UNIVERSIDAD DE COSTA RICA
SISTEMA DE ESTUDIOS DE POSGRADO

COMPARACIÓN DE MODELOS DE IDENTIFICACIÓN AUTOMÁTICA DE ODIO EN COMENTARIOS DE MICROTEXTOS EN ESPAÑOL

Trabajo final de investigación aplicada sometido a la consideración de la Comisión del Programa de Estudios de Posgrado en Computación e Informática para optar al grado y título de Maestría Profesional en Computación e Informática

NOELIA NAVARRO MURILLO

Ciudad Universitaria Rodrigo Facio, Costa Rica

2021

Dedicatoria

Le dedico este trabajo a mi madre, padre, hermano y hermana por su apoyo incondicional.

Agradecimientos

Le agradezco al Dr. Edgar Casasola Murillo, por su valiosa guía, tiempo, apoyo y paciencia durante todo este proceso del TFIA.

Gracias a todas las personas que ayudaron en la investigación, por su tiempo, aportes y observaciones realizadas.

Gracias a la vida y a Dios por permitirme concluir esta etapa.

Este trabajo final de investigación aplicada fue aceptado por la Comisión del Programa de Estudios de Posgrado en Computación e Informática de la Universidad de Costa Rica, como requisito parcial para optar al grado y título de Maestría Profesional en Computación e Informática

VANESSA SMITH CASTRO (FIRMA)
Firmado digitalmente por VANESSA SMITH CASTRO (FIRMA)
Fecha: 2021.09.21 19:38:38 -06'00'

Dra. Vanessa Smith Castro
Representante del Decano
Sistema de Estudios de Posgrado

EDGAR ENRIQUE CASASOLA MURILLO (FIRMA)
Firmado digitalmente por EDGAR ENRIQUE CASASOLA MURILLO (FIRMA)
Fecha: 2021.09.20 20:28:21 -06'00'

Dr. Edgar Casasola Murillo
Profesor Guía

GABRIELA MARIN RAVENTOS (FIRMA)
Firmado digitalmente por GABRIELA MARIN RAVENTOS (FIRMA)
Fecha: 2021.09.21 18:12:19 -06'00'

Dra. Gabriela Marín Raventós
Lectora

GUSTAVO LOPEZ HERRERA (FIRMA)
Firmado digitalmente por GUSTAVO LOPEZ HERRERA (FIRMA)
Fecha: 2021.09.22 07:55:44 -06'00'

Dr. Gustavo López Herrera
Lector

ALLAN FRANCISCO BERROCAL ROJAS (FIRMA)
PERSONA FISICA, CPF-02-0553-0149.
Fecha declarada: 23/09/2021 10:41:13 AM
Esta representación visual no es fuente de confianza. Valide siempre la firma.

Dr. Allan Berrocal Rojas
Representante de la Directora del Programa de Posgrado en Computación e Informática

NOELIA MILENA NAVARRO MURILLO (FIRMA)
PERSONA FISICA, CPF-03-0450-0885.
Fecha declarada: 20/09/2021 06:35:07 PM
Esta representación visual no es fuente de confianza. Valide siempre la firma.

Noelia Navarro Murillo
Sustentante

Índice general

Resumen en español

Esta investigación se enfoca en la detección de odio en comentarios en español extraídos de Twitter. Se analiza la efectividad de los modelos de SVM (*Support Vector Machine*) y CNN (Convolutional Neural Networks) en la identificación del odio en los textos. Se analizan los resultados con características de frecuencia de términos y *word embeddings* para SVM, así mismo el efecto de aplicar sobremuestro. Mientras, para las CNN se utilizan los *word embeddings*. La investigación provee un *corpus* de textos anotados, para el cual se utilizó la guía de anotación para la identificación de odio en el texto. Este trabajo busca colaborar con la investigación en español sobre la detección del odio, proveyendo el *corpus* anotado y el análisis de efectividad de los modelos e SVM y CNN para la identificación del odio.

Resumen en inglés

This research focuses on detecting hate in comments in Spanish taken from Twitter. The effectiveness of the SVM (*Support Vector Machine*) and CNN (Convolutional Neural Networks) models in identifying hate in texts is analyzed. The results are analyzed with characteristics of frequency of terms and *word embeddings* for SVM, as well as the effect of oversampling. Meanwhile, for CNN *word embeddings* are used. The research provides a *corpus* of texts annotated by people following an annotation guide for the manual identification of hate speech. This work has the aim to collaborate with the Spanish research on hate speech. It provides the annotated *corpus* and the performance analysis for the SVM and CNN models used to identify hate in text.

Lista de tablas

Lista de figuras

UNIVERSIDAD DE COSTA RICA

SEP Sistema de Estudios de Posgrado

Autorización para digitalización y comunicación pública de Trabajos Finales de Graduación del Sistema de Estudios de Posgrado en el Repositorio Institucional de la Universidad de Costa Rica.

Yo, Noelia Navarro Murillo, con cédula de identidad 304500885, en mi condición de autor del TFG titulado COMPARACION DE MODELOS DE IDENTIFICACION AUTOMÁTICA DE ODIO EN COMENTARIOS DE MICROTEXTOS EN ESPAÑOL

Autorizo a la Universidad de Costa Rica para digitalizar y hacer divulgación pública de forma gratuita de dicho TFG a través del Repositorio Institucional u otro medio electrónico, para ser puesto a disposición del público según lo que establezca el Sistema de Estudios de Posgrado. **SI** [X] **NO** * [ ]

***En caso de la negativa favor indicar el tiempo de restricción: _____ año (s).**

Este Trabajo Final de Graduación será publicado en formato PDF, o en el formato que en el momento se establezca, de tal forma que el acceso al mismo sea libre, con el fin de permitir la consulta e impresión, pero no su modificación.

Manifiesto que mi Trabajo Final de Graduación fue debidamente subido al sistema digital Kerwá y su contenido corresponde al documento original que sirvió para la obtención de mi título, y que su información no infringe ni violenta ningún derecho a terceros. El TFG además cuenta con el visto bueno de mi Director (a) de Tesis o Tutor (a) y cumplió con lo establecido en la revisión del Formato por parte del Sistema de Estudios de Posgrado.

NOELIA MILENA NAVARRO MURILLO (FIRMA)
PERSONA FISICA, CPF-03-0450-0885.
Fecha declarada: 20/09/2021 06:36:22 PM
Esta representación visual no es fuente
de confianza. Valide siempre la firma.

FIRMA ESTUDIANTE

Nota: El presente documento constituye una declaración jurada, cuyos alcances aseguran a la Universidad, que su contenido sea tomado como cierto. Su importancia radica en que permite abreviar procedimientos administrativos, y al mismo tiempo genera una responsabilidad legal para que quien declare contrario a la verdad de lo que manifiesta, puede como consecuencia, enfrentar un proceso penal por delito de perjurio, tipificado en el artículo 318 de nuestro Código Penal. Lo anterior implica que el estudiante se vea forzado a realizar su mayor esfuerzo para que no sólo incluya información veraz en la Licencia de Publicación, sino que también realice diligentemente la gestión de subir el documento correcto en la plataforma digital Kerwá.

1 Introducción

La detección del discurso de odio posibilita la identificación de comentarios con lenguaje abusivo y degenerativo en contra de uno o varios grupos de personas. La masificación de las redes sociales, así como los canales virtuales de comunicación, exponen a los usuarios a interacciones de todo tipo, incluyendo interacciones perjudiciales en contra de grupos de personas o de ellos mismos. La agresión y la victimización de los usuarios por el discurso de odio en línea es una situación común en usuarios jóvenes. Según (Oksanen et al., 2014), el 67% de los usuarios entre 15 y 18 años han estado expuestos al odio cibernético en Facebook y YouTube, donde el 21% de ellos habría sido víctima de comentarios de odio. Los medios de interacción virtual carecen o poseen limitados controles dedicados a la detección del odio para el bloqueo o alerta de aquellas interacciones que agreden a los usuarios.

De acuerdo con (Dorris et al., 2020), el 70% de los usuarios en Internet están expuestos a expresiones de odio y al lenguaje ofensivo de frecuentemente. En el estudio previo realizado por (Navarro-Murillo et al., 2019), sobre la identificación de contenido inadecuado para niños en foros de videojuegos, se evidenció la presencia de contenido abusivo, palabras ofensivas y temáticas inadecuadas en foros de videojuegos en línea. En estos foros participaban usuarios de todas las edades. A pesar de contar con usuarios mediadores, estos no son capaces de detectar todo el espectro de contenido inadecuado para la audiencia.

A pesar de esfuerzos realizados en plataformas de video juegos y redes sociales por evitar este tipo de interacción, estas acciones han sido implementadas en mayor medida en el idioma inglés, no así en español. Según (Fortuna & Nunes, 2018), la mayoría de los estudios previos sobre la detección del discurso del odio están enfocados al inglés. Esto debido a la disponibilidad de *datasets* y fuentes de información en ese idioma. No obstante, se han realizado algunas investigaciones en otros idiomas. En el caso de los estudios en español, estos han sido dirigidos en gran parte por las competencias de SemEval[1] (*International Workshop on Semantic Evaluations*) cuando incluyen los desafíos en español. Así mismo, Twitter es la red social preferida para realizar estos análisis y el inglés el idioma más común de estudio. De acuerdo con

[1] SemEval http://alt.qcri.org/semeval2020/index.php?id=tasks

(Fortuna & Nunes, 2018) en su investigación en el 2018, Twitter era la red social más utilizada para la investigación en informática e ingeniería.

A fin de colaborar con la investigación en español sobre la detección del odio, se plantea las siguientes preguntas de investigación: ¿Cuál es la efectividad que se obtiene al identificar contenido de odio en un texto en español mediante el uso de clasificadores automáticos? A fin de responder esta pregunta, se propone la investigación de comparación de modelos de identificación automática de odio en comentarios en español extraídos de Twitter. En la sección de Planteamiento del problema, se expone con mayor detalle la problemática y los objetivos de la investigación. Los principales beneficiaros de esta investigación son los investigadores o desarrolladores interesados en el desempeño de modelos automáticos para la detección de odio con el idioma en español. Debido a la reducida cantidad de estudios de este tipo realizados para el idioma español.

El alcance de esta investigación está delimitado a comentarios en español extraídos de Twitter. El resultado de esta investigación es un *corpus* anotado en español y un análisis comparativo entre dos técnicas de clasificación automática de texto. Ambos resultados sirven como insumo en estudios posteriores para técnicas de detección de odio en textos en español. Así mismo, se proveen recomendaciones y consideraciones necesarias para preparar el *corpus* y las configuraciones de los modelos de cada clasificador automático evaluado.

Este documento detalla el planteamiento, el desarrollo, resultados y conclusiones de la investigación. En la siguiente sección se presentan los conceptos y definiciones relacionados a la detección automática de texto, el discurso de odio y técnicas de procesamiento del lenguaje natural. Más adelante, se presentan la sección de antecedentes acerca de los esfuerzos realizados en inglés y español para la detección de odio. Posteriormente se detalla los objetivos de la investigación y la metodología ejecutada para obtener los resultados del desarrollo de la investigación. Finalmente, se presentan los resultados obtenidos y las conclusiones en las últimas secciones del documento.

2 Marco Teórico

Esta sección define los principales conceptos que permiten la comprensión de los términos incluidos en la pregunta de investigación y la metodología propuesta. Se incluye la definición del discurso del odio, a fin de introducir el tema e identificar el alcance que tiene el etiquetado del odio. Así mismo, a raíz de ser un problema de procesamiento del lenguaje natural, esta investigación, se incluye la definición de este término y de las técnicas que se pueden utilizar para la clasificación automática de contenido como lo son: los clasificadores de aprendizaje automático supervisado y los clasificadores aprendizaje profundo. Además de la descripción de las propiedades necesarias para la configuración de los clasificadores. Estos conceptos son indispensables para comprender la complejidad del estudio a realizar y las técnicas a utilizar para responder la pregunta de investigación. A continuación, se presentan las definiciones de los términos anteriores.

Discurso del odio

El discurso de odio comprende el lenguaje que ataca, minimiza, e incita a la violencia o al odio. En concreto, contra grupos de individuos y en función de características específicas. Así, por ejemplo: la apariencia física, la religión, la ascendencia, el origen nacional o étnico, la orientación sexual, la identidad de género u otras. Este puede ocurrir en diferentes idiomas, estilos, incluso en formas sutiles o cuando se usa el humor (Fortuna & Nunes, 2018). Así también, se le conoce por odio en línea u *online hate,* el cual por definición, se refiere al lenguaje que utiliza formas abusivas, agresivas, de ciber acoso, odio, insultos, ataques personales, provocación, racismo, sexismo, amenazas, o toxicidad (Salminen et al., 2020). Otra definición relacionada a este término es el discurso ofensivo, el cual es un discurso abusivo dirigido a un grupo particular de personas (Dorris et al., 2020). De esta manera, las referencias hacia el discurso del odio, odio en línea y el discurso ofensivo se refieren al mismo término y la acción de utilizar discurso de odio en contra de un individuo o un grupo de individuos. El análisis de la detección de odio en comentarios en línea es posible por medio de técnicas procesamiento de lenguaje natural, las cuales permiten analizar la composición de las oraciones, y por medio técnicas de aprendizaje automático para procesar el contenido a fin de generar clasificadores automáticos de texto que contiene o no contiene odio.

Natural Language Processing o Procesamiento de Lenguaje Natural

Los autores (Sintoris & Vergidis, 2017), definen el procesamiento del lenguaje natural (NLP) como un campo interdisciplinario que estudia y desarrolla algoritmos y sistemas que permiten a las computadoras comprender y realizar tareas relacionadas con el lenguaje humano. El NLP puede denominarse lingüística computacional, que permite el procesamiento del lenguaje humano para ser utilizado por la computadora o alguna tecnología de interacción. El lenguaje natural es inherentemente impreciso; por lo tanto, puede que no sea posible garantizar la confianza del programa entrenado con un *corpus* específico (Desai et al., 2016). El procesamiento del lenguaje natural es aplicado en la recuperación y extracción de información, traducción automática, sistemas de búsquedas de respuestas, generación de resúmenes automáticos, minería de datos y análisis de sentimientos (Hernández & Gómez, 2013).

El análisis de textos que contienen odio requiere de técnicas y herramientas de procesamiento de lenguaje natural. Esto con el objetivo de definir las características necesarias para entrenar un modelo de clasificación automática, el cual identifique el contenido con odio y sin odio, basado en ejemplos y contraejemplos. Para efectos de la investigación los algoritmos de clasificación de interés corresponden al de aprendizaje automático y los de aprendizaje profundo.

Supervised machine learning classifiers o clasificadores aprendizaje automático supervisado

Estos algoritmos corresponden a modelos de aprendizaje automática utilizados para la clasificación del texto. Por ejemplo, estos pueden ser de *Support Vector Machine* (SVM), Gaussian Naive Bayes (GNB), Complement Naive Bayes (CNB), Decision Tree (DT), Nearest Neighbors (KN), Random Forest (RF) y Neural Network (NN)(Sohn & Lee, 2019). Los anteriores por medio de un proceso de entrenamiento y clasificación, determinan las propiedades o características que definen si un objeto o sujeto pertenece a una categorización brindada (Hernández & Gómez, 2013). Una vez que el *corpus* ha sido utilizado para generar el modelo, los algoritmos procesan las nuevas entradas para discernir, basados en evidencia, si la entrada pertenece a alguno de los grupos previamente identificados en el *corpus*. La definición de las características y propiedades es indispensable para la configuración del algoritmo de clasificación.

Features (propiedades) *o características*

Las propiedades y características aplicadas a un algoritmo de procesamiento de lenguaje natural hacen referencia a los caracteres escritos. Estas pueden ser utilizados por medio de n-gramas (unigramas, bigramas, trigramas), palabras de función n-gramas y regionalismos. Así mismo, en algunos casos se utiliza la ponderación de frecuencia de términos para la creación de las características (Carmona et al., 2018). Este método es utilizado para definir las variables del modelo de los clasificadores automáticos de texto. La generación de estas propiedades puede generarse en diferentes representaciones, por ejemplo la representación de Bag-of-Words, TF-IDF, Word2Vec, BERT, y sus combinaciones (Sohn & Lee, 2019). Las propiedades son utilizadas por algoritmos de clasificación de aprendizaje, no obstante, para algoritmos de *Deep Learning*, se requiere un enfoque diferente.

*Deep learning (*aprendizaje profundo)

El *Deep Learning* usa múltiples capas para representar las abstracciones de datos para construir modelos computacionales. Algunos algoritmos de *Deep Learning* son las redes de confrontación generativas, las redes neuronales convolucionales y las de transferencias de aprendizaje (Pouyanfar et al., 2018). Estos algoritmos han introducido un cambio en la forma de procesamiento de la información y por ende el procesamiento del lenguaje natural. Así mismo, existen arquitecturas de aprendizaje rápido y de *Deep Learning* como lo son: *FastText*, Redes neuronales convolucionales (CNN) y Redes de memoria a largo plazo (LSTM) utilizadas para problemas de clasificación de texto (Badjatiya et al., 2017). Adicionalmente, se pueden utilizar las redes neuronales profundas (DNN). Estas últimas, por su capacidad de encontrar representaciones de datos, son útiles en problemas de clasificación y son ampliamente exploradas para realizar tareas de procesamiento del lenguaje natural. Las DNN pueden ser de dos tipos: redes neuronales de convolución (CNN) y redes neuronales recurrentes (RNN). Ambos enfoques de DNN han beneficiado el procesamiento del lenguaje natural (Jiang & Suzuki, 2019). Con el propósito de utilizar algoritmos de Deep Learning a diferencia de utilizar propiedades o características, se utilizan representaciones vectoriales como los *word embeddings.*

Word embeddings

Los *word embeddings* corresponden a un conjunto de vectores de incrustaciones de palabras que representan el espacio semántico de las palabras en un espacio vectorial continuo de valor real. La intersección entre los vectores de los vectores de palabras reflejan las relaciones lingüísticas de las palabras (Plaza-Del-Arco et al., 2020). Los *word embeddings* asignan cada palabra a un vector de alta dimensión que captura el contexto de una palabra en un documento y una similitud semántica y sintáctica con otra palabra. Word2Vec es una de las técnicas más populares para utilizar *word embeddings* en redes neuronales poco profundas y en implementación particular de algoritmos basados en redes neuronales (Pujari et al., 2019). No obstante, para algoritmos basados en aprendizaje de transferencia se utilizan otros enfoques diferentes a los *word embeddings*.

Transfer Learning o aprendizaje de transferencia

El aprendizaje por transferencia ha sido reconocido como un tema importante en la investigación del aprendizaje automático (Dai et al., 2007). En él se utilizan datos etiquetados y no etiquetados en los dominios de origen y destino que se están analizando (Weiss et al., 2016). Las dos técnicas de transferencia de aprendizaje más comunes en el procesamiento del lenguaje natural corresponden a la transferencia basada en características y ajuste fino (Houlsby et al., 2019). Los ejemplos de algoritmos basados en modelos de transferencia corresponden a BERT y ELMo, los cuales son modelos de representación del lenguaje que demuestran resultados prometedores en tareas de procesamiento de lenguaje natural de propósito general (Peng et al., 2019).

El proceso de clasificación de texto se puede ejecutar utilizando modelos convencionales de aprendizaje automático o por medio de *Deep Learning*. Sin embargo, la variedad de algoritmos y parámetros de configuración es un espectro muy amplio. Para efectos de esta investigación, el enfoque se encuentra en la utilización de un modelo de cada categoría y comparar los resultados de la clasificación obtenida. En la siguiente sección se presenta el estado del arte acerca de los estudios realizados para la clasificación de texto de odio utilizando modelos convencionales y de *Deep Learning* en textos en español y en inglés.

3 Estado del Arte

El estudio científico del discurso del odio en los medios digitales es reciente, y el número de estudios enfocados en esta temática es considerablemente reducido (Fortuna & Nunes, 2018). La mayoría de los estudios previos sobre la detección de odio están en inglés, esto debido a la disponibilidad de conjuntos de datos anotados en inglés. Al mismo tiempo, se han realizado esfuerzos en otros idiomas. Sin embargo, la mayor parte de los estudios ha sido realizada sobre *datasets* en inglés. Así mismo, la detección automática del discurso de odio ha tenido un gran interés en los últimos diez años, esto debido que la fuente de datos para estas investigaciones proviene de comentarios y publicaciones de los usuarios. El problema de la detección del odio corresponde a un desafío del procesamiento del lenguaje natural (Ruwandika & Weerasinghe, 2018).

Los autores (Fortuna & Nunes, 2018), realizaron una revisión sistemática de la literatura acerca de la detección del discurso de odio, e identificaron que esta área tiene un potencial incuestionable para el impacto social, especialmente en las comunidades en línea y las plataformas de medios digitales. La colaboración, que se puede realizar en esta área de estudio, incluye desde el desarrollo de recursos compartidos, lineamientos, algoritmos y conjunto de datos anotados en diferentes idiomas para promover el conocimiento en la detección automática del de odio.

La popularidad de las redes sociales en línea, como Facebook, Twitter, Instagram y YouTube, aumentan la comunicación y el intercambio de información. Así también, incrementa la visibilidad y la presencia del discurso de odio. Este tipo de discurso, puede dañar a los usuarios e incrementar la hostilidad entre los grupos étnicos, o incluso conducir a ataques bélicos (Teh et al., 2018). El discurso de odio es un fenómeno complejo asociado a las relaciones entre grupos que por medio de las redes sociales y medios digitales ejercen influencia en comportamientos de grupos de individuos. El interés en esta área no es únicamente de interés académico, sino también de interés social debido a la injerencia que tiene el odio en los individuos y en la sociedad. Por ejemplo, en el caso de Costa Rica, según el estudio de (*Digital 2020*, s. f.), los usuarios de redes sociales en Costa Rica a enero 2020 suman 3.70 millones, mostrando un aumento en 146 mil (+ 4.1%) entre abril de 2019 y enero de 2020. La cantidad de comentarios y publicaciones es enorme en estas plataformas y su influencia aún mayor.

De la misma manera que sucede en Costa Rica, en el mundo las publicaciones en línea tienen efectos en grupos sociales generados en los medios digitales. Por lo tanto, la detección del discurso del odio es necesaria y su automatización aún más para prevenir la proliferación de este discurso. En algunos países se cuentan con regulaciones para limitar el discurso del odio, por ejemplo, (Teh et al., 2018) comenta que en Países Bajos, Australia, Gran Bretaña y Alemania se prohíbe y sanciona la utilización del discurso que ofende, insulta, humilla o intimida a individuos o grupos. La detección automática del discurso del odio es necesaria para control del efecto de estas publicaciones en los medios digitales, a pesar de que esta detección conlleva a desafíos relacionados con procesamiento del lenguaje natural.

3.1 Desafíos de la detección del discurso

La detección automática del discurso del odio posee desafíos acerca del reconocimiento de palabras relacionadas a temáticas de odio, discriminación, vocabulario obsceno y el entendimiento del contexto del mensaje, el cual no siempre contiene vocabulario obsceno. Así mismo, está presente el problema de interpretación del mensaje y la clasificación de este de acuerdo con temáticas sociales. En referencia a (Teh et al., 2018), los léxicos de palabras negativas son recursos necesarios para extraer las características del discurso de odio, utilizando el supuesto de que los mensajes de odio generalmente contienen palabras específicamente dañinas. Además, propone las siguientes categorías para la clasificación del odio: etnia, comportamiento, física, orientación sexual, clase, género, discapacidad, religión y otras.

Los autores (Salminen et al., 2020) , definieron los desafíos de la detección del odio en línea en tres categorías:

- Problemas de falsos positivos: ocurre cuando los modelos identifican expresiones de odio debido a la presencia de palabras y frases en el texto. Se identifican mensajes de odio cuando no hay evidencia suficiente para realizar categorizar el mensaje en esa categoría.
- Problema de los de falsos negativos: sucede cuando los modelos no identifican el odio en las expresiones.
- Subjetividad: ocurre en la creación del *ground truth* por medio de *source crowding*. El etiquetado manual puede verse sesgado por los valores y creencias de los anotadores. Además, esta subjetividad se incrementada por la interpretación de expresiones que

incluyen sarcasmo y humor. Además del principal problema de la diversidad lingüística del odio, que no está totalmente captada por un diccionario.

- Polisemia: sucede cuando una palabra cambia su significado cuando el contexto cambia.

Estos desafíos deben ser tomados en cuenta para la selección de modelos ideales de clasificación. Si bien cualquier modelo estará sujeto a los dos últimos desafíos, los dos primeros desafíos pueden ser medibles y controlables en un modelo de detección automática.

3.2 Complejidad temática del discurso del odio

Adicionalmente a los desafíos anteriores, se suma la complejidad temática. Esta radica en el entendimiento de qué se clasifica como odio y qué no. Por su parte (Teh et al., 2018) definieron 7 categorías para anotar textos con contenido de odio las cuales son: orientación sexual, discapacidad, género, religión, raza, comportamiento, clase y otros. Sin embargo, estas categorías varían entre investigaciones realizadas sobre el discurso de odio. Este problema se relaciona con el desafío de la subjetividad, el cual consiste en el alcance sobre qué se entiende por discurso de odio. Así mismo, hay diferentes referencias para el discurso del odio, por su parte (Fortuna & Nunes, 2018) los conceptos encontrados en la literatura fueron odio, acoso cibernético, lenguaje abusivo, discriminación, blasfemias, toxicidad, *flaming*, extremismo y radicalización. Cada investigación utiliza su propia escala para generar las reglas de clasificación y anotación de los textos que contienen odio. Estas reglas son necesarias para definir los criterios de anotación de los textos del *corpus* del *grounded truth*, el cual será utilizado en los clasificadores automáticos de odio.

3.3 Detección automática del discurso del odio

La detección automática del odio en línea se puede abordar desde tres perspectivas según (Salminen et al., 2020), estas son: clasificadores simples basados en léxico o palabras clave, clasificadores que usan semántica distribuida o clasificadores de aprendizaje profundo utilizando características lingüísticas avanzadas. Las características por utilizar pueden ser la forma de bolsa de palabras *(Bag of words BOW)*, la bolsa de caracteres (BOC), los caracteres n-gramas y las palabras n-gramas. Todos estos pueden ser utilizados para predecir el discurso de odio según (Sohn & Lee, 2019). Además, la detección automática puede ser de dos tipos (Sohn & Lee, 2019) :

- Métodos convencionales de aprendizaje automático: Utilizando ingeniería de características y algoritmos de clasificación como lo son la máquina de soporte de vectores (SVM), Naive Bayes y la regresión logística.
- Métodos de aprendizaje profundo (*Deep learning*): Aplicando algoritmos de redes neuronales de tipo de red neuronal convolucional (CNN), la red neuronal recurrente (RNN), el modelo de memoria a corto plazo (LSTM) y la unidad recurrente cerrada (GRU). Estos se han utilizado para aumentar el rendimiento en la detección y para aprender automáticamente de características ocultas, no predefinas al inicio del modelo de clasificación.

La detección automática del odio ha sido abordada por medio de ambos métodos anteriores y con configuraciones distintas de características. Un ejemplo de ello es el HateDefender (Dorris et al., 2020), el cual utiliza una red neuronal profunda de memoria a largo plazo (LSTM) y un modelo de explicación basado en la notoriedad de palabras calculadas a partir de las señales de del LSTM. Es decir, las palabras que son responsables del discurso de odio o del lenguaje ofensivo pueden tener gran relevancia. Así mismo, (Plaza-Del-Arco et al., 2020) definieron un espacio de vectores de características para entrenamiento y evaluación, que está compuesto por vectores no supervisados de *word embeddings* utilizando *FastText*. Por su parte (Corazza et al., 2020) propone utilizar MicroT, el cual se basa un enfoque de clasificación de texto respaldado por un modelo basado en léxico que tiene en cuenta la presencia de palabras agresivas y afectivas, aplicando FastText. Con un enfoque similar (Herwanto et al., 2019) implementa *word embeddings* con una bolsa continua de palabras (CBOW) y FastText.

Por su parte (Ruwandika & Weerasinghe, 2018), construyeron cinco modelos. Ellos utilizaron cuatro algoritmos de aprendizaje supervisados y un algoritmo de aprendizaje no supervisado. Utilizaron la máquina de vectores de soporte, la regresión logística, el algoritmo Naïve Bayes, el algoritmo del árbol de decisión para los modelos de aprendizaje supervisado y el algoritmo de agrupación K-Means. Este último se usó para el modelo de aprendizaje no supervisado. En la investigación, los modelos de aprendizaje supervisados funcionaron mejor que el modelo de aprendizaje no supervisado.

Se debe agregar que un análisis comparativo realizado por (Modha et al., 2020), identificó que los modelos de transferencia de aprendizaje como BERT y los modelos neuronales aprendizaje profundo basados en LSTM y CNN, funcionan de manera similar pero mejor que los clasificadores tradicionales como SVM. Esto basado en los resultados de las pruebas ejecutadas

con *tweets* en inglés y en alemán. No obstante, el beneficio obtenido en el idioma indio no era representativo. Entre las razones para este comportamiento, se identificó que la cantidad de *tweets* con contenido de odio para el entrenamiento y prueba provocaba una afectación en el despliegue de cada modelo.

Otro estudio por (Almatarneh et al., 2019) compara los resultados obtenidos por algoritmos supervisados para la detección de odio en *tweets* en español e inglés. Este estudio mostró que el algoritmo de SVM supera a Naive Bayes y los árboles de decisión, en la tarea de búsqueda de opiniones muy negativas. Para esta investigación se utilizaron características de Ngrams y Doc2Vec.

Finalmente, en referencia a la revisión de la literatura por (Fortuna & Nunes, 2018), al 2017 los siguientes algoritmos de la Tabla 1 son los más referenciados en las investigaciones relacionadas a la identificación de odio. Esta lista no diferencia el idioma en el cual se realizaron las investigaciones.

Tabla 1 Algoritmos utilizados en los documentos de "Informática e Ingeniería"

Algoritmo	**Frecuencias**
SVM	10
Random forests	5
Decision trees	4
Logistic regression	4
Naive bayes	3
Deep learning	1
DNN	1
Ensemble	1
GBDT	1
LSTM	1
Non-supervised	1
One-class classifiers	1
Skip-bigram model	1

Fuente:(Fortuna & Nunes, 2018)

Además, la revisión de la literatura de (Fortuna & Nunes, 2018) incluía un análisis comparativo de los resultados obtenidos por cada algoritmo, sin embargo la comparación sirve de referencia más no de comparación. Esto debido que las características utilizadas por cada algoritmo son diferentes, así como los *datasets* utilizados.

En la siguiente sección se define el planteamiento del problema utilizando de insumo el estado del arte brindado en esta sección.

4 Planteamiento del problema

El estudio de detección del discurso del odio en español ha crecido considerablemente, sin embargo, en comparación con el inglés, las investigaciones han sido menores. Los desafíos para la investigación en español no son solo incluye desafíos técnicos como los mencionados en la sección de los antecedentes sino que adicionalmente, existe la limitante de disponibilidad de los *datasets* en español para realizar estudios (Fortuna & Nunes, 2018). A pesar de ser Facebook, WhatsApp e Instagram las aplicaciones con mayor utilización en América Latina y en nuestro país (*Digital 2020*, s. f.), el acceso a la extracción de interacciones de los usuarios de forma automática es restrictiva. No obstante, existe la posibilidad de extracción de comentarios de Twitter y es por esta razón es la fuente de datos más utilizada para la creación de *datasets*. Esto se puede identificar en la revisión de la literatura de (Fortuna & Nunes, 2018). A pesar de utilizar Twitter como fuente de datos para el ejercicio de recolección de textos, esta plataforma no es tan utilizada como lo es Facebook o WhatsApp en América Latina.

El estudio del discurso del odio ha aumentado, esto debido a la necesidad de prevenir el ataque y discriminación por medios virtuales. Además de la relevancia que esta prevención ha tomado. No es únicamente el hecho de identificar el odio, sino tomar medidas contra estos usuarios y promover en los medios virtuales un lugar seguro para los usuarios. A fin de colaborar con la investigación en español y con la detección automática del odio para protección de los usuarios, se plantean la siguiente pregunta de investigación ¿Cuál es la efectividad que se obtiene al identificar contenido de odio en un texto en español mediante el uso de clasificadores automáticos? Esta pregunta es muy amplia, se espera al menos comparar dos métodos de clasificación automática (convencional y de aprendizaje profundo) y construir un corpus anotado para obtener un primer insumo para responder la pregunta de investigación. El resultado de esta investigación aportará al campo de investigación con: un *corpus* anotado de identificación de odio en español y una comparación de entre dos técnicas de procesamiento de lenguaje natural, que servirán como insumo en estudios posteriores para técnicas de detección de odio en textos en español.

4.1 Objetivos de la investigación

Con el propósito de responder a la pregunta de investigación se plantean los siguientes objetivos.

Objetivo general

Comparar el uso de un clasificador convencional de SVM (*Support Vector Machine*) y un clasificador de *Deep learning* CNN (*Convolucional Neural Networks*) para la detección automática de odio a partir de comentarios de Twitter en español de Costa Rica.

Como parte de la comparación se procederá a analizar el desempeño de cada modelo de clasificación respecto a las métricas de: precisión promedio, *recall* promedio (recuperación), *accuracy* promedio (exactitud) y macro F1. Estas evaluaciones se realizarán en los conjuntos de datos de entrenamiento, desarrollo y prueba del *corpus* etiquetado que se desarrolla como parte de esta investigación. Los objetivos específicos de la investigación son los siguientes:

Objetivos específicos

1. Crear un *corpus* etiquetado de *tweets* para detección de odio en español.
2. Determinar las características y las reglas de preprocesamiento a utilizar para los modelos de SVM y CNN en este contexto.
3. Comparar los resultados obtenidos de la implementación del modelo SVM y el modelo CNN.

4.2 Alcances y limitaciones

El alcance de esta investigación se limita la a extracción de *tweets* en español en Costa Rica. La extracción se enfocará en *tweets* de interés público sobre comentarios realizados por figuras públicas, periodistas y terceros sobre las noticias de interés nacional.

5 Metodología

Con el propósito de lograr los objetivos colocados en la sección anterior, en esta sección se expone la metodología a implementar. Este trabajo está relacionado con el curso de Procesamiento de Lenguaje Natural y en complemento al proyecto de laboratorio sobre la *Identificación de contenido inadecuado para niños en foros de videojuegos*. Los resultados obtenidos fueron presentados en las IV Jornadas Costarricenses de investigación en computación e informática JOCICI del 2019 y publicados en la IEEE (Navarro-Murillo et al., 2019). El cual identificó contenido inapropiado en el foro en línea de un videojuego, en dónde se encuentran usuarios de todas las edades.

La investigación realizada corresponde a un caso de estudio. Lo anterior, debido que no se pretende llevar a cabo un muestreo o generalización de los resultados obtenidos. El dominio de aplicación se centra en un conjunto de datos específico y se seleccionan algunos factores fijos para la investigación siguiendo un criterio de oportunidad. La razón para este procedimiento es debido que en la actualidad no se cuenta con todos los recursos lingüísticos, como por ejemplo un *dataset* en español con anotaciones de odio. La construcción de este recurso es uno de los aportes de esta investigación. El cual, está dirigido a la implementación de un método convencional y otro no convencional para el análisis de micro textos en español para detectar odio. Se procedió a identificar los *features* o características que benefician la identificación de odio en micro textos (comentarios de Twitter), utilizando ambos modelos y analizando los resultados obtenidos en las ejecuciones de los modelos. A fin de lograr la comparación de los modelos de clasificación, se explican a continuación los siguientes métodos para el logro de cada uno de los objetivos específicos de la investigación. Así mismo, la Figura 1 muestra la estructura de la metodología y métodos utilizados.

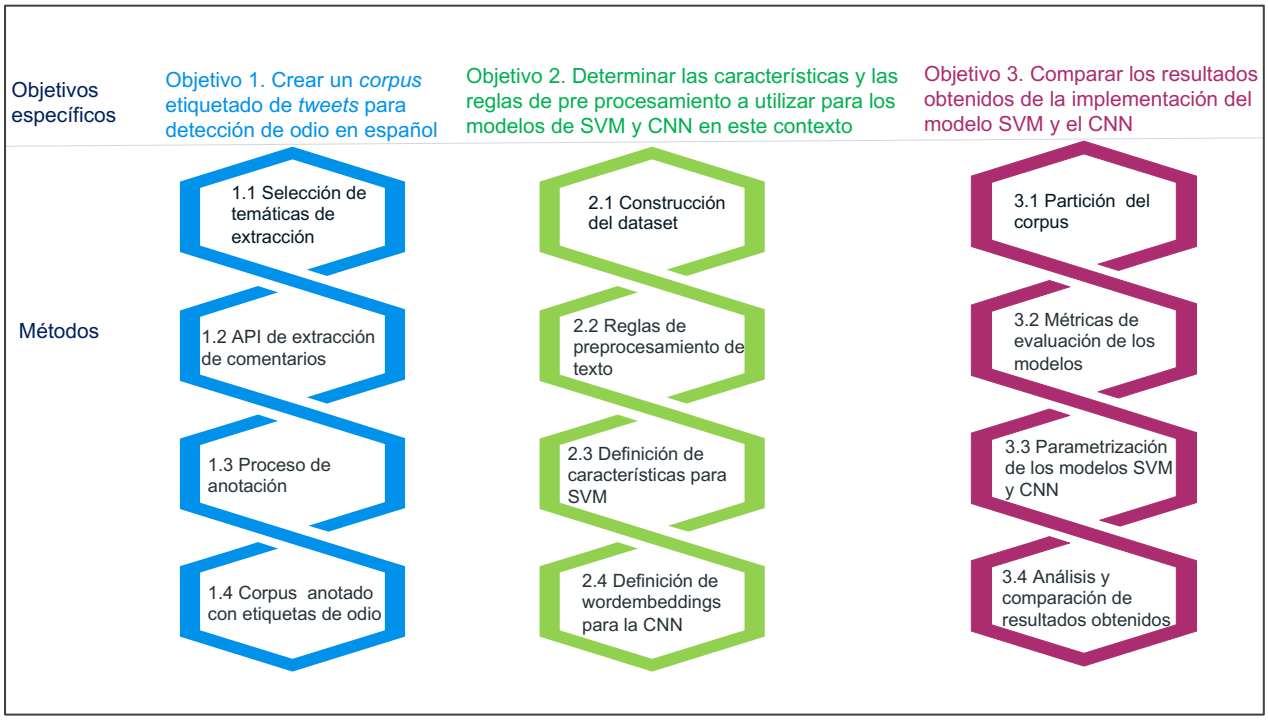


Figura 1 Estructura de la metodología
Fuente: Propia

Objetivo específico 1

Crear un *corpus* etiquetado de *tweets* para detección de odio en español.

Métodos:

1.1 Selección de temáticas de extracción: Se identifican las temáticas de interés del momento que generan comentarios negativos, se procedió a identificar las fuentes de las cuales se extraerán *tweets*. Así mismo, se creó una lista de palabras de tendencias para extraer los *tweets.*

1.2 API de extracción de comentarios: Configuración del API de extracción de comentarios, una vez identificados las palabras y temáticas de interés se procedió con la extracción de los *tweets*.

1.3 Proceso de anotación: Definición de las reglas de anotación de odio. Estas reglas serán utilizadas por las personas que colaborarán con la anotación de los textos de los *tweets*. Estas reglas deben son claras, precisas y de fácil aplicación. Las reglas deben servir para discernir si un texto posee un comentario con lenguaje abusivo, ofensivo contra un individuo o un grupo de individuos. Así también, se debe generar la herramienta a utilizar para realizar la anotación, en este caso un archivo de Excel, el cual será utilizado por cada anotador.

Se realiza la anotación manual por medio de *crowd sourcing.* Utilizando la herramienta y las reglas de anotación. Se selecciona un grupo de personas para realizar el etiquetado manual de los micro textos. Las personas identificaron si el comentario contiene odio o no basado en las reglas de anotación definidas previamente.

1.4 Corpus anotado con etiquetas de odio: Se procede con la revisión de la anotación realizada por los usuarios. Posterior a la anotación, se verifica la calidad de la anotación realizada por las personas y se asegura la creación de un *corpus* representativo para el clasificador automático.

Productos:

Los productos obtenidos al completar el objetivo son los siguientes:

- Reglas de anotación manual.
- Herramienta de anotación de odio en micro textos.
- *Corpus* anotado con las etiquetas de odio. Los comentarios son anotados en español identificando sí o no poseen odio en su contenido.

Objetivo específico 2

Identificar las características y las reglas de preprocesamiento a utilizar para los modelos de SVM y CNN en este contexto.

Métodos:

2.1 Construcción del *dataset*: Corresponde al proceso de consolidación de los resultados obtenidos en el proceso de anotación manual. Se crea el corpus a ser consumido por los modelos.

2.2 Reglas de preprocesamiento de texto: Se construye el preprocesador del texto de los comentarios. Se procedió a definir las reglas de preprocesamiento a utilizar para la estandarización del texto de los tweets y configurar el preprocesador de texto.

2.3 Definición de características para SVM: Se determinan y configuran los *features* a utilizar para el modelo de SVM.

2.4 Definición de *word embeddings* para la CNN: Se selecciona el modelo de *word embeddings* para representar las palabras de los comentarios, utilizando un criterio de oportunidad y se procede con la configuración de los *embeddings*.

Productos

Los productos obtenidos al completar el objetivo son los siguientes:

- Definición de *features* y *embeddings* construidos para cada modelo.
- *Corpus* preprocesado.

Objetivo específico 3

Comparar los resultados obtenidos de la implementación del modelo SVM y el CNN.

Métodos:

3.1 Partición del corpus: Se procede a dividir el *corpus* en forma aleatoria para efectos de entrenamiento, desarrollo y evaluación. De esta forma se generaron 3 *datasets* respectivamente.

3.2 Métricas de evaluación de los modelos: Se realiza una validación cruzada para determinar la calidad de cada modelo, con las métricas: precisión promedio, recall promedio (recuperación), *accuracy* promedio (exactitud) y F1.

3.3 Parametrización de los modelos SVM y CNN: Corresponde a la implementación del modelo de SVM y el CNN para el análisis de odio de comentarios en español, utilizando el *dataset* de entrenamiento. Se procede a realizar ajustes a los modelos, mediante evaluaciones de su efectividad sobre el conjunto de datos de desarrollo. Particularmente, se implementan evaluaciones de sobreajuste a los datos de entrenamiento. Así mismo, se configuran la lógica de ejecución para realizar las evaluaciones cruzadas de cada modelo, guardar los resultados de las métricas de evaluación y las gráficas de desempeño.

3.4 Análisis y comparación de resultados obtenidos: Posterior a la implementación de los modelos, se evalúan los resultados de acuerdo con las métricas definidas para evaluación de los modelos. Se procede a realizar la comparación de los resultados de ejecución de los modelos y determinar los hallazgos obtenidos. Se documentan las diferencias en caso de que existan y se definen las recomendaciones para la implementación de los modelos. Además de analizar los resultados obtenidos en comparación con resultados observados en estudios similares de detección de odio en inglés.

Productos

Los productos obtenidos al completar el objetivo son los siguientes:

- Script de ejecución de los modelos de clasificación.
- Resultados de ejecución de los modelos de clasificación los cuales corresponden a las métricas de evaluación de modelos de *machine learning* de precisión, *recall* y medida F1.
- Análisis y comparación de resultados obtenidos.
- Síntesis de los resultados obtenidos con respecto a otros estudios mencionados en los antecedentes para el idioma inglés.

6 Productos y resultados de la investigación

Esta sección detalla el trabajo realizado en la implementación de la metodología anterior. Se detalla los pasos realizados, así como los resultados obtenidos en la implementación de los modelos de CNN y SVM. Se presentan las consideraciones y decisiones tomadas para la ejecución de los modelos de acuerdo con los objetivos de la investigación. El primer paso, en el desarrollo de la investigación consistió en la identificación de los comentarios a utilizar. La siguiente subsección muestra el proceso de selección de la temática y extracción de los textos desde Twitter. Más adelante, se explica el proceso de preprocesamiento de los comentarios y su utilización en los modelos de CNN y SVM.

6.1 Etiquetado de *tweets* para detección de odio en español

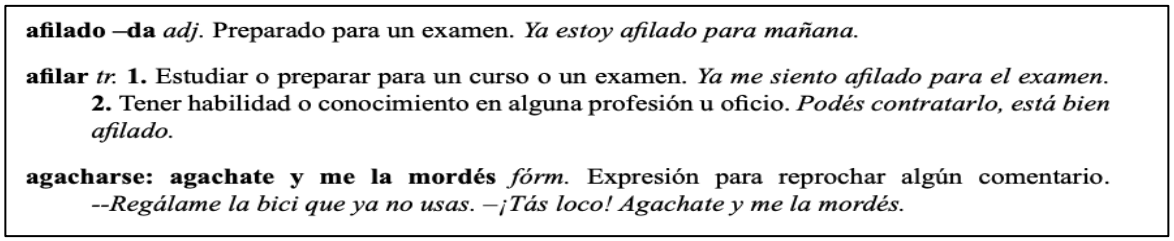
La identificación de los *tweets* para construir el *corpus* consistió en la creación de una lista de palabras que se utilizó para extracción de *tweets* desde la API de *Twitter*. Posteriormente, se procedió con la creación de conjuntos de *tweets* para ser revisados y etiquetados por individuos. La lista final de *tweets* anotados corresponde al *corpus* a utilizar para el ejercicio de identificación de odio por medio de los modelos SVM y CNN.

6.1.1 Temáticas de interés del *corpus*

Con el propósito de delimitar la extracción de lo *tweets* a posibles comentarios en español y de Costa Rica se procedió con el levantamiento de una lista de nombres de personajes políticos y figuras públicas de Costa Rica para el 2020. Sin embargo, en una primera extracción la cantidad de *tweets* y la presencia de odio era mínima. A fin de aumentar la cantidad de *tweets* y la posibilidad de extraer *tweets* con odio, se procedió con la inclusión de palabras provenientes del libro *Léxico juvenil costarricense* de (Ríos González, 2017), además de adicionar los vocablos fraseológicos que pueden construir frases peyorativas del estudio realizado por (Durán, 2015).

En referencia a las palabras provenientes del diccionario *Léxico juvenil costarricense* de (Ríos González, 2017), se procedió con la extracción del libro en formato PDF del texto y se construyó un procesador de texto propio para extraer las palabras de definición del diccionario. El procesador únicamente extrae la primera palabra o frase de la definición. Así mismo, se incluyeron las versiones masculinas y femeninas de los sustantivos y adjetivos extraídos. El procesador de texto espera de entrada un archivo de texto plano del archivo PDF original y entrega una lista de términos extraídos. Un total de 941 palabras y frases fueron identificadas

utilizando el *Léxico juvenil costarricense* de (Ríos González, 2017). En la Figura 2 se observa un extracto del diccionario, sobre el cual se extraen las palabras: afilado, afilada, afilar y agacharse.



afilado –da *adj*. Preparado para un examen. *Ya estoy afilado para mañana.*

afilar *tr*. **1.** Estudiar o preparar para un curso o un examen. *Ya me siento afilado para el examen.* **2.** Tener habilidad o conocimiento en alguna profesión u oficio. *Podés contratarlo, está bien afilado.*

agacharse: agachate y me la mordés *fórm*. Expresión para reprochar algún comentario. *--Regálame la bici que ya no usas. –¡Tás loco! Agachate y me la mordés.*

Figura 2 Extracción de palabras del diccionario de Léxico Juvenil Costarricense
Fuente: Léxico Juvenil Costarricense (Ríos González, 2017).

Adicionalmente, a la lista de palabras del léxico juvenil, se sumaron los vocablos utilizados para generar oraciones peyorativas. El lenguaje peyorativo se compone de términos étnicos o sociales neutros utilizados como insultos (Benito et al., 2020), además de las groserías. Estas últimas, pueden ser de dos tipos: vocablos, unidades simples en la lengua, por ejemplo, “Pendejo” o las unidades fraseológicas, compuestas por dos vocablos que constituyen una sola unidad semántica(Durán, 2015). En la investigación de (Durán, 2015) se listan los vocablos que pueden construir unidades fraseológicas, estos vocablos fueron agregados a la lista de extracción de *tweets*. Un total de 162 vocablos se sumaron a la lista de palabras para extracción.

Finalmente, para la lista de figuras políticas de Costa Rica, se revisó la lista de diputados de la Asamblea Legislativa que poseen *Twitter* a Setiembre 2020. Además, se agregaron las cuentas de expresidentes y de figuras públicas involucradas en temas políticos. Se identificaron 86 entidades y personajes de esta forma. La Tabla 2 detalla la composición final de la lista de extracción de *tweets*. Esta lista fue utilizada posteriormente en la API de Twitter, como parámetro de búsqueda y extracción de los tweets que conforman el *corpus* de análisis de la investigación. La lista final de extracción se incluye en el Anexo 1 Lista de extracción.

Tabla 2 Lista de fuentes extracción de *tweets*

Fuente de extracción	**Tipo de palabra clave**	**Clasificación morfológica**	**Cantidad**
Políticos y figuras públicas	Nombres de personas e instituciones	Sustantivos	86
Vocablos de groserías	Vocablos	Sustantivos, adjetivos, verbos	162
Diccionario Léxico Juvenil	Definiciones	Sustantivos, adjetivos, verbos, frases	941
Total de frases o palabras a buscar			1189
Eliminación de repetidos			-52
Total de frases o palabras a buscar sin repeticiones			**1137**

Fuente: Propia

6.1.2 API de extracción de *tweets*

Utilizando la lista generada en la sección anterior se procedió a extraer los *tweets* utilizando las palabras y frases en la API de Twitter. Twitter posee APIs que permiten la extracción de *tweets,* utilizando palabras clave, menciones y localización, entre otros parámetros. En esta investigación se utilizó el paquete de Python Tweepy (*Tweepy Documentation — tweepy 3.10.0 documentation*, s. f.). Este paquete, permite utilizar el API *Search Tweets* de búsqueda Twitter. La versión del API utilizado corresponde a una versión *Standard,* que permite buscar y obtener una muestra de tweets recientes publicados en los últimos 7 días basados en los criterios de la búsqueda (Twitter, 2020). Los parámetros utilizados para la extracción de los tweets corresponden a: español como idioma del *tweet*, extraer al menos 200 *tweets* por llamado al API y utilizar la geo ubicación en el cuadrante [9.0655537º,-84.5919495º] en un radio de 500km. Este último parámetro, permite delimitar la zona geográfica del *tweet* en dónde se emitió, si la información geográfica está disponible para el Tweet.

A pesar de brindar la ubicación por medio de latitud y longitud cerca del territorio de Costa Rica, la API de Twitter no reduce los *tweets* por ubicación en la mayoría de los casos. Esto debido que no todos los usuarios poseen una ubicación asociada para sus *tweets*. Por lo tanto, se utilizó el parámetro de localización y la lista de palabras como criterio de búsqueda para completar la extracción. Esta lista es conformada por palabras y frases propias de Costa Rica e incluso de regiones cercanas al país.

El proceso de extracción de los *tweets* se realizó durante el 27 y 28 de setiembre del 2020. Durante este periodo se extrajeron entre 0 a 100 *tweets* cada día, generados en un umbral de los últimos 7 días calendario dónde las palabras de la lista de extracción aparecieron en *tweets*. Los 7 días corresponden al periodo comprendido entre el primer día de la ejecución de la extracción hasta la fecha menos 7, es decir los días 20 y 21 de setiembre. Finalmente, el total de *tweets* extraídos por el API y bajo los parámetros descritos fueron 398 466. Se observa en la Figura 3*,* la composición final del *DataFrame* resultado de la extracción. Así mismo, se nota la situación de los campos de *geo* y *location* que no están presentes para todas las líneas extraídas.

```
IPdb [15]: vfolderExtraction ='../TweetsExtraction/'
   ...: dtSamples =read_directory_load_df(vfolderExtraction)

IPdb [16]: dtSamples.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 398466 entries, 0 to 393
Data columns (total 16 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   SearchQuery      398466 non-null  object
 1   Datetime         398466 non-null  object
 2   Tweet Id         398466 non-null  object
 3   Text             398466 non-null  object
 4   Languages        398466 non-null  object
 5   Retweeted        398466 non-null  object
 6   hashtags         398466 non-null  object
 7   user_mentions    398466 non-null  object
 8   screen_names     398466 non-null  object
 9   result_type      398466 non-null  object
 10  geo              261 non-null     object
 11  place            8260 non-null    object
 12  location         247299 non-null  object
 13  user             0 non-null       object
 14  user screenname  398466 non-null  object
 15  Unnamed: 0       398466 non-null  float64
dtypes: float64(1), object(15)
memory usage: 51.7+ MB
```

Figura 3 Detalle del *DataFrame* de extracción
Fuente: Propia

Adicionalmente, fue necesario analizar la cantidad de *tweets* encontrados por cada elemento de la lista de extracción. En algunos casos, la API extrajo menos de 100 *tweets* esperados por día. La Figura 4, muestra en tres visualizaciones distintas el volumen de tweets extraídos por cada una de las 1137 palabras durante la extracción. Se observa además dos fenómenos, en el primero hay extracciones con un volumen alto de *tweets* entre 300 y 600 por extracción. En el segundo, hay casos donde la extracción fue menor entre 0 y 100 *tweets* por extracción. Para efectos de selección del *corpus* de tamaño de 3000 muestras, la muestra fue seleccionadas aleatoriamente.

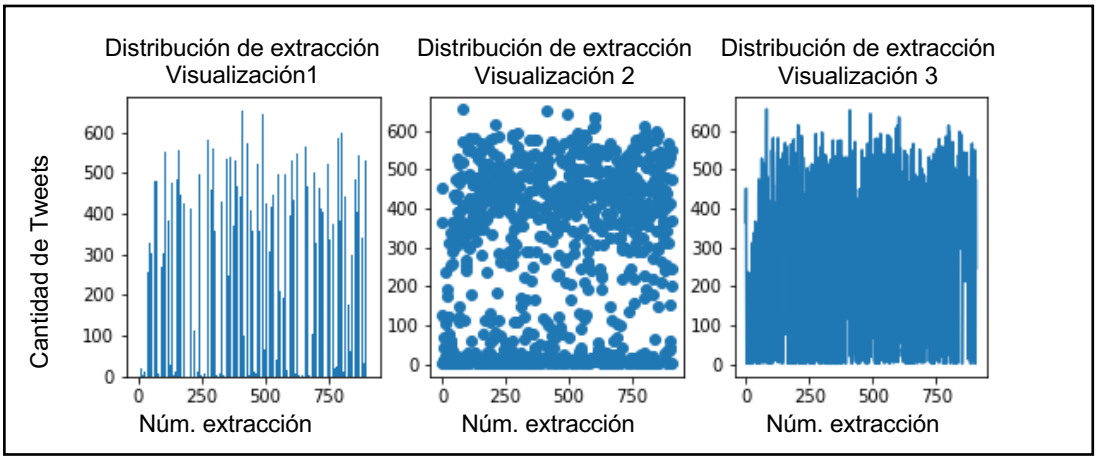


Figura 4 Cantidad *tweets* por extracción
Fuente: Propia

En razón a la gran cantidad de *tweets* extraídos se aplicaron reglas de revisión de la calidad del texto del *tweet*, previo a la selección del *dataset* a utilizar en la investigación. Las reglas de calidad que se aplicaron fueron las siguientes:

a. Eliminar duplicados. Se procedió a eliminar la posibilidad de tener Tweets repetidos.
b. Eliminar *retweets*. En algunos casos se observó en la extracción los *retweets* contenían texto incompleto.
c. El texto debe estar en español. A pesar de forzar la extracción en español con el API de Twitter, se observó, *tweets* en portugués que debían ser removidos. Con la meta de asegurar únicamente textos en español, se utilizó la identificación de idioma por cada uno de los *tweets* utilizando FastText (*Language Identification · FastText*, s. f.)(Joulin, Grave, Bojanowski, & Mikolov, 2016)(Joulin, Grave, Bojanowski, Douze, et al., 2016).

Posterior al aplicar las reglas anteriores se seleccionó un conjunto de 3000 textos de forma aleatoria. Este subconjunto se distribuye en 600 palabras o frases de la lista de extracción original. La lista del subconjunto se puede consultar en Anexo 2. El mínimo de tweets por palabra o frase de extracción es 1 y su máximo es de 16 tweets. La Figura 5 muestra la distribución de la frecuencia de la cantidad de tweets presentes en la muestra de 3000 textos. Mientras que la Figura 6 muestra la cantidad de tweets por las 600 extracciones realizadas.

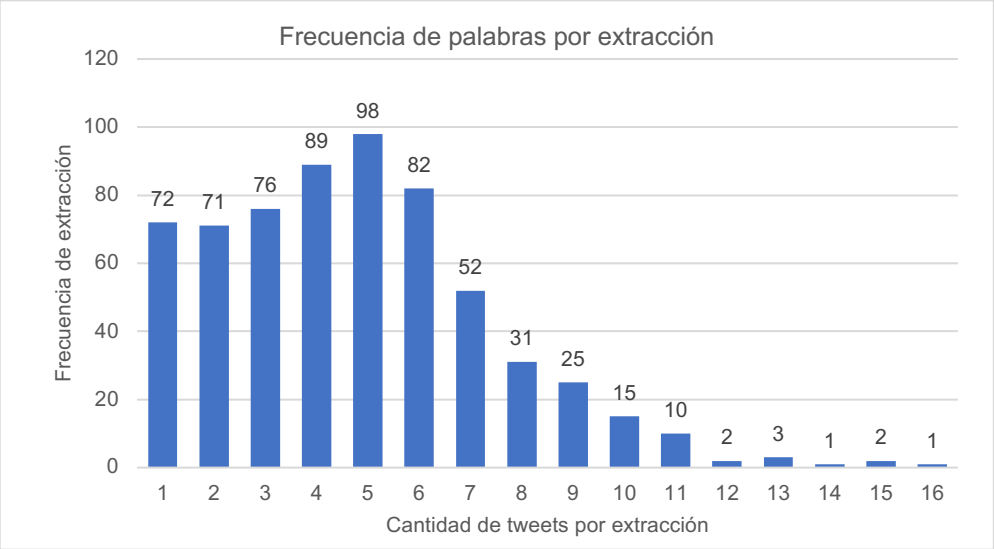


Figura 5 Frecuencia de palabras por extracción
Fuente: Propia

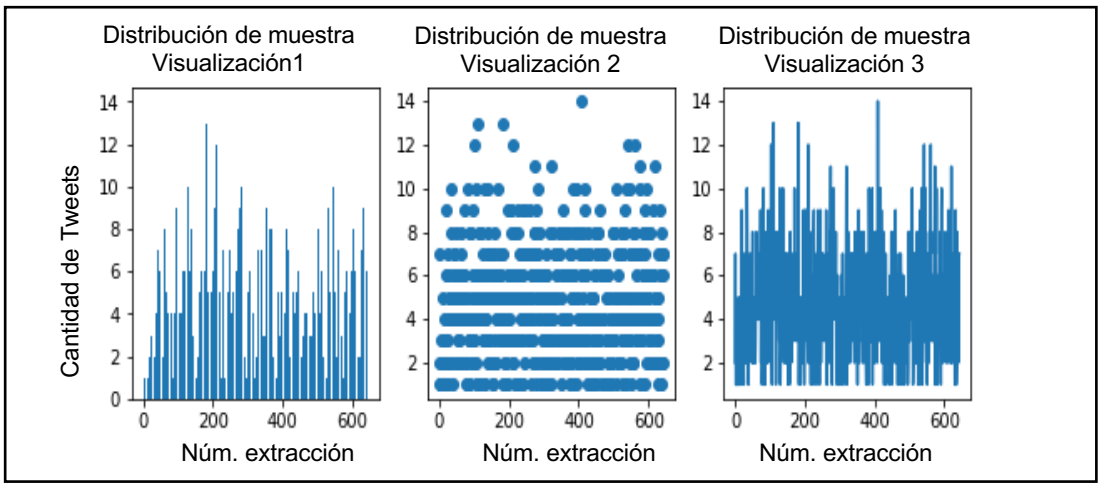


Figura 6 Cantidad *tweets* por extracción de la muestra
Fuente: Propia

6.1.3 Herramienta de creación de anotadores

Utilizando la muestra de 3000 textos se procede a la construcción de los subconjuntos de anotación. Para efectos de este estudio se procedió con la definición de 15 anotadores de 600 textos. El proceso de creación de los anotadores se realizó aleatoriamente, para asegurar que cada tweet fuese anotado por 3 personas diferentes y todos los anotadores revisaran grupos disjuntos y aleatorios de tweets, para dispersar cualquier sesgo de cansancio al anotar o del mismo anotador. Se creó el script de Samples.py para aleatorización y creación de las muestras utilizando la siguiente lógica:

a. Crear 3 órdenes aleatorios de los 3000 textos para la anotación manual.
b. Crear subconjuntos para cada anotador a partir de los órdenes, de forma que un texto sea anotado un máximo de 3 veces por 3 anotadores manuales distintos.
c. Cada texto es revisado y etiquetado por 3 anotadores manuales diferentes.
d. Los conjuntos de texto para cada anotador poseen un orden aleatorio.
e. Se creó la función getAnnotationSets(vqtySamples,vdfSample) en el script de Samples.py, el cual recibe la cantidad de muestras/anotadores que se requiere construir y el *DataFrame* que se utiliza para crear los anotadores.

El resultado de la creación de los anotadores es un *DataFrame* con la estructura de la Tabla 3. Este *DataFrame* es utilizado para la creación de los 15 documentos de anotación, filtrando su contenido por el número de anotador. Cada archivo es revisado por un anotador humano a fin de identificar el nivel de odio observado en el texto.

Tabla 3 Ejemplo de creación de anotadores

Annotator Index	Annotator	Order	Tweet ID	Query Search	Datetime	Text	Totalmente en desacuerdo	En desacuerdo	De acuerdo	Totalmente de acuerdo
A3-148	3	1	1	afilar	9/19/20 22:32	ya está me cansé de afilar la vereda con mis cuernos mañana le hago llegar la carta de divorcio				
A9-236	9	2	1	afilar	9/19/20 22:32	ya está me cansé de afilar la vereda con mis cuernos mañana le hago llegar la carta de divorcio				
A10-439	10	3	1	afilar	9/19/20 22:32	ya está me cansé de afilar la vereda con mis cuernos mañana le hago llegar la carta de divorcio				

Fuente: Propia

Una vez que los anotadores humanos identificaron la existencia de odio en los tweets, se procede a consolidar los datos de las anotaciones. Esto con el objetivo de asignar una categoría de odio final a cada tweet. Este proceso de consolidación brinda como resultado el corpus final de anotación, el cual es utilizado en la prueba de los modelos SVM y CNN. La Figura 7 detalla el proceso de construcción de los anotadores hasta la creación del corpus final. En la siguiente sección se detalla el proceso y las reglas utilizadas para la anotación manual de odio.

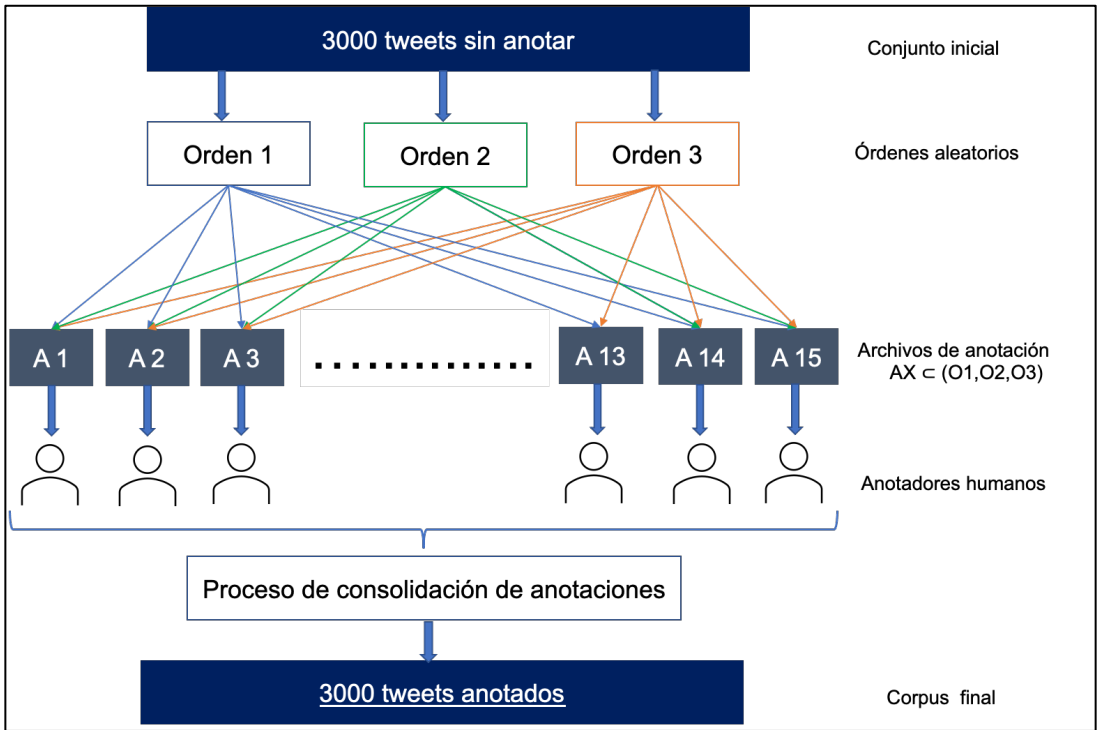


Figura 7 Proceso de anotación
Fuente: Propia

6.1.4 Proceso de anotación manual

Posterior a la generación de los 15 archivos anotadores, se procede con el proceso de la anotación manual. El cual, se basa en la revisión de cada uno de los textos de los *tweets* con el objetivo de revisar si posee o no una denotación de odio. Para este ejercicio, se creó el instrumento de *Instrucciones de anotación de tweets para la identificación de odio* el Anexo 3. Este instrumento provee las pautas para la anotación del odio en los textos. El anotador deberá elegir una opción de la escala: Totalmente en desacuerdo, En desacuerdo, De Acuerdo, o Totalmente de Acuerdo, si el comentario posee odio. Siendo Totalmente de Acuerdo la calificación más alta de odio y Totalmente en desacuerdo con la calificación más baja en donde el odio es inexistente. Entre más sea el odio presente en el *tweet* más de acuerdo el anotador deberá marcar la escala. Así mismo, el individuo debe tomar las siguientes consideraciones para identificar el odio y la calificación:

- Totalmente de acuerdo = Se identifica odio en el texto. Se identifica la persona, entidad o grupo ofendido por la expresión de odio.
- De acuerdo = El texto muestra odio, sin embargo, no identifica la persona, entidad o grupo al cual es dirigido.
- En desacuerdo = El texto no muestra odio, pero está sujeto a interpretaciones y podría contener odio.
- Totalmente en desacuerdo = Esta seguro de que el texto no muestra ningún tipo de odio hacia una persona, entidad o grupo.

Para la actividad de anotación se necesitaba la participación de 15 anotadores, que debían cumplir con el requisito de poseer un grado académico de bachillerato o superior. Se procedió a levantar la lista de personas 20 que podían cumplir con el requisito y la disponibilidad de realizar el ejercicio de anotación. Por lo tanto, la selección de los anotadores fue realizado siguiendo un criterio de conveniencia y disponibilidad. Al final, la actividad de anotación manual fue realizada por 17 personas que cumplían el requisito, 4 de los anotadores era mujeres y 13 hombres. En un inicio, se brindó un archivo de anotación para cada persona. Sin embargo, existieron 2 casos en donde la totalidad de la revisión se realizó entre 2 personas. El proceso de anotación se ejecutó entre el 8 de noviembre del 2020 y el 8 de diciembre el 2020. La descripción del criterio de selección de la población se muestra en la Tabla 4, en la cual se detalla el último grado académico obtenido o en curso por la persona.

Tabla 4 Descripción de la población de anotadores manuales

Ultimo nivel académico completado o en proceso	Cantidad de individuos
Estudiante Maestría Computación UCR	11
Bachillerato Computación UCR	2
Maestría Administración de Proyectos UNA	1
Doctorado en Computación UCR	1
Licenciatura Tecnologías de Información TEC	1
Estudiante de Maestría - Posgrado en Lingüística	1
Total de participantes en el proceso de anotación	17

Fuente: Propia

Cada persona recibió un archivo de Excel con 600 textos por revisar y anotar, utilizando la escala de identificación de odio descrita anteriormente. Una vez finalizado el ejercicio de anotación, se procedió con la revisión de la completitud de la anotación. En caso de identificar textos sin evaluación, se remitió el archivo incompleto a los mismos anotadores para que completaran los casos no evaluados. Finalmente, se obtuvieron los 15 archivos de anotación completos, los cuales constituirían el *corpus* inicial para la etapa de análisis y procesamiento del texto.

6.2 Definición de características y reglas de preprocesamiento de los *tweets*

A partir de los textos anotados manualmente, se procedió con la consolidación de los archivos de anotación, obteniendo una calificación de odio por texto. La calificación indica el grado de odio identificado, el resultado es un *data set* binario con la presencia de si o no hay odio en el texto. Las siguientes secciones detallan las reglas y procesamiento realizado para la preparación del *corpus* de entrenamiento.

6.2.1 Construcción del *corpus*

Posterior al etiquetado de odio de los anotadores, todos los textos contaban con 3 calificaciones. Por lo tanto, fue necesario consolidar la calificación asignando un valor a la escala de evaluación del odio. La escala con sus equivalencias numéricas es:

- Totalmente en desacuerdo = -2
- En desacuerdo = -1
- De acuerdo = 1
- Totalmente de acuerdo = 2

La Figura 8 muestra la asignación de odio, entre más odio identificado más alta la calificación consolidada. En caso contrario menor es la calificación. Posterior a la consolidación, se asigna la etiqueta de odio, si el valor está entre [-6, 0] es sin odio *(No Hate*) y si está entre [1,6] es con odio (*Hate*). La escala utilizada se aprecia en la Figura 8.

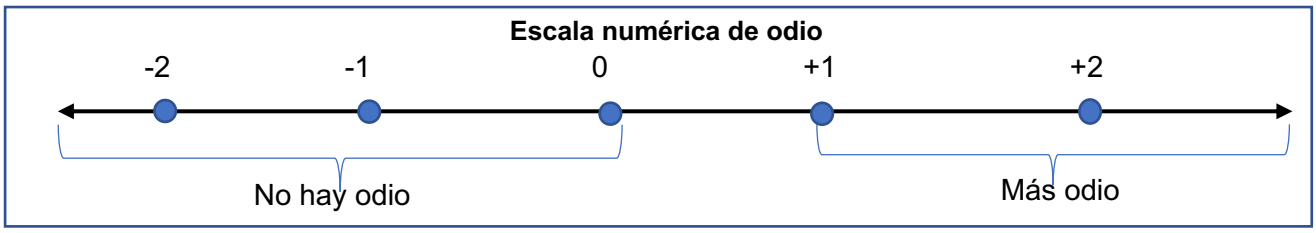


Figura 8 Escala numérica de odio
Fuente: Propia

Un ejemplo de aplicación de la escala y la calificación de odio se aprecia en la Tabla 5. En la cual, dos de los anotadores coincidieron que el texto no posee odio y el tercer anotador identificó un nivel menor de sin odio. Por lo tanto, se le asigna un -5 como calificación final.

Tabla 5 Ejemplo de calificación numérica de odio

Texto	**Calificación numérica**
@usuario Jajajajaja pobres tipos que batazo	-2
@usuario Jajajajaja pobres tipos que batazo	-2
@usuario Jajajajaja pobres tipos que batazo	-1
Calificación numérica	**-5** **No hay Odio**

Fuente: Propia

Al completar el proceso de revisión de las anotaciones y consolidación de resultados, la composición de *corpus* es de 17% de los textos poseen odio y un 83% no posee odio. La Tabla 6 muestra la cantidad de textos por cada una de las etiquetas, resultado de la anotación.

Tabla 6 Total de etiquetas de odio en el *corpus*

Etiquetas	**Cantidad de textos**	**Porcentaje**
Sin Odio	2500	83%
Odio	500	17%
Total	**3000**	**100%**

Fuente: Propia

6.2.2 Reglas de preprocesamiento implementadas

Las reglas de procesamiento consisten en la estandarización de texto antes de ser procesado para generar las características de los modelos de SVM y CNN. Las reglas aplicadas al *corpus* son las siguientes:

- Pasar todo el texto a minúsculas.
- Remover URLs contenidas en el texto.
- Remplazar menciones de otros usuarios por @MENTION.
- Remover emojis por medio del paquete demoji.
- Remover puntuación.
- Remover innecesarios saltos de línea.

Para efectos de esta investigación se decidió no eliminar los *stopwords*, debido al limitado texto disponible para hacer el análisis. Posterior a la implementación de estas reglas se continuó con la configuración de las características que son utilizadas para los modelos de SVM y CNN. Estos modelos esperan procesar vectores representativos de las palabras en los textos del *corpus* a fin de identificar similitudes y generar el clasificador binario. El proceso de *tokenización* se realizó para cada uno de los tipos de características a utilizar, por lo tanto, ese paso no se incluye en este proceso.

6.2.3 Definición de características para los modelos de SVM y CNN

La selección de las características y la parametrización de los modelos se realizó de acuerdo con la revisión de la literatura. Se procedió a realizar una tabla comparativa de la información de los estudios revisados. En ellos se identificó la utilización del modelo SVM con características de tipo unigrama para establecer el punto de referencia con los demás modelos en comparación. La Tabla 7 contempla los modelos utilizados y los mejores resultados obtenidos según la literatura.

Tabla 7 Resultados obtenidos en otros estudios en inglés

Estudio	**(Badjatiya et al., 2017)**	**(Davidson et al., 2017)**	**(Yuan et al., 2016)**	**(Liu & Forss, 2014)**	**(Kim, 2014)**	**(Greevy & Smeaton, 2004)**
Modelo con mejor puntuación	LSTM+Random Embedding+GBDT	Regresión Logística con regularización de L2	LSTM y regresión logística	Identificador de odio modelo propuesto	CNN static y word2vec	Polynomial SVM con BOW
Precisión	0.93	0.91	Únicamente métrica de exactitud de 0.91	0.64.43	Sin información	0.9278
Recall	0.93	0.9	Únicamente métrica de exactitud de 0.92	0.9628	Sin información	0.9
F1 Score	0.93	0.9	Únicamente métrica de exactitud de 0.93		89.6	0.9137
Features	ngrams, TDF-IDF, BoWV, word2vec_model	TF-IDF, POS, sentimiento, hashtags, menciones, *retweets*, URLs, número de caracteres, palabras, y sílabas	word2vec - word emdeddings for LSTM, (1 gram, 2 gram for SVM)	TF-IDF, N-grams, topic similarity, análisis de sentimiento	Word2vec aplicado a varios *dataset*s	BOW, N-grams, POS
Algoritmos	Regresión logística, Random Forest, SVM, GBDT, DNN, CNN	Regresión logística, SVM	Regresión logística, LSTM y SVM	Naive Bayes - NB Classifier	CNN	SVM
Validación	10-Fold Cross Validation	Validación cruzada de 5 Folds, manteniendo el 10% de la muestra para evaluación para ayudar a prevenir un ajuste excesivo.	Validación cruzada de 5 folds para evaluar el rendimiento de la clasificación.	Validación cruzada, no indica cantidad de folds.	No se menciona, únicamente se indica descenso de gradiente estocástico sobre mini-lotes mezclados con la regla de actualización de Adadelta	No se menciona
Resultados	CNN funcionó mejor que LSTM	La regresión logística y la SVM lineal tendían a funcionar significativamente mejor que otros modelos, sin embargo el 40% del discurso de odio fue clasificado incorrectamente.	El modelo de aprendizaje profundo de regresión logística y LSTM supera significativamente a los clasificadores SVM y Naive Bayes y el entrenamiento previo mejora aún más la precisión.	Los modelos basados en unigrama, muestran su valor y efectividad únicos en la clasificación de contenido web.	Una CNN simple con una capa de convolución funcionó notablemente bien.	Polynomial demostró ser la función de *kernel* más efectiva tanto para BOW como para POS, mientras que sigmoid tanh resultó ser la mejor para bigrams. BOW resultó en una alta recuperación, mientras que los bigrams mejoraron la precisión. POS también dio una alta recuperación, pero en general se desempeñó peor que BOW y bigrams y es computacionalmente costoso de hacer.

Fuente: Propia

Las características elegidas para el modelo de SVM corresponden a los unigramas y los vectores de tipo word2vec, de acuerdo con los estudios anteriores. Para los unigramas se utilizó el método de *scikit-learn de TfidfVectorizer* (*sklearn.feature_extraction.text.TfidfVectorizer — scikit-learn 0.24.2 documentation*, s. f.), que permite calcular una matriz de frecuencia de términos provenientes del vocabulario del texto de los *tweets*. Las matrices se calculan para los conjuntos de entrenamiento y validación. Para el conjunto de entrenamiento la cantidad de términos incluidos es de 31 961 mientras que conjunto de validación es de 12 982 términos. Así mismo, para la configuración de los *word embeddings,* se procedió con el cálculo *del bag of words* que contiene 10 875 palabras. Utilizando este diccionario de palabras se calculan los vectores de *word embeddings* de dimensión de 300. Los textos utilizados para generar los *word embeddings* tiene la estructura de la Tabla 8.

Tabla 8 Ejemplo de textos para calcular los *word embeddings*

Ejemplo de textos para calcular los *word embeddings*
['estoy', 'durmiendo', 'con', 'l', 'puerta', 'abierta', 'y', 'tengo', 'miedo']
['MENTION', 'MENTION', 'MENTION', 'dejé', 'de', 'darle', 'bola', 'cuando', 'se', 'hizo', 'una', 'cuenta', 'para', 'acosarme', 'pero', 'ya', 'cansa']
['MENTION', 'algúna', 'varilla', 'tengo', 'en', 'el', 'haber', 'jeje']
['tiene', 'las', '3', 'ps', 'pinche', 'pelón', 'puñal']
['rt', 'para', 'que', 'me', 'ayuden', 'a', 'pegarlos', 'con', 'goma']

Fuente: Propia

Los *word embeddings* utilizados para esta investigación el de (Cardellino, 2019). Esta investigación no contempla la creación del *word embeddings* a partir de los textos de los *tweets* extraídos. En cambio se utiliza los *word embeddings* pre entrenados de (Cardellino, 2019), estos poseen un *corpus* de 1 000 653 *word embeddings* de dimensión de 300. Los mismos son utilizados para crear las características de los modelos a utilizar SVM y CNN, incluyendo mayor cantidad de relaciones con potenciales palabras que se encuentran relacionadas a las palabras encontradas en los *tweets*. De esta manera la definición del *embedding* se beneficia de un vocabulario más extenso a diferencia de utilizar únicamente las palabras de los textos. Este beneficio también se conoce como transferencia de conocimiento, al utilizar un *embedding* pre entrenado para definir las características de un modelo, dado un vocabulario inicial del *corpus* a

utilizar. Por ejemplo, la palabra 'durmiendo' tiene mayor similaridad con las siguientes palabras de la Tabla 9 ,utilizando el *word embedding* de (Cardellino, 2019):

Tabla 9 Palabras con mayor similitud de durmiendo

Palabras con mayor similitud de durmiendo	
roncando	0.715
dormido	0.696
holgazaneando	0.691
dormir	0.678
dormidos	0.674
dormían	0.670
despertándola	0.668
comiendo	0.668
bebiendo	0.668
duchando	0.667

Fuente: Propia

El tamaño de 300 valores en el vector permite estandarizar la matriz final para ser consumida por los modelos. Las representaciones vectoriales del modelo se configuran extrayendo la representación promedio del *word embedding* original y se construye un nuevo vector por cada texto de los *tweets*. Las palabras que no son identificadas en el *word embedding* original se representan como un vector de 300 posiciones con valores de 0.00001. Esta situación es experimentada en el *token* 'MENTION'. Los *word embeddings* permiten una representación vectorial densa de palabras a partir de un vocabulario más extenso. Para efectos del entrenamiento a realizar en SVM se generó la matriz de 1785 x 300 a partir de los *word embeddings.* Así mismo, para el modelo de CNN se utilizó el Word *embeddings* de (Cardellino, 2019), la diferencia está en la configuración de los vectores que se realizó utilizando la capa de *Embedding* de *Keras* (Team, s. f.). Keras es una API de *Deep learning* de Python, que se ejecuta sobre la plataforma de aprendizaje automático *TensorFlow*, este fue desarrollado para permitir la experimentación rápida. La configuración del *embedding* para el modelo de CNN consistió en los siguientes parámetros.

- Tokens únicos 9375.
- Tamaño de las secuencias para el modelo del conjunto de entrenamiento y del conjunto validación: (2040, 70) (510, 70)
- Tamaño de las etiquetas para tensor de entrenamiento y validación: (2040, 2) (510, 2).
- Palabras mapeadas al *embedding* 7945 palabras y no mapeadas 1430.

A partir de las características definidas para SVM y CNN, se procede con las configuraciones de los modelos y su parametrización.

6.3 Implementación del modelo SVM y el CNN

Esta investigación utiliza 16 variantes del modelo de *Support Vector Machine* y 4 para la red convolucional CNN. El primer modelo de SVM, establece la línea base para analizar la ejecución de los 6 modelos configurados. Esta decisión fue definida de acuerdo con lo observado en los estudios de la Tabla 7, en su mayoría utilizan la configuración básica del modelo SVM como primer modelo de evaluación. Para la ejecución de los modelos se realizó una valoración inicial correspondiente a una primera ejecución. Este proceso permitió la revisión de la configuración de los modelos y observar su desempeño. Posteriormente, se procedió con el proceso de validación cruzada y analizar los resultados obtenidos. Finalmente, se identificaron los modelos objetivos para realizar el último análisis sobre un conjunto de datos no utilizados hasta la etapa final. Esto con el objetivo de evaluar la efectividad de la predicción. La Figura 9 muestra el proceso de evaluación ejecutado para la implementación de los modelos de SVM y CNN.

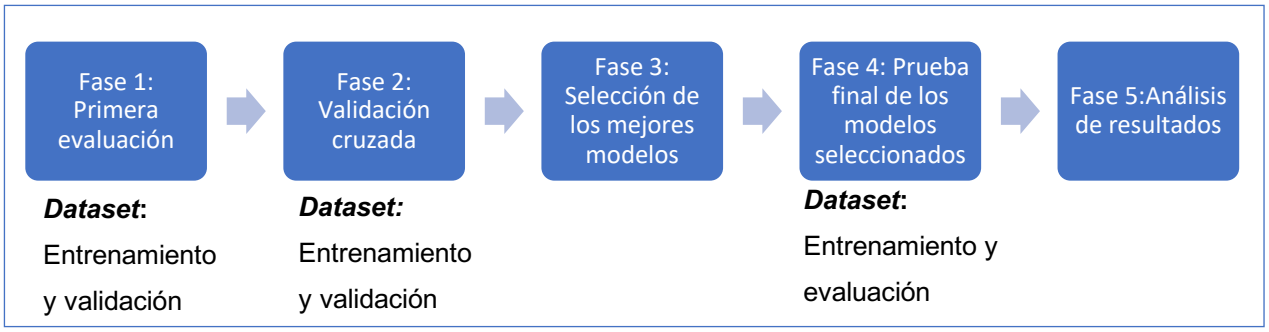


Figura 9 Proceso de implementación de modelos
Fuente: Propia

6.3.1 Partición del *corpus* para entrenamiento, desarrollo y evaluación

El *corpus* inicial se dividió en forma aleatoria para efectos de entrenamiento, validación y evaluación (*testing*). De esta forma, se generaron 3 *datasets* respectivamente (evaluación, entrenamiento y validación). Los *datasets* de entrenamiento y validación fueron utilizados para validar y configurar los modelos. El *dataset* de evaluación se dejó por aparte para la evaluación final de la predicción de los modelos. La distribución del *corpus* consistió en asignar un 15% del *corpus* para evaluación, del 85% restante se utilizó un 80% para entrenamiento y 20% para validación. La distribución de muestras se realizó como lo indica la Figura 10.

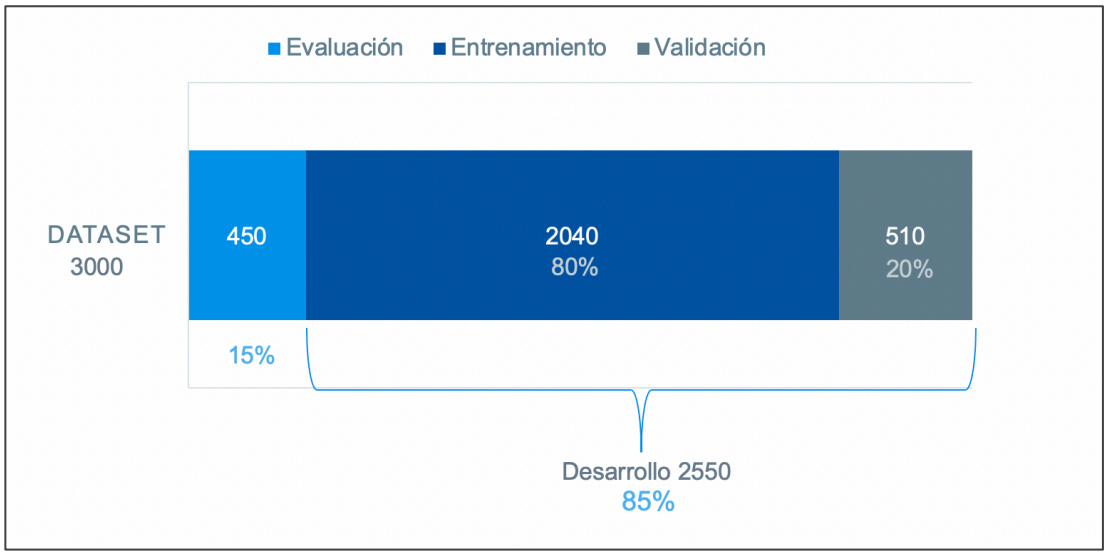
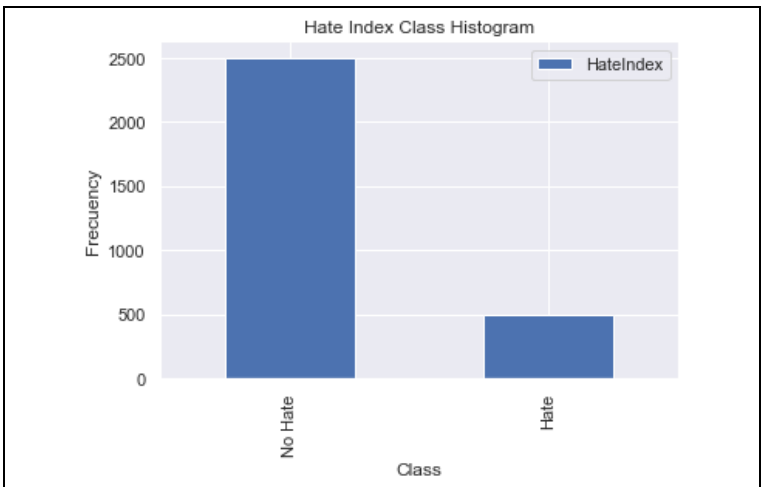


Figura 10 Distribución de los sets de entrenamiento, validación y evaluación
Fuente: Propia

6.3.1.1 Validación cruzada

Los subconjuntos de entrenamiento y validación son utilizados para la primera evaluación, la cual tiene un enfoque de tabla de entrenamiento y tabla de validación. En donde, únicamente se verifica el entrenamiento y configuración del modelo para una única ejecución. A diferencia del conjunto de desarrollo, el cual se utiliza en su totalidad para realizar el ejercicio de validación cruzada. Este último es utilizado para generar subconjuntos aleatorios para entrenamiento y validación para un número específico de iteraciones. Para efectos de esta investigación, se procedió a utilizar la validación cruzada estratificada. Este tipo de validación fue seleccionada debido al desbalance de las clases de odio y no odio presente en el *corpus*. La validación estratificada corresponde a un tipo de validación cruzada que es utilizado para situaciones de desbalanceo de clases (scikit-learn developers, 2021). Esta validación permite generar los subconjuntos de cada iteración, manteniendo la proporción de clases original del conjunto inicial. En este ejercicio, el conjunto inicial es equivalente al conjunto de desarrollo de la Figura 10. La distribución de las clases de odio se identifican en la Figura 11.



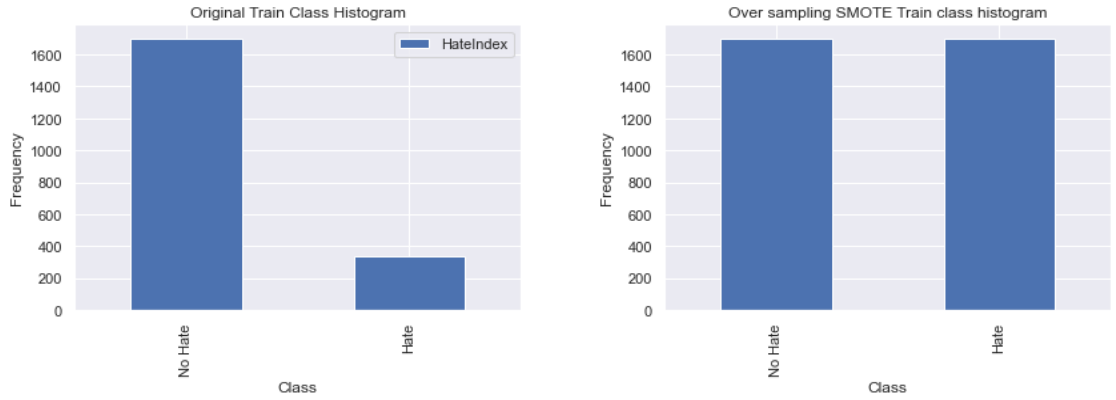
Class	Hate Index Count	Percentage
No Hate	2500	83.3%
Hate	500	16.67%

Figura 11 Distribución de odio en el *corpus*
Fuente: Propia

6.3.1.2 Sobremuestreo SMOTE

En relación con la distribución de clases de odio y no odio en los textos, es necesario considerar el sobremuestro a fin de balancear las clases y observar si se obtienen mejores resultados en la ejecución del modelo. Este enfoque se sustenta en (Solís, 2020), el cual identificó que utilizando el método de sobremuestro de SMOTE se obtenían mejores resultados en la clasificación en comparación con los resultados de sub muestreo en un *corpus* con clases desbalanceadas. A partir de esta afirmación, se procede a configurar dos modelos de SVM utilizando el sobremuestro SMOTE. La implementación de SMOTE, es básica sin ningún parámetro especial de configuración. Así mismo, se configuran las validaciones cruzadas de los modelos que utilizan esta técnica, considerando el cálculo de SMOTE en cada iteración. De esta manera, cada conjunto de entrenamiento es ajustado antes de realizar la evaluación de la iteración. SMOTE se aplica para las características basadas en TF-IDF y de *word embeddings*. Adicionalmente, en la implementación de SMOTE se utilizó el paquete *imbalanced-learn*, este proporciona herramientas para trabajar clasificaciones con clases desequilibradas (imbalanced-learn developers, 2021).

Conjunto de entrenamiento antes y después de SMOTE



Class	Hate Index Count
No Hate	1701
Hate	339

Class	Hate Index Count
No Hate	1701
Hate	1701

Figura 12 Distribución de clases antes y después de SMOTE
Fuente: Propia

En la Figura 12 se muestra la distribución de las etiquetas del *dataset* de entrenamiento, la cual también está desbalanceada. Aplicando SMOTE a este *dataset* se obtiene los resultados de la Figura 12, en donde se balancea la cantidad de muestras por la clase de odio y no odio. Así mismo, este enfoque es aplicado en la validación cruzada de los modelos que utilizan SMOTE. Se observa en la Figura 12, la cantidad de muestras con odio (*Hate*) aumenta al mismo nivel de la clase sin odio (*No Hate*). Posterior a la definición de los *datasets* para cada modelo, se procede con la ejecución de esto y su evaluación en referencia a las métricas de desempeño. Estas se explican en la siguiente sección.

6.3.2 Métricas de evaluación de desempeño de los modelos

Los resultados de ejecución de los modelos de clasificación se analizan de acuerdo con las predicciones realizadas con el conjunto de validación y el conjunto de entrenamiento. Las clases de no odio y de odio fueron sustituidas por los valores 1 y 0, definiendo el tipo de modelo de clasificación. Los resultados de los modelos se analizan utilizando las métricas de la matriz de confusión y las métricas calculadas sobre la matriz(Fawcett, 2006). La matriz de confusión se constituye como se muestra en la Figura 13 .

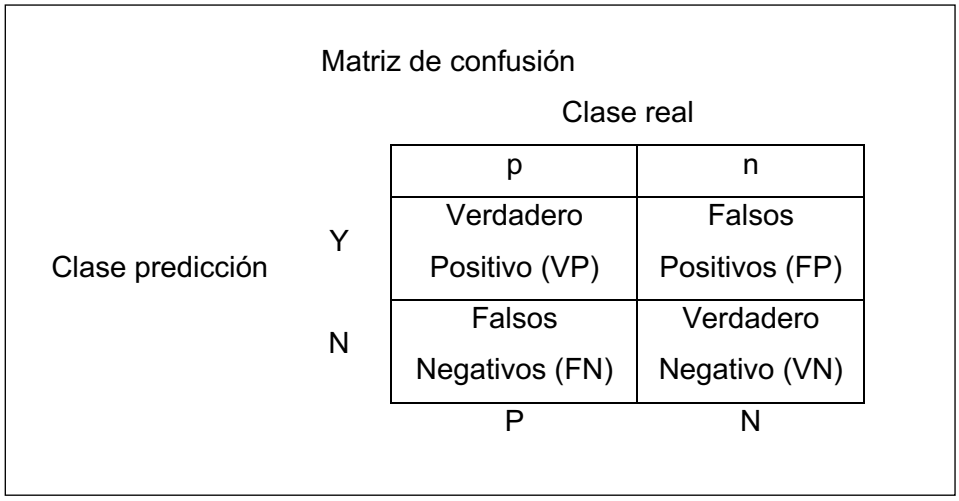


Figura 13 Matriz de confusión
Fuente: Propia

Las métricas derivadas de la matriz anterior con sus fórmulas correspondientes son las siguientes(Fawcett, 2006):

Precisión: $\frac{VP}{VP + FP}$ **Exactitud:** *(Accuracy)* $\frac{VP+VN}{P+N}$

Recall: $\frac{VP}{P}$ **Medida F1:** *(F1 Score)* $\frac{2 * (\text{precisión} * \textit{recall})}{\text{precisión} + 1/\textit{recall}}$

La métrica de precisión identifica la calidad del modelo en la detección correcta de la clasificación de la clase esperada, mientras que la medida de *recall* nos indica la cantidad de muestras positivas identificadas correctamente. La medida F1 corresponde al promedio ponderado de la precisión y el *recall*. Esta métrica alcanza su mejor valor en 1 y la peor puntuación en 0 (*sklearn.metrics.f1_score — scikit-learn 0.24.2 documentation*, s. f.). Por su parte la exactitud (*accuracy*) mide el porcentaje de muestras que fueron clasificados correctamente.

En la validación cruzada se calcula el promedio de las métricas de *recall*, precisión y medida de F1 score, obtenidas en todas las iteraciones. Estos resultados son analizados y se procede a escoger los 2 modelos con mejor puntuación. Así mismo, se utiliza la medición del error en cada iteración para analizar el desempeño de cada iteración el error corresponde al resultado 1 – exactitud. En este punto de la investigación, se tiene identificadas las características, *datasets* y métricas requeridas para analizar los modelos de SVM y CNN. El siguiente paso corresponde a

la definición de la parametrización de los modelos. La siguiente sección detalla la parametrización propuesta para el análisis.

6.3.3 Parametrización de los modelos SVM

Se implementó el modelo de SVM y el CNN para el análisis de odio de comentarios en español, utilizando las características utilizando términos frecuentes (TF-IDF) y los *word embeddings*. Para el modelo SVM se identificó el *kernel* como el único parámetro a probar con sus cuatro variaciones. Se realizaron ejecuciones con los *kernels: linear, polynomial, rbf y sigmoid* para el SVM con cada una de las combinaciones de la Tabla 10. El total de modelos de SVM por evaluar corresponde a 16 modelos, 4 ejecuciones para cada modelo.

Tabla 10 Parametrización de los modelos SVM

Núm.	Nombre del modelo	Kernel	Método de características	SMOTE	Justificación
1	SVM TF-IDF	Sigmoid Poly Rbf linear	TF-IDF	No	Línea base de comparación. unigrama para línea de comparación observado en (Liu & Forss, 2014)(Greevy & Smeaton, 2004)
2	SVM WORD EMBEDDINGS	Sigmoid Poly Rbf linear	*word embeddings*	No	Línea base de comparación del modelo con *word embeddings*
3	SVM SMOTE TF-IDF	Sigmoid Poly Rbf linear	TF-IDF	Sí	Analizar si el sobremuestro beneficia la clasificación.
4	SVM SMOTE WORD EMBEDDINGS	Sigmoid Poly Rbf linear	*word embeddings*	Sí	Analizar si el sobremuestro beneficia la clasificación.

Fuente: Propia

6.3.4 Parametrización de los modelos CNN

Para el ejercicio de la configuración de la red convolucional CNN, se procedió con la construcción de 4 redes con parámetros distintos. El detalle de la configuración de las redes se observa en la Tabla 11. Las redes convolucionales utilizan únicamente como características los vectores de tipo de *word embeddings*. Además, no se implementó la técnica de SMOTE para la evaluación de las redes. No obstante, se aplicó el método de validación cruzada. Los parámetros configurado implementación de CNN son: número de clases es 2, máxima cantidad de palabras para el tokenizador del *word embeddings* es 20 000. El tamaño de los *embeddings* es 300 y el largo máximo de la secuencia es 70. Este tamaño de secuencia se definió en base al tamaño del texto de los *tweets* del *corpus* los cuales tienen una extensión de hasta 70 palabras. Las

secuencias para la red se definen de máximo de 70, convirtiendo el texto en secuencias de enteros. La configuración de los tensores queda como se muestra en la Figura 14.

```
Found 9375 unique tokens.
Pad Shape of X train and X validation tensor: (2040, 70) (510, 70)
Shape of label train and validation tensor: (2040, 2) (510, 2)
Pad Shape of X  tensor: (2550, 70)
Shape of labels X tensor: (2550, 2)
```

Figura 14 Configuración de las secuencias de los tensores
Fuente: Propia

Un ejemplo de la configuración de los tensores para el *dataset* de entrenamiento para los 10 primeros textos y 20 posiciones dentro de la matriz se muestra en la Figura 15. Las representaciones de cada palabra se observan en la Figura 14, cada celda representa el valor otorgado por el *embedding*, los valores de 0 se asignan después de la última palabra del texto hasta completar las 70 posiciones. Es decir, la extensión del *tweet* con más palabras.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	1	1	12	1311	1847	31	3067	28	1312	15	3068	6	968	33	742	2	1848	33	31	8	3069
1	1	3070	3071	3072	7	743	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	21	31	8	1313	189	65	2	1850	5	177	744	6	83	0	0	0	0
3	969	1314	1311	1315	7	745	2	3073	3074	3075	15	3076	6	66	0	0	0	0	0	0	0
4	3077	28	1851	6	970	5	27	600	3078	746	3079	229	0	0	0	0	0	0	0	0	0
5	3080	971	3081	13	3082	5	3083	3084	73	3	31	12	3085	4	1852	2	3086	3	116	1316	27
6	1	52	393	129	1317	441	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3087	3	48	601	7	747	44	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	13	3088	3	199	1	442	602	4	200	6	4	1318	16	3089	3090	0	0	0	0	0	0
9	1	213	230	10	3091	6	11	748	1854	603	0	0	0	0	0	0	0	0	0	0	0
10	1	1	17	138	2	749	25	3	394	2	1319	3092	8	3093	9	26	3094	0	0	0	0

Figura 15 Ejemplo de la matriz de los tensores de *dataset* de entrenamiento 20 x 10
Fuente: Propia

Este estudio incluye la prueba de 2 configuraciones de redes convolucionales de referencia. Para cada modelo se realizaron 2 ejecuciones: una con *batch* de 32 y otra con *batch* de 128. Los parámetros utilizados para para cada red, se detalla en la siguiente Tabla 11.

Tabla 11 Parámetros de configuración de CNN

Núm.	Nombre del modelo	Método de características	Tipo de red	Parámetros	Justificación
1	CNN 1D Word embeddings	*word embeddings*	CNN – 1D conv net	Max Sequence Length = 70 filters = 100 class = 2 *batch* size = 32 y 128 Dim embedding = 300 ***Epochs*= 5** **Filter sizes =5** **Dropout = 0.5** **Optimizer = Adam** **Loss= Binary cross entropy**	Línea base de comparación del desempeño de las CNN. Se utilizó como base la implementación de la configuración del (Mark Omernick, Francois Chollet, 2019) Así mismo, se realizó la validación del parámetro de pérdida a utilizar y se seleccionó el *binary cross entropy.*
2	CNN 1D Diff Kernel Word embeddings	*word embeddings*	CNN – 1D conv net	Max Sequence Length = 70 filters = 100 class = 2 *batch* size = 32 y 128 Dim embedding = 300 ***Epochs*= 5** **Filter sizes = [3,4,5]** **Dropout = 0.5** **Optimizer = Adam** **Loss= Binary cross entropy**	Tunning de la red básica 1 con los parámetros utilizados por (Kim, 2014). El cual utilizó ventanas de filtro de 3, 4, 5 con 100 mapas de características y con un *dropout* de 0.5 y tamaño de *batch* de 50.

Fuente: Propia

Los principales parámetros de configuración de la de red CNN son el tamaño del *batch*, *epochs*, filtros y clases. El tamaño del *batch,* define el número de muestras a utilizar para realizar las predicciones. El tamaño puede ser del tamaño de todo el conjunto de entrenamiento o puede ser del mismo tamaño de una muestra del conjunto de entrenamiento. Los valores por defecto para el tamaño de *batch*, corresponde a 32, 64 y 128. El valor de 32 es considerado de referencia y default para la ejecución de redes neuronales (Bengio, 2012). Para efectos de esta investigación se procedió a realizar la ejecución con los valores de *batch* de 32 y 128. Las cuales corresponden a un análisis de tipo *Mini-batch Gradient Descent*. Este enfoque espera que el valor del *batch* oscile entre 1 y el tamaño del conjunto de entrenamiento.

Adicionalmente, se determinó utilizar 5 *epochs,* que corresponde a la cantidad de iteraciones a realizar dentro de la red. Se realizaron pruebas con 5,10 y 50 *epochs*. Los resultados mostraron que era mejor mantener en 5 las *epochs,* debido que no se muestra una mejora en el entrenamiento con mayor número de *epochs.* En la Figura 16 se aprecia el decremento observado en la ejecución con 50 *epochs* sobre la red básica con *batch* de 32.

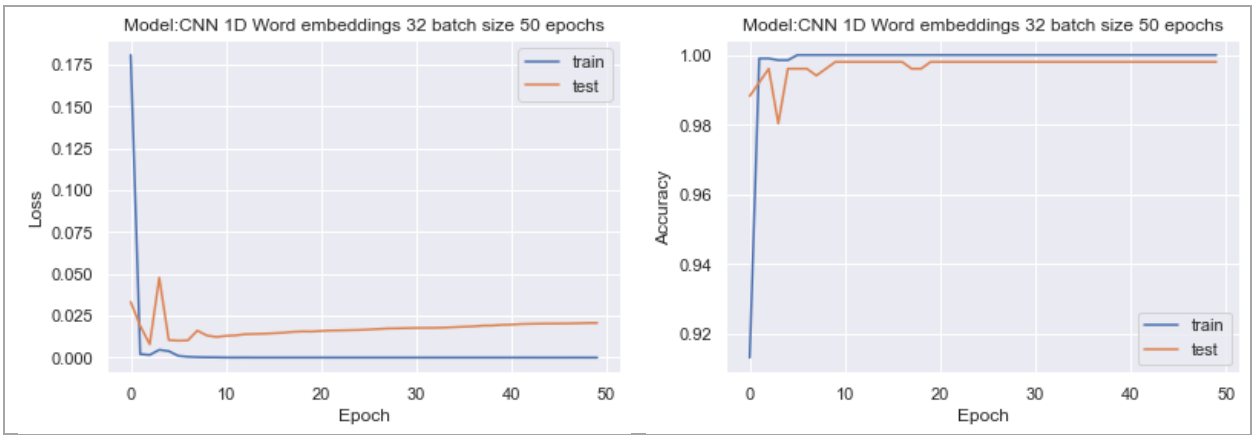


Figura 16 Ejemplo CNN con 100 *epochs*
Fuente: Propia

Finalmente, el optimizador a utilizar es Adam. Este optimizador es un método que calcula las tasas de aprendizaje adaptativo individuales para diferentes parámetros a partir de estimaciones del primer y segundo momento de los gradientes (Kingma & Ba, 2017). Además, el tipo de CNN a utilizar es el de 1D, al lugar de utilizar el CNN 2D. Estás últimas son utilizadas para análisis de imágenes o con características que posean más de una dimensión. Las CNN 2D se recomiendan para el análisis de video e imágenes. Las CNN 1D, por su parte, han sido preferidas para el reconocimiento del discurso a diferencia de las 2D de acuerdo con (Kiranyaz et al., 2021). En consecuencia, se decidió utilizar las CNN de 1D. Además, de acuerdo con (Masters & Luschi, 2018) el uso de *batch* pequeños logra la mejor estabilidad de entrenamiento y rendimiento de generalización. Por lo tanto, se procede a realizar las ejecuciones con *batch* de 32 y de 128 para analizar el impacto del tamaño del *batch*. Los resultados de ejecuciones de los modelos del CNN y SVM se detallan en el siguiente apartado.

6.4 Análisis y comparación de resultados obtenidos

Esta sección detalla los resultados obtenidos en las 3 etapas de ejecución de los modelos. La primera evaluación con el conjunto de entrenamiento y el conjunto de validación. La segunda evaluación se obtiene con los resultados de la validación cruzada. Por último, la tercera evaluación que ejecuta las predicciones de los mejores valores obtenidos en la medida de F1 de la etapa 2.

6.4.1 Resultados de ejecución de los modelos con 1 ejecución

Los resultados obtenidos en la primera ejecución se muestran en la Tabla 12. Los resultados se organizaron de mayor a menor valor de la medida de F1 (Columna 5 de izquierda a derecha). En esta ejecución de 1 iteración con el conjunto de datos de entrenamiento y validación, se observa que las 3 primeras posiciones con mejor F1 corresponde a resultados de los modelos de CNN. Procederemos a enfocar esta primera etapa en los modelos que cumplan con una evaluación de superior a la línea base del SVM, la cual corresponde a 0.918 de la ejecución *Model: SVM TF-IDF linear*. Esta ejecución es la mejor obtenida para la línea base. En la Tabla 12 la se resalta la línea base en color verde.

De acuerdo con los resultados obtenidos en la primera evaluación, se observa que el modelo de referencia de SVM obtiene su mejor calificación con el *kernel* linear, obteniendo la mejor sexta posición. La Tabla 12 se encuentra ordenada de mayor a menor valor de la calificación obtenida en la medida F1. Con el objetivo, de enfocar el análisis en los mejores modelos, se establece estudiar las primeras mejores posiciones en la tabla sobre el mejor resultado del modelo de base. Así mismo, se observa que para esta evaluación los modelos que poseen SMOTE son los TDF tienen mejor resultados que la línea base. En la Figura 17 se observa los resultados de las matrices de confusión. Adicionalmente, se percibe que la mayor parte de los textos son clasificados sin odio. Esto corresponde a una buena clasificación debido que, de los 510 textos del subconjunto de validación, únicamente 81 textos tenían odio. Particularmente 2 de las redes CNN obtuvieron un *recall* alto, únicamente mal clasificando un texto. Las dos primeras matrices de confusión muestran este comportamiento en la Figura 17.

Tabla 12 Resultados primera evaluación

Rank	Execution	Accuracy	Precision	Recall	Mean Absolute Error	F1 Score
1.	Model: CNN 1D Word embeddings 128 *batch* size 5 *epochs*	0.998	0.998	1.000	0.002	0.999
2.	Model: CNN 1D Diff Kernel 128 *batch* Word embeddings 5 *epochs*	0.998	0.998	1.000	0.002	0.999
3.	Model: CNN 1D Diff Kernel 32 *batch* Word embeddings 5 *epochs*	0.982	0.988	0.991	0.018	0.990
4.	Model: SVM SMOTE TF-IDF linear	0.871	0.890	0.965	0.129	0.926
5.	Model: SVM SMOTE TF-IDF sigmoid	0.869	0.908	0.939	0.131	0.923
6.	Model: SVM TF-IDF linear	0.851	0.852	0.995	0.149	0.918
7.	Model: SVM WORD EMBEDDING poly	0.847	0.846	1.000	0.153	0.917
8.	Model: SVM SMOTE TF-IDF poly	0.847	0.846	1.000	0.153	0.917
9.	Model: SVM WORD EMBEDDING rbf	0.845	0.844	1.000	0.155	0.916
10.	Model: SVM TF-IDF rbf	0.843	0.843	1.000	0.157	0.915
11.	Model: SVM WORD EMBEDDING linear	0.841	0.841	1.000	0.159	0.914
12.	Model: SVM TF-IDF poly	0.841	0.841	1.000	0.159	0.914
13.	Model: SVM SMOTE TF-IDF rbf	0.841	0.843	0.998	0.159	0.914
14.	Model: SVM TF-IDF sigmoid	0.841	0.848	0.988	0.159	0.913
15.	Model: CNN 1D Word embeddings 32 *batch* size 5 *epochs*	0.839	0.897	0.914	0.161	0.905
16.	Model: SVM WORD EMBEDDING sigmoid	0.822	0.849	0.958	0.178	0.900
17.	Model: SVM SMOTE WORD EMBEDDING rbf	0.798	0.914	0.839	0.202	0.875
18.	Model: SVM SMOTE WORD EMBEDDING poly	0.761	0.932	0.772	0.239	0.844
19.	Model: SVM SMOTE WORD EMBEDDING linear	0.716	0.949	0.699	0.284	0.805
20.	Model: SVM SMOTE WORD EMBEDDING sigmoid	0.596	0.905	0.580	0.404	0.707

Fuente: Propia

Matrices de confusión de la primera evaluación (conjunto de datos y validación)

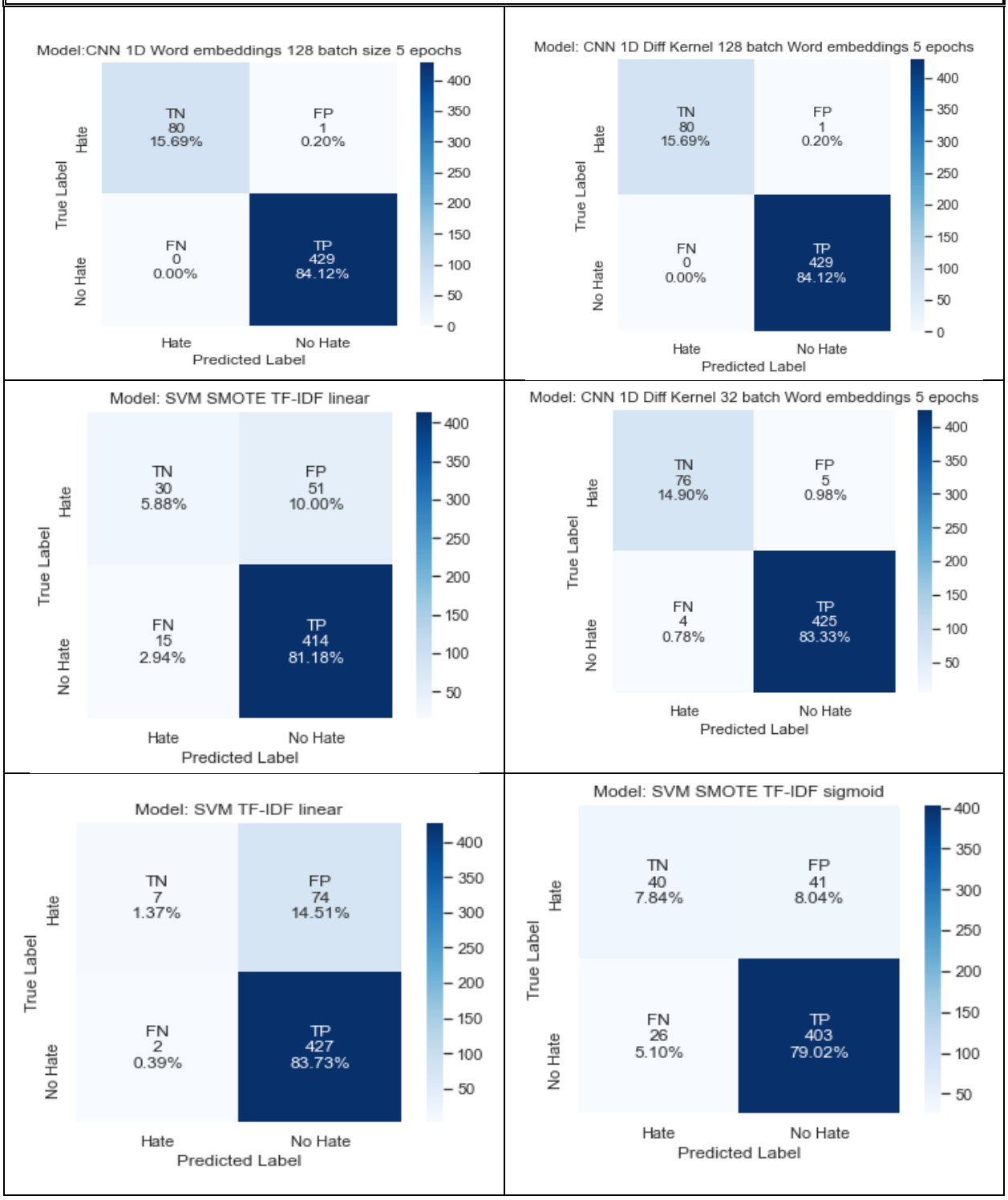


Figura 17 Matrices de confusión de la primera evaluación
Fuente: Propia

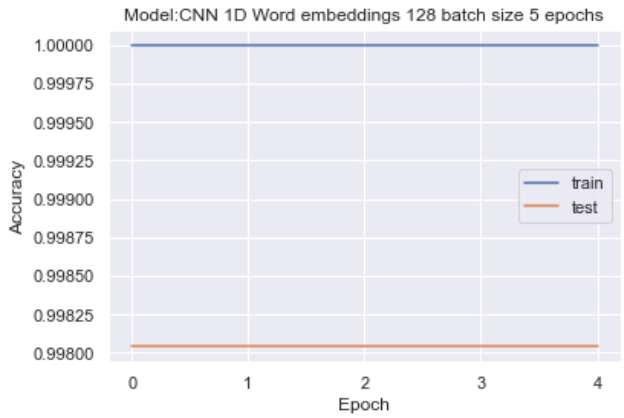
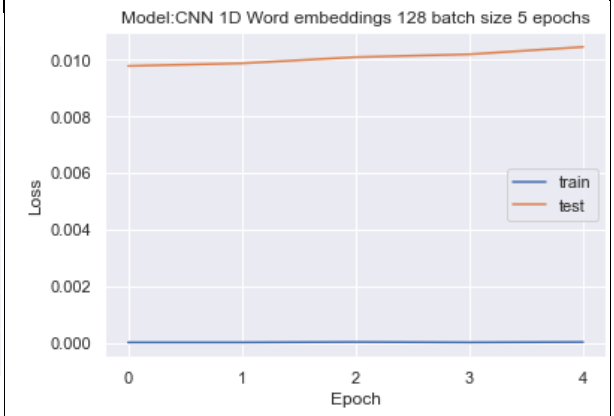
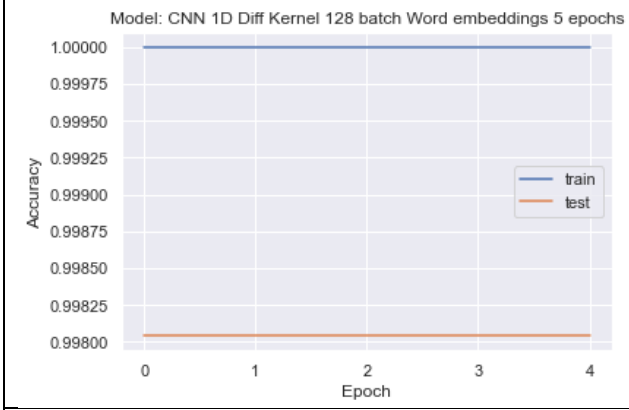
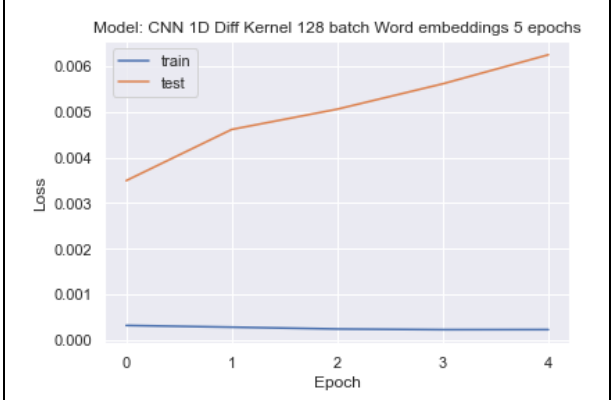
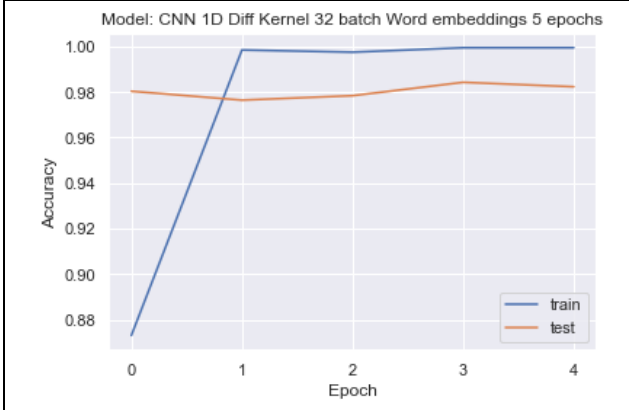
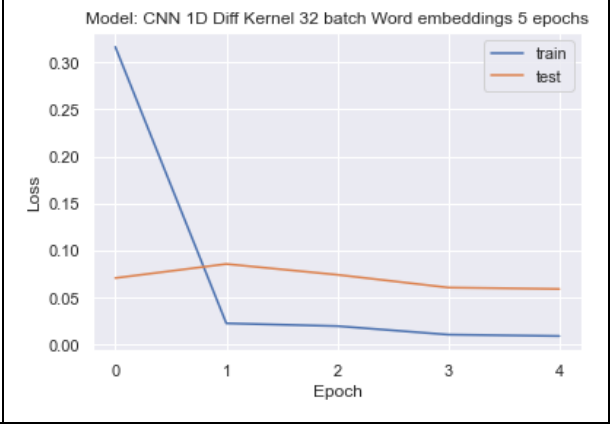
CNN Accuracy	CNN Loss
	
	
	

Figura 18 Desempeño de modelos de CNN
Fuente: Propia

En referencia del desempeño de las redes convolucionales de las primeras 3 mejores posiciones, estas se muestran en la Figura 18. La diferencia en las métricas de pérdidas y de precisión son mínimas entre el conjunto de entrenamiento y el conjunto de validación. No obstante, la pérdida tiende a incrementar en las ejecuciones con *batch* de tamaño de 128. Sin embargo, el incremento corresponde a una distancia de milésimas. La configuración que muestra mayor estabilidad

corresponde a la ejecución con diferentes tamaños de filtros y *batch* de 32 bits, tanto la precisión como la pérdida se mantiene con un comportamiento similar. Se procederá a comprobar su efectividad en la validación cruzada, tomando todo el conjunto de entrenamiento y de validación en un solo conjunto, sobre el cual se itera n veces para obtener resultados de los posibles resultados del modelo. Este procedimiento se detalla en la siguiente sección.

6.4.2 Resultados de ejecución de la validación cruzada

Se ejecutaron 10 validaciones cruzadas para cada una de las combinaciones de SVM y CNN explicadas en las secciones anteriores. Los resultados por iteración se observan en la gráfica de Figura 19 y la Figura 20. La gran mayoría de las combinaciones obtiene un desempeño superior al 0.8 en la métrica de F1. Prevalece el desempeño de F1 observado anteriormente en la evaluación del entrenamiento y conjunto de validación. Únicamente la CNN con *word embeddings* de 32 *batch*, muestra un comportamiento atípico y no constante en las iteraciones, su F1 score no es constante. Al menos, seis configuraciones mantienen un comportamiento estable en las 10 iteraciones. La configuración de SVM SMOTE *word embedding* con el núcleo Sigmoid, fue la ejecución con el desempeño más mínimo y con el máximo valor de error.

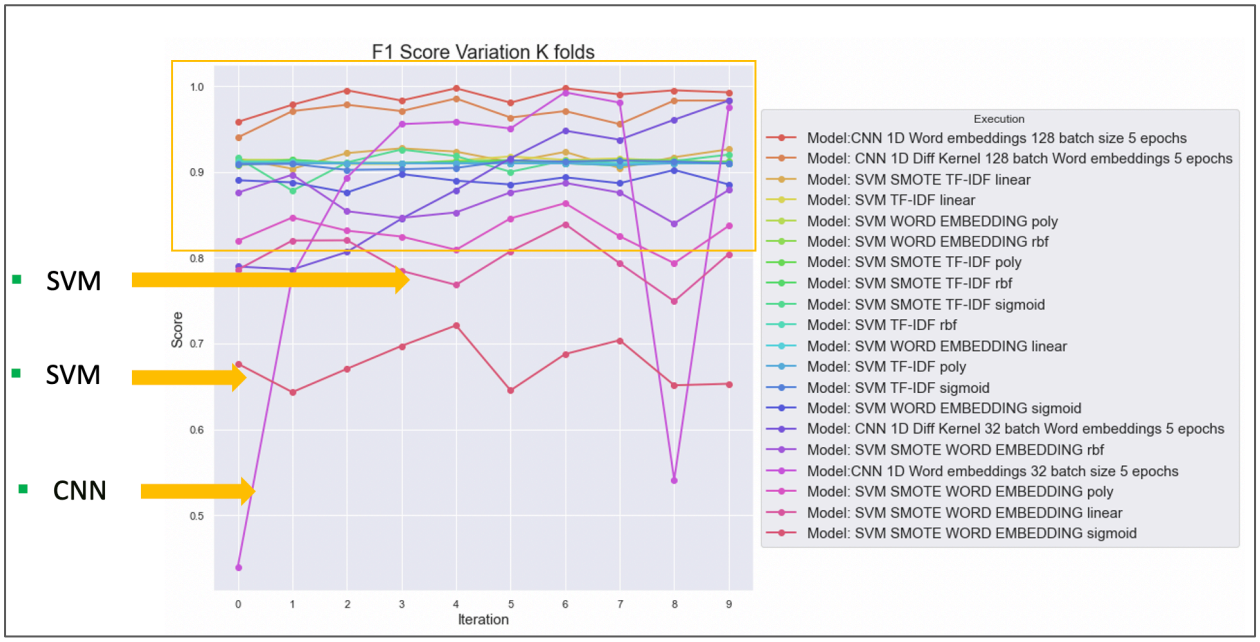


Figura 19 Variación de la métrica F1 en la validación cruzada
Fuente: Propia

En la siguiente gráfica se observa el error observado en cada iteración, se observa que la CNN con *word embeddings* de 32 *batch con 5 epochs*, obtiene la mayor variación del error, oscilando entre [0.0,0.6] en las iteraciones de la validación cruzada.

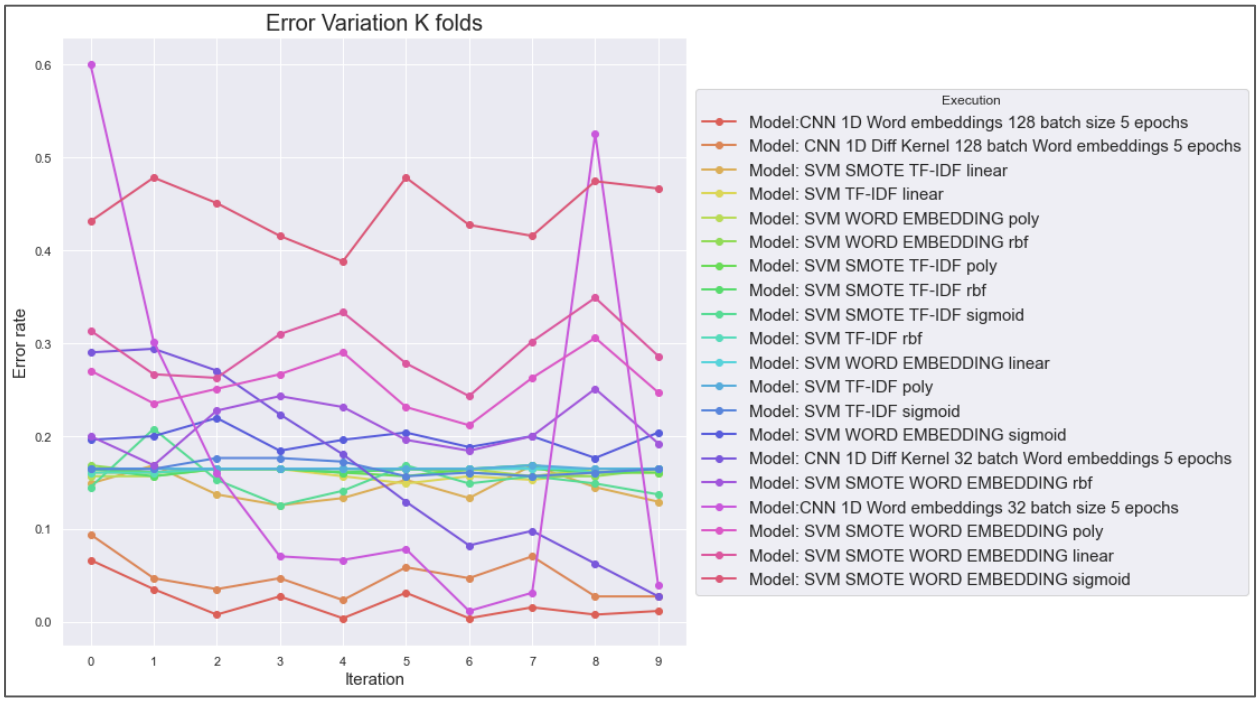


Figura 20 Variación del error en la validación cruzada
Fuente propia

Las métricas de *recall*, precisión, exactitud y finalmente F1 para la validación cruzada, corresponden al promedio de los resultados de las iteraciones. La Tabla 13 detalla los resultados obtenidos para todas combinaciones de las ejecuciones. Se observa, que la línea base del modelo SVM TF-IDF linear se encuentra en la cuarta posición en esta ocasión. Únicamente tres modelos quedan por encima del resultado de la línea base. Así mismo, se puede observar que hay 10 modelos que se encuentran muy cerca del 0.91 para la métrica de F1. Las posiciones 5 y 6 de la Tabla 13, corresponden a los modelos con SVM con *word embeddings* , los cuales tienen una diferencia mínima por debajo de la línea base. Además, las ejecuciones con *batch* de 32 de las redes CNN, descendieron hasta las posiciones 15 y 17 de la tabla, a diferencia de las redes CNN con *batch* de 128, las cuales se posicionaron en las primeras 2 posiciones de la tabla de resultados. Por lo tanto, las CNN de *batch* de 128 obtienen mejores resultados a diferencia de utilizar *batch* pequeños para este ejercicio.

Tabla 13 Resultados de la validación cruzada

Rank	Execution	CV Mean Accuracy Score	CV Mean Absolute Error	CV Mean Recall	CV Mean Precision	CV Mean F1 Score
1.	Model: CNN 1D Word embeddings 128 *batch* size 5 *epochs*	0.979	0.021	0.975	1.000	0.987
2.	Model: CNN 1D Diff Kernel 128 *batch* Word embeddings 5 *epochs*	0.952	0.048	0.943	1.000	0.970
3.	Model: SVM SMOTE TF-IDF linear	0.856	0.144	0.962	0.877	0.918
4.	Model: SVM TF-IDF linear	0.841	0.159	0.995	0.843	0.913
5.	Model: SVM WORD EMBEDDING poly	0.838	0.162	0.999	0.838	0.912
6.	Model: SVM WORD EMBEDDING rbf	0.837	0.163	1.000	0.837	0.911
7.	Model: SVM SMOTE TF-IDF poly	0.837	0.163	0.999	0.838	0.911
8.	Model: SVM SMOTE TF-IDF rbf	0.837	0.163	0.999	0.837	0.911
9.	Model: SVM SMOTE TF-IDF sigmoid	0.847	0.153	0.938	0.886	0.911
10.	Model: SVM TF-IDF rbf	0.836	0.164	1.000	0.836	0.911
11.	Model: SVM WORD EMBEDDING linear	0.835	0.165	1.000	0.835	0.910
12.	Model: SVM TF-IDF poly	0.835	0.165	1.000	0.835	0.910
13.	Model: SVM TF-IDF sigmoid	0.835	0.165	0.990	0.841	**0.909**
14.	Model: SVM WORD EMBEDDING sigmoid	0.803	0.197	0.950	0.837	0.890
15.	Model: CNN 1D Diff Kernel 32 *batch* Word embeddings 5 *epochs*	0.834	0.166	0.801	1.000	0.885
16.	Model: SVM SMOTE WORD EMBEDDING rbf	0.791	0.209	0.830	0.912	0.868
17.	Model: CNN 1D Word embeddings 32 *batch* size 5 *epochs*	0.811	0.189	0.774	1.000	0.847
18.	Model: SVM SMOTE WORD EMBEDDING poly	0.743	0.257	0.752	0.927	0.830
19.	Model: SVM SMOTE WORD EMBEDDING linear	0.705	0.295	0.696	0.935	0.797
20.	Model: SVM SMOTE WORD EMBEDDING sigmoid	0.557	0.443	0.551	0.871	0.675

Fuente: Propia

A fin de identificar el conjunto de modelos, por evaluar la efectividad en la predicción, se procede a analizar las gráficas de error y de F1. Se observa en la Figura 21, que hay 6 modelos que mantienen una convergencia entre 0.15 y 0.17 de la variación del error en las iteraciones 2 y 3 de la validación cruzada. A diferencia de las 3 mejores ejecuciones, que poseen mayor varianza entre las iteraciones y aun así obtienen valores más altos de F1. Adicionalmente, se observa que los modelos de CNN tienen un menor nivel de error que los demás modelos. Finalmente, se procede con la selección final de los 13 mejores modelos, de acuerdo con el nivel del F1 obtenido en la validación cruzada. Así mismo, se incluye la evaluación de todos los modelos que poseen un F1 mayor o igual a 0.90, debido a la convergencia observada y la cercanía con los resultados de la línea base.

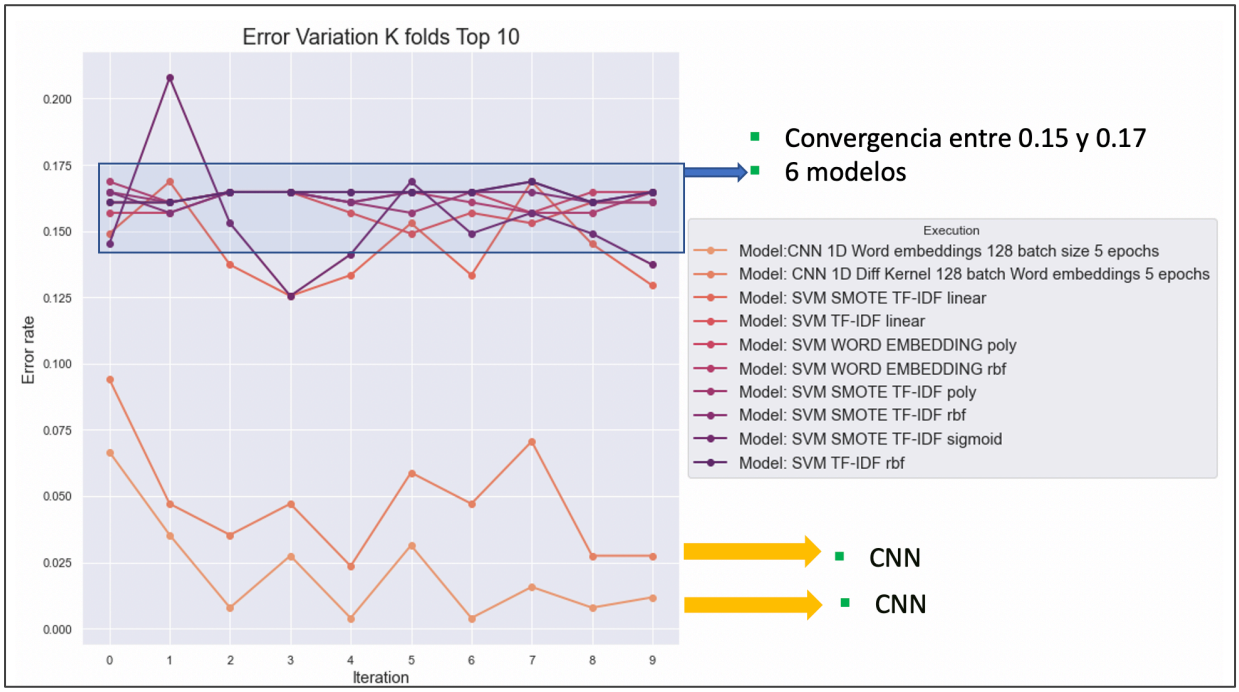
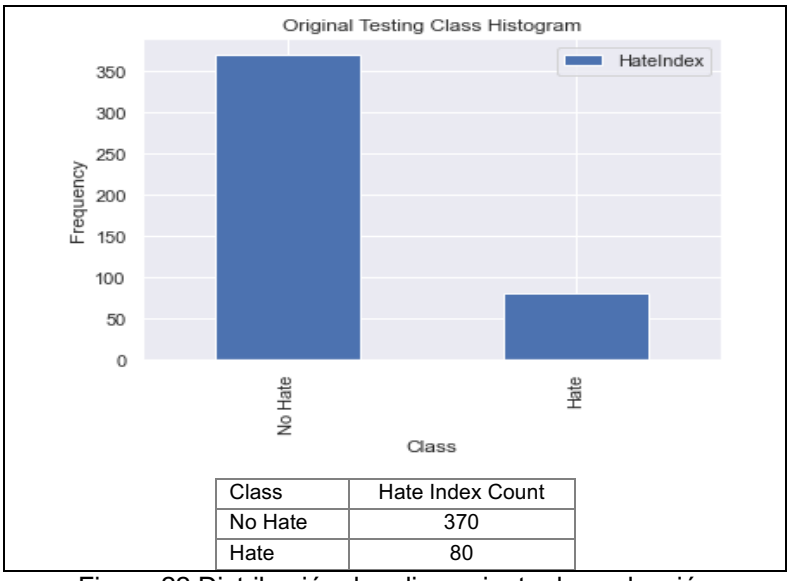


Figura 21 Nivel de error en las diez primeras posiciones de la validación cruzada
Fuente: Propia

6.4.3 Resultados de evaluación final

En referencia con los resultados de la validación cruzada, y utilizando el criterio de filtrado, se ejecutaron las predicciones de los modelos de interés. Por consiguiente, se utilizaron el conjunto de desarrollo para el entrenamiento y el conjunto de evaluación para medir la efectividad de la predicción. El conjunto de evaluación no ha sido utilizado para la validación y ni calibración de los modelos en las dos evaluaciones anteriores. El conjunto de evaluación (*testing*) tiene una distribución de 370 textos sin odio y 80 con odio, la Figura 22 muestra la distribución de las clases.



Class	Hate Index Count
No Hate	370
Hate	80

Figura 22 Distribución de odio conjunto de evaluación
Fuente: Propia

Los resultados obtenidos en esta última evaluación muestran que la línea base fue el modelo que obtuvo el mayor nivel para la medida F1. Adicionalmente, la línea base con el método de SMOTE obtuvo el segundo mejor nivel para la medida del F1. En general, los modelos con SVM lograron mejor desempeño en la precisión, *recall* y medida del F1, a diferencia de los resultados de las redes convolucionales. Estas últimas disminuyeron su posición privilegiada en la primera y segunda posición de los resultados de la validación cruzada hasta las últimas posiciones del resultado de las predicciones. Su resultado disminuyó un 27% para la CNN-1 y un 33% para la CNN-2 con tamaño de núcleos diferente. Así también, los modelos de SVM con mejores desempeños consisten en los basados en la frecuencia de términos TF-IDF, a diferencia de los que utilizaron los *word embeddings*. No obstante, tienen un rendimiento cercano similar, debido que la diferencia es mínima de 1%. En la Figura 23 se muestra en detalle el comportamiento explicado, incluyendo los mejores y peores resultados obtenidos en la predicción y en comparación al desempeño observado en el ranking de la validación cruzada.

Prediction Ranking	Cross Validation Ranking	Execution	Ngrams TF-IDF	Word embed dings	SMOTE	Precision	Recall	Mean Absolute Error	F1 Score
1	4	Model: Predict **SVM** TF-IDF linear	√			0.839	0.997	0.160	**0.911**
2	3	Model: Predict **SVM** SMOTE TF-IDF linear	√		√	**0.881**	0.941	0.153	0.910
3	6	Model: Predict **SVM** SMOTE TF-IDF rbf	√		√	0.830	1.000	0.169	0.907
12	1	Model: Predict **CNN** 1D Word embeddings 128 *batch* size 5 *epochs*		√		0.895	0.576	0.404	0.701
13	2	Model: Predict **CNN** 1D Diff Kernel 128 *batch* Word embeddings 5 *epochs*		√		0.910	0.495	0.456	0.641

- Mejor desempeño
- Frecuencia de términos TF-IDF
- Rendimiento menor
- Ultimas posiciones

Figura 23 Resultados comparativos de la predicción
Fuente: Propia

Así mismo en la Tabla 14, se detallan todos los resultados obtenidos para cada uno de los modelos evaluados en esta tercera y final evaluación. Se aprecia en la Tabla 14, la línea base resaltada en verde y en amarillo los resultados para los modelos CNN. Además, se observa el modelo SVM SMOTE TF-IDF linear obtuvo el mejor nivel de precisión con un 88%. Se observa que la detección es más precisa para detección los comentarios con odio con este modelo. Además, este modelo detecta la mayor cantidad de comentarios de odio, en comparación a los demás, a pesar de tener un F1 score 0.91. Este fenómeno ocurre de forma similar con el modelo SVM SMOTE TF-IDF sigmoid, el cual posee una precisión de 0.88 y un *recall* de 0.89. La detección de odio es mucho mayor para este modelo también, sin embargo, posee mayor cantidad de falsos positivos y falsos negativos, afectado su efectividad. La matriz de confusión para este modelo se observa en la Figura 24.

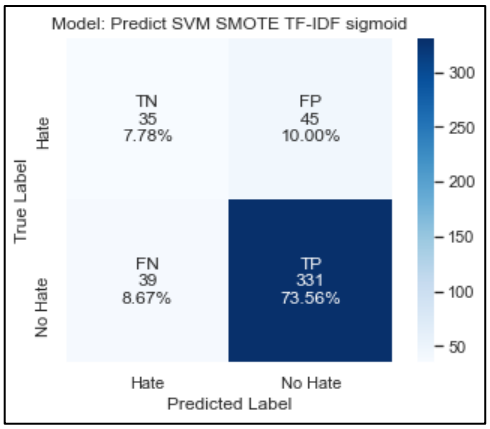


Figura 24 SVM SMOTE TF-IDF sigmoid
Fuente: Propia

Tabla 14 Resultados de predicciones

Ranking Prediction	Ranking Cross Validation	Execution	Accuracy	Precision	Recall	Mean Absolute Error	F1 Score
1	4	Model: Predict SVM TF-IDF linear	0.840	0.839	0.997	0.160	**0.911**
2	3	Model: Predict SVM SMOTE TF-IDF linear	0.847	**0.881**	0.941	0.153	0.910
3	6	Model: Predict SVM SMOTE TF-IDF rbf	0.831	0.830	1.000	0.169	0.907
4	10	Model: Predict SVM TF-IDF rbf	0.827	0.826	1.000	0.173	0.905
5	7	Model: Predict SVM SMOTE TF-IDF poly	0.827	0.830	0.992	0.173	0.904
6	8	Model: Predict SVM WORD EMBEDDING rbf	0.824	0.824	1.000	0.176	0.904
7	11	Model: Predict SVM WORD EMBEDDING linear	0.822	0.822	1.000	0.178	0.902
8	12	Model: Predict SVM TF-IDF poly	0.822	0.822	1.000	0.178	0.902
9	5	Model: Predict SVM WORD EMBEDDING poly	0.822	0.825	0.995	0.178	0.902
10	13	Model: Predict SVM TF-IDF sigmoid	0.816	0.828	0.978	0.184	0.897
11	9	Model: Predict SVM SMOTE TF-IDF sigmoid	0.813	**0.880**	0.895	0.187	0.887
12	1	Model: Predict CNN 1D Word embeddings 128 *batch* size 5 *epochs*	0.596	0.895	0.576	0.404	0.701
13	2	Model: Predict CNN 1D Diff Kernel 128 *batch* Word embeddings 5 *epochs*	0.544	0.910	0.495	0.456	0.641

Fuente: Propia

En la Figura 25 se muestran las matrices de confusión para las mejores 4 predicciones de los modelos evaluados. Así también, se observa en la matriz de confusión del modelo SVM SMOTE TF-IDF linear, que este tiene mayor precisión en la detección de textos con odio, logrando identificar 33 comentarios con odio que representa un 7.33% del total de los textos anotados con odio.

Matrices de confusión de las mejores 4 predicciones

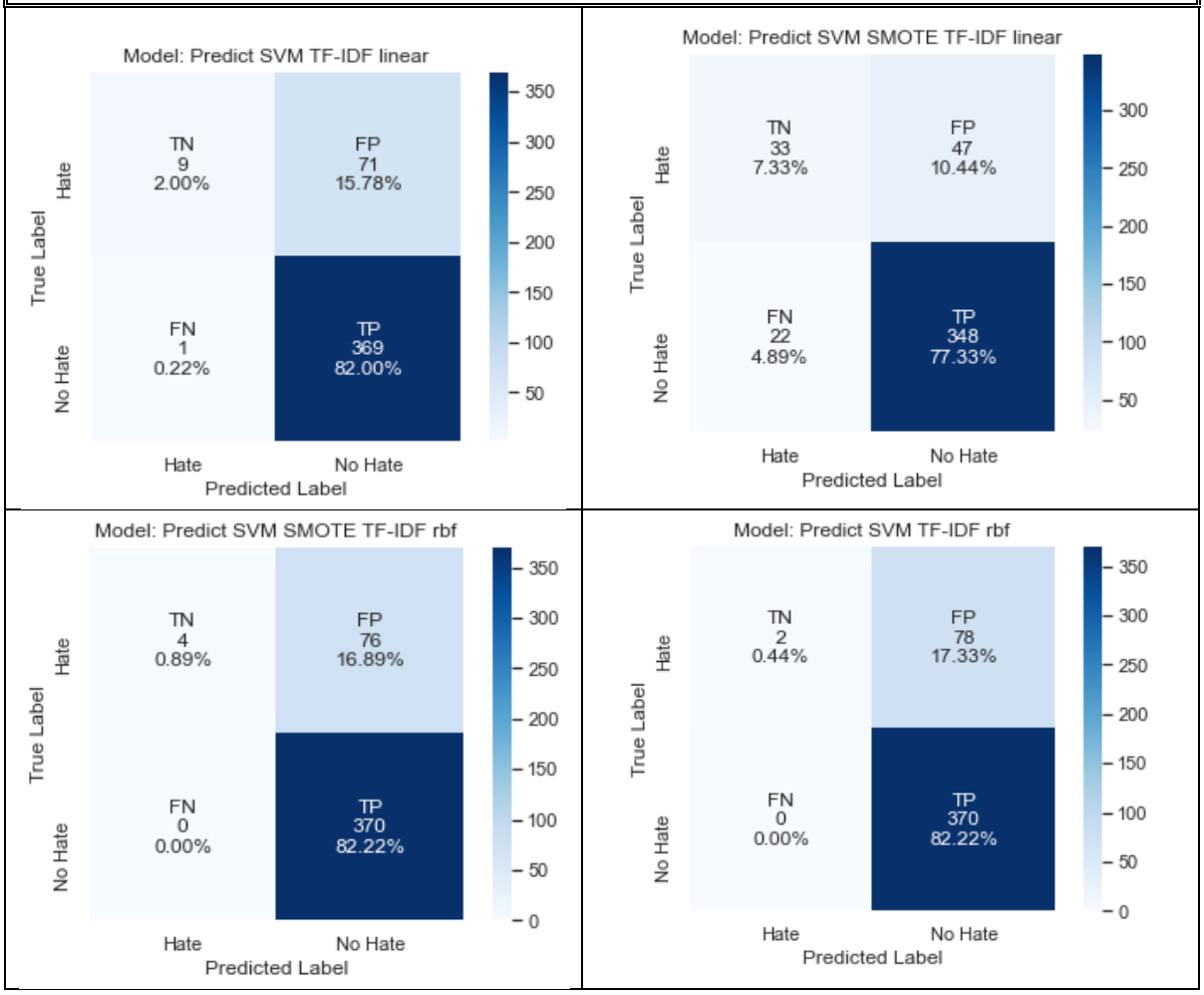


Figura 25 Matrices de confusión de las mejores 4 predicciones
Fuente: Propia

En referencia a los textos identificados correctamente con odio del modelo SVM TF-IDF linear, estos se listan en Tabla 15. Esta tabla incluye el texto preprocesador para la ejecución del modelo de clasificación, la etiquete de *MENTION* corresponde a cuentas de otros usuarios de Twitter. Estos textos evidencian el lenguaje de odio utilizado hacia otras cuentas de usuarios a los cuales hacen referencia. Así mismo, estos 9 textos con odio fueron identificados en el modelo SVM

SMOTE TF-IDF linear, sin embargo, este modelo logró identificar un total de 33 textos con odio correctamente. La lista completa de estos textos clasificados se encuentra en el Anexo 4.

Tabla 15 Resultado de predicción con modelo SVM TF-IDF linear

SVM TF-IDF linear			
Núm.	**Anotación**	**Texto**	**Resultados de Predicción**
1.	Odio	MENTION cada vez estas más pndjo pinche mariguano	Odio
2.	Odio	anda calmandote vientito xq me vas a volar el techo hdp	Odio
3.	Odio	MENTION MENTION MENTION busca otro lugar donde llamar la atención gorda desempleada	Odio
4.	Odio	MENTION otra puta rata callejera pero encima mas puta que la otra puta rata callejera es que eres bien reputa	Odio
5.	Odio	MENTION agarralo a las piñas por puto buchón marica	Odio
6.	Odio	MENTION te distes cuenta que es un hijo e puta comemierda el papa ese	Odio
7.	Odio	MENTION yo sospecho de los que callan ante los atropellos y arbitrariedades de un demente y que además justifican a capa y espada lo que antes acusaban con pasión a mi no me engañas pinche corrupta tu y el peje me pelan la riata	Odio
8.	Odio	MENTION pues de broma también te mandaron ya a chingar a tu puta madre de la ibero pero no te preocupes es broma	Odio
9.	Odio	MENTION MENTION MENTION MENTION pues un ladrillo evidentemente es mucho mas útil que un marica como usted	Odio

Fuente: Propia

6.4.4 Comparación con estudios anteriores para el idioma inglés

Finalmente, el desempeño observado en los modelos evaluados es cercano al desempeño identificado en los estudios anteriores en inglés de la Tabla 7 *Resultados obtenidos en otros estudios en inglés.* En los cuales, para los modelos de SVM la precisión y medidas de F1 se encuentra entre [0.91,0.93]. Las predicciones para los modelos SVM evaluados brindaron medidas de precisión entre [0.89,0.91]. Por lo tanto, un rendimiento menor al visto en los estudios en inglés para los modelos SVM. Respecto a las CNN, la métrica de referencia es F1 de 89.6, mientras que el resultado obtenido en las predicciones para esta métrica en este estudio oscila entre [0.64,0.70], siendo este un desempeño menor. Los mejores resultados se obtuvieron para los modelos de SVM que están más cercanos a los resultados obtenidos en la literatura de referencia. La Figura 26 muestra la comparación de los resultados con investigaciones realizadas en inglés.

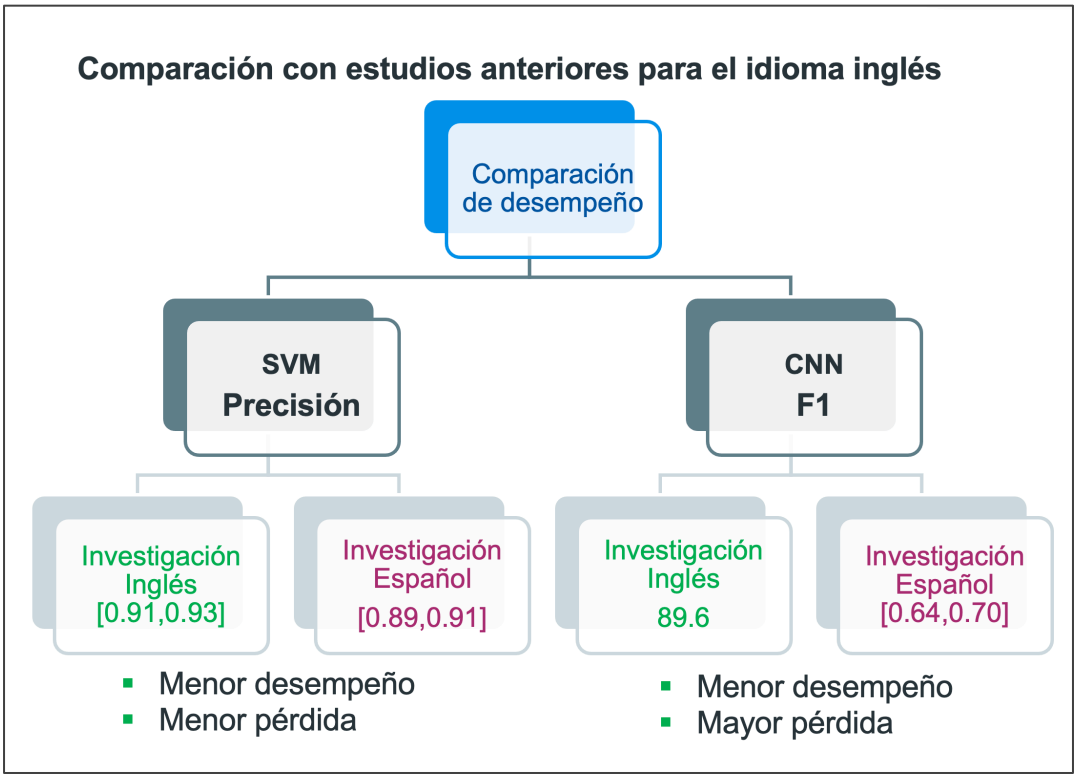


Figura 26 Resultados de referencia
Fuente: Propia

6.5 Script de ejecución de los modelos de clasificación

El código utilizado en para esta investigación se encuentra en https://git.ucr.ac.cr/NOELIA.NAVARROMURILLO/hatespanishanalyzer. El código incluye la lógica para la extracción de *tweets*, creación de muestras para la anotación manual, procesamiento de los resultados de la anotación, configuración de modelos y almacenaje de resultados de las tres evaluaciones incluidas en la investigación. El código está en Python 3.6.8. Se utilizó *Sypder* de *Anaconda* para la construcción de los scripts. La estructura de carpetas del código se muestra en la Figura 27.

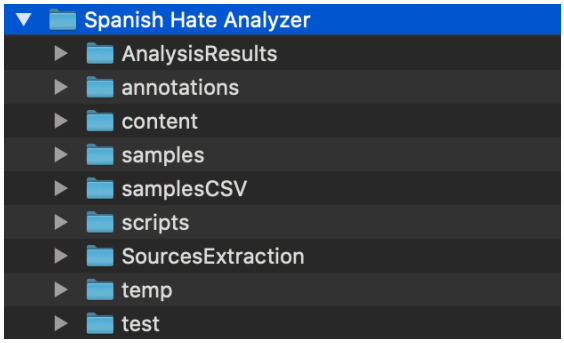


Figura 27 Estructura de carpetas del código
Fuente: Propia

Las carpetas contienen el siguiente contenido:

- AnalysisResults: Posee los archivos detallaos de la ejecución del código, incluyendo matrices de confusión, histogramas, gráficas de análisis y archivos de Excel con los *data frames* almacenados en estos.
- annotations: Incluye los archivos de muestra creados para anotación manual con las anotaciones realizadas.
- Content: folder donde se almacenan los *word embeddings* en español de referencia, por ejemplo el de Google.
- Samples: carpeta en donde se generan las muestras aleatorias y almacenan en formato xls.
- samplesCSV: carpeta en donde se generan las muestras aleatorias y son almacenadas en formato csv.
- Scripts: Lugar donde se encuentran los scripts de ejecución. Los scripts del programa son:
    - extractWordsDictionary.py : extractor de palabras del diccionario en pdf para ser utilizada como palabras de búsqueda en el extractor de *tweets* .
    - Samples.py: generador de muestras aleatorias para crear los archivos de anotación manual.
    - *tweets*.py: extractor de *tweets* en español de *Tweeter*.
    - TweetsAnalysis.py: Ejecutor de los modelos de SVM y CNN, incluye la configuración de las características, preprocesamiento del texto, configuración de los modelos, validación cruzada y evaluación final de los modelos.
- SourceExraction: contiene las listas de palabras clave a utilizar para realizar la extracción de *tweets*.
- Temp: Modelos de lenguaje de *FastText* para la validación de palabras en español.
- Test: carpeta para realizar pruebas sobre archivos anotados manualmente.

Finalmente el programa cuenta con un archivo de readme.txt y de Environment Packages.txt, este último detallas las versiones de los paquetes instalados para hacer la ejecución de los modelos. La versión del código corresponde a la 1.0. El código fue desarrollado a partir de Agosto del 2020.

7 Conclusiones

En la investigación realizada se logró identificar la efectividad de los modelos de SVM es superior que los modelos CNN evaluados para la identificación de odio en textos. La línea base de los modelos SVM obtuvo mejor rendimiento que los modelos utilizando sobremuestro y *word embeddings* para sus características. En comparación con los resultados obtenidos en la revisión de literatura, los resultados obtenidos por esta investigación poseen un rendimiento similar. Esto debido que se logró obtener el desempeño de F1 entre [0.89,0.91] en referencia a los mejores resultados de la literatura que se sitúan entre [0.91,0.93] para los modelos SVM. Al menos se obtuvo un rendimiento en común 0.91 para la medida de F1. Las redes neuronales mostraron buenos rendimientos en las validaciones y en el *training set*, sin embargo, al ejecutar la tercera evaluación de predicción su desempeño se vio afectado y se obtuvo el menor de los rendimientos en comparación a todos los modelos utilizados en la tercera evaluación para las medidas de F1 y *recall*.

En referencia a los resultados obtenidos se logró cumplir con los objetivos propuestos y dar respuesta a la pregunta de investigación, que pretendía identificar la efectividad que se obtiene al identificar contenido de odio en un texto en español mediante el uso de clasificadores automáticos. Por supuesto, limitado al alcance de la investigación. El trabajo realizado consistió en la elaboración del *corpus* de los textos a utilizar para la identificación de odio hasta la configuración y evaluación de los modelos de SVM y CNN recomendados por la literatura revisada. Se identificó que los modelos SVM poseen mejor desempeño que los CNN dado el corpus creado de 3000 muestras. Así mismo, el sobre muestreo beneficia la identificación de mayor cantidad de textos con odio para el SVM linear, sin embargo, la métrica de F1 es menor que no implementarlo. No obstante, si se quiere un escenario más optimista se recomienda usar el SVM linear con SMOTE, en caso de ser más conservadores se recomienda utilizar SVM sin SMOTE.

Además, como parte de la investigación se generó un corpus anotado con comentarios de odio que puede ser utilizado en investigaciones futuras sobre identificación de odio. La creación del *corpus* de textos en español requirió de la anotación manual. A fin de establecer un proceso estándar, se diseñó y utilizó el instrumento *Instrucciones de anotación de tweets para la identificación de odio.* Esta herramienta sirvió para estandarizar la identificación de odio en un texto y evitar la inherencia de los valores de los anotadores a la hora de identificar el odio. Sin embargo, en los resultados de anotación se obtuvieron textos anotados con valor neutral. Esto

como el resultado de la sumatoria de la anotación de tres personas sobre un mismo texto dio como resultado 0, anulando de esta manera la identificación del odio. Para efectos de esta investigación, se clasificaron los neutros dentro de la categoría sin odio. En conclusión, a pesar de tomar las medidas para evitar el juicio basado en los valores de los anotadores, se observa que es necesario incrementar los ejemplos de odio en las instrucciones. Así también, se observó que la anotación de odio está sesgada a la percepción de las personas y de acuerdo con su contexto puede afectar la efectividad de la anotación basada en reglas. En este trabajo se limitó el equipo de anotación a personas con un grado de bachillerato o mayor a fin de contar con un equipo anotador objetivo que entendiera el ejercicio y siguiera las reglas de anotación. Finalmente, el *corpus* final anotado con las etiquetas de odio o sin odio, puede ser utilizado para investigaciones futuras sobre la identificación de odio en español. Así mismo, la herramienta de extracción de *tweets* se puede utilizar para obtener *tweets* de acuerdo con una lista de términos personalizable.

Respecto a la definición de las características se utilizaron los *word embeddings* y las frecuencias de términos. La más efectiva de acuerdo con los resultados fueron las frecuencias de términos, los *word embeddings* son más beneficiosos en la implementación de los modelos de CNN. Así mismo, se recomienda utilizar *word embeddings* pre entrenados al lugar de crear los *word embeddings* desde cero. Esta investigación utilizó *word embeddings* pre entrenados, que fueron ajustados a los términos del vocabulario de los textos de los *tweets*. De esta manera, hay mayor transferencia de conocimiento, al incluir en los *embeddings* vocabulario asociado a palabras que no pertenecen únicamente al vocabulario de los textos. Así mismo, en la composición semántica de las frases con odio se componen de monosílabos que son importantes para la identificación de odio. Por lo tanto, remover los *stopwords* del preprocesamiento, afecta la identificación de frases con odio, por esta razón no se realizó la exclusión de los *stopwords* del vocabulario de los textos.

Respecto al desempeño de las redes CNN, el enfoque de *mini gradient descent* con mínimas iteraciones, obtiene resultados positivos en la validación cruzada, sin embargo, en la evaluación final con el conjunto de datos que no se había sido utilizado, los resultados fueron bastante bajos en comparación a los modelos de CNN. De esta forma, resultó interesante verificar que la red con *batch* de 128, obtiene mejores resultados en comparación a los *batches* de 32. Así mismo, esta investigación corresponde a un problema de clasificación binaria, al convertir las categorías de odio y no odio en 1, 0. Por lo tanto, la entropía utilizada en las CNNs, fue *binary cross entropy.*

Si se quiere analizar las CNNs con la *categorical cross entropy* las entradas a la red deben ser modificados para predecir sobre 2 vectores binarios. Los resultados obtenidos en esta investigación solo contemplaron la predicción sobre un vector binario.

Este trabajo colabora con la investigación de identificación de odio en textos en español. Aportando un *corpus* anotado para futuros estudios, además de proveer una referencia del desempeño de los modelos SVM (*Support Vector Machine*) con características de frecuencia de términos, *word embeddings* y el efecto de utilizar sobremuestro en el entrenamiento. Además, de la referencia del desempeño de las CNN con *word embeddings* utilizando *batches* de 32 y 128 y 5 *epochs*, definiendo una referencia para futuras ajustes de las configuraciones de las redes CNN.

Finalmente, esta investigación provee una herramienta inicial que identifica odio en textos en español, proporcionando ayuda a la investigación de la identificación de odio en contenido en español. El principal resultado de la investigación corresponde a los fundamentos de la herramienta de detección de odio automática y científicamente probada con una precisión de 0.84 para textos en español. La implementación de este tipo de herramienta en el análisis de comentarios o en micro textos en las plataformas de interacción, fortalecería las políticas antidiscriminatorias y anti-odio. Esto debido a identificación automática de odio en los comentarios. Lo cual debería iniciar las medidas de eliminación o y prohibición de la divulgación de los textos cuando estos son clasificados con odio. En definitiva, la identificación automática de odio en español necesita de estas investigaciones para potenciar herramientas de detección en tiempo real, que protejan a los usuarios de discursos de odio y de discriminación. Así mismo, de la definición de reglas de identificación de odio que eviten el sesgo de los valores y creencias que poseen los anotadores humanos sobre el discurso de odio. Por último, la investigación del odio en español tiene mucho camino por correr y esta investigación aporta una referencia específica realizada sobre textos en español de Costa Rica y la región. En el proceso de esta investigación se lograron identificar importantes hallazgos para futuras investigaciones dirigidas a la identificación automática de odio en español de Costa Rica para la protección de los usuarios del discurso de odio en línea.

8 Trabajo Futuro

A partir de esta investigación, se recomienda continuar con el análisis de la identificación de odio en español. Utilizando el *corpus* para la identificación de frases peyorativas que agreguen un mayor criterio en la clasificación automática de odio. Sería importante explorar la construcción de *word embeddings* con estas frases idiomáticas y frases peyorativas, para obtener *word embeddings* personalizados y verificar si este enfoque beneficia la identificación de odio. Así mismo, otra opción es continuar con la misma línea de la investigación y enfocarse en el ajuste de los *hyper* parámetros de las redes convolucionales, hasta identificar la configuración con mejores resultados de predicción y en donde su desempeño sea constante. En un ajuste futuro del trabajo realizado sobre las CNNs, sería ideal verificar si con mayor cantidad de *epochs* y *batches* de 128 se obtienen mejores resultados. Así también, sería indispensable construir una API para la detección de odio, utilizando los hallazgos y herramientas de esta investigación, para colocarla a la disposición de la comunidad científica de investigación de procesamiento de lenguaje natural y desarrolladores web que busquen implementar herramientas de prevención de odio.

9 Bibliografía

1. Almatarneh, S., Gamallo, P., Pena, F. J. R., & Alexeev, A. (2019). Supervised Classifiers to Identify Hate Speech on English and Spanish Tweets. En A. Jatowt, A. Maeda, & S. Y. Syn (Eds.), *Digital Libraries at the Crossroads of Digital Information for the Future* (pp. 23-30). Springer International Publishing. https://doi.org/10.1007/978-3-030-34058-2_3
2. Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep Learning for Hate Speech Detection in Tweets. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, 759-760. https://doi.org/10.1145/3041021.3054223
3. Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. *arXiv:1206.5533 [cs]*. http://arxiv.org/abs/1206.5533
4. Benito, A. V., Fraser, K., & Miró, E. C. (2020). Lenguaje peyorativo en español: Términos despreciativos y términos neutros usados como insultos. *Teorema: Revista internacional de filosofía*, *39*(2), 63-85.
5. Cardellino, C. (2019). *Spanish Billion Words Corpus and Embeddings*. https://crscardellino.github.io/SBWCE/
6. Carmona, M. A., Guzmán-Falcón, E., Montes, M., Escalante, H. J., Villaseñor-Pineda, L., Reyes-Meza, V., & Rico-Sulayes, A. (2018, agosto 1). *Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets*.
7. Corazza, M., Menini, S., Cabrio, E., Tonelli, S., & Villata, S. (2020). A Multilingual Evaluation for Online Hate Speech Detection. *ACM Transactions on Internet Technology*, *20*(2), 10:1-10:22. https://doi.org/10.1145/3377323
8. Dai, W., Yang, Q., Xue, G.-R., & Yu, Y. (2007). Boosting for transfer learning. *Proceedings of the 24th International Conference on Machine Learning - ICML '07*, 193-200. https://doi.org/10.1145/1273496.1273521
9. Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). *Automated Hate Speech Detection and the Problem of Offensive Language*. https://arxiv.org/abs/1703.04009v1
10. Desai, A., Gulwani, S., Hingorani, V., Jain, N., Karkare, A., Marron, M., R, S., & Roy, S. (2016). Program Synthesis Using Natural Language. *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, 345-356. https://doi.org/10.1145/2884781.2884786
11. *Digital 2020: Costa Rica*. (s. f.). DataReportal – Global Digital Insights. Recuperado 26 de abril de 2020, de https://datareportal.com/reports/digital-2020-costa-rica

12. Dorris, W., Hu, R. (Roger), Vishwamitra, N., Luo, F., & Costello, M. (2020). Towards Automatic Detection and Explanation of Hate Speech and Offensive Language. *Proceedings of the Sixth International Workshop on Security and Privacy Analytics*, 23-29. https://doi.org/10.1145/3375708.3380312
13. Durán, M. A. P. (2015). Estudio de los campos semánticos que sirven en la construcción de la unidad fraseológica del tipo peyorativo. *Forma y función*, *28*(1), 157-182.
14. Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861-874. https://doi.org/10.1016/j.patrec.2005.10.010
15. Fortuna, P., & Nunes, S. (2018). A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys*, *51*(4), 1-30. https://doi.org/10.1145/3232676
16. Greevy, E., & Smeaton, A. F. (2004). Classifying racist texts using a support vector machine. *Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval - SIGIR '04*, 468. https://doi.org/10.1145/1008992.1009074
17. Hernández, M. B., & Gómez, J. M. (2013). Aplicaciones de Procesamiento de Lenguaje Natural. *Revista Politécnica*, *32*. https://revistapolitecnica.epn.edu.ec/ojs2/index.php/revista_politecnica2/article/view/32
18. Herwanto, G. B., Maulida Ningtyas, A., Nugraha, K. E., & Nyoman Prayana Trisna, I. (2019). Hate Speech and Abusive Language Classification using fastText. *2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 69-72. https://doi.org/10.1109/ISRITI48646.2019.9034560
19. Houlsby, N., Giurgiu, A., Jastrzebski, S., & Morrone, B. (2019). *Parameter-Efficient Transfer Learning for NLP*. 13.
20. imbalanced-learn developers. (2021). *imbalanced-learn documentation—Version 0.8.0*. imbalanced-learn documentation. https://imbalanced-learn.org
21. Jiang, L., & Suzuki, Y. (2019). Detecting hate speech from tweets for sentiment analysis. *2019 6th International Conference on Systems and Informatics (ICSAI)*, 671-676. https://doi.org/10.1109/ICSAI48974.2019.9010578
22. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., & Mikolov, T. (2016). FastText.zip: Compressing text classification models. *arXiv:1612.03651 [cs]*. http://arxiv.org/abs/1612.03651
23. Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of Tricks for Efficient Text Classification. *arXiv:1607.01759 [cs]*. http://arxiv.org/abs/1607.01759

24. Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *arXiv:1408.5882 [cs]*. http://arxiv.org/abs/1408.5882
25. Kingma, D. P., & Ba, J. (2017). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*. http://arxiv.org/abs/1412.6980
26. Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2021). 1D convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, *151*, 107398. https://doi.org/10.1016/j.ymssp.2020.107398
27. *Language identification · fastText*. (s. f.). Https://Fasttext.Cc/. Recuperado 26 de abril de 2021, de https://fasttext.cc/index.html
28. Liu, S., & Forss, T. (2014). Combining N-gram based Similarity Analysis with Sentiment Analysis in Web Content Classification: *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*, 530-537. https://doi.org/10.5220/0005170305300537
29. Mark Omernick, Francois Chollet, T. K. (2019, noviembre 6). *Keras documentation: Text classification from scratch*. Keras. https://keras.io/examples/nlp/text_classification_from_scratch/
30. Masters, D., & Luschi, C. (2018). Revisiting Small Batch Training for Deep Neural Networks. *arXiv:1804.07612 [cs, stat]*. http://arxiv.org/abs/1804.07612
31. Modha, S., Mandl, T., Majumder, P., & Patel, D. (2020). Tracking Hate in Social Media: Evaluation, Challenges and Approaches. *SN Computer Science*, *1*(2), 105. https://doi.org/10.1007/s42979-020-0082-0
32. Navarro-Murillo, N., Calvo-Vargas, P., & Casasola-Murillo, E. (2019). Identification of Unsuitable Content for Children in Video Gaming Forums. *2019 IV Jornadas Costarricenses de Investigación en Computación e Informática (JoCICI)*, 1-6. https://doi.org/10.1109/JoCICI48395.2019.9105201
33. Oksanen, A., Hawdon, J., Holkeri, E., Näsi, M., & Räsänen, P. (2014). Exposure to Online Hate among Young Social Media Users. En *Sociological Studies of Children and Youth* (Vol. 18, pp. 253-273). https://doi.org/10.1108/S1537-466120140000018021
34. Peng, Y., Yan, S., & Lu, Z. (2019). Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets. *arXiv:1906.05474 [cs]*. http://arxiv.org/abs/1906.05474
35. Plaza-Del-Arco, F.-M., Molina-González, M. D., Ureña-López, L. A., & Martín-Valdivia, M. T. (2020). Detecting Misogyny and Xenophobia in Spanish Tweets Using Language

Technologies. *ACM Transactions on Internet Technology*, *20*(2), 12:1-12:19. https://doi.org/10.1145/3369869

36. Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., Shyu, M.-L., Chen, S.-C., & Iyengar, S. S. (2018). A Survey on Deep Learning: Algorithms, Techniques, and Applications. *ACM Computing Surveys*, *51*(5), 92:1-92:36. https://doi.org/10.1145/3234150
37. Pujari, A. K., Mittal, A., Padhi, A., Jain, A., Jadon, M., & Kumar, V. (2019). Debiasing Gender biased Hindi Words with Word-embedding. *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*, 450-456. https://doi.org/10.1145/3377713.3377792
38. Ríos González, G. (2017). *Léxico juvenil costarricense* (1 ed). Imprenta Nacional. https://www.imprentanacional.go.cr/editorialdigital/libros/literatura%20costarricense/lexico_juvenil_costarricense_edincr.pdf
39. Ruwandika, N. D. T., & Weerasinghe, A. R. (2018). Identification of Hate Speech in Social Media. *2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer)*, 273-278. https://doi.org/10.1109/ICTER.2018.8615517
40. Salminen, J., Hopf, M., Chowdhury, S. A., Jung, S., Almerekhi, H., & Jansen, B. J. (2020). Developing an online hate classifier for multiple social media platforms. *Human-centric Computing and Information Sciences*, *10*(1), 1. https://doi.org/10.1186/s13673-019-0205-6
41. scikit-learn developers. (2021). *3.1. Cross-validation: Evaluating estimator performance—Scikit-learn 0.24.2 documentation*. scikit-learn. https://scikit-learn.org/stable/modules/cross_validation.html
42. Sintoris, K., & Vergidis, K. (2017). Extracting Business Process Models Using Natural Language Processing (NLP) Techniques. *2017 IEEE 19th Conference on Business Informatics (CBI)*, *01*, 135-139. https://doi.org/10.1109/CBI.2017.41
43. *sklearn.feature_extraction.text.TfidfVectorizer—Scikit-learn 0.24.2 documentation*. (s. f.). Recuperado 2 de mayo de 2021, de https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
44. *sklearn.metrics.f1_score—Scikit-learn 0.24.2 documentation*. (s. f.). Recuperado 10 de mayo de 2021, de https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score

45. Sohn, H., & Lee, H. (2019). MC-BERT4HATE: Hate Speech Detection using Multi-channel BERT for Different Languages and Translations. *2019 International Conference on Data Mining Workshops (ICDMW)*, 551-559. https://doi.org/10.1109/ICDMW.2019.00084
46. Solís-Fonseca, C. (2020). *Efecto del balanceo de clases al evaluar el f-score para un clasificador de texto en análisis de sentimiento* [Trabajo Final de Investigación Aplicada sometido a la consideración de la Comisión del Programa de Estudios de Posgrado en Computación e Informática para optar al grado y título de Maestría Profesional en Computación e Informática]. Universidad de Costa Rica.
47. Team, K. (s. f.). *Keras documentation: Embedding layer*. Recuperado 3 de mayo de 2021, de https://keras.io/api/layers/core_layers/embedding/
48. Teh, P. L., Cheng, C.-B., & Chee, W. M. (2018). Identifying and Categorising Profane Words in Hate Speech. *Proceedings of the 2nd International Conference on Compute and Data Analysis*, 65-69. https://doi.org/10.1145/3193077.3193078
49. *Tweepy Documentation—Tweepy 3.10.0 documentation*. (s. f.). Recuperado 26 de abril de 2021, de https://docs.tweepy.org/en/latest/index.html
50. Twitter. (2020). *Search Tweets: Standard v1.1*. Search Tweets: Standard v1.1. https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/overview
51. Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, *3*(1), 9. https://doi.org/10.1186/s40537-016-0043-6
52. Yuan, S., Wu, X., & Xiang, Y. (2016). *A Two Phase Deep Learning Model for Identifying Discrimination from Tweets* [Data set]. OpenProceedings.org. https://doi.org/10.5441/002/EDBT.2016.92

10 Anexos

10.1 Anexo 1 Lista de extracción

Núm.	Palabra -frase	Núm.	Palabra -frase	Núm.	Palabra -frase	Núm.	Palabra -frase
0	cerda	101	mensa	201	@c_avendanocr	301	arroz con mango
1	perra	102	tarada	202	@Maroo_Lean	302	asiado
2	perro	103	zoquete	203	@Cruickshankedu	303	atacado
3	pájaro	104	bueno	204	@silhersa	304	atacada
4	gata	105	babosa	205	@wagnerjimenez1	305	aterro
5	animal	106	bruto	206	@FloriaSagot	306	aturusar
6	bestia	107	cabrón	207	@HarllanPaez	307	no nacer ayer
7	buey	108	culera	208	@NielsenPerez	308	azotar
8	zorra	109	desgraciado	209	Víctor Manuel Morales Mora	309	azote
9	rata	110	estúpido	210	paolavegar	310	babosada
10	burro	111	hocicón	211	@PedroMunoz_cr	311	bajar
11	madre	112	huevón	212	@Diputada2018	312	bajonazo
12	mariguano	113	idiota	213	@josemvillalta	313	bajoneado
13	mayate	114	imbécil	214	@otto_2022	314	bajoneada
14	meco	115	inservible	215	@mariajocc14	315	baldazo
15	piruja	116	maldito	216	@DanielUlate21	316	banano
16	joto	117	malparido	217	@nidia_cespedes	317	bañazo
17	zángano	118	pelotudo	218	@CaroHHe	318	barra
18	golfa	119	pendejo	219	@lcarranzacc	319	barrabasada
19	güey	120	pinche	220	@MasisErwen	320	barridita
20	marica	121	Follar	221	@diputado_erick	321	basureador
21	maricón	122	defecar	222	Dragos Dolanescu Valenciano	322	basureadra
22	prostituta	123	cagar	223	Mario Eduardo Castillo Méndez	323	basurear
23	cadáver	124	coger	224	@PaoValladaresRo	324	batazo
24	feto	125	chingar	225	@dipchaconmonge	325	bate
25	bato	126	joder	226	Pablo Heriberto Abarca Mora	326	bateada
26	hombre	127	picar	227	Catalina Montero Gómez	327	bateadera
27	ramera	128	mamar	228	Aracelly Salas Eduarte	328	bateador
28	bolas	129	cerdo	229	@jonprendas	329	bateadra
29	cabeza	130	mecos	230	@AidaMontielCR	330	batear
30	agasajo	131	zángana	231	Mileyde Alvarado Arias	331	biblia
31	carajo	132	meón	232	@rodolfo_pusc	332	bicha
32	chingaderas paja	133	pitos	233	@PastorMelvincr	333	bichilla
33	puñetas cojones	134	penes	234	Giovanni Alberto Gómez Obando	334	bichillo
34	chocha	135	reata	235	@GourzongDip	335	bichillla
35	concha	136	tonta	236	@figuerescr	336	bichito
36	coño	137	puta	237	@oariascr	337	bicho
37	culo	138	reputa	238	@PizaRodolfo	338	biccha
38	cola	139	reputo	239	@alvarez_desanti	339	bici
39	diablo	140	marrana	240	@Nuria_MarinR	340	bicicletear
40	hocico	141	mendiga	241	@JDiegoCastroCR	341	bicla
41	huevos	142	arrabalero	242	@ClaudiaDobles	342	billullo
42	ovarios	143	putrefacto	243	@marioredondo	343	birra
43	mión	144	floja	244	@FabriAlvarado7	344	blanco
44	pito	145	gorda	245	@AlejandraMoraM	345	blíster
45	pitón	146	desmadrosa	246	abierta	346	bola

Núm.	Palabra -frase	Núm.	Palabra - frase	Núm.	Palabra -frase	Núm.	Palabra -frase
46	punta	147	fastidiosa	247	abrirse	347	Se usa en plural. dar Obs.
47	escroto	148	mamona	248	achantado	348	boleta
48	fundillo	149	mantenida	249	achantada	349	seas boli
49	panocha	150	menso	250	achante	350	bollo
50	pene	151	tarado	251	sentir el ácido	351	ser bolsa
51	pepa	152	baboso	252	acois	352	bombeta
52	picha	153	bruta	253	aculrse	353	ponerle bonito
53	pichón	154	cabrona	254	afilado	354	bostezo
54	polla	155	culero	255	afilada	355	botar
55	pucha	156	desgraciada	256	afilar	356	qué braza
56	riata	157	estúpida	257	agarrado	357	brete
57	verga	158	maldita	258	agarrada	358	breteada
58	mocos	159	malparida	259	agarrarse	359	breteadera
59	nalga	160	pendeja	260	agarronazo	360	breteador
60	polainas	161	#PACNuncamas	261	aguantar	361	breteadra
61	Asqueroso	162	#PAC	262	aguante	362	bretear
62	ignorante	163	#blackface	263	agüevada	363	bro
63	inepto	164	#TuCaraMeSuena	264	agüevado	364	brocha
64	infame	165	#AsambleaLegislativaCR	265	agüevar	365	bronca
65	infeliz	166	#DespiertaCostaRica	266	agüevazón	366	buchón
66	inútil	167	#CarebarroCR	267	ahora	367	buchona
67	maje	168	#EsPorVos	268	ahorcarse	368	cabrearse
68	tonto	169	INCOPESCA	269	ahorita	369	cabro
69	lelo	170	#PacCorrupto	270	ahuevado	370	cabbra
70	mamalón	171	@zoilarosavolio	271	ahuevada	371	pedir cacao
71	obeso	172	@CRBenavidesJ	272	dar aire	372	cachar
72	pandroso	173	@VitaMonge	273	alborotado	373	1. a cachete
73	puto	174	@DiputadoErick	274	alborotada	374	como cachiflín
74	marrano	175	Diputado	275	alborotazón	375	cacho
75	mendigo	176	Diputada	276	alcancía	376	caemal
76	mongol	177	@DiputadaA	277	alegrón de burro	377	caer
77	promiscua	178	@karinenino	278	aleta	378	cagado
78	puñal	179	@AcunaCabrera	279	aletazo	379	cagada
79	ojete	180	@ChanDiputada	280	seas mi amor	380	caja de leche
80	semental	181	@FranggiDiputada	281	andar	381	salírsele el caldo
81	arrabalera	182	@Iguido	282	antro	382	camarón
82	negro	183	@GustavoVialesCR	283	apearse	383	camellar
83	gay	184	@WelmerRamos	284	apretar	384	camello

Núm.	Palabra -frase	Núm.	Palabra -frase	Núm.	Palabra -frase	Núm.	Palabra -frase
84	homosexual	185	@OttoGuevaraG	285	aprete	385	camellla
85	sucia	186	@OrtizFabrega	286	apretecillo	386	caminar
86	torpe	187	@CarlosHernndezA	287	apuntado	387	camote
87	zonza	188	@misolisq	288	apuntada	388	campanear
88	flojo	189	@Michael Soto_CR	289	¡arajo@	389	agarrar cancha
89	gordo	190	@marcovargasd	290	argolla	390	canchudo
90	inmunda	191	Bernardo Alfaro Banco Nacional	291	argollero	391	canchuda
91	cachonda	192	Elián Villegas	292	argollera	392	candado
92	barata	193	#CostaRica	293	aro	393	mojársele la canoa
93	chingón	194	@epsycampbell	294	arrastrada	394	cantar
94	cobarde	195	@asambleacr	295	arratada	395	cañazo
95	desmadroso	196	#hartosdelpac	296	arratado	396	carajada
96	fastidioso	197	#Covidiotas	297	arrecho	397	carajeada
97	gacha	198	@Laura_Ch	298	arreccha	398	carajear
99	hipócrita	199	@jlfonsecaf	299	arreglar	399	carajillo
99	mamón	200	@aluciadelgado	300	arrepellarse	400	carajillla
100	mantenido						
401	caraja	501	chupada	601	encagonarse	701	a la hora de los balazos
402	¡carajo!	502	chupado	602	enchanchado	702	hornazo
403	meter carbón	503	chupagüevos	603	enchanchada	703	horneado
404	carbonear	504	chupamedias	604	enchancharse	704	horneada
405	carbonero	505	chupar	605	enchilado	705	pegar hueco
406	carbonera	506	chupas	606	enchilada	706	jalame el hule
407	carebanano	507	chupetazo	607	enganchado	707	inventar
408	carebarro	508	chupo	608	enganchada	708	ir a darle
409	carechapi	509	churuco	609	enganchar	709	patear con la izquierda
410	careculo	510	chuzo	610	engarrotado	710	jacha
411	carepez	511	cito	611	engarrotada	711	jalar
412	carepicha	512	cita	612	engarrotarse	712	jaleas
413	carepinga	513	clavar	613	enjachar	713	jalón
414	carepistola	514	claveadera	614	enjache	714	jama
415	caretanda	515	clavear	615	ensartar	715	jartar
416	caretorta	516	dejar con el clavo	616	entrador	716	jeta
417	carga	517	clean	617	entradra	717	jetear
418	carne molida	518	cleta	618	entucada	718	jetón
419	carracazo	519	cletear	619	entucar	719	jetona
420	carraco	520	cletero	620	envenenado	720	jetonada
421	cartucho	521	cletera	621	envenenada	721	jodido
422	cáscara	522	codo	622	enviajarse	722	jodida

Núm.	Palabra -frase	Núm.	Palabra -frase	Núm.	Palabra -frase	Núm.	Palabra -frase
423	cascarudo	523	cogida	623	enyucado	723	jueputa
424	cascaruda	524	cogido	624	enyucada	724	jugar de intrépido
425	llegar tarde a la repartición de cerebro	525	colchón	625	enyucar	725	juma
426	cero	526	colocho	626	del otro equipo	726	jumarse
427	cerote	527	color	627	escocherar	727	jumas
428	chamaco	528	colorazo	628	espantarse	728	jupa
429	chamaca	529	comecaca	629	espeso	729	jupón
430	chance	530	comemierda	630	espesa	730	jupona
431	chancecito	531	compa	631	sacarla del estadio	731	tenérsela jurada a alguien
432	chanchada	532	cómper	632	estrilar	732	lactar
433	chanchos	533	compita	633	estrilón	733	laja
434	chancleta	534	conchada	634	estrilona	734	lamegüevos
435	chancletudo	535	concho	635	me extrañar	735	lance
436	chancletuda	536	conccha	636	faroleado	736	lancecillo
437	chanear	537	conocher	637	faroleada	737	lanzado
438	chante	538	cool	638	hacer feo	738	lanzada
439	chantoso	539	cortar	639	filo	739	lata
440	chantosa	540	cosi	640	finguerear	740	lavagüevos
441	chapa	541	con cosito -a. Generalmente dicho por mujeres. 2.	641	póngale la firma	741	con toda la leche
442	chaperón	542	ni costra	642	firmarme aquí	742	legal
443	chaperona	543	cagar cuadrado	643	fuck	743	lento
444	chapi	544	cuadrar	644	follamigo	744	hecho leña
445	charraleado	545	cuentear	645	follamiga	745	leñateada
446	charraleada	546	cuento	646	follaring	746	leñatear
447	charralearse	547	cuerazo	647	forro	747	leñazo
448	charralero	548	cuerero	648	fresa	748	leva
449	charralera	549	cuerera	649	fresco	749	la ley
450	chata	550	cuero	650	fresquillo	750	ligar
451	chatear	551	culazo	651	funcar	751	limpio
452	chato	552	culeada	652	furris	752	limppia
453	chavalo	553	culeadera	653	gacho	753	linda
454	chavala	554	culeado	654	gaccha	754	llorar
455	chela	555	culeador	655	gajo	755	loca
456	chema	556	culeadra	656	galleta	756	locazo
457	chepear	557	culear	657	juntársele el ganado	757	loco
458	chepito	558	culeolada	658	ganar	758	lora
459	chepita	559	culeolo	659	ganja	759	loser
460	chic	560	métase el dedo	660	garrote	760	andar en la luna
461	chicha	561	descocherar	661	gaysazo	761	madrazo
462	chicharrón	562	desestrés	662	comer gente	762	¡qué madre!
463	chichero	563	desestresante	663	meter un gol	763	madreada
464	chichera	564	desestresar	664	goma	764	madrear

Núm.	Palabra -frase	Núm.	Palabra - frase	Núm.	Palabra -frase	Núm.	Palabra -frase
465	¡tome chichi!	565	despanocharse	665	gorrear	765	mae
466	chichoso	566	despapayarse	666	gorro	766	mafufo
467	chichosa	567	despechugado	667	gozar	767	mafufa
468	chiliguili	568	despechugada	668	grifo	768	pegar una majada
469	chillarse	569	despechugar	669	grone	769	andar de malas
470	chimar	570	despelote	670	guaba	770	mamada
471	chinamo	571	despichado	671	guabero	771	mamado
472	chingue	572	despichada	672	guabera	772	mamador
473	chiva	573	despichar	673	guachear	773	mamadra
474	chivearse	574	despiche	674	guachi	774	mamapichas
475	chivo	575	despichingarse	675	guachimán	775	mami
476	¡cho!	576	despijiante	676	guacho	776	man
477	chobi	577	di	677	guamazo	777	no manches
478	chochera	578	diay	678	guasapiar	778	mandar
479	chochosca	579	¿puros diezes?	679	guaso	779	meterle mano
480	chopo	580	la doce	680	guasa	780	ser mantequilla
481	choricear	581	doña	681	guataso	781	mapri
482	choricero	582	dopar	682	guau	782	maría
483	choricera	583	dope	683	güeiso	783	maricas
484	chorizo	584	dormir conmigo	684	güelepedos	784	maricona
485	chorpa	585	drogo	685	güevazo	785	maripepino
486	chota	586	embarcada	686	güeviar	786	matadero
487	chotear	587	embarcador	687	a güevo	787	matizado
488	choteo	588	embarcadra	688	güevón	788	matizada
489	choza	589	embarcar	689	güevona	789	matizar
490	chucear	590	dar una embolia	690	güevonada	790	mecha
491	chuchinga	591	embollada	691	güila	791	mejenga
492	chueco	592	embollar	692	guineo	792	mejenguear
493	chueca	593	embollado	693	hablada	793	hacerse una melcocha
494	chulear	594	embolsarse	694	hachazo	794	meme
495	chulo	595	emborrachada	695	harina	795	meneíto
496	chula	596	emigrar	696	hembrilla	796	mensajear
497	chunche	597	empalidar	697	mi herma	797	mero
498	chunchereco	598	empanada	698	heteroflexible	798	mera
499	chuncherequear	599	enano	699	hijo de papi	799	metiche
500	chupaculo	600	enaona	700	hijueputa	800	miau
801	meterse una mica	901	perica	1001	rocolo	1101	vara
802	mico	902	perico	1002	rocola	1102	varilla
803	¿quién dijo miedo?	903	perol	1003	rojo	1103	venirse
804	mier	904	perreo	1004	rola	1104	verde
805	comer mierda	905	perrera	1005	roncale la gana	1105	hasta la verga
806	seas mío	906	calmado pescado	1006	levantar roncha	1106	vergazo

Núm.	Palabra -frase	Núm.	Palabra - frase	Núm.	Palabra -frase	Núm.	Palabra -frase
807	mita	907	picado	1007	rulear	1107	vergueada
808	moncha	908	picada	1008	saico	1108	verguear
809	monchar	909	picarle	1009	saica	1109	viajado
810	mongolito	910	picarita	1010	salida	1110	viajada
811	mongolita	911	pichacear	1011	salir rascando	1111	darle viaje
812	montado	912	pichazo	1012	salvada	1112	cantar viajera
813	montada	913	nan	1013	sapazo	1113	buscar vida
814	montarse	914	pichudo	1014	sapaza	1114	viejo
815	monte	915	pichuda	1015	sapear	1115	vieja
816	mop	916	volar pico	1016	sapo	1116	violar
817	mopri	917	piedrero	1017	sasa	1117	violento
818	morada	918	pijiado	1018	mojar la sardina	1118	violenta
819	morderse	919	pijiada	1019	sembrar	1119	traerla viva
820	mordido	920	pijiarse	1020	sep	1120	póngase vivo
821	mordida	921	pinga	1021	serruchapisos	1121	volado
822	mortal	922	pinta	1022	sip	1122	volar
823	mostacilla	923	pipa	1023	sipe	1123	ve vos
824	mota	924	pipazo	1024	sacarle el sirope	1124	vuelta
825	mozote	925	pipi	1025	sobar	1125	wachear
826	muerdealmohadas	926	pisar	1026	sobo	1126	watsapear
827	musgo	927	mover el piso	1027	sobrada	1127	weed
828	naaa	928	pisón	1028	sofis	1128	weiso
829	naco	929	pisona	1029	váyase por la sombra	1129	yeguada
830	naca	930	hecho pistola	1030	sombreado	1130	yeyo
831	nada que ver	931	volar pisucho	1031	sombreada	1131	yodo
832	nave	932	cagar plata	1032	sonador	1132	yolo
833	ninis	933	platanazo	1033	sonadra	1133	zacate
834	¡qué nivel!	934	plátano	1034	sonar	1134	zaguate
835	nombres	935	platudo	1035	soplado	1135	zarpe
836	nop	936	platuda	1036	soplada	1136	zocar
837	nope	937	playada	1037	soplanucas		
838	buena nota	938	playo	1038	soplapichas		
839	bajarse de la nube	939	plumas	1039	sorompo		
840	darle por la nuca	940	plumero	1040	suave		
841	nudo de mortadela	941	polada	1041	subir		
842	tener el ombligo	942	pollo	1042	tabo		
843	aplanchar la oreja	943	polo	1043	tachar		
844	pa	944	pola	1044	taco		
845	paca	945	polvo	1045	tacuen		
846	pachanga	946	tr. ponerle	1046	tamarindo		
847	pacho	947	popi de carne	1047	tanda		
848	pachuco	948	porte	1048	tandeada		
849	paco	949	hecho un poste	1049	tandeado		
850	paisa	950	presa	1050	tandear		
851	hablar paja	951	pretilear	1051	tapis		
852	pajariolo	952	primo	1052	taratúpido		
853	pajariola	953	pro	1053	taratúpida		
854	pajoso	954	prometer	1054	tarro		
855	pajosa	955	pucho	1055	tata		
856	irse de pálido	956	puesto	1056	comérselo la tecnología		

Núm.	Palabra -frase	Núm.	Palabra - frase	Núm.	Palabra -frase	Núm.	Palabra -frase
857	palmada	957	pul	1057	teja		
858	palmarse	958	pulpe	1058	templado		
859	palmatón	959	purete	1059	templada		
860	detrás del palo	960	putear	1060	templarse		
861	paloma	961	quebrar	1061	templazón		
862	paniquear	962	quemado	1062	tenis		
863	panocho	963	quemada	1063	ser una teta		
864	panza	964	quemón	1064	darle tetica		
865	panzona	965	queque	1065	tierrosa		
866	hacer el papel	966	quihubo	1066	tigra		
867	papelón	967	quiubas	1067	pal tigre		
868	papi	968	rábano	1068	timba		
869	papillo	969	majar el rabo	1069	tipo		
870	papudo	970	rajado	1070	tipa		
871	papuda	971	rajar	1071	tirar		
872	paque	972	rajón	1072	tirars		
873	paqueteado	973	rajona	1073	tiro		
874	paqueteada	974	ramalazo	1074	tita		
875	manda la parada	975	rancharse	1075	tololas		
876	parado	976	rancho	1076	tomatinga		
877	parársele	977	randon	1077	tombo		
878	pargo	978	rasta	1078	¡tome!		
879	parga	979	ratear	1079	hacer tonto		
880	pata	980	rayado	1080	toque		
881	patas	981	regada	1081	torre		
882	pato	982	regarla	1082	torta		
883	tenga paz	983	reinita	1083	tortillera		
884	pega	984	rejo	1084	tostado		
885	pegado	985	volar rejo	1085	tostada		
886	pegada	986	reloj	1086	tramar		
887	pegar	987	relojazo	1087	tranqui		
888	pegue	988	repellar	1088	trolear		
889	pegona	989	reventado	1089	trollear		
890	pelada	990	reventada	1090	tuanis		
891	pelarle	991	reventar	1091	tuca		
892	peli	992	revolcada	1092	tuco		
893	pellizcado	993	revolcarse	1093	tur		
894	pellizcada	994	revolcón	1094	turra		
895	pellizcarse	995	revolquea dera	1095	ser pura tusa		
896	la pelona	996	rico	1096	vacas		
897	verla peluda	997	rica	1097	vacilada		
898	pepiado	998	dar ride	1098	vacilar		
899	pepiada	999	roco	1099	vacilón		
900	perderse	1000	roca	1100	vaquera		

Fuente: Propia

10.2 Anexo 2 Subconjunto de 3000 muestras para anotación

Conjunto de 3000 muestras para el *dataset* de anotación				
grone	pachanga	rancho	chata	quemón
malparida	ir a darle	carajillo	me extrañar	paisa
aro	perderse	cabeza	enganchar	cañazo
hablada	cartucho	arrastrada	agarrarse	Diputada
camello	chupamedias	caer	despiche	tarro
seas mi amor	puto	chueco	@marioredondo	nombres
barra	bato	colocho	camote	hecho pistola
ve vos	reventado	¡qué nivel!	panzona	paco
cuero	agarrada	marrano	meme	hacer el papel
polo	rajar	@silhersa	rayado	baldazo
guau	zángano	goma	grifo	clean
rajón	filo	perra	parga	maricona
emigrar	tostado	huevos	lento	carajo
harina	gorda	no manches	¡cho!	güeviar
@asambleacr	violento	antro	tipo	chichero
chamaca	¡tome!	prostituta	tombo	matadero
ser una teta	arroz con mango	tur	rica	embolsarse
mordida	bruta	cleta	pucha	@ClaudiaDobles
subir	chicha	chopo	banano	mayate
bichito	barrabasada	ahora	mop	mami
golfa	guacho	cantar	tuca	bicicletear
perreo	morada	candado	reloj	camellar
maje	tipa	apretar	bate	Diputado
picado	estúpida	quemada	chupado	monte
naca	desestresante	carne molida	naaa	mendiga
guachi	joder	mota	aterro	jetona
desestrés	comer mierda	meón	paloma	pajoso
venirse	bicho	verguear	embarcar	vacilada
verga	sip	meterle mano	viejo	@karinenino
culero	mamador	dar aire	papudo	violenta
floja	mortal	rajado	pendeja	follamiga
botar	caja de leche	bollo	pachuco	montado
pitos	bajoneado	tuanis	panza	atacada
gorro	sobo	picada	chema	sombreado
guasa	coger	chavala	jodida	babosada
roco	cerote	mandar	cómper	enganchada
revolcada	papuda	gorrear	agasajo	rata
hablar paja	promiscua	infame	cabrona	@CaroHHe
menso	reventar	viajado	nave	infeliz
montarse	compa	mamar	pegar	mamada
batazo	mae	chapi	feto	chingar
pipazo	picarle	pega	lelo	fundillo
hocico	bostezo	mamado	chatear	vieja
compita	chupada	maricas	chaperón	jupa

Conjunto de 3000 muestras para el *dataset* de anotación				
bateador	bronca	vergazo	madre	jodido
linda	laja	aguantar	putrefacto	pito
gozar	cagada	jeta	torta	azote
weed	caraja	morderse	hachazo	sapo
loca	buey	hijo de papi	mantenida	nop
arrecho	bajar	mier	leva	jama
hecho leña	güevón	forro	cacho	color
di	sucia	inventar	@rodolfo_pusc	concha
brocha	Follar	homosexual	animal	reinita
reputa	biblia	mendigo	chocha	afilado
mero	cerda	la doce	Asqueroso	tachar
desgraciada	pinga	maldita	chic	puta
hacer feo	nope	pata	tortillera	pa
madrazo	la pelona	carga	apuntado	fresco
dormir conmigo	cortar	espeso	gay	alcancía
chivo	abierta	suave	tonto	mión
comer gente	culear	lance	semental	chuzo
#DespiertaCostaRica	jetón	lata	palmada	vacilar
dope	boleta	darle por la nuca	plátano	aleta
naco	culeado	paca	caminar	pacho
yeyo	matizado	defecar	engarrotada	afilar
meter un gol	comemierda	chicharrón	burro	piruja
maricón	chorizo	buena nota	parado	chupar
mariguano	porte	cuadrar	cita	vara
volado	pellizcado	carbonero	arrabalera	bolas
gata	madreada	concho	cosi	guaso
@misolisq	trolear	sofis	tostada	quebrar
templado	tarado	culo	doña	codo
blíster	picha	chiva	basurear	pegado
putear	roca	jartar	choza	rejo
peli	chance	toque	güey	pene
cagado	azotar	zángana	@josemvillalta	mico
despelote	pájaro	chula	nalga	miau
cáscara	ser bolsa	chanchos	chupo	revolcarse
bola	salvada	agarrado	cabrearse	aguante
bajonazo	ignorante	timba	hombre	guineo
arreglar	tuco	buchona	del otro equipo	pato
tanda	cito	mongol	@zoilarosavolio	enganchado
ahorita	sonar	polainas	carepicha	garrote
papi	obeso	chingue	pichón	regada
mocos	seas mío	picar	espesa	pargo
ovarios	revolcón	salida	escroto	rola
#Covidiotas	bajoneada	trollear	plumero	pegada
tranqui	fuck	zorra	primo	cagar plata
tonta	vaquera	enchilada	torre	drogo
afilada	pelarle	paolavegar	cachar	meco

Conjunto de 3000 muestras para el *dataset* de anotación				
pitón	teja	@FabriAlvarado7	quemado	tamarindo
apuntada	vuelta	chante	cerdo	chela
chulo	ninis	hacer tonto	purete	inútil
tramar	gacho	pepa	tirar	reventada
ramera	loco	prometer	pucho	cuento
baboso	reata	fastidiosa	ahuevado	llorar
rojo	violar	bestia	tata	cogido
pinta	pelada	@CRBenavidesJ	perrera	chapa
póngale la firma	diay	soplado	sobrada	metiche
chamaco	cabro	chato	cagar	montada
chancleta	regarla	perol	mopri	plumas
mamalón	gajo	pisar	maría	mamona
mozote	joto	pro	guachimán	sasa
marica	varilla	yolo	cool	sapear
paque	clavar	ahorcarse	riata	marrana
lora	argolla	loser	abrirse	tiro
pegue	cletear	pipi	vacilón	limpio
horneado	tenga paz	vacas	envenenado	bateada
ligar	papelón	pipa	volar	alborotada
rico	mamapichas	pollo	bici	aprete
man	galleta	tabo	jalar	@FranggiDiputada
colchón	enano	musgo	carbonera	cletero
pola	taco	tenis	polvo	culeada
yodo	chavalo	¿quién dijo miedo?	sep	#CostaRica
fresa	jalón	alborotado	patas	mordido
coño	batear	jueputa	cola	brete
penes	buchón	chota	tita	lanzada
turra	con toda la leche	mecos	chueca	güevona
sembrar	hasta la verga	queque	nada que ver	verde
envenenada	bro	rábano	pisón	chupas
@VitaMonge	ojete	birra	rasta	carebarro
camarón	diablo	punta	chulear	andar
mera	hijueputa	mecha	ratear	bicha
buscar vida	bruto	panocha	cero	sobar
ponerle bonito	puñal	madrear	atacado	inepto
culazo	sipe	matizar	@jonprendas	polla

Fuente: Propia

10.3 Anexo 3 Instrucciones de anotación de tweets para la identificación de odio

Instrucciones de anotación de *tweets* para la identificación de odio

1. Objetivo del ejercicio

El objetivo es identificar el discurso de odio en comentarios extraídos de Twitter en español. El discurso de odio comprende el lenguaje que ataca, minimiza, e incita a la violencia o al odio. En concreto, contra un individuo o grupos de individuos y en función de características específicas. Así, por ejemplo: la apariencia física, la religión, la ascendencia, el origen nacional o étnico, la orientación sexual, la identidad de género, discapacidad, orientación política, clase o comportamiento. El odio se identifica cuando las personas usan uno o varios términos despreciativos para referirse a un determinado grupo de personas o individuo de acuerdo con su forma de sentir y pensar con el fin de denigrar, herir, desvalorizar o atacar a los sujetos.

2. Instrucciones de la anotación (revisión de *tweets*)

Se le ha brindado un archivo de Excel con una lista de *tweets* que deben ser revisados por su persona. Para cada *tweet* se debe identificar si el texto posee o no discurso de odio. En donde, se identifique el sujeto o grupo afectado por el comentario. Deberá elegir una opción de la escala: *Totalmente en desacuerdo, En desacuerdo, De Acuerdo, o Totalmente de Acuerdo,* si el comentario posee odio. Siendo ***Totalmente de Acuerdo*** la calificación **más alta** de odio y ***Totalmente en desacuerdo*** la calificación **más baja** en donde el odio es inexistente.

Entre más sea el odio presente en el *tweet* más de acuerdo deberá marcar la escala.

3. Escala de calificación

Deberá colocar una "x" en la casilla que corresponda para cada *tweet*.

Totalmente en desacuerdo	En desacuerdo	De Acuerdo	Totalmente de Acuerdo
			x

No hay odio — Más odio

Por ejemplo en este caso la "x" indica que el comentario sí posee odio.

Ejemplos de Tweets para la categoría de Totalmente de acuerdo:

"A esta vieja es mejor no verla! No la soporto"

"Otro viejo Parrilli, tan ridículo el ansiano este...."

Así mismo, debe tomar las siguientes consideraciones para identificar el odio y la calificación:

- Totalmente de Acuerdo = Se identifica odio en el texto. Se identifica la persona, entidad o grupo ofendido por la expresión de odio.
- De Acuerdo = El texto muestra odio, sin embargo no identifica la persona, entidad o grupo al cual es dirigido.
- En desacuerdo = El texto no muestra odio, pero está sujeto a interpretaciones y podría contener odio.
- Totalmente en desacuerdo = Esta seguro de que el texto no muestra ningún tipo de odio hacia una persona, entidad o grupo.

Todo Tweet debe ser revisado. Por favor seleccione en la escala según su criterio, cual categoría representa más la presencia de odio en el Tweet. El **archivo de Excel** contiene 3 páginas que deben ser revisadas **Order1,Order2 y Order3**

¡Muchas gracias por su participación!

Fuente: Propia

10.4 Anexo 4 Micro textos clasificados por modelo SVM TF-IDF Linear con y sin SMOTE

Núm.	Anotación	Texto	Resultado de Predicción	Modelo SVM TF-IDF linear	
				Sin SMOTE	Con SMOTE
1.	Odio	MENTION cada vez estas más pndjo pinche mariguano	Odio	√	√
2.	Odio	anda calmandote vientito xq me vas a volar el techo hdp	Odio	√	√
3.	Odio	MENTION MENTION MENTION busca otro lugar donde llamar la atención gorda desempleada	Odio	√	√
4.	Odio	MENTION otra puta rata callejera pero encima mas puta que la otra puta rata callejera es que eres bien reputa	Odio	√	√
5.	Odio	MENTION agarralo a las piñas por puto buchón marica	Odio	√	√
6.	Odio	MENTION te distes cuenta que es un hijo e puta comemierda el papa ese	Odio	√	√
7.	Odio	MENTION yo sospecho de los que callan ante los atropellos y arbitrariedades de un demente y que además justifican a capa y espada lo que antes acusaban con pasión a mi no me engañas pinche corrupta tu y el peje me pelan la riata	Odio	√	√
8.	Odio	MENTION pues de broma también te mandaron ya a chingar a tu puta madre de la ibero pero no te preocupes es broma	Odio	√	√
9.	Odio	MENTION MENTION MENTION MENTION pues un ladrillo evidentemente es mucho mas útil que un marica como usted	Odio	√	√
10.	Odio	ha sido un año difícil dijo pooh y todavía no pateamos al ex de hyojung respondió pigglet su perra madre exclamó pooh ohmygirl	Odio		√
11.	Odio	MENTION te duele desde octubre no vago ignorante vividor hijo de papi con plata chorro de guante blanco blanco villegas	Odio		√
12.	Odio	recuerdo que antes de venir a vivir a suecia recibi comentarios como negro te vas a cagar de hambre a donde vas a ir vos que apenas hablas español te vas a quedar sin plata y en tres meses vas a pegar la vuelta nunca escuchen a esos imbecilxs	Odio		√
13.	Odio	ni ti milisti qui si hibli cn sis ixs no te molesta ser tan pelotuda y arrastrada lástima que el ya lo sabía chusma 2	Odio		√
14.	Odio	MENTION tipico de macho escroto abusador tóxico	Odio		√
15.	Odio	MENTION MENTION con esta comparación demuestras que eres una mierda de persona con razón trabajas para el reforma no tienes moral ética o inteligencia para poner a cada una en su justo lugar cuando truene reforma tendrás que jubilarte obeso idiota nadie te contratara	Odio		√
16.	Odio	MENTION tienen tantas patas ganas de chulear guau increíble este pendejo	Odio		√

Núm.	Anotación	Texto	Resultado de Predicción	Modelo SVM TF-IDF linear	
				Sin SMOTE	Con SMOTE
17.	Odio	MENTION que buen periodicazo te dieron en el puro hocico aparte de ciego eres p3ndejo encharcamiento no es lo mismo que inundación sarna es lo que tienes perro sarniento ah y trata de hablar como hombrecito y sácate la papa o es camote lo que te metes al hocico	Odio		√
18.	Odio	MENTION MENTION MENTION MENTION claro el pendejito hijo de papi se dedica a destruir los autos y lo premian dándole las mejoras a él solito para que se luzca	Odio		√
19.	Odio	MENTION MENTION MENTION pues serás de los poquitosestos ya no representan a los trabajadores ni de coña si no a okupas ninis indepes etarras progresetc	Odio		√
20.	Odio	el fascismo es la única ideología política viable sólo un grupo selecto de personas preparadas y entrenadas para ello debería acceder al poder político no un cerote sin personalidad y sin puta idea d cómo se gobierna como ese MENTION culero por mucho q les caiga bien	Odio		√
21.	Odio	MENTION maduro digale al mongolico trump q c ocupe d la tumba d su sucia madre q esos huesos estan llenos d telaraña q saque la cochinada esa d la caja dond la metieron y meta esos resto x la poceta x dond ella debe ir como los mojones	Odio		√
22.	Odio	MENTION MENTION MENTION ya has perdido con ese argumento de mierda de las fiestas que eres mas tonto como se nota tu que no has asistido a fiestas por que pareceque soloo has ido a escuchar a esa basura y a coger de manera muy facil total musica q habla solo de follar x dios	Odio		√
23.	Odio	MENTION MENTION MENTION que vergüenza que ud sea caleña que bruta entonces es culpa de petro las masacres e injusticias del gobierno duque la gente sale a marchar porq está mamada del sistema no porq petro los obligue o instigue lea un poco más señora	Odio		√
24.	Odio	hoy vi a un maje que no lo soporto y aun tengo unas ganas de meterle unos sus buenos cuentazos en esa su cara de jorobado de notre dame	Odio		√
25.	Odio	veterino t da asco meterle la mano por el culo a una vaca aria	Odio		√
26.	Odio	MENTION fake este viejo güevón sigue con sus falsos positivos y sus súbditos le siguen comiendo cuento	Odio		√
27.	Odio	yo lo llamaba el niño rata chungo mientras que envy era el niño rata pesado	Odio		√
28.	Odio	estos son los anarquistas del grupo reventado del dictador compartan la imagen para que estén atentos los compañeros de frenaa	Odio		√
29.	Odio	este hijueputa no representa al liguismo	Odio		√
30.	Odio	MENTION la cabecilla del frente farc europa coordinando desde su piso en parís malparida buena para nada	Odio		√
31.	Odio	si el niño no existiera ahí anduvieras de indigente pidiendo dinero pendeja ganas de agarrarte a putazossss	Odio		√

Núm.	Anotación	Texto	Resultado de Predicción	Modelo SVM TF-IDF linear	
				Sin SMOTE	Con SMOTE
32.	Odio	MENTION MENTION MENTION esta mantenida que va saber lo que sufrir cuando todo se lo tienen facil este pais de msolo favorece a los grandes clar la sunat y sunafil no quiere que le hagan competencia a ricos de mx tanta injusticia yo tambien me volveria slque problema hay	Odio		√
33.	Odio	MENTION MENTION por eso idiota pendejos como tu no progresan para que alquilar una puta golfa si tengo una mujer que me ama en casa o es que a ti te hace falta y necesitas comprar lo mas pésimo del mercado	Odio		√