifica un script en Python que obtiene las características representadas en el Anexo C, en las tablas C.6 C.7 y C.8. Este analizador extrae más información de los usuarios a través de los campos de los tweets de los usuarios además de las características textuales que poseen. Los métodos de extracción y la descripción de los mismos también se encuentran representados en dichas tablas.

3.3.3. Análisis de las características de los usuarios

Debido a que el conjunto de ACHaterNet (3391 tweets) se encuentra extraído de manera por búsqueda de palabras clave y manual, los usuarios no se encuentran fuertemente conectados. Es por ello que se realiza una nueva extracción de un grupo fuertemente conectado de usuarios, **Little743**, de cara a comprender si tanto las medidas de centralidad como el resto de atributos pueden servir a la hora de detectar odio en la red. Según Libert [49], un conjunto fuertemente interconectado de usuarios, aunque sea más pequeño, puede ser de mucha más relevancia en términos de propagación de la

influencia. Esto, además, puede servir de cara a futuros trabajos relacionados con la viralidad del odio en la red.

El análisis completo de esta red de usuarios se encuentra explicado con detalle en el Anexo D.

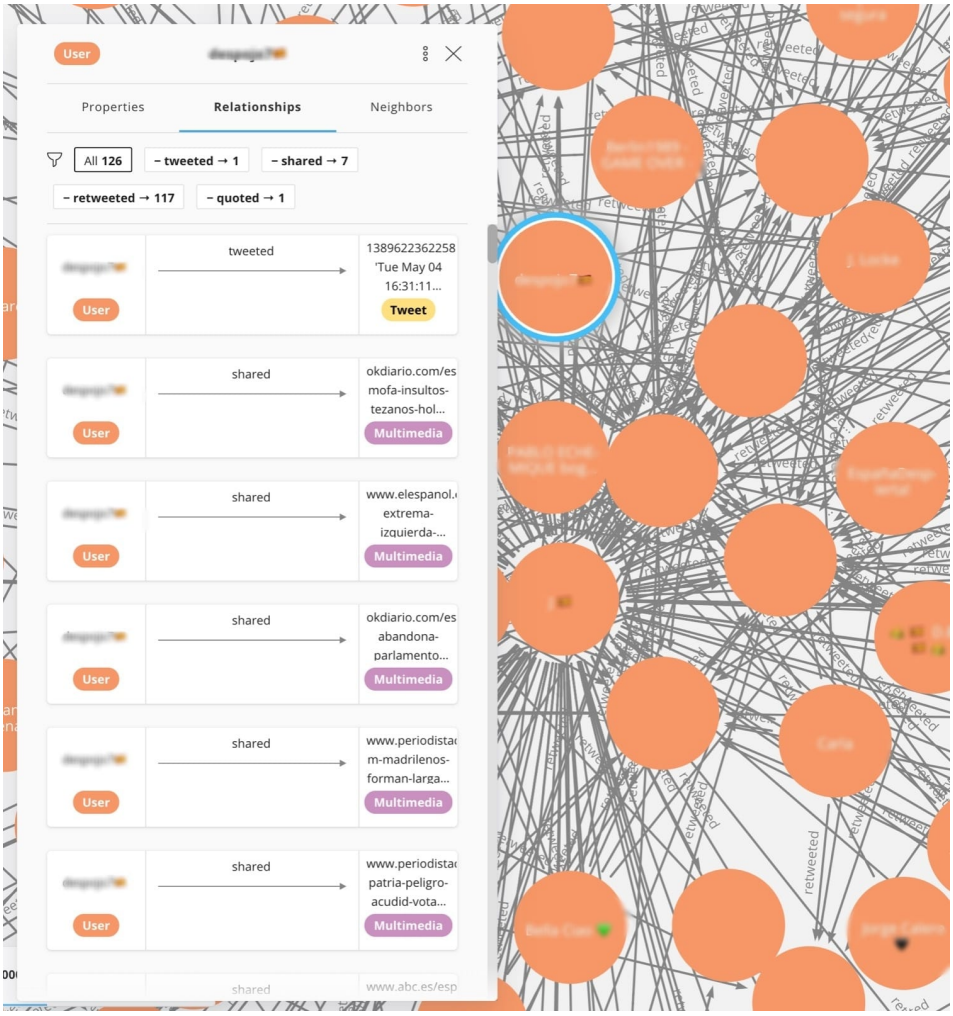


Figura 3.4: Detalle de un nodo la red de usuarios fuertemente conectada. Se muestra por la relación `User – retweeted – User`, además del detalle de otras relaciones entre otros nodos. De la misma forma se pueden visualizar otras opciones como las propiedades o los vecinos del nodo.

3.3.4. Manipulación y transformación de las características de los usuarios

Para poder incorporar a SocialHaterBERT un conjunto de características es imprescindible tratarlas previamente. Es preciso destacar que los diferentes tipos de atributos poseen un tipo en concreto de datos (ver Anexo C) y a cada uno de ellos se le debe aplicar su transformación correspondiente. De esta manera, se distribuyen los atributos en características numéricas y categóricas que se probarán en el modelo siguiente. En concreto, la mayoría de las variables categóricas requieren de una extensa

transformación bajo la técnica de Topic Modeling, ya que se necesita resumir las categorías más altas en la jerarquía de, por ejemplo, las top 5 categorías más utilizadas por el usuario en sus tweets, o las de los usuarios a los que siguen y a los que mencionan.

Se prueban, por tanto, tres modelos para el modelado de temas con Gensim: Latent Dirichlet Allocation (LDA) , Latent Semantic Analysis (LSI) y Hierarchical Dirichlet Process (HDP) . Generalmente, LDA es el mejor modelo en cuestiones Topic Modeling, pero en el caso presente con textos cortos, sobre todo cuando no se desea especificar temas de antemano, HDP puede ofrecer una solución mucho más coherente [50].

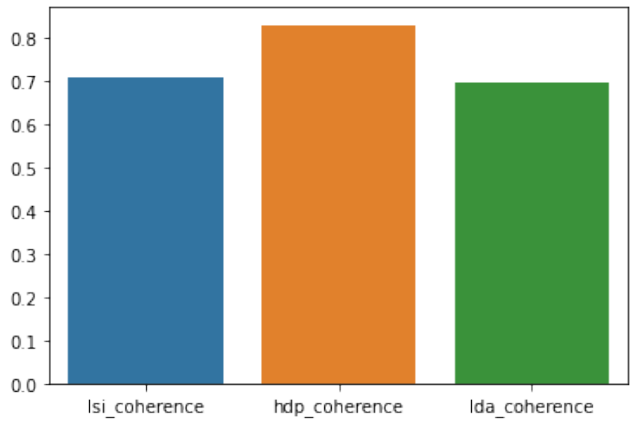


Figura 3.5: Coherencia de los modelos utilizados para Topic Modeling. Se refleja que el más coherente es HDP, seguido por LSI y LDA.

La Figura 3.5 muestra un ejemplo de medición de la coherencia. Cabe destacar que se mide la coherencia de los tres modelos en todos los casos probados que requieren de modelado de temas con Gensim, siendo HDP el que ofrece mejor puntuación de todos.

Por otro lado, se necesita estandarizar las variables numéricas. A dichas variables se le aplica un `StandardScaler` de manera que la distribución tenga un valor medio 0 y una desviación estándar de 1. Es importante centralizar los datos para que el ajuste sea consistente a toda la distribución.

Finalmente se obtiene el dataset de ACHaterNet transformado, cuya formalización se describe en el Cuadro 3.1. Se destaca que se ha realizado una corrección del campo del texto del tweet, ya que estos últimos estaban escritos por los usuarios con el alfabeto leet, tratando de camuflar insultos o palabras malsonantes.

Si se desea visualizar los detalles de todas estas transformaciones se encuentran explicados en el Anexo E. Además, los resultados finales sobre la valoración de la utilidad de las características totales recogidas se encuentran representados en el Capítulo 4, bajo la aportación de la red de usuarios fuertemente conectada, Little743.

El conjunto total de atributos $\mathcal{A}$, queda definido formalmente como:

$$\mathcal{A} = X_{profile} \times X_{activity} \times X_{centrality} \times \mathcal{L} \times \mathcal{T}$$

Donde:

- $X_{profile} \cong \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_4 \times \mathbb{Z}_4 \times \mathbb{Z}_{20} \times \mathbb{Z}_3 \times \mathbb{Z}_9$
  Denota el espacio de variables asociadas a la información intrínseca al perfil de un usuario (nombre, tipo de imagen ...). El espacio está formado por 7 variables categóricas pudiendo tomar cada una de ellas el número de valores i asociada al grupo multiplicativo Z_i.
- $X_{activity} \cong \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_4 \times \mathbb{Z}_{12} \times \mathbb{Z}_3 \times \mathbb{Z}_{12} \times \mathbb{Z}_{12} \times \mathbb{R}^{61}$
  Denota el espacio de variables asociadas a la actividad que un usuario realiza en la red y sus estadísticas agregadas obtenidas a partir de estos (porcentaje de tweets cada hora, número de tweets totales...). El espacio está formado por 7 variables categóricas pudiendo tomar cada una de ellas el número de valores i asociada al grupo multiplicativo Z_i asociado y 61 variables numéricas.
- $X_{centrality} \cong \mathbb{R}^7$
  Denota el espacio conformado por 7 variables numéricas asociadas a medidas de centralidad.
- $\mathcal{L} \cong \mathbb{Z}_2$ *Denota la etiqueta de clasificación correspondiente al registro (Tweet) que se define como Hate (1) /Non-Hate (0).*
- $\mathcal{T}$ *Denota el texto del tweet asociado.*

Como consecuencia de la definición se trabajará con un espacio de características de dimensión:

$$\begin{aligned} |\mathcal{A}| &= |X_{profile}| + |X_{activity}| + |X_{centrality}| + |\mathcal{L}| + |\mathcal{T}| \\ &= 7 + 68 + 7 + 1 + 1 = 84 \end{aligned}$$

Y con 3391 registros de datos cada uno correspondiente a un tweet, por lo que el dataset construido se puede definir formalmente como una matriz $\mathcal{D}$ de dimensión 3391 x 84.

Cuadro 3.1: Formalización del nuevo dataset creado, ACHaterNet.

3.3.5. Arquitectura de SocialHaterBERT

Con el objetivo de mejorar BERT se construye un modelo multimodal que incluye las características relativas a la red social.

Para la construcción del modelo final se hace uso de la librería de Multimodal Transformers, un toolkit que sirve para incorporar datos multimodales sobre datos de texto para tareas de clasificación y regresión. Se eligió dicha librería dado que utiliza los transformadores de la librería HuggingFace como modelo base para características de texto y además, agrega un módulo de combinación que toma las salidas del transformador junto con las características categóricas y numéricas, lo que produce características multimodales ricas para la clasificación correspondiente. De esta manera, un transformador preentrenado, los parámetros del módulo de combinación y el transformador se pueden entrenar en función de la tarea supervisada. Este concepto está representado visualmente en la Figura 3.6.

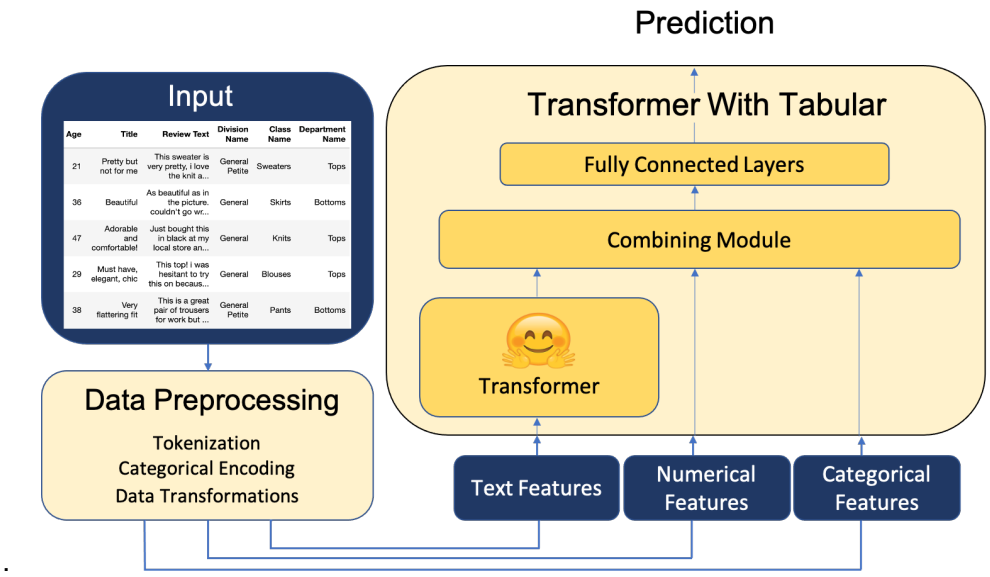


Figura 3.6: Detalle de la estructura de Multimodal Transformers. Este toolkit permite la elaboración de modelos multimodales que combinan transformadores con variables categóricas y numericas extra. Fuente github.com/georgian-io/Multimodal-Toolkit/

Como se puede incluir cualquier modelo compatible, se hace uso de BETO nuevamente, `dccuchile/bert-base-spanish-wwm-cased`. Para poder distribuir los datos para la clasificación, se puede especificar en un diccionario qué columnas son las columnas de texto, las numéricas, las categóricas y la de predicción. Gracias a la librería de transformers, bajo las clases `AutoTokenizer` y `AutoConfig` se crean instancias de `BertTokenizer` y `BertForSequenceClassification` respectivamente, lo que también permite el *Fine-Tuning* del mismo.

En el *Combining Module* en cuestión se crea una MLP de dos capas ocultas con una función de activación ReLU que mejora el entrenamiento de la misma. El detalle de este módulo se encuentra dibujado en la Figura 3.7.

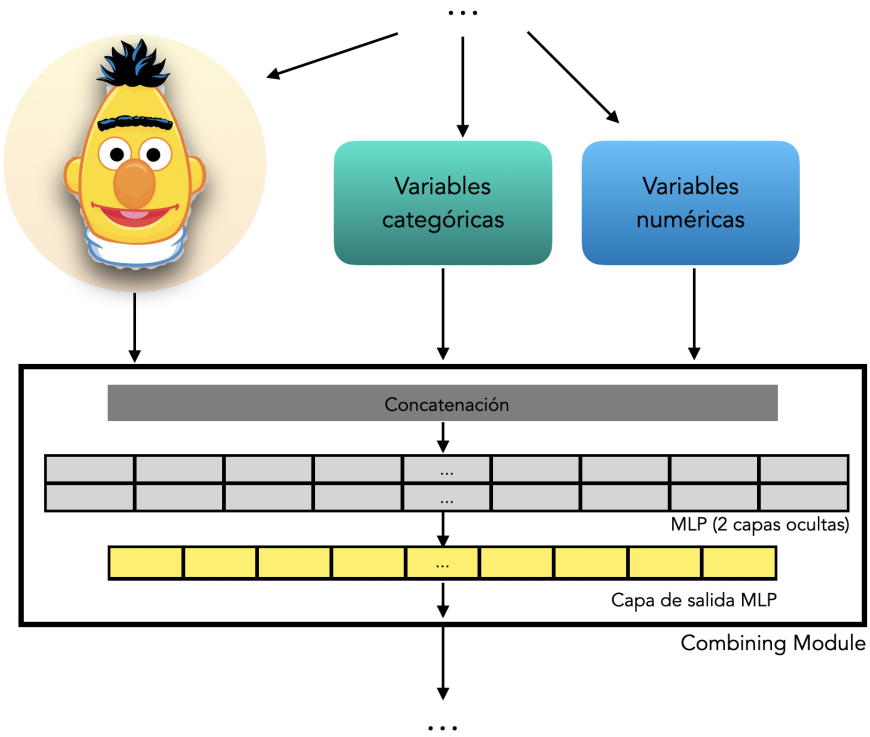


Figura 3.7: Detalle del Combining Module de Multimodal Transformers. Se muestran además las entradas al mismo.

Es preciso destacar que antes de la capa de salida (Figura 3.7) se realizan diferentes estrategias que son proporcionadas por el toolkit. Estas se describen a continuación en la Tabla 3.1, etiquetados por número de experimento asociado.

SHAT-1	Sólo texto
SHAT-2	Suma basada en la atención antes de la capa de salida
SHAT-3	Suma ponderada antes de la capa de salida
SHAT-4	Suma lógica antes de la capa de salida

Tabla 3.1: Detalle de estrategias realizadas para el Combining Module de SocialHaterBERT.

Los resultados de estos experimentos se detallan en el Capítulo 4, donde se refleja que la mejor estrategia es la tomada por el experimento SHAT-4.

4

Pruebas y resultados

En este capítulo se detallan y se analizan los resultados obtenidos en los diferentes experimentos realizados sobre los modelos HaterBERT y SocialHaterBERT. En primer lugar, se muestra la optimización de hiperparámetros para cada uno de los modelos, siguiendo tras ello la exposición de los diferentes resultados obtenidos. Para el caso de HaterBERT se expone, además, una comparativa con el estado del arte en español y en inglés, con la configuración de hiperparámetros correspondiente. En segundo lugar, se presentan los resultados obtenidos a través del análisis realizado en el Anexo D, con el fin de mostrar la utilidad de SocialGraph. Finalmente se exponen los hiperparámetros probados para SocialHaterBERT, así como los que mejor se adaptan al modelo, además del resultado con las diferentes estrategias probadas y la comparativa con HaterBERT. De esta manera, se prueba el rendimiento de incluir características multimodales al problema de la detección de odio.

Es importante resaltar que las valoraciones de los resultados se realizan tomando en cuenta como principal métrica F1-score, ya que los falsos negativos y los falsos positivos son más cruciales en diversos temas [51] como la detección de odio, si bien se valoran otras métricas.

4.1. Entorno de pruebas

Las pruebas realizadas para HaterBERT y SocialHaterBERT se han llevado a cabo en Google Colaboratory Pro con las siguientes características representadas en la Tabla 4.1.

Recursos	Características
GPU	Tesla P100-PCIE-16GB
CUDA version	11.2
RAM	30GB
Python version	3.7.10

Tabla 4.1: Detalle de las características del entorno de pruebas para los modelos HaterBERT y SocialHaterBERT.

Las pruebas realizadas para recoger y analizar SocialGraph se han llevado a cabo en un ordenador

personal con las siguientes características representadas en la Tabla 4.2.

Recursos	Características
CPU	2,3 GHz Intel Core i9 (8 núcleos)
GPU	Intel UHD Graphics 630 1536 MB
RAM	16 GB 2667 MHz DDR4
Python version	3.8.3
S.O.	macOS Big Sur 11.4 (20F71)

Tabla 4.2: Detalle de las características del entorno de pruebas para SocialGraph.

4.2. HaterBERT: optimización de hiperparámetros

Con el fin de recoger los mejores resultados posibles, se realizan diferentes experimentos con distintos hiperparámetros. Los hiperparámetros probados se muestran en la Tabla 4.3.

Hiperparámetros	Opciones
Épocas	[2,3,4,5]
Learning Rate	[2e-5, 3e-5, 5e-5]
Random Seed	[**2018**, 2019, 2020, 2021, 2022, 2023]
Batch Size	[16, 32]
Epsilon	[1e-6, 1e-8]
Max Length	256

Tabla 4.3: Detalle de los hiperparámetros probados para el modelo de HaterBERT.

4.3. HaterBERT: comparativa con el estado del arte

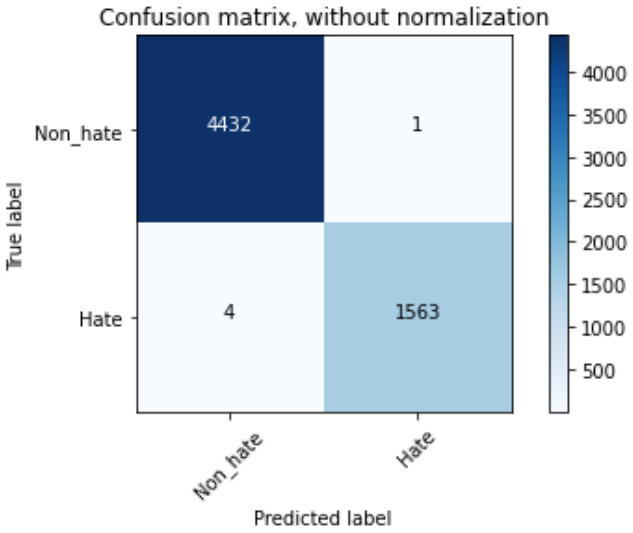
A continuación se detallan los experimentos realizados con HaterBERT, además de una comparativa con el estado del arte. Es importante destacar que se reflejan los mejores resultados, ya que por limitaciones relacionadas con la longitud del presente trabajo no se pueden detallar todos.

En la Tabla 4.4 se detallan los resultados obtenidos con el dataset de HaterNet (6000 tweets, ver Tabla 2.2), además de la comparativa del rendimiento obtenido respecto a Pereira *et al.* [1]. Se detalla, además, el método de validación utilizado (LOOCV).

Experimento	Modelo	Autor	Validación	Precisión	Recall	F1	AUC	Accuracy	Errores	Hiperparámetros
–	LSTM+MLP	Pereira *et al.* [1]	LOOCV	0.6250	0.5980	0.6110	0.8280	–	–	–
HAT-1171	HaterBERT	Esta propuesta	LOOCV	0.9299	0.9196	0.9246	0.9196	0.9425	345	Épocas: 2 Batch Size: 16 Learning Rate: 2e-5 Epsilon: 1e-6
HAT-1183	HaterBERT	Esta propuesta	LOOCV	0.9984	0.9986	0.9985	0.9986	0.9988	7	Épocas: 5 Batch Size: 16 Learning Rate: 5e-5 Epsilon: 1e-6
HAT-1185	HaterBERT	Esta propuesta	LOOCV	0.9992	0.9986	**0.9989**	0.9986	0.9992	5	Épocas: 5 Batch Size: 32 Learning Rate: 5e-5 Epsilon: 1e-6

Tabla 4.4: Resultados y comparativa con el estado del arte de HaterBERT con Pereira *et al.* [1]. Se utiliza el dataset de HaterNet (Tabla 2.2).

Se puede comprobar que se alcanzan buenos resultados para el dataset y que BETO mejora considerablemente el problema. En la Figura 4.1 se ilustra la matriz de confusión del experimento HAT-1185, donde se reflejan que 5 de 6000 tweets se clasifican erróneamente. Si bien cabe valorar la relevancia de los resultados, se ha de entender que LOOCV emplea $n-1$ datos para entrenar el modelo, lo que implica prácticamente todo el dataset. En términos prácticos lo que realiza es un mayor ajuste del modelo a los datos disponibles y reduciendo el sesgo, lo que puede conllevar a un mayor riesgo de overfitting (sobre-ajuste) y de varianza. Es por ello que se realizan más comparaciones respecto al estado del arte con otros tipos de validación: Stratified Sampling, con diferentes divisiones, y K-Fold Cross Validation.




```
Accuracy: 0.99917
Fscore: 0.99892
AUC: 0.99861
Precision: 0.99923
Recall: 0.99861
```

Figura 4.1: Matriz de confusión del experimento HAT-1185 realizado con LOOCV.

En la siguiente página, en la Tabla 4.5 se detallan los resultados obtenidos con una división (Stratified Sampling) del dataset de HaterNet de 70 % para el entrenamiento, 20 % para el test y 10 % para la validación del modelo. Además, se comparan los resultados con Aluru *et al.* [4], donde utilizan mBERT.

Nuevamente se observa que los resultados mejoran con la utilización de BETO, además de que a una mayor entrada del dataset se mejora la salida. Es preciso destacar que aunque mBERT pueda ofrecer buenos resultados a nivel general, el problema que tiene es que está pre-entrenado sobre un conjunto de corpus monolingües de diferentes idiomas, por lo que no proporciona un mecanismo de detección del idioma en cuestión, y el token se puede confundir con otro idioma fácilmente. Sin embargo, BETO fue pre-entrenado con un conjunto de datos específicamente en el idioma español, por lo que es mucho más apropiado en conjuntos de datos en español.

También se realiza una comparativa con el estudio de Plaza-del-Arco [5], el cual se muestra en la Tabla 4.7. En ella se realizan diferentes experimentos con una validación de 10k-Fold CV sobre el dataset de HaterNet. Este método es también un proceso iterativo al igual que LOOCV, aunque emplea menos observaciones como entrenamiento. Sin embargo, obtiene una estimación más precisa del error de test gracias a un mejor balance entre el sesgo y la varianza, ya que emplea $k - 1$ grupos para entrenar el modelo y el restante para la validación. Gracias a esto, se obtiene una mejor perspectiva de los resultados obtenidos, valorando así la calidad de los mismos.

	Modelo	Autor	Validación	F1 (Training Size)						Hiperparámetros
				16	32	64	128	256	Total	
	mBERT	Aluru *et al.* [4]	70–20–10	0.4395	0.4285	0.4048	0.4861	0.5999	0.7329	–, Max Length: 128
	HaterBERT	Esta propuesta	70–20–10	0.5025	0.5787	0.65401	0.6906	0.7459	**0.7667**	Épocas: 5, Batch Size: 32, Learning Rate: 5e-5, Epsilon: 1e-6, Max Length: 256
Experimento				HAT-743	HAT-758	HAT-773	HAT-881	HAT-816	HAT-1001	

Tabla 4.5: Resultados y comparativa con el estado del arte de HaterBERT con Aluru *et al.* [4]. Se utiliza el dataset de HaterNet (Tabla 2.2).

Exp.	Modelo	Autor	Precisión	Recall	F1	AUC	Accuracy	Errores	Hiperparámetros
–	BETO	Plaza-del-Arco *et al.* [5]	0.6928	0.8303	0.7553	–	–	359	E: 3, BS: 16, LR: 2e-5, ML: 80
HAT-1188	HaterBERT	Esta propuesta	0.8666	0.8710	**0.8673**	0.8709	0.8680	66	E: 3, BS: 16, LR: 2e-5, ML: 256

Tabla 4.6: Resultados y comparativa con el estado del arte de HaterBERT con Plaza-del-Arco *et al.* [5]. Se utiliza el dataset de HatEval en español (Tabla 2.3).

En el estudio de Plaza-del-Arco [5] también se realizaron pruebas sobre el dataset de HatEval en español con el mismo método de validación (10K-Fold CV). En la Tabla 4.6 se muestra la comparativa de la presente propuesta con su estudio.

Experimento	Modelo	Autor	Validación	Precisión	Recall	F1	AUC	Accuracy	Errores	Hiperparámetros
–	BETO	Plaza-del-Arco *et al.* [5]	10 k-Fold	0.7045	0.6282	0.6580	–	–	106	Épocas: 2, Batch Size: 16, Learning Rate: 2e-5, Max length: 80
HAT-1191	HaterBERT	Esta propuesta	10 k-Fold	0.9165	0.9100	0.9132	0.9101	0.9335	99	Épocas: 2, Batch Size: 16, Learning Rate: 2e-5, Epsilon: 1e-6, Max length: 256
HAT-1188	HaterBERT	Esta propuesta	10 k-Fold	0.9766	0.9791	**0.9778**	0.9701	0.9828	73	Épocas: 5, Batch Size: 32, Learning Rate: 5e-5, Epsilon: 1e-6, Max length: 256

Tabla 4.7: Resultados y comparativa con el estado del arte de HaterBERT con Plaza-del-Arco *et al.* [5]. Se utiliza el dataset de HaterNet (Tabla 2.2).

A modo general, se observa que se mejora el resultado en español, confirmando la Hipótesis 1 (ver 3.1), a través del ajuste de BERT y de una óptima configuración de hiperparámetros. Además de esto, a diferencia de Aluru *et al* [4] y Plaza-del-Arco *et al.* [5], se ha elegido un max length de 256. Es cierto que esto hace que el algoritmo tarde más tiempo en procesar la entrada, pero también puede mejorar la clasificación.

Si se desea ver pruebas adicionales sobre datasets en inglés, se debe consultar el Anexo G.

4.4. SocialGraph: detección de haters con Little743

Con el objetivo de probar la Hipótesis 2 (ver 3.1), se recogen en esta tabla diversas pruebas realizadas con modelos tradicionales de Machine Learning y un MLP Classifier, con los que se pueden trabajar fácilmente en Scikit-Learn.

Modelo	Precisión	Recall	F1	Accuracy
Naive Bayes	0.8677	0.8618	0.8563	0.8565
Logistic Regression	0.8995	0.8831	0.8858	0.8879
KNN	0.6794	0.6713	0.6708	0.6771
SVM	0.8801	0.8709	0.8728	0.8744
Random Forest	0.9958	0.9952	0.9955	0.9955
MLP	0.8411	0.8296	0.8314	0.8341

Tabla 4.8: Pruebas realizadas para la detección de haters con la red social Little743 recogida previamente.

Por norma general se puede observar que en este caso Random Forest es el que mejor se comporta con diferencia. Además, se puede valorar a modo general que las características encontradas en Little743 pueden ser determinantes a la hora de detectar odio en la red, lo que se puede aplicar para SocialGraph en general.

4.5. SocialHaterBERT: optimización de hiperparámetros

De la misma manera que en HaterBERT, se prueban diferentes configuraciones de hiperparámetros con el objetivo de encontrar el potencial máximo de SocialHaterBERT.

En la Tabla 4.9 se describen estos hiperparámetros, siendo los valores en negrita los que finalmente mejoraban el rendimiento del modelo.

Hiperparámetros	Opciones
Épocas	[1,2,3,4,5,6,**7**,8,9]
Learning Rate	[1e-5, **2e-5**, 3e-5, 4e-5, 5e-5]
Activation	ReLU
Batch Size	[4, 6, **16**, 32]
Epsilon	[**1e-5**, 1e-12]

Tabla 4.9: Diferentes hiperparámetros probados para SocialHaterBERT.

4.6. SocialHaterBERT: resultados y comparativa

Finalmente se muestra el rendimiento de SocialHaterBERT, entrenado con el dataset descrito en la Sección 3.3, ACHaterNET (ver Cuadro 3.1). Si bien se exceptúan las variables `activity_hourly_X` ($\texttt{X}\epsilon[0-23]$) y `activity_weekly_X` ($\texttt{X}\epsilon[0-6]$), ya que en pruebas anteriores se descubre que solo aportan ruido en este dataset en concreto y no cambian apenas los resultados. Por tanto, bajo estos resultados se escogen finalmente 54 atributos de 84, uno de los cuales es la etiqueta a predecir. Para ello, se prueban las diferentes estrategias comentadas en la Sección 3.3.5, cuyos resultados se pueden encontrar en la Tabla 4.10. Es preciso destacar que para estas pruebas se tiene en cuenta un Stratified Sampling de 80–10–10.

Experimento	Descripción	AUC	Recall	Precisión	F1
SHAT-1	Solo texto	0.7791	0.6943	0.5121	0.6222
SHAT-2	Suma basada en la atención antes de la capa de salida	0.8394	0.7653	0.7364	0.7501
SHAT-3	Suma ponderada antes de la capa de salida	0.8536	0.5952	0.8928	0.7142
SHAT-4	Suma lógica antes de la capa de salida	0.8923	0.7826	0.7031	**0.8023**

Tabla 4.10: Resultados obtenidos para las diferentes estrategias tomadas en SocialHaterBERT.

Finalmente, se vuelve a entrenar HaterBERT con el dataset de ACHaterNet, el cual como se mencionó en la Sección 3.1 se redujo a 3391 tweets. Tras ello, se compara con SocialHaterBERT. Se puede comprobar que HaterBERT alimentado por SocialGraph es más fuerte que sin él.

Modelo	Accuracy	F1	AUC	Precisión	Recall
HaterBERT	0.8343	0.7645	0.7354	0.8506	0.7354
SocialHaterBERT	0.8472	**0.8023**	0.8923	0.7031	0.7826

Tabla 4.11: Comparativa realizada HaterBERT vs. SocialHaterBERT

5

Conclusiones y trabajo futuro

En este capítulo se reflejan las conclusiones extraídas del proyecto, valorando los resultados obtenidos y el potencial detectado para la elaboración de futuros estudios.

5.1. Conclusiones

En un mundo ya polarizado las redes sociales muestran ser un arma de doble filo con la aparición de fenómenos como el discurso de odio. En el presente trabajo se ha detectado y analizado la presencia del mismo en la red social Twitter. Para ello se ha realizado primero un algoritmo base, HaterBERT, que mejora de un 3 % a un 27 % los resultados actuales en español, cumpliendo el objetivo inicial del proyecto.

Además se ha analizado la presencia del discurso de odio en la red social Twitter a través de un amplio estudio que ha servido para extrapolar características esenciales de la misma. Para ello, se ha elaborado un procedimiento para la extracción y manipulación de dichas características, SocialGraph, lo cual se ha demostrado con un F1 del 99 % (Random Forest) que aportan datos de valor para la clasificación hater o no hater.

Estas conclusiones han servido para incluir más información a la clasificación textual a través de SocialHaterBERT, un modelo final multimodal que combina variables categóricas y numéricas de la red social con la entrada de texto de los tweets, ofreciendo no solo una nueva manera de entender el discurso de odio en las redes sociales en general, sino una aportación de valor que muestra que el contexto de la red social mejora el problema de la clasificación textual. En concreto se ha realizado una mejora de un 4 % en el algoritmo inicial de HaterBERT, un 18 % en el algoritmo base (solo texto) y un 19 % en el original, Pereira *et al.* [1].

5.2. Trabajo futuro

Gracias al potencial que nos ofrece SocialHaterBERT, se abre un nuevo caso de estudio centrado en la exploración y la creación de modelos multimodales para la detección de odio.

En primer lugar, sería interesante superar los límites que se han sorteado en este trabajo con los datasets, ya que sería relevante obtener toda la información posible acerca de la red social con el objetivo de no perderla con el paso del tiempo. Para ello, se debería obtener conjuntos de datos que no solo tengan el identificador del tweet, sino poder automatizar las características que ofrece SocialGraph.

Aunque SocialGraph demuestra aportar una mejoría en la clasificación, habría que realizar una revisión de los métodos que se han utilizado para la obtención de las características. Por ejemplo, se podrían refinar métodos como el reconocimiento de la entidad del usuario o el de su descripción. Además, se debería investigar si la utilización de BETO sin sensibilidad en mayúsculas pueda ser o no más apropiado que sin ella.

Además de esto, habría que incluir un clasificador de imágenes para el multimedia que comparten los usuarios, ya que en muchas ocasiones los usuarios no necesitan de lenguaje verbal para lanzar un mensaje. También sería relevante extender algunas de las características obtenidas que sean únicas al propio tweet y no a todos los tweets recogidos para analizar al usuario.

Otra tarea futura sería obtener a través de Group Lasso qué características son las que realmente mejoran el modelo, ya que, aunque la gran mayoría aporten datos de valor, muchas de ellas pueden aportar nada más que ruido en según qué casos. Por ello es relevante estudiar cómo estas se comportan en diferentes datasets y finalmente obtener las que realmente hagan un cambio en SocialHaterBERT.

También se podría estudiar el contexto de no solo la red social, sino del propio tweet en sí, esto es, conocer a qué responde exactamente en caso de ser una respuesta, y además, obtener todos los tweets conectados en caso de ser un hilo. Para ello se podría crear un árbol de respuestas con herramientas como PHEME.

Es destacable que el toolkit de Multimodal Transformers ofrece una buena solución para unificar atributos y crear en consecuencia un modelo multimodal. Sin embargo, este funciona como una caja negra y a pesar de que se puede ajustar con parámetros de entrada, tiene sus límites en la creación y ajuste del MLP creado en el *Combining Module*. Esto se podría explotar creando nuevas soluciones, como la aportación de otros tipos de redes y/o configuraciones, que amplíen los beneficios de dicha herramienta.

De manera paralela se podría estudiar la aportación de optimizadores como LAMB (NVLAMB) [52] con el fin de escalar el entrenamiento con BERT con las GPU de NVIDIA, ya que el tiempo de

entrenamiento de estos modelos es alto a medida que el dataset incrementa.

Finalmente, ya que se ha alentado a encontrar relaciones con la difusión y la viralidad del odio en la red, sería relevante estudiar aspectos como la revisión de un histórico de odio, tanto de tendencias como de personajes públicos y usuarios anónimos afectados por el odio, o de los propios agresores. Tras ello, se podría estudiar cómo se comportan entre sí con una ampliación de atributos en SocialGraph.

Bibliografía

[1] J. C. Pereira-Kohatsu, L. Quijano-Sánchez, F. Liberatore, and M. Camacho-Collados, "Detecting and monitoring hate speech in twitter," *Sensors*, vol. 19, no. 21, 2019.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[3] F. Poletto, V. Basile, M. Sanguinetti, C. Bosco, and V. Patti, "Resources and benchmark corpora for hate speech detection: a systematic review," *Language Resources and Evaluation*, 2020.

[4] S. S. Aluru, B. Mathew, P. Saha, and A. Mukherjee, "Deep learning models for multilingual hate speech detection," 2020.

[5] F. M. P. del Arco, M. D. Molina-González, L. A. Ureña-López, and M. T. Martín-Valdivia, "Comparing pre-trained language models for spanish hate speech detection," *Expert Systems with Applications*, vol. 166, 2021.

[6] M. del Interior, "Informe sobre la evolución de los delitos de odio en españa 2019." http://www.interior.gob.es/documents/642012/3479677/informe+evolucion+2019/631ce020-f9d0-4feb-901c-c3ee0a777896.

[7] B. Mathew, N. Kumar, Ravina, P. Goyal, and A. Mukherjee, "Analyzing the hate and counter speech accounts on twitter," 2018.

[8] M. H. Ribeiro, P. H. Calais, Y. A. Santos, V. A. Almeida, and W. Meira, "Characterizing and detecting hateful users on twitter," 2018.

[9] L. B. Martínez, P. P. D. S. Ortega, M. Ángel Martínez Miró, and M. S. R. Hidalgo, "Discursos de odio: una epidemia que se propaga en la red. estado de la cuestión sobre el racismo y la xenofobia en las redes sociales," *Mediaciones Sociales*, vol. 18, pp. 25–42, may 2019.

[10] J. A. C. Donaire, "Libertad de expresión y "discurso del odio" religioso," *Revista De Fomento Social*, vol. 278, pp. 205–243, 2015.

[11] A. M. Founta, C. Djouvas, D. Chatzakou, I. Leontiadis, J. Blackburn, G. Stringhini, A. Vakali, M. Sirivianos, and N. Kourtellis, "Large scale crowdsourcing and characterization of twitter abusive behavior," 2018.

[12] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? predictive features for hate speech detection on twitter," 2016.

[13] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, and N. Farra, "Predicting the type and target of offensive posts in social media," pp. 1415–1420, 01 2019.

[14] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. R. Pardo, P. Rosso, and M. Sanguinetti, "Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter," 2019.

[15] P. Mathur, R. R. Shah, R. Sawhney, and D. Mahata, "Detecting offensive tweets in hindi-english code-switched language," pp. 18–26, 01 2018.

[16] E. Fersini, P. Rosso, and M. Anzovino, "Overview of the task on automatic misogyny identification at ibereval 2018," in *IberEval@SEPLN*, 2018.

[17] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," 2017.

[18] F. Del Vigna, A. Cimino, F. Dell'Orletta, M. Petrocchi, and M. Tesconi, "Hate me, hate me not: Hate speech detection on facebook," 01 2017.

[19] J. Salminen, H. Almerekhi, M. Milenkovic, S.-G. Jung, J. An, H. Kwak, and J. Jansen, "Anatomy of online hate: Developing a taxonomy and machine learning models for identifying and classifying hate in online news media," 03 2019.

[20] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. L. Bhamidipati, "Hate speech detection with comment embeddings," *Proceedings of the 24th International Conference on World Wide Web*, 2015.

[21] T. Zia, M. Akram, M. S. Nawaz, B. Shahzad, M. Abdullatif, R. Mustafa, and M. I. Lali, "Identification of hatred speeches on twitter," 11 2016.

[22] L. Silva, M. Mondal, D. Correa, F. Benevenuto, and I. Weber, "Analyzing the targets of hate in online social media," 2016.

[23] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, 2017.

[24] Z. Waseem, "Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter," pp. 138–142, 01 2016.

[25] J. Park and P. Fung, "One-step and two-step classification for abusive language detection on twitter," in *ALW@ACL*, 2017.

[26] B. Gambäck and U. K. Sikdar, "Using convolutional neural networks to classify hate-speech," in *Proceedings of the First Workshop on Abusive Language Online*, (Vancouver, BC, Canada), pp. 85–90, Association for Computational Linguistics, Aug. 2017.

[27] Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on twitter using a convolution-gru based deep neural network," 03 2018.

[28] Z. Zhang and L. Luo, "Hate speech detection: A solved problem? the challenging case of long tail on twitter," 2018.

[29] M. Mozafari, R. Farahbakhsh, and N. Crespi, "A bert-based transfer learning approach for hate speech detection in online social media," vol. 881 SCI, 2020.

[30] G. Kovács, P. Alonso, and R. Saini, "Challenges of hate speech detection in social media," *SN Computer Science*, vol. 2, 2021.

[31] A. Olteanu, C. Castillo, J. Boy, and K. R. Varshney, "The effect of extremist violence on hateful speech online," 2018.

[32] G. A. de Oliveira, R. de Oliveira Albuquerque, C. A. B. de Andrade, R. T. de Sousa, A. L. S. Orozco, and L. J. G. Villalba, "Anonymous real-time analytics monitoring solution for decision making supported by sentiment analysis," *Sensors (Switzerland)*, vol. 20, 2020.

[33] A. Arango, J. Pérez, and B. Poblete, "Hate speech detection is not as easy as you may think: A closer look at model validation," 2019.

[34] P. Badjatiya, M. Gupta, and V. Varma, "Stereotypical bias removal for hate speech detection task using knowledge-based generalizations," 2019.

[35] S. MacAvaney, H. R. Yao, E. Yang, K. Russell, N. Goharian, and O. Frieder, "Hate speech detection: Challenges and solutions," *PLoS ONE*, vol. 14, 2019.

[36] M. A. H. Taieb, T. Zesch, and M. B. Aouicha, "A survey of semantic relatedness evaluation datasets and procedures," *Artificial Intelligence Review*, vol. 53, 2020.

[37] A. Muneer and S. M. Fati, "A comparative analysis of machine learning techniques for cyberbullying detection on twitter," *Future Internet*, vol. 12, 2020.

[38] P. Vijayaraghavan, H. Larochelle, and D. Roy, "Interpretable multi-modal hate speech detection," 2021.

[39] K. Perifanos and D. Goutsos, "Multimodal hate speech detection in greek social media," 3 2021.

[40] J. Cañete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang, and J. Pérez, "Spanish pre-trained bert model and evaluation data," in *PML4DC at ICLR 2020*, 2020.

[41] D. Battistelli, C. Bruneau, and V. Dragos, "Building a formal model for hate detection in french corpora," vol. 176, 2020.

[42] K. Florio, V. Basile, M. Polignano, P. Basile, and V. Patti, "Time of your hate: The challenge of time in hate speech detection on social media," *Applied Sciences (Switzerland)*, vol. 10, 2020.

[43] J. Garland, K. Ghazi-Zahedi, J.-G. Young, L. Hébert-Dufresne, and M. Galesic, "Countering hate on social media: Large scale classification of hate and counter speech," 2020.

[44] J. Garland, K. Ghazi-Zahedi, J.-G. Young, L. Hébert-Dufresne, and M. Galesic, "Impact and dynamics of hate and counter speech online," 2020.

[45] K. Sreelakshmi, B. Premjith, and K. Soman, "Detection of hate speech text in hindi-english code-mixed data," *Procedia Computer Science*, vol. 171, pp. 737–744, 2020. Third International Conference on Computing and Network Communications (CoCoNet'19).

[46] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[47] F. Grando, D. Noble, and L. Lamb, "An analysis of centrality measures for complex and social networks," pp. 1–6, 12 2016.

[48] S. Rajeh, M. Savonnet, E. Leclercq, and H. Cherifi, *Investigating Centrality Measures in Social Networks with Community Structure*, pp. 211–222. 01 2021.

[49] K. Libert, "Your network's structure matters more than its size." https://hbr.org/2016/02/your-networks-structure-matters-more-than-its-size, Oct 2017.

[50] E. C. Sroka, "Don't be afraid of nonparametric topic models (part 2: Python)," Sep 2020.

[51] P. Huilgol, "Accuracy vs. f1-score," Aug 2019.

[52] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, “Large batch optimization for deep learning: Training bert in 76 minutes,” 2020.

[53] F.-M. Plaza-Del-Arco, M. D. Molina-González, L. A. Ureña-López, and M. T. Martín-Valdivia, “Detecting misogyny and xenophobia in spanish tweets using language technologies,” *ACM Transactions on Internet Technology (TOIT)*, vol. 20, no. 2, pp. 1–19, 2020.

Definiciones

alfabeto leet escritura compuesta que sustituye letras por números del 0-9.

Centroid Embedding método para la tarea de resumir textos que aprovecha la composición de las incrustaciones de palabras bajo el concepto de pseudo-documento.

counter speech fenómeno que contrarresta el discurso de odio con más discurso.

crowdsourcing herramienta colaborativa por la cual se convoca a una gran cantidad de público para aportar ideas con la finalidad de encontrar la solución de una tarea.

cyberbullying uso de medios digitales para molestar o acosar a una persona o grupo de personas mediante ataques personales, divulgación de información confidencial o falsa entre otros medios.

discurso de odio tipo de discurso que consiste en la denigración de un grupo o individuos basados en una condición de pertenencia a un grupo, normalmente contextualizado en raza, etnia, orientación sexual, identidad de género, religión o en creencias políticas.

F1-score medida de precisión que tiene un test, la cual realiza una ponderación sobre la precisión y exhaustividad.

fake news tipo de bulo que consiste en un contenido pseudoperiodístico difundido a través de portales de noticias, prensa escrita, radio, televisión y redes sociales y cuyo objetivo es la desinformación.

función softmax función exponencial normalizada, es decir, una generalización de la función logística, que se emplea para asignar probabilidades decimales a cada clase en un caso de dos o más clases.

Group Lasso método de análisis de regresión que realiza una selección de variables y regularización con el fin de mejorar la interpretabilidad y la exactitud del modelo.

Kappa de Fleiss medida estadística para evaluar la fiabilidad del acuerdo entre un número fijo de evaluadores al asignar clasificaciones categóricas a un número de elementos o clasificarlos.

K-Fold Cross Validation método de validación iterativo que se realiza dividiendo los datos de manera aleatoria en k grupos del mismo tamaño aproximadamente, mientras que k-1 grupos se emplean para entrenar el modelo y otro como validación.

LOOCV método de validación iterativo que se realiza empleando como conjunto de entrenamiento todas las observaciones disponibles menos una, la cual se incluye como dato de validación.

max pooling proceso de discretización basado en muestras cuyo objetivo es hacer un mues-

treo descendente de una representación de entrada, reduciendo la dimensionalidad y permitiendo que se hagan suposiciones sobre las características contenidas en las subregiones agrupadas.

misoginia actitud y comportamiento de odio, repulsión y aversión por parte de un individuo hacia las mujeres.

nodo virtual nodo no original, representa una entidad sobre una relación, como usuarios a los que comparten o retuitean.

overfitting efecto que ocurre cuando se sobre-entrena un modelo con unos ciertos datos para los que se conoce el resultado.

racismo exacerbación del sentido racial de un grupo étnico que suele motivar la discriminación o persecución de otro u otros con los que convive.

ReLU función de activación no lineal que se utiliza en redes neuronales multicapa o redes neuronales profundas.

Sentiment Analysis tarea de clasificación NLP de frases o textos en función de la connotación positiva o negativa del lenguaje ocupado en los mismos.

Stratified Sampling técnica de división realizada por investigadores que se basa en dividir los datos totales en grupos para su posterior entrenamiento y validación.

tensor matriz n-dimensional genérica que se utiliza para cálculos numéricos arbitrarios que puede ser ejecutada tanto en CPU como en GPU.

TF-IDF medida numérica que expresa cuán relevante es una palabra para un documento en una colección basándose en la frecuencia inversa.

Topic Modeling tarea de clasificación NLP de frases o textos en función de los temas abstractos que los conciernen.

word2vec técnica que utiliza un modelo de red neuronal para aprender asociaciones de palabras de un gran corpus de texto.

xenofobia fobia o rechazo al extranjero o inmigrante.

Acrónimos

BERT Bidirectional Encoder Representations from Transformers.

BiLSTM Bidirectional Long Short-Term Memory.

BOW Bag of Words.

CNN Convolutional Neural Network.

GloVe Global Vectors for Word Representation.

GRU Gated Recurrent Unit.

HDP Hierarchical Dirichlet Process.

kNN k – Nearest Neighbor.

LDA Latent Dirichlet Allocation.

LR Logistic Regression.

LSI Latent Semantic Analysis.

LSTM Long Short-Term Memory.

mBERT Multilingual BERT.

MLP MultiLayer Perceptron.

NB Naive Bayes.

NER Named Entity Recognition.

NLP Natural Language Processing.

OSCE Organización para la Seguridad y la Cooperación en Europa.

POS Part-Of-Speech.

RBF Radial Basis Function.

SGD Stochastic Gradient Descent.

SVM Support Vector Machine.

Apéndices

A

Protocolo de Revisión Bibliográfica

Con el objetivo de revisar de manera completa la literatura relacionada con la propuesta presente, se lleva a cabo una búsqueda de acuerdo al esquema que se presenta a continuación. En primer lugar, se selecciona una serie de palabras clave, que posteriormente se organizarán en una *query*, buscando así en diversas bases de datos de búsqueda bibliográfica. A estas palabras clave se le añaden variaciones como plurales o palabras relacionadas que tienen que ver con la búsqueda. La consulta enviada a Mendeley se muestra en la Tabla A.1.

Sección	Bloque
TITLE–ABS–KEY	“hate speech detection” OR “counter speech detection”
TITLE–ABS–KEY	“social network” OR “Twitter” OR “social media” OR “social graph” OR “social graphs”
ALL	“hate” OR “hater” OR “haters” OR “hateful user” OR “hateful users” OR “aggresive” OR “offensive”
ALL	“multimodal” OR “tabular”
TITLE–ABS–KEY	“misogyny” OR “against women” OR “xenophobia” OR “racism” OR “inmmigrants” OR “cyberbullying”
ALL	“BERT”

Tabla A.1: Consulta final realizada a Mendeley. Cada fila representa uno de los bloques separados por AND de la consulta, correspondiente a su sección de búsqueda.

Con ellos agrupados en diferentes bloques, se elabora una consulta para las bases de datos ResearchGate, arXiV, Mendeley y PapersWithCode, con diversas variaciones que respetan la sintaxis propia. Dados los resultados obtenidos, y considerando que el campo de la detección del discurso de odio es bastante nuevo, se considera oportuno conservar únicamente los artículos posteriores al año 2016. Con esta primera búsqueda, se obtienen 216 resultados en ResearchGate, 173 en arXiV, 250 en Mendeley y 56 en PapersWithCode, con coincidencias entre ellos. Tras eliminar los resultados duplicados, se procede a una lectura somera de cada uno de ellos para verificar que realmente sean relevantes para el proyecto, por lo que finalmente se encuentran 47 artículos.

B

Funcionamiento de BERT

BERT (Bidirectional Encoder Representations from Transformers) [2] es un modelo creado por Google, en principio, para comprender de manera más natural la intención detrás de las consultas de búsqueda. Aprende de las relaciones contextuales entre palabras en función de todo el entorno gracias a su característica bidireccional, el ocultamiento de palabras por máscaras y su asistencia al lenguaje desde la atención.

Lo que hace realmente asombroso a BERT es su arquitectura transformer bidireccional multicapa. A diferencia de otros modelos que leen la entrada de texto de manera secuencial, BERT se considera bidireccional porque utiliza un codificador que es capaz de aprender el contexto de una palabra en función de todo su entorno, aunque sería más preciso decir directamente que no es direccional. La base BERT consta de 12 capas de transformador, donde cada capa incluye una entrada de texto, la cual está formada por una secuencia de tokens, que primero se incrustan en vectores y luego se procesan en el modelo. La salida es una secuencia de vectores en los que cada vector corresponde a un token de entrada con el mismo índice.

Para superar los límites del aprendizaje contextual direccional, esto es, la predicción de la siguiente palabra nada más que a partir de la secuencia previa, BERT utiliza dos estrategias de entrenamiento. La primera es el enmascarado, que reemplaza el 15 % de las palabras de cada secuencia con un token `[MASK]` y tras ello, el modelo trata de predecir el valor original de los mismos basándose en el contexto de las que no están enmascaradas. En términos prácticos, la predicción de los tokens enmascarados requiere de una capa de clasificación que se encuentra en la parte alta de la salida del codificador. Acto seguido, se obtiene la dimensión del vocabulario con la multiplicación de los vectores de salida con la matriz de incrustación de palabras. Finalmente se obtiene la probabilidad de cada palabra en el vocabulario con la función softmax.

Otra de las estrategias de BERT es la de, en vez de predecir la palabra siguiente, predecir la siguiente oración. Para ello, BERT recibe en la fase de entrenamiento pares de oraciones y aprende a predecir si la segunda del par es también la oración posterior en la entrada original. Lo que hace realmente interesante a BERT en este caso para la tarea que nos ocupa, es cómo se procesa el texto

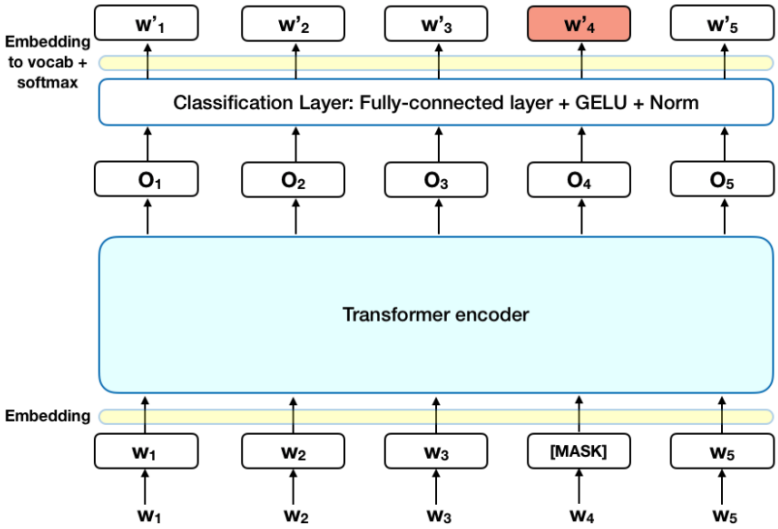


Figura B.1: Representación de la estrategia de enmascarado de BERT. Fuente [2].

antes de la entrada al modelo, de la misma manera que se presentó el concepto. Para ello se inserta un token `[CLS]` al principio de la primera oración y un token al final de cada una, `[SEP]` y se distingue de oración `A` y oración `B`. Finalmente, se agrega una incrustación a cada token con el fin de identificar su posición en toda la secuencia.

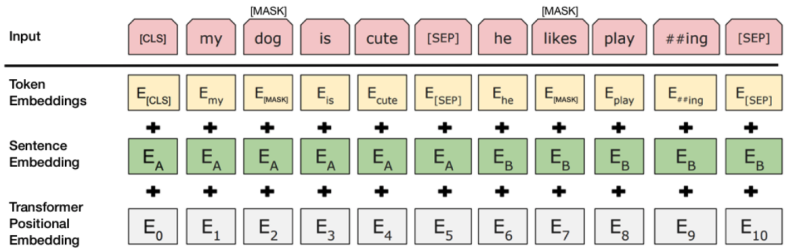


Figura B.2: Representación de la estrategia de predicción de la siguiente oración de BERT. Fuente [2].

Toda esta entrada pasa por el modelo transformer y la salida del token `[CLS]` se transforma en un vector de 2x1, lo que facilita el cálculo de la probabilidad de saber si la oración `B` está conectada a la oración `A`. Lo bueno de BERT es que es posible reajustar el modelo (Fine-Tuning) agregando una capa de clasificación en la parte superior para el token `[CLS]` de esta última estrategia comentada. Esto es muy sencillo gracias al mecanismo de atención, en el cual simplemente se pueden ajustar las entradas y salidas, a coste bajo.

C

Extracción de las características de los usuarios

En el presente anexo se detallan las tablas de la extracción de características realizada en la Sección 3.3.

author_id	identificador del usuario
author_nick	nombre de usuario
tweet_id	identificador del tweet
text	texto del tweet
favorite_count	número de favoritos
retweet_count	número de retweets
created_at	fecha de creación
source	fuente/origen
reply_to_status	id del tweet al que responde
reply_to_user	id del usuario/s al que responde
user_mentions	usuarios mencionados en el tweet
links	multimedia compartida
label	etiqueta de clasificación

Tabla C.1: Atributos de la primera extracción de tweets (4154 tweets).

Relación	Descripción
– quoted –	– citado –
– retweeted –	– retuiteado –
– shared –	– compartido –
– quoted –	– tuiteado –

Tabla C.2: Relaciones entre nodos de SocialGraph.

Nodo	Atributo	Tipo	Descripción
Usuario	user_id	int	identificador del usuario
	uname	str	nombre de perfil del usuario
	virtual	boolean	nodo virtual
	screen_name	str	nombre de usuario
	description	str	biografía o descripción
	location	str	localización si la hubiera
	verified	boolean	cuenta verificada
	profile_image_url	str	url de la imagen de perfil
	default_profile	boolean	actualización del perfil
	default_image_profile	boolean	actualización de la imagen de perfil
	geo_enabled	boolean	localización real habilitada
	created_at	datetime	fecha de creación de la cuenta
	statuses_count	int	número de tweets del usuario
	listed_count	int	número de listas
	followers_count	int	número de seguidores
	followees_count	int	número de seguidos
	favorites_count	int	número de favoritos
Multimedia	netloc	str	parte network location
	path	str	parte path jerárquica
	url	str	url completa

Continúa en la siguiente página

Tabla C.3: Atributos de SocialGraph: Parte I.

Comienza en la anterior página			
Nodo	**Atributo**	**Tipo**	**Descripción**
Tweet	user_id	int	identificador del usuario
	screen_name	str	nombre de usuario
	tweet_id	int	identificador del tweet
	tweet_text	str	texto del tweet
	tweet_creation_at	datetime	fecha de creación del tweet
	n_favs	int	número de favoritos
	n_rts	int	número de retweets
	is_rt	boolean	el tweet es retweet
	rt_id_user	int	id del usuario al que retuitea
	rt_id_status	int	id del tweet al que retuitea
	rt_text	str	texto del tweet al que retuitea
	rt_creation_at	datetime	fecha de creación del tweet al que retuitea
	rt_fav_count	int	número de favoritos (si es retweet)
	rt_rt_count	int	número de retweets (si es retweet)
	is_reply	boolean	el tweet es una respuesta
	reply_id_status	int	id del tweet al que responde
	reply_id_user	int	id del usuario al que responde
	is_quote	boolean	el tweet es una cita de otro
	quote_id_status	int	id del tweet al que cita
	quote_id_user	int	id del usuario al que cita
	quote_text	str	texto del tweet al que cita
	quote_creation_at	datetime	fecha de creación del tweet al que cita
	quote_fav_count	int	número de favoritos al que cita
	quote_rt_count	int	número de retweets al que cita

Tabla C.4: Atributos de SocialGraph: Parte II.

Medida	**Descripción**
betweenness	computa el camino más corto a la centralidad del grafo
eigenvector	medida de la influencia de un nodo en la red
in-degree	número de aristas apuntando al nodo
out-degree	numero de aristas apuntando afuera del nodo
clustering	fracción de pares de nodos vecinos adyacentes entre sí
degree	número de aristas adyacentes al nodo
closeness	distancia promedio de todos los nodos alcanzables al nodo

Tabla C.5: Medidas de centralidad de SocialGraph.

Atributo	Tipo	Clasificador	Nombre del clasificador	Descripción
categories_profile _image_url	dict(dict(categoría, puntuación, jerarquía=None))	Cliente Watson Visual Recognition (IBM)	VisualRecognitionV3	categorías de la imagen de perfil del usuario
negativos positivos neutros	int	Clasificador sentiment analysis (transformers)	finiteautomata/beto-sentiment-analysis	número de negativos número de positivos número de neutros
negativos_score positivos_score neutros_score	float	Clasificador sentiment analysis (transformers)	finiteautomata/beto-sentiment-analysis	puntuación de negativos puntuación de positivos puntuación de neutros
hate non_hate	int	Clasificador propio	HaterBERT (ver 3.2.2)	número de tweets de odio número de tweets de no odio
hate_score non_hate_score	float	Clasificador propio	HaterBERT (ver 3.2.2)	puntuación de odio puntuación de no odio
top_categories	dict(categoría(str), cuenta(int))	Clasificador de categorías en español (Librería Python)	subject_classification _spanish	top 15 categorías de los tweets
misspelling_counter	int	Corrector ortográfico en español	pyspellchecker (Librería Python)	número de erratas que el usuario comete

Tabla C.6: Características extra de SocialGraph (Parte I).

Atributo	Tipo	Descripción
status_retrieving	int	número de tweets guardados
status_start_day	datetime	fecha de comienzo de extracción de los tweets
status_end_day	datetime	fecha de fin de extracción de los tweets
status_average_tweets_per_day	float	media de tweets por día
activity_hourly_X	int	número de tweets a cada hora del día, 24 atributos siendo X∈[00-23]
activity_weekly_X	int	número de tweets por día de la semana, 7 atributos siendo X∈[0-6]
rt_count	int	numero de retweets del total guardados
geo_enabled_tweet_count	int	número de tweets con geolocalización activada
num_hashtags	int	número de hashtags utilizados
num_mentions	int	número de menciones
num_urls	int	número de dominios compartidos por el usuario
baddies	list(str)	palabras malsonantes o insultos utilizados por el usuario
n_baddies	int	número de baddies
n_baddies_tweet	float	número de baddies por tweet
len_status	float	longitud media de los tweets
times_user_quotes	int	número de veces que cita a otros usuarios
num_rts_to_tweets	int	número de veces que los tweets del usuario son retuiteados
num_favs_to_tweets	int	número de veces que los tweets del usuario son favoriteados
leet_counter	int	número de veces que el usuario utiliza el alfabeto leet

Continúa en la siguiente página

Tabla C.7: Características extra de SocialGraph (Parte 2).

Comienza en la anterior página		
Atributo	**Tipo**	**Descripción**
top_languages	dict(idioma(str), cuenta(int))	top 5 de idiomas más usados por el usuario por número de tweets
top_sources	dict(vía(str), cuenta(int))	top 5 de vías a través de la cual tuitea por número de tweets
top_places	dict(lugar(str), cuenta(int))	top 10 de lugares más habilitados por el usuario por número de tweets
top_hashtags	dict(hashtag(str), cuenta(int))	top 10 de hashtags más utilizados por el usuario por número de tweets
top_retweeted_users	dict(usuario(str), cuenta(int))	top 5 de usuarios más retuiteados por el usuario por número de tweets
top_mentioned_users	dict(usuario(str), cuenta(int))	top 5 de usuarios más mencionados por el usuario por número de tweets
top_referenced_domains	dict(dominio(str), cuenta(int))	top 6 de dominios más compartidos por el usuario por número de tweets

Tabla C.8: Características extra de SocialGraph (Parte III).

D

Análisis de las características de los usuarios

En este apéndice se detalla el análisis realizado para Little743 con el fin de valorar las características extraídas para SocialGraph. Para ello se valorarán en diferentes clasificadores que se mostrarán en el Capítulo 4.

Con el fin de encontrar usuarios que utilicen textualmente palabras malsonantes, se realiza previamente una recolección de tweets que utilicen este tipo de léxico. El léxico utilizado es el provisto por Plaza-del-Arco en [53], el cual se puede encontrar en su GitHub.

Este léxico está formado por los siguientes archivos:

- `xenophobia_lexicon.txt`: léxico de odio hacia inmigrantes. Contiene 44 palabras.
- `immigrant_lexicon.txt`: léxico de odio que se refiere hacia la nacionalidad de un inmigrante. Contiene 250 palabras.
- `misogyny_lexicon.txt`: léxico de odio hacia las mujeres. Contiene 183 palabras.
- `insults_lexicon.txt`: léxico de insultos generales.

Es preciso destacar que comprende una amplia variedad de palabras procedentes de diversos lugares del mundo, tanto en castellano como en latioamericano.

Además de la recogida por léxico, se realiza también por temas polémicos con la intención de encontrar temas históricos relevantes de los que se hable hoy en día. Para ello, se parsea la página Wikipedia:List of controversial issues con BeautifulSoup traducida al español. Se guarda con ello un diccionario de páginas con sus correspondientes enlaces y tras ello, se limpian las claves del mismo. Estas claves sirven para la extracción de tweets, filtrando los tweets escritos en español.

Esta recogida de tweets está comprendida entre el 17 y 19 de marzo de 2021. En total se recogen 20119 tweets con sus correspondientes campos. Como este análisis se realizó dos meses más tarde se filtran los usuarios comprobando los que quedaban existentes, de los cuales finalmente se quedaron en 10543. Tanto como si es un tweet, un retweet o una cita, éstos se filtran por el modelo base, HaterBERT, con el fin de recoger los candidatos a odio. Para afinar la limpieza de la recogida, se pasan por el clasificador `finiteautomata/beto-sentiment-analysis`, y se guardan los que posean un valor mayor a 0.75 de puntuación de connotación negativa. Tras ello, se quedan unos 539

usuarios. Gracias a la colaboración de la Oficina Nacional Contra los Delitos de Odio, se añaden 52 usuarios más que se disponen hasta la fecha, siendo éstos autores de tweets de odio.

Con el objetivo de encontrar relaciones entre los mismos y ampliar el grafo con usuarios que no sean de odio para encontrar diferencias, se busca de manera aleatoria en 5 de sus followers 5 followers más y así recursivamente en los nuevos. Este proceso se realiza 6 veces. Finalmente se obtuvienen 743 usuarios.

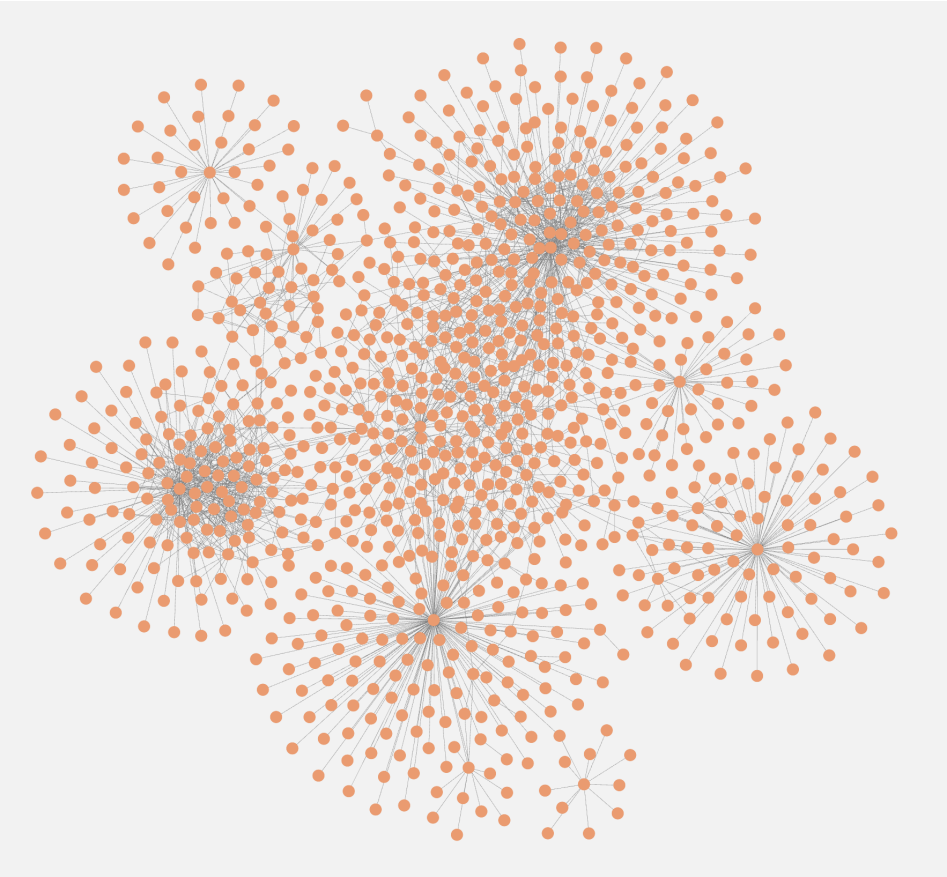


Figura D.1: Vista global de Little743. Se observa que esta red está fuertemente conectada.

Estos usuarios se introducen en SocialGraph, guardando 200 tweets por cada uno de ellos. De la misma manera que en el diseño (ver Capítulo 3), se extraen sus características asociadas (ver Anexo C).

Con el objetivo de valorar estas características se crea otra nueva, que sirve como etiqueta para las pruebas futuras sobre Little743. Esta nueva variable será `hater`. Para ello se ilustra la distribución del odio en la Figura D.2 y se decide que aquellos usuarios que tengan más de un 5 % de odio en sus tweets son candidatos a `hater`. A través de esto, se calcula, ayudándose de herramientas de NetworkX, si cada uno de estos usuarios tiene al menos un vecino hater, por lo que se crea la variable `vecino_hater`. Estas se incluyen en el Anexo E.

Finalmente, se obtienen 335 haters y 407 usuarios normales.

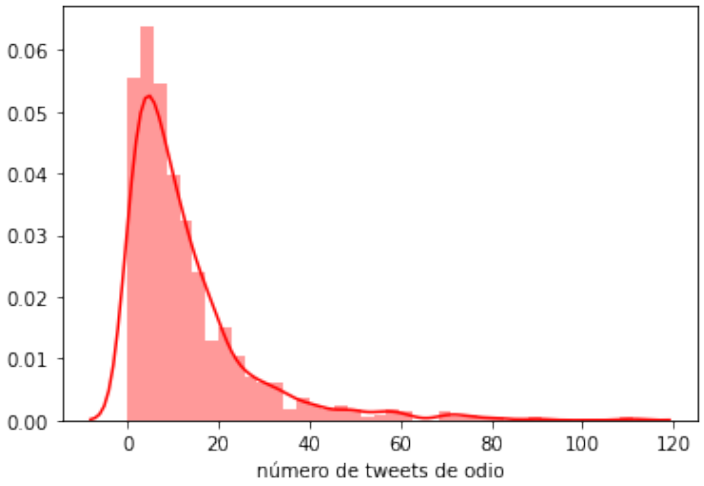


Figura D.2: Distribución del odio en la red Little743, se observa el número de tweets de hate en función de la densidad. Se elije un 5 % como corte.

Con el fin de extraer conclusiones acerca de estas variables, se realiza un análisis dependiendo de si el usuario es hater o normal. A continuación se representan algunas de las variables en gráficas que se analizan de cara a entender el comportamiento de los usuarios. Los usuarios haters se representan en color rojo, mientras que los no haters en color azul.

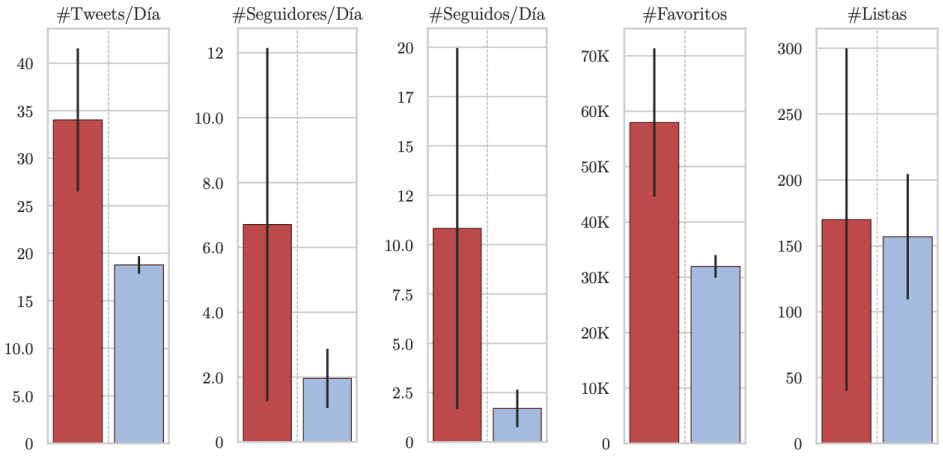


Figura D.3: Gráficas que representan el comportamiento de los usuarios haters (rojo) frente a los normales (azul). Se observan la media de tweets por día, los seguidores por día, los seguidos por día, el número de favoritos y número de listas en las que están unidos.

En la Figura D.3 se puede observar que los usuarios haters parecen tener un comportamiento más activo que los que no lo son. Si bien tampoco se puede determinar que se comporten como *spammers* en todas las ocasiones a modo genérico, tienen más actividad que los usuarios normales.

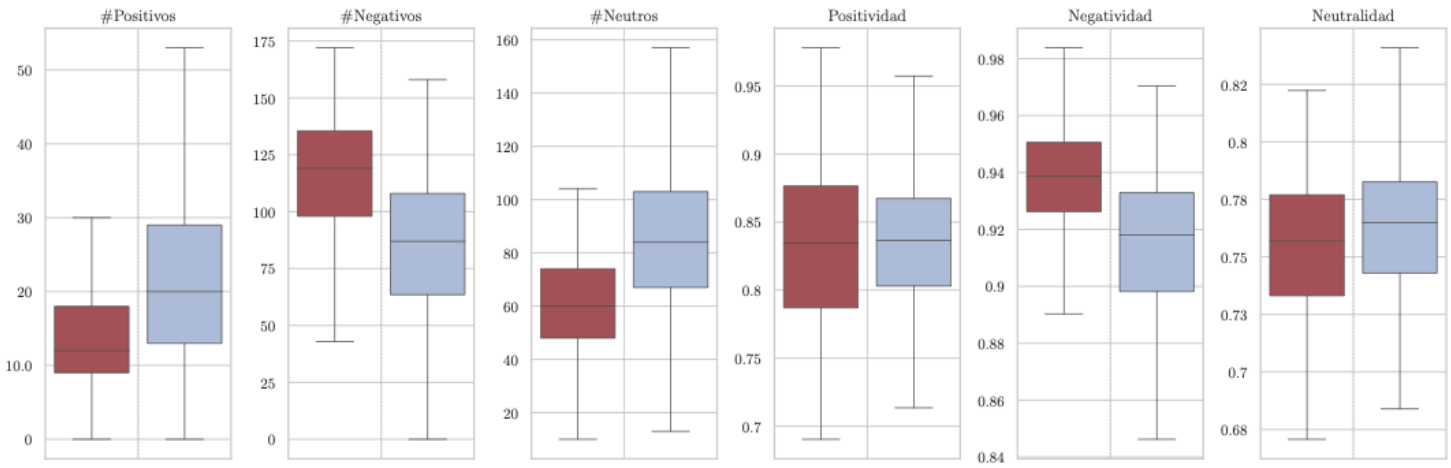


Figura D.4: Gráficas que representan el comportamiento de los usuarios haters (rojo) frente a los normales (azul). Se observan el número promedio de tweets con connotación positiva, negativa y neutra, además de la media de puntuaciones correspondientes.

En la Figura D.4 se puede ver que por norma general los usuarios haters son menos positivos, si bien se refleja que son mucho menos neutrales y ofrecen su opinión tanto en la parte positiva como en la negativa. Además, son bastante más negativos que los usuarios normales. Estos usuarios muestran un claro comportamiento distinto a los haters.

En la Figura D.5 se confirma que la estructura de la red importa [49]. Los usuarios haters parecen estar más interconectados que los usuarios normales.

Mientras tanto, en la Figura D.6 se ratifica que los usuarios no se comportan como *spammers*, ya que ni comparten demasiados enlaces ni tampoco utilizan más hashtags que los usuarios normales. Si que es destacable que se muestra que los haters son más activos por que siguen más y son más seguidos que los normales, además de que responden mucho más a otros usuarios, esto es, contestando a otros por sus tweets. Parece que el tamaño del tweet es relativamente despreciable en cuestiones hater o no hater.

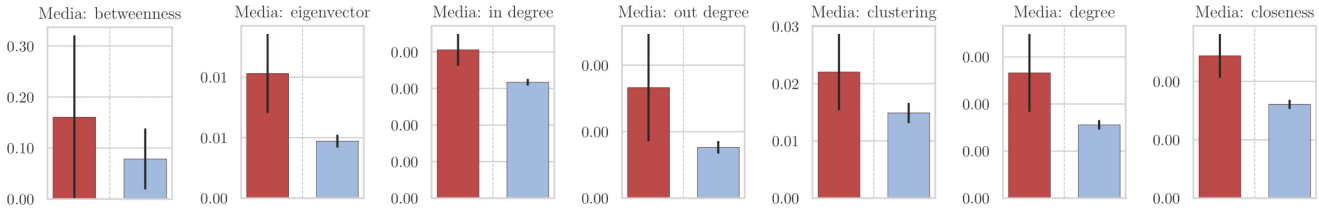


Figura D.5: Gráficas que representan el comportamiento de los usuarios haters (rojo) frente a los normales (azul). Se muestra la media de las medidas de centralidad recogidas.

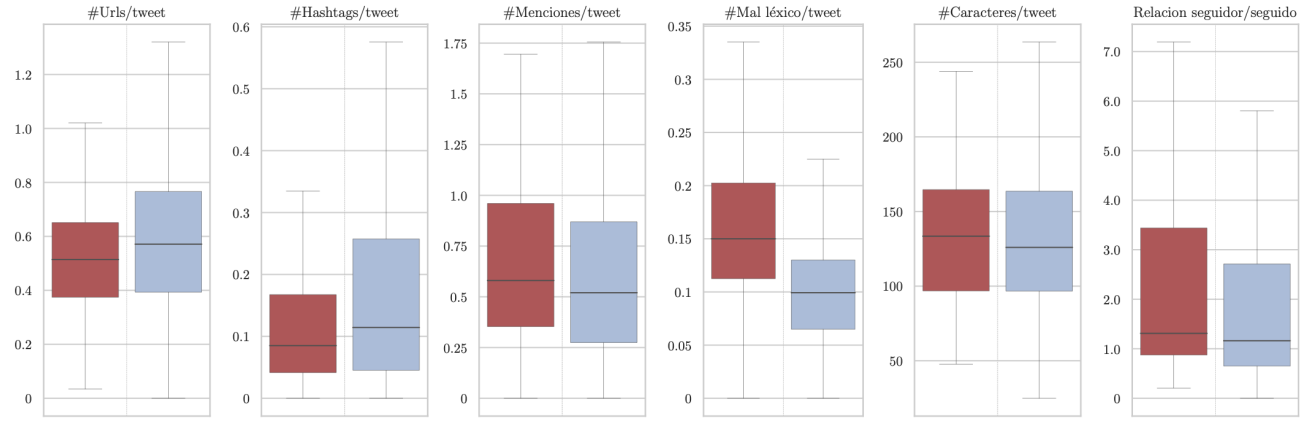


Figura D.6: Gráficas que representan el comportamiento de los usuarios haters (rojo) frente a los normales (azul). Se muestra el número de urls, hashtags, menciones, mal léxico y caracteres por tweet, además de la relación follow por follow.

En cuanto a la Figura D.7 se observa que hay tres regiones diferenciadas de usuarios, si bien los haters son más nuevos que los normales.

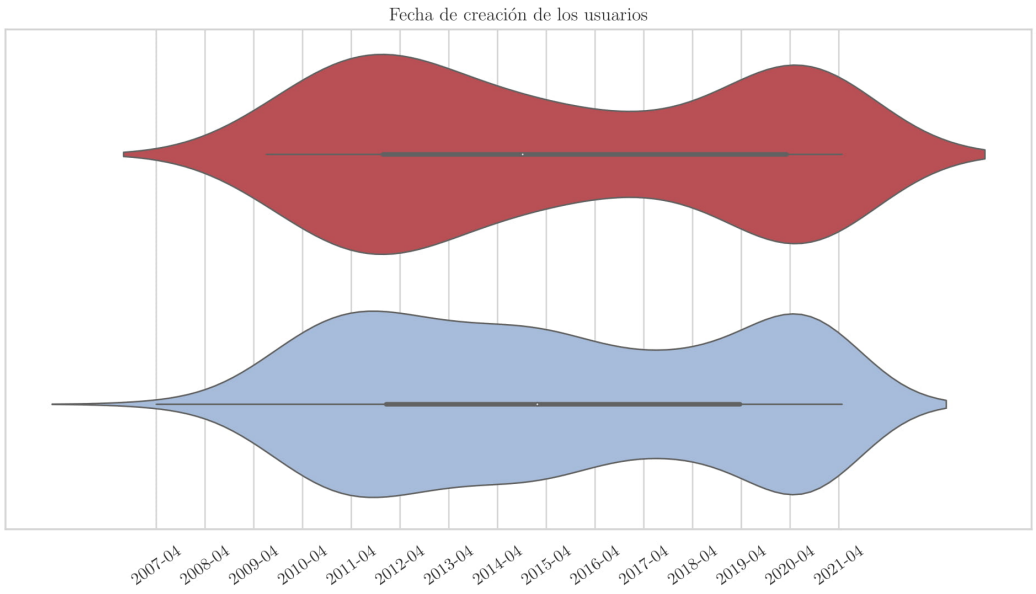


Figura D.7: Gráficas que representan el comportamiento de los usuarios haters (rojo) frente a los normales (azul). Se muestra la distribución de la fecha de creación de los usuarios.

Tras realizar la manipulación de las características de los usuarios (Anexo E) se obtienen otras conclusiones que se reflejan también en este apartado por razones de interés.

En concreto, en la Figura D.8 se observa que los perfiles de los usuarios haters no se identifican tanto como personas, esto puede significar que se creen cuentas falsas con nombres falsos o bajo el nombre de personajes abstractos. Además, en la Figura D.9 se muestra que los usuarios haters, cuando utilizan hashtags, son éstos en mayor medida de política. Al contrario que en temas de deportes y actualidad, parece que el odio puede pronunciarse más a raíz de eventos políticos.

Es preciso destacar que en esta red social no se han recogido usuarios verificados, por lo que esta variable no se ha podido corroborar en este análisis. Sin embargo, Mathew *et al.* [7] muestran que esto puede ser relevante a la hora de identificar odio, si bien los usuarios haters no son generalmente verificados, también son relevantes en términos de audiencia.

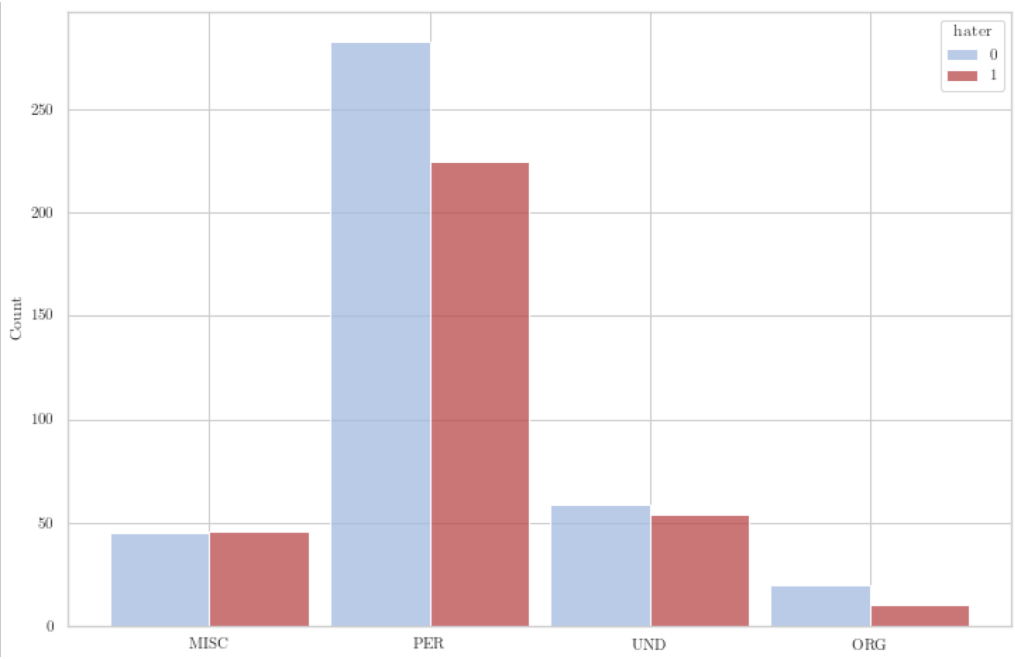


Figura D.8: Gráfica que representa el comportamiento de los usuarios haters (rojo) frente a los normales (azul). En concreto se representa el análisis NER (Named Entity Recognition) en el cual se observan tres categorías: MISC (industrias/empresas), PER (persona), UND (indeterminado) y ORG(organización)

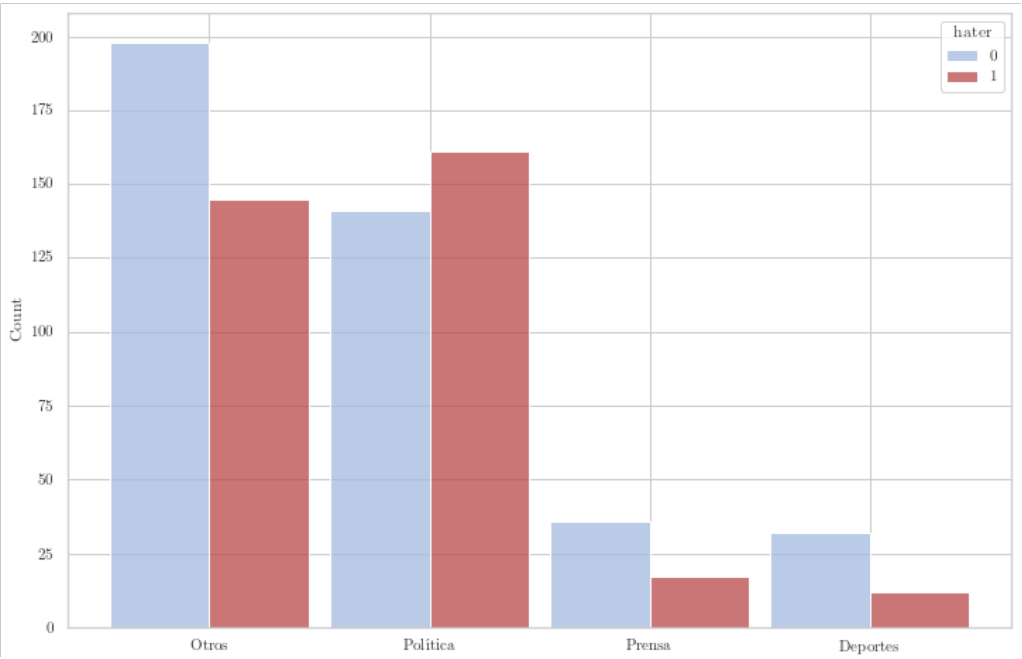


Figura D.9: Gráfica que representa el comportamiento de los usuarios haters (rojo) frente a los normales (azul). Se diferencia claramente que los temas políticos sobresalen frente a otros en los usuarios haters.

E

Manipulación y normalización de las características de los usuarios

En este anexo se explica la transformación realizada de los atributos de los usuarios (ver Anexo C) dado el formato en el que se encuentran, ya que es estrictamente necesario prepararlos como entrada para el modelo de SocialHaterBERT.

Con el fin de presentar una mejor división del trabajo, se exponen las características en dos grupos: variables categóricas y variables numéricas. Además se recogen las que tienen información consistente de cara al modelo. Se entiende por esta información aquellas que no son meramente informativas al investigador como `status_start_day` o que sirven para calcular otras como `status_retrieving`.

A continuación se detallan en las tablas E.1, E.2, E.3 y E.4 las transformaciones realizadas con sus métodos asociados. En la Figura E.1 se muestra la matriz de correlación de los hashtags más utilizados por los usuarios. Un análisis previo que ha sido útil para hacer Topic Modeling manual, ya que en este caso los hashtags no son interpretables por los modelos HDP , LSI o LDA . En el caso de los dominios utilizados por los usuarios se pensó en realizar una búsqueda por wikipedia, obteniendo las palabras claves de la descripción del dominio y siendo estas categorizadas en función de un modelado de temas manual.

Variable categórica	Variable(s) previas	Grupo	Método	Clases	Descripción
verificado	NC	perfil	clasificación booleana	0: No, 1: Sí	el usuario está verificado
hater	NC	actividad	clasificación booleana	0: No, 1: Sí	el usuario tiene más de un 5 % de tweets de odio
vecino_hater	NC	actividad	clasificación booleana	0: No, 1: Sí	el usuario tiene al menos algún vecino con más de un 5 % de tweets de odio
profile_changed	default_profile	perfil	clasificación booleana	0: No, 1: Sí	el usuario actualizó alguna vez su perfil
clase_NER	screen_name + uname	perfil	búsqueda de NER tag (Spacy)	0: PER, 1: MISC, 2: ORG, 3: UND	tipo de nombre
clase_DESCR	description	perfil	limpieza (NLTK) + Topic Modeling (Gensim)	0: opinión, 1: estudios, 2: política, 3: actividades	tipo de descripción
clase_LOC	location	perfil	limpieza + diccionario propio + pycountry	0-19: Zonas del mundo y provincias en caso de ser España	zona geográfica habilitada por el usuario
clase_FECHA	created_at	perfil	división en tres regiones según Figura D.7	0: <2015, 1: [2015-2019], 2: >2019	momento temporal de creación del usuario
clase_IMG	categories_profile_image_url	perfil	Topic Modeling (Gensim)	0: people, 1: clothing, 2: building, 3: animal, 4: nature, 5: technology, 6: sports, 7: objects, 8: food	tipo de imagen de perfil

Continúa en la siguiente página

Tabla E.1: Detalle de las transformaciones realizadas para las variables categóricas (Parte I). NC = no cambia.

Comienza en la anterior página

Variable categórica	Variable(s) previas	Grupo	Método	Clases	Descripción
clase_HASHTAGS	top_hashtags	actividad	Matriz correlación + Topic Modeling (manual)	0: política, 1: prensa, 2: deportes, 3: otros	tipo de hashtag
clase_CATS	top_categories	actividad	Topic Modeling (Gensim)	0: españa, 1: cultura, 2: arte, 3: sociedad 4: viñetas, 5: cataluña, 6: artes gráficas, 7: dibujo, 8: opinión, 9: ilustración, 10: política, 11: otros	categorías más repetidas por el usuario en sus tweets
clase_DOMS	top_referenced_domains	actividad	wikipedia + Topic Modeling (manual)	0: redes sociales, 1: información, comunicación y actualidad, 2: entretenimiento	tipo de dominio más compartido por el usuario
clase_RTSCAT	top_retweeted_users	actividad	Topic Modeling (Gensim)	0: españa, 1: cultura, 2: arte, 3: sociedad 4: viñetas, 5: cataluña, 6: artes gráficas, 7: dibujo, 8: opinión, 9: ilustración, 10: política, 11: otros	tipo de usuario al que más retuitea
clase_MENCAT	top_mentioned_users	actividad	Topic Modeling (Gensim)	0: españa, 1: cultura, 2: arte, 3: sociedad 4: viñetas, 5: cataluña, 6: artes gráficas, 7: dibujo, 8: opinión, 9: ilustración, 10: política, 11: otros	tipo de usuario al que más menciona

Tabla E.2: Detalle de las transformaciones realizadas para las variables categóricas (Parte II). NC = no cambia.

Variable numérica	Variable(s) previas	Grupo	Método	Descripción
n_LESP	top_languages	actividad	Función propia	porcentaje de tweets de odio en español
n_LENG	top_languages	actividad	Función propia	porcentaje de tweets de odio en inglés
n_LOTR	top_languages	actividad	Función propia	porcentaje de tweets de odio en otro idioma (no español o inglés)
activity_hourly_X	NC	actividad	Función propia	porcentaje de tweets a cada hora
activity_weekly_X	NC	actividad	Función propia	porcentaje de tweets por día de la semana
negativos	NC	actividad	Función propia	porcentaje de tweets con connotación negativa
positivos	NC	actividad	Función propia	porcentaje de tweets con connotación positiva
neutros	NC	actividad	Función propia	porcentaje de tweets con connotación neutra
n_hate	NC	actividad	Función propia	porcentaje de tweets de odio
n_nohate	NC	actividad	Función propia	porcentaje de tweets de no odio
n_baddies	NC	actividad	Función propia	porcentaje de baddies por tweet
eigenvector	NC	centralidad	–	puntuación eigenvector
in_degree	NC	centralidad	–	puntuación in degree
out_degree	NC	centralidad	–	puntuación out degree
degree	NC	centralidad	–	puntuación degree
clustering	NC	centralidad	–	puntuación clustering
closeness	NC	centralidad	–	puntuación closeness
betweenness	NC	centralidad	StandardScaler	número de caminos más cortos hasta él
status_average_tweets_per_day	NC	actividad	StandardScaler	número de veces promedio que tuitea al día
times_user_quotes	NC	actividad	StandardScaler	número de veces que cita a otros

Continúa en la siguiente página

Tabla E.3: Detalle de las transformaciones realizadas para las variables numéricas (Parte I). NC = no cambia.

Comienza en la anterior página

Variable numérica	Variable(s) previas	Grupo	Método	Descripción
negativos_score	NC	actividad	–	puntuación media de tweets negativos
positivos_score	NC	actividad	–	puntuación media de tweets positivos
neutros_score	NC	actividad	–	puntuación media de tweets neutros
hate_score	NC	actividad	–	puntuación media de tweets de odio
no_hate_score	NC	actividad	–	puntuación media de tweets de no odio
statuses_count	NC	actividad	StandardScaler	número de tweets totales
followers_count	NC	actividad	StandardScaler	número de followers totales
followees_count	NC	actividad	StandardScaler	número de followees totales
favorites_count	NC	actividad	StandardScaler	número de favoritos totales
listed_count	NC	actividad	StandardScaler	número de listas en las que está
num_hashtags	NC	actividad	StandardScaler	número de hashtags utilizados
rt_count	NC	actividad	StandardScaler	número de retweets totales
num_mentions	NC	actividad	StandardScaler	número de menciones que realiza
num_urls	NC	actividad	StandardScaler	número de urls compartidas
len_status	NC	actividad	StandardScaler	longitud media de los tweets
num_rts_to_tweets	NC	actividad	StandardScaler	número de veces que sus tweets son retuiteados
num_favs_to_tweets	NC	actividad	StandardScaler	número de veces que sus tweets son favoriteados
misspelling_counter	NC	actividad	StandardScaler	número de veces que comete faltas o erratas
leet_counter	NC	actividad	StandardScaler	número de veces que utiliza el alfabeto leet

Tabla E.4: Detalle de las transformaciones realizadas para las variables numéricas (Parte II). NC = no cambia.

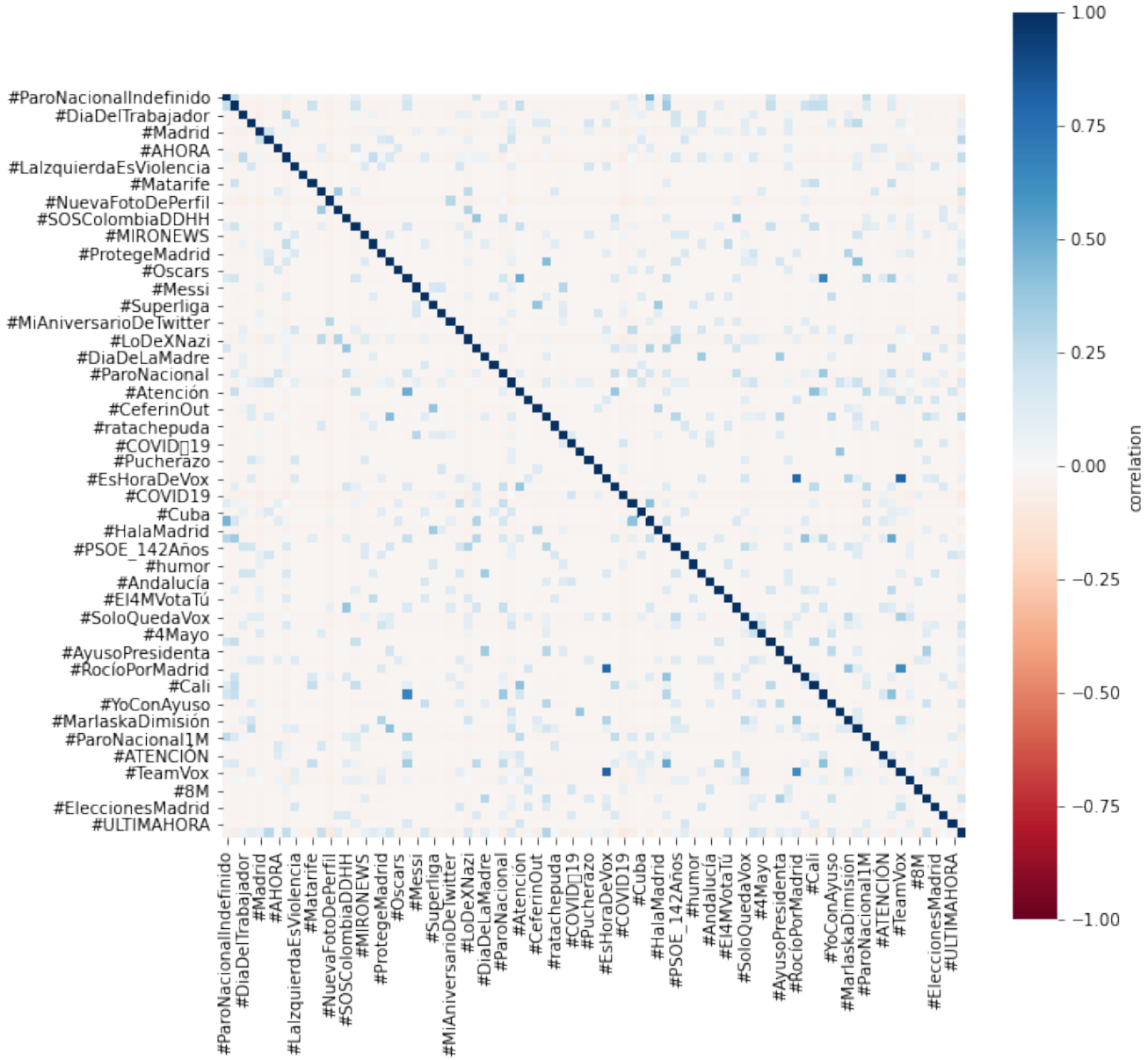


Figura E.1: Matriz de correlación de hashtags que más utilizan los usuarios en sus tweets. Se observa que los hashtags de los mismos temas están conectados entre sí.

Apéndice F

Diagrama de Gantt del proyecto

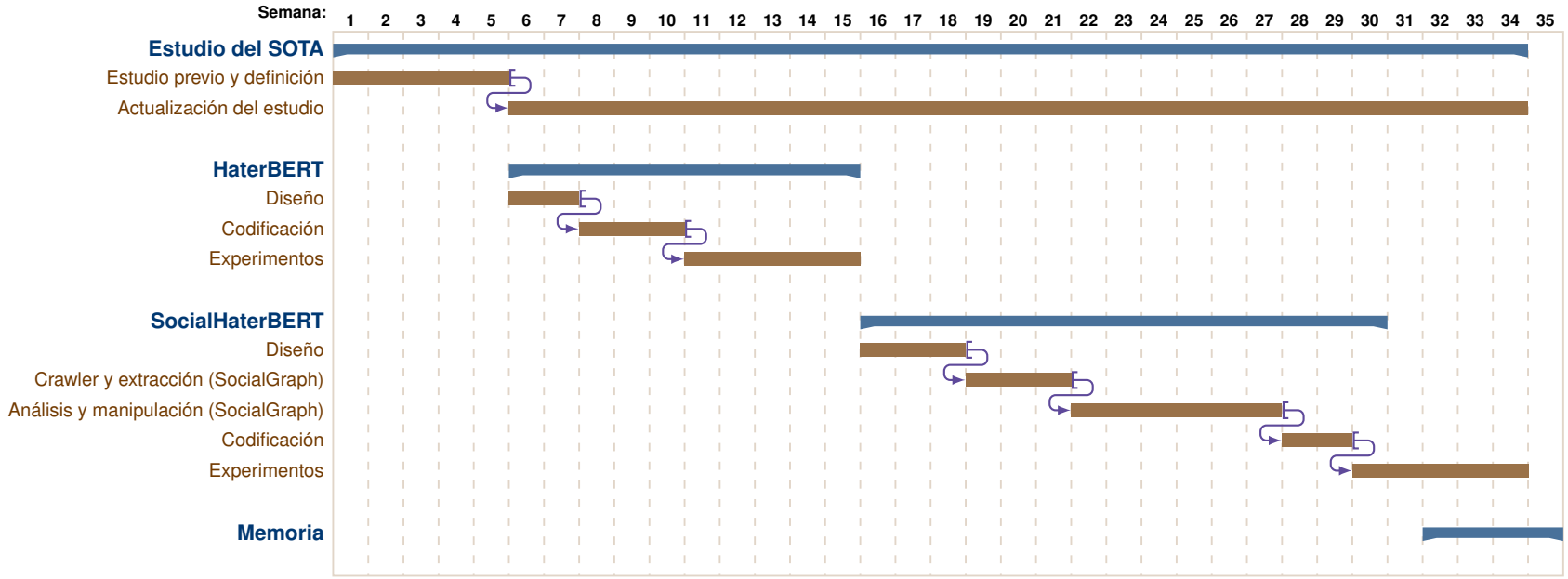


Figura F.1: Diagrama de Gantt del proyecto, donde se detalla la realización y distribución del proyecto a lo largo del mismo.

G

Pruebas con otros datasets de la literatura

En este anexo se detallan las pruebas realizadas con datasets en inglés.

	Train		Test	
	Mujeres	Inmigrantes	Mujeres	Inmigrantes
Hate	44.44 %	39.76 %	42 %	42 %
Non hate	55.56 %	60.24 %	58 %	58 %

Tabla G.1: División del dataset de HatEval en inglés

En la Tabla G.2 se realiza una comparativa con el estudio de MacAvaney *et al.* [35]. En él se utiliza el dataset de HatEval en inglés, el cual está formado por 13000 tweets. El detalle de este dataset se muestra en la Tabla G.1. Es preciso destacar que su estudio no se detalla la configuración de hiperparámetros pero en esta propuesta se eligen 3 épocas, un batch size de 16 y un learning rate de $2e^{-}5$.

Exp.	Modelo	Autor	Precisión	Recall	F1	AUC	Accuracy	Errores
–	BERT	MacAvaney *et al.* [35]	–	–	0.7481	–	0.7470	–
HAT-1206	HaterBERT	Esta propuesta	0.7816	0.7887	**0.7799**	0.7877	0.7810	219

Tabla G.2: Resultados y comparativa con [35].

	Total
Train	19832
Test	2475
Validation	2475
Total	24783

(a) División 80–10–10.

	Total
Hate	1430
Offensive	19190
Neither	4143
Total	24783

(b) Clases del dataset.

Tabla G.3: División del dataset de Davidson.

En la Tabla G.4 se realiza una comparativa con el estudio de Mozafari *et al.* [29]. Se realizan diversas pruebas con la CNN y sin ella para ver su utilidad. Es preciso destacar que se utiliza el dataset de Davidson, cuyo detalle se describe en la Tabla G.3. Es preciso destacar que tanto en [29] como en esta propuesta se eligen 3 épocas, un batch size de 32 y un learning rate de $2e^{-}5$. Se observa que en este caso la utilización de una CNN puede mejorar el problema propuesto. Si bien en este caso no se ha alcanzado la mejor puntuación para el inglés.

Exp.	Tipo val.	Modelo	Autor	Precisión	Recall	F1	AUC	Accuracy
–	weighted	BERT (uncased) + CNN	Mozafari *et al.* [29]	0.92**	0.92**	0.92**	–	–
HAT-1208*	macro	BERT (uncased) + CNN	Mozafari *et al.* [29]	0.7812	0.7723	0.7743	–	–
HAT-1301	macro	HaterBERT	Esta propuesta	0.7268	0.6470	0.6771	0.6473	0.9534
HAT-1302	macro	HaterBERT(uncased) + CNN	Esta propuesta	0.7643	0.6424	0.6677	0.6566	0.9378
HAT-1303	macro	HaterBERT + CNN	Esta propuesta	0.7677	0.6934	0.7231	0.6725	0.9494

Tabla G.4: Resultados y comparativa con [29].(*) Esta prueba se ha realizado con su código para observar qué macro se obtenía. Los valores con ** en los decimales no se conocen.


UAM
UNIVERSIDAD AUTONOMA
DE MADRID