UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

DETECCIÓN DE NOTICIAS FALSAS MEDIANTE LA COMBINACIÓN DE TÉCNICAS DE APRENDIZAJE PROFUNDO Y CARACTERÍSTICAS ESTILOMÉTRICAS

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRA EN CIENCIAS E INGENIERÍA DE LA COMPUTACIÓN

PRESENTA:
CLAUDIA PORTO CAPETILLO

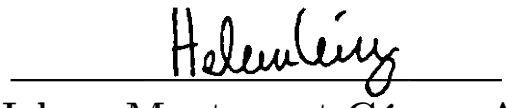
DIRECTORA DE TESIS:
DRA. HELENA MONTSERRAT GÓMEZ ADORNO
INSTITUTO DE INVESTIGACIONES EN MATEMÁTICAS APLICADAS Y EN SISTEMAS

CIUDAD UNIVERSITARIA, CD. MX
ENERO, 2023

Jurado asignado:

Presidente:	Dr. Héctor Benítez Pérez
Vocal:	Dra. Helena Montserrat Gómez Adorno
Secretario:	Dra. Wendy Elizabeth Aguilar Martínez
Suplente:	Dr. Gemma Bel Enguix
Suplente:	Dr. Ignacio Arroyo Fernández

Directora de tesis:



Dra. Helena Montserrat Gómez Adorno

Este trabajo se ha realizado con el apoyo ofrecido por CONACYT mediante los recursos de cómputo brindados a través de la Plataforma de Aprendizaje Profundo de Tecnologías del Lenguaje del Laboratorio de Supercomputación del INAOE. Además, agradezco a CONACYT por el apoyo económico brindado durante la realización de este proyecto (CVU 1084833).

A mis padres con mucho amor y cariño.

Agradecimientos

"Un individuo solo, sólo no puede transcender."

Carlos Salinas de Gortari

A mis padres por ser mi eterna guía, por su confianza, apoyo incondicional, por estar siempre presente y por su dedicación y entrega. Por ser las personas más especiales de mi vida, este trabajo es para ellos.

A CONACYT por el apoyo económico brindado durante la realización de este proyecto

A la Dra. Helena Gómez, por aceptar dirigir el presente trabajo y por guiarme durante el desarrollo de esta tesis, por tanto conocimiento y experiencia brindada, por sus recomendaciones y por toda su ayuda y dedicación.

A todos los profesores quienes de una forma u otra han contribuido con mi formación, en especial a los de la facultad.

A mi pareja por su ayuda y apoyo incondicional en todo momento, por animarme en mis momentos de tristezas y brindarme tantas alegrías. A su familia por todo su cariño y preocupación.

Muchas gracias a todas las personas que de una forma u otra me han ayudado en la realización de este trabajo y a lo largo de esta carrera. A todos, mis más sinceros agradecimientos, muchísimas gracias!

Ciudad Universitaria, CD. MX, 2022

Índice general

Índice de figuras

Índice de tablas

Detección de noticias falsas mediante la combinación de técnicas de aprendizaje profundo y características estilo métricas

Tesis de Maestría

Claudia Porto Capetillo
Posgrado en Ciencia e Ingeniería de la Computación
Universidad Nacional Autónoma de México

Resumen

Las noticias falsas o *fake news* han existido desde los inicios de la humanidad, pero con el surgimiento del internet y los avances tecnológicos en las últimas décadas se ha visto una proliferación de las mismas a lo largo del planeta. En la actualidad el fácil acceso a las redes sociales provoca que los usuarios divulguen información falsa, la cual en muchas ocasiones es compartida miles de veces en cortos plazos de tiempo. Además de la fácil divulgación de noticias falsas en las redes sociales, el uso de las mismas también facilita el consumo por parte de los usuarios de noticias engañosas, falsas o fabricadas, provocando de esta forma la desinformación entre las personas. Estas noticias falsas generalmente tienen la intención de manipular a los usuarios con diferentes propósitos, por ejemplo, pueden ser una propaganda contra un individuo, sociedad, organización o partido político, o entrando en el contexto actual en presencia de la pandemia provocada por el COVID 19 pueden ser noticias respecto a temas relacionados con la enfermedad. En este proyecto de tesis se aborda el problema de la detección de noticias falsas en particular para el lenguaje español y se propone una arquitectura basada en técnicas de aprendizaje profundo combinadas con características estilo-métricas del texto. La arquitectura propuesta es evaluada en el corpus *The Spanish Fake News Corpus* tomando en cuenta la métrica F_1 obteniéndose el valor 0.744.

Palabras claves: Aprendizaje Profundo, Noticias Falsas, Redes Neuronales Convolucionales, Redes Neuronales Recurrentes, Arquitecturas Transformers, *BERT*, Procesamiento del Lenguaje Natural, Características Estilo-Métricas.

Detección de noticias falsas mediante la combinación de técnicas de aprendizaje profundo y características estilo métricas

MSc Thesis

Claudia Porto Capetillo
Posgrado en Ciencia e Ingeniería de la Computación
Universidad Nacional Autónoma de México

Abstract

Fake news has existed since the beginning of humanity, but with the emergence of the internet and technological advances in the last decades have been a proliferation of them throughout the planet. At present, easy access to social networks causes users to spread false information, which on many occasions is shared thousands of times in short periods of time. In addition to the easy dissemination of news false in social networks, the use of the same also facilitate the consumption by users of misleading, false or fabricated news, causing thus misinformation among people. These fake news generally have the intention of manipulating users for different purposes, for example, they can be a propaganda against an individual, society, organization or political party, or entering the current context in the presence of the pandemic caused by COVID 19 they can be news regarding topics related to the disease. In this thesis project, the problem of fake news detection is addressed, particularly for the Spanish language, and an architecture based on deep learning techniques combined with stylistic-metric characteristics of the text is proposed. The proposed architecture is evaluated in the corpus *The Spanish Fake News Corpus* considering the metric F_1 obtaining the value 0.744.

Key words: Deep Learning, Fake News, Convolutional Neural Networks, Long Short Term Memory, Transformers Architectures, *BERT*, Natural Language Processing, Stylo-Metrics Features.

Capítulo 1

Introducción

"Imagination is the discovering faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science".

Ada Lovelace

Las noticias falsas o *fake news* son definidas como artículos de noticias intencionalmente y de manera verificable falsas [Shu et al., 2017]. Esta definición principalmente posee dos características claves. La primera característica está dada por que las noticias falsas incluyen información falsa que se puede verificar como tal y la segunda característica consiste en que estas noticias son creadas con la intención deshonesta de engañar a los consumidores.

Un hecho bien conocido es que debido a los avances tecnológicos cada vez son más las personas con acceso a plataformas digitales. Datos señalan que cerca del 61 % de la población mundial es usuario de internet y aproximadamente el 58 % son usuarios activos de redes sociales [Europa Press, 2021]. Estos porcentajes provienen de varias razones, entre ellas se encuentra la facilidad de interacción y comunicación entre los usuarios debido a que pueden compartir con amigos u otros usuarios sus criterios respecto a alguna noticia, además generalmente es más económico producir y consumir noticias desde plataformas digitales en comparación con medios de comunicación tradicionales, como periódicos o canales televisivos de noticias.

Estas ventajas de las plataformas digitales permiten la divulgación de manera muy rápida entre miles de usuarios de noticias falsas, provocando de esta forma la desinformación entre los usuarios de las mismas. Ejemplo de la proliferación de noticias falsas en redes sociales se evidenció durante el inicio de la pandemia sobre la cual en las redes sociales se difundieron muchas noticias falsas referentes al origen, tratamiento y transmisión del SARS-Cov-2 [van Der Linden et al., 2020]. Ejemplos de estas noticias falsas son que *"Comer ajo previene el contagio"*, *"Bill Gates el creador del coronavirus."*, entre otras.

Una solución para evitar la proliferación de noticias engañosas o falsas en las redes las cuales tienen un gran impacto en la sociedad sería apoyarnos de profesionales como es el caso de los periodistas, para que verifiquen si la veracidad de las noticias en base a hechos publicados en periódicos o sitios de confianza. Esta solución es poco viable debido a que tiende a ser muy lenta y costosa producto de la cantidad de información circulando en las redes.

A raíz de esta problemática en el área del procesamiento del lenguaje natural se han desarrollado múltiples investigaciones encaminadas a la detección automática de noticias falsas, debido a que mediante el uso de la inteligencia artificial somos capaces de reducir el tiempo

y esfuerzo necesario que inviertan los humanos para la clasificación de las noticias, y de esta forma frenar la propagación de las mismas en las plataformas digitales.

Relacionado con la tarea de detección de noticias falsas existen otros problemas estrechamente relacionados a esta tarea como son la Comprobación de hechos *(Fact-Checking)*, esta tarea consiste en evaluar la veracidad de afirmaciones hechas por figuras públicas como políticos, expertos, entre otros [Vlachos and Riedel, 2014]. Otra de las tareas es la Detección de Rumores *(Rumor Detection)*, la cual consiste en identificar declaraciones cuya veracidad no se confirma en plataformas digitales, rápidamente en el momento de ser publicada o un tiempo después. También se encuentra la tarea de Detección de postura *(Stance Detection)* la cual se basa en la extracción de la reacción de un sujeto a una afirmación hecha por un actor principal, esta tarea constituye una parte central de un conjunto de enfoques para la evaluación de noticias falsas. Por último se encuentra Análisis de sentimiento *(Sentiment Analysis)*, la cual consiste en clasificar la polaridad de un texto.

1.1. Motivación

La problemática referente a la rápida propagación de noticias falsas ha propiciado numerosas investigaciones en el área del procesamiento del lenguaje natural enfocadas a la clasificación de noticias, principalmente dirigidas al desarrollo de herramientas computacionales en particular clasificadores de aprendizaje automático que permitan verificar la veracidad de las noticias publicadas en la web de forma automática. A pesar de que este problema se encuentra presente en todos los idiomas, existen muy pocas investigaciones donde se realicen experimentos evaluando arquitecturas de aprendizaje automático o aprendizaje profundo para la detección de noticias falsas en español. Esto debido a que no existen en el estado del arte una gran cantidad de corpus conformados con noticias etiquetadas en español, a pesar de ser este idioma el segundo con más hablantes nativos y el cuarto de los idiomas más hablados Santander Universidades [2021]. A raíz de lo mencionado anteriormente en el presente trabajo de tesis nos planteamos como meta proponer una arquitectura capaz de poder identificar contenido de noticias falsas principalmente en el idioma español, para de esta forma ayudar a minimizar la difusión y consumo de dichas noticias a través de plataformas digitales. En particular las arquitecturas de aprendizaje profundo como son las redes neuronales recurrentes, en particular la arquitectura *Long Short Term Memory (LSTM)*, las redes neuronales convolucionales y arquitecturas basadas en transformers como es el caso de *Bidirectional Encoder Representation from transformers (BERT)* han mostrado muy buenos resultados en el estado del arte al ser aplicadas a corpus de noticias en lenguaje inglés, debido a que permiten extraer características más abstractas de los textos de entrada. Es por lo que en este trabajo de tesis buscamos combinar el poder de las arquitecturas basadas en aprendizaje profundo junto con características estilométricas presentes en las noticias.

1.2. Objetivos de la Tesis

Objetivo General

Proponer un modelo de detección de noticias falsas basado en una arquitectura de aprendizaje profundo potenciado con características estilométricas identificadas en las noticias falsas.

Objetivos Específicos

- Implementación de arquitecturas neuronales profundas bases para la detección de noticias falsas como *Long Short Term Memory (LSTM)*, *Convolutional Neural Networks (CNN)*, *Bidirectional Encoder Representation from transformers (BERT).*
- Identificación de características estilométricas que presentan las noticias falsas.
- Proponer una arquitectura neuronal profunda que integre las características estilométricas identificadas, para la detección de noticias falsas.
- Evaluar los modelos de detección de noticias falsas en corpus etiquetados.
- Evaluar los resultados de los modelos propuestos y seleccionar el modelo con los mejores resultados.
- Comparar el modelo desarrollado con métodos existentes en el estado del arte.

1.3. Hipótesis

Detectar las noticias falsas antes de que sean difundidas en las plataformas digitales es de gran importancia. Es por es todo, que con este trabajo se busca demostrar que es posible combinar características estilométricas con técnicas aprendizaje profundo para mejorar el rendimiento de los modelos de detección de noticias falsas principalmente en el lenguaje español.

1.4. Preguntas de investigación

Las preguntas de investigación que surgen en este trabajo son:

1. ¿Qué efecto tienen las técnicas de preprocesamiento de textos en el rendimiento de los modelos de clasificación obtenidos?
2. ¿Cuáles de los modelos de aprendizaje profundo obtenidos permiten alcanzar mejores resultados en base a la métrica F_1?
3. En base al tópico de las noticias ¿Qué categorías obtuvieron mayor porcentaje de noticias clasificadas correctamente con el modelo que mostró mejor desempeño?
4. ¿Qué efecto tiene el uso de las características estilométricas en los resultados de las clasificaciones en conjunto con las arquitecturas de aprendizaje profundo implementadas?

1.5. Metodología

Para el desarrollo de la presente investigación, en la etapa inicial se realizó una revisión general referente al área del aprendizaje profundo y el área del procesamiento del lenguaje natural. En particular se realizó una investigación sobre las diferentes técnicas de preprocesamiento de textos existentes, las posibles representaciones vectoriales de textos empleadas en la literatura. De igual forma se revisaron los enfoques presentes en el estado del arte para la clasificación de textos y en particular para la tarea de detección de noticias falsas. Basándonos

en los resultados presentes en el estado del arte por los modelos *CNN*, *LSTM* y el modelo preentrenado *BERT* consideramos que era un buen punto de partida para realizar experimentos. En paralelo se realizó una revisión de los corpus para determinar características estilométricas relevantes que pudieran ser combinadas con las arquitecturas de aprendizaje profundo. Seguidamente se trabajó en la propuesta de la arquitectura y finalmente se realizaron experimentos y se analizaron los resultados.

Para la etapa de experimentos fueron considerados dos corpus, uno en español *"Spanish Fake News Corpus"* [1], en el cual las noticias se encuentran clasificadas como verdaderas y falsas. El segundo corpus utilizado contiene noticias en inglés, y las mismas se encuentran clasificadas en verdaderas, falsas, parcialmente falsas y otro.

Referente a las tecnologías utilizadas para las implementaciones del proyecto, se seleccionó el lenguaje de programación *Python*, el cual posee una amplia documentación. En particular para las implementaciones de las arquitecturas se seleccionó *Pytorch*.

1.6. Contribución

Los resultados obtenidos en este trabajo poseen gran trascendencia científica, debido a que mediante la arquitectura propuesta como resultado de esta investigación es posible clasificar noticias publicadas en la web, en falsas y verdaderas principalmente en el idioma español.

Las principales contribuciones de este trabajo se pueden enumerar de la siguiente forma:

1. Se propuso una arquitectura novedosa la cual combina características estilométricas y técnicas de aprendizaje profundo para la clasificación de noticias en falsas y verdaderas.

2. Los resultados obtenidos mediante la evaluación de la arquitectura en varios corpus demuestran que la arquitectura propuesta obtiene resultados comparables a los del estado del arte.

1.7. Organización de la tesis

La investigación realizada para la implementación de una arquitectura basada en aprendizaje profundo combinada con características estilométricas, capaz de detectar noticias falsas se presenta en este documento de tesis, dicho documento se encuentra estructurado en seis capítulos que se describen a continuación.

En el Capítulo 2 se presentan los conceptos fundamentales necesarios para la comprensión del trabajo de tesis, los cuales incluyen nociones sobre las diferentes técnicas empleadas para procesar el lenguaje natural, además de las diferentes maneras de representar información textual de manera vectorial. También en este capítulo se describen las diferentes arquitecturas basadas en aprendizaje profundo utilizadas para la clasificación de textos. Entre las redes neuronales que se describen se encuentran las redes neuronales densas, redes neuronales recurrentes, redes neuronales convolucionales y el modelo *BERT*. En la última sección del capítulo se describen las diferentes métricas de evaluación de los clasificadores de textos como son la exactitud, la precisión, la exhaustividad y el F1.

En el Capítulo 3 se presenta un panorama global del estado del arte en el área de la detección de noticias falsas. En particular se presentan las diferentes perspectivas abordadas para la solución de esta tarea.

[1] `https://github.com/jpposadas/FakeNewsCorpusSpanish`

En el Capítulo 4 se describe el marco metodológico de este trabajo de tesis. En la primera sección se describe la arquitectura *baseline* utilizada en la fase de experimentación para comparar con la arquitectura basada en técnicas de aprendizaje profundo combinado con características estilométricas propuesta en este trabajo de investigación. En la segunda sección se presenta y detalla la arquitectura propuesta para la detección de noticias falsas en el lenguaje español.

En el Capítulo 5 se describe el conjunto de datos utilizados durante la fase de experimentación para la evaluación de la arquitectura propuesta.

Seguidamente en el Capítulo 6 se describe la metodología experimental utilizada en este trabajo de investigación. Además, se presentan los resultados experimentales obtenidos al utilizar la arquitectura propuesta basada en técnicas de aprendizaje profundo combinadas características estilométricas para la detección de noticias falsas en un corpus en español. Además, el capítulo presenta un estudio de ablación sobre cómo los componentes de la arquitectura propuesta influyen en los resultados finales.

El documento finaliza presentando en el Capítulo 7 las conclusiones, contribuciones del trabajo de tesis y las recomendaciones para trabajos futuros. Luego se muestran las referencias bibliográficas consultadas durante la investigación.

Capítulo 2

Marco Teórico

"Imagination is the discovering faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science."

Ada Lovelace

La clasificación de textos es uno de los campos más estudiados en el área del procesamiento del lenguaje natural y la minería de textos. Esta tarea de aprendizaje supervisado consiste en asignar una categoría a una oración o documento. Los textos pueden provenir de *tweets*, artículos de noticias, opiniones de usuarios, entre otros. Las categorías dependen del conjunto de datos elegido y pueden variar en temas. Esta clasificación se puede realizar mediante anotación manual o mediante etiquetado automático, pero producto del aumento de textos en aplicaciones digitales se ha vuelto cada vez más importante realizar investigaciones sobre la clasificación automática.

La tarea de clasificación consiste en aprender de un conjunto de características o un conjunto de datos de entrenamiento etiquetados para clasificar posteriormente un conjunto de características sin etiquetar.

Entre los problemas de clasificación de textos se encuentra la clasificación de noticias, clasificación de emociones, detección de sátira, análisis de tópicos, análisis de paráfrasis, entre otros.

En este capítulo se describen conceptos y términos que se utilizarán en el desarrollo del presente trabajo de investigación. En la Sección 2.1 se describen las técnicas desarrolladas en el estado del arte para el procesamiento de los textos. En particular se describe la técnica de tokenización, transformación de los textos a minúsculas, eliminación de palabras funcionales o *stopwords*, la eliminación de signos de puntuación, la corrección ortográfica, el *stemming* y la lematización. Seguidamente en la Sección 2.2 se resumen los métodos populares más utilizados para representar documentos y palabras de forma vectorial. Entre los descritos en esta sección se encuentran el modelo *one-hot encoding*, el modelo de bolsa de palabras, el modelo de n-gramas, el esquema de pesado *TfIdf*, el modelo *Word2Vec* y el modelo *GloVe*. Luego en la Sección 2.3 se presentan las diferentes arquitecturas basadas en redes neuronales utilizadas para la clasificación de textos. Entre las redes neuronales descritas se encuentran las redes neuronales densas, las redes neuronales recurrentes, las redes neuronales convolucionales y las arquitecturas basadas en *transformers*. Seguidamente en la Sección 2.4 donde se describen los diferentes análisis estilométrico que se pueden realizar en textos. A continuación, en la

Sección 2.5 se presentan las diferentes métricas de evaluación utilizadas para la clasificación de textos, como son, la exactitud, la precisión, exhaustividad y el F1. Por último, en el presente capítulo en la Sección 2.6 se presentan las diferentes librerías exclusivamente dedicadas para la resolución de tareas del procesamiento del lenguaje natural y la creación de modelos para la clasificación de textos mediante redes neuronales.

2.1. Preprocesamiento del texto

El procesamiento de los textos es una de las fases más aplicadas durante una tarea de clasificación de textos, antes de utilizar un modelo de aprendizaje automático. Esta fase consiste en realizar el proceso de tokenización, transformación de los textos a minúsculas, eliminación de signos de puntuación y etiquetas, eliminación de palabras funcionales, en el caso de textos procedentes de redes sociales como es el caso de Twitter también es necesario la eliminación de emoticonos, también se puede aplicar técnicas de corrección ortográfica, *stemming* y lematización. A continuación, se describen en detalles los diferentes procesamientos mencionados anteriormente.

2.1.1. Tokenización:

Este método de preprocesamiento permite dividir una secuencia de texto en palabras, frases, símbolos u otros elementos significativos llamados *tokens*.

2.1.2. Transformación de los textos a minúsculas:

Esta técnica es aplicada para proyectar todos los textos al mismo espacio de características, debido a que en un documento es posible que aparezca la misma palabra con mismo significado, escrita tanto en mayúscula como en minúsculas. Por ejemplo, si tenemos la palabra “Casa” y “casa” escritas de ambas formas las mismas se representan como dos palabras diferentes en el modelo de espacio vectorial lo que provocará más dimensiones.

2.1.3. Eliminación de *stopwords*:

Las palabras funcionales o *stopwords* son aquellas palabras frecuentes tales como “la", “el", “los", “de", “y", “etc.", es decir, artículos, conjunciones, preposiciones, entre otras, que no poseen contenido semántico importante para la etapa de clasificación debido a que no ayudan a diferenciar dos documentos. Generalmente estas palabras junto con los signos de puntuación son eliminadas antes de proceder a entrenar un modelo de aprendizaje automático

2.1.4. Eliminación de signos de puntuación

Los signos de puntuación (‘,’, ‘;’, ‘.’, ‘:’, ‘(’, ‘?’, ‘!’, entre otros) generalmente aparecen en los textos para ayudar a evitar errores o ambigüedades del lenguaje. Generalmente es una de las técnicas empleadas durante el procesamiento del lenguaje natural debido a que estos signos no añaden significado a los textos. Además, en caso de utilizar modelos de lenguajes que computan *word embeddings*.

2.1.5. Corrección ortográfica

Esta técnica generalmente es utilizada en textos procedentes de redes sociales, debido a que en estos tipos de textos generados por usuarios son más comunes estos errores. Para la corrección de los errores

2.1.6. *Stemming*

Esta técnica consiste en transformar una palabra a su forma raíz (*stem*), removiendo todos los afijos y en algunos casos se realiza alguna transformación a la base. Por ejemplo, la raíz de la palabra "pequeñitas" es "pequeñ".

2.1.7. Lematización

A diferencia de la técnica de *stemming* el procesamiento de lematización consiste en reducir las palabras a una palabra existente en el idioma en cuestión, es decir a una entrada existente en el diccionario. Por ejemplo, el lema de la palabra "pequeñitas" es "pequeño".

2.2. Representación vectorial de documentos y palabras

Los textos son fuentes de datos no estructurados debido a que el lenguaje es ambiguo, para poder ser utilizados en un modelo de aprendizaje automático o aprendizaje profundo es necesario convertirlos a un formato estandarizado. Es decir, en una representación numérica. Este proceso de transformación de texto a valores numéricos es realizado generalmente mediante modelos de lenguaje. A modo general en estos modelos se asignan frecuencias, probabilidades u otros valores a las palabras, secuencias de palabras, o documentos. Una forma sencilla de representar información textual es mediante el uso del Modelo de Espacio Vectorial (*Vector Space Model*). Este enfoque representa cada elemento textual como un vector disperso de identificadores, que pueden referirse a palabras individuales o n-gramas de palabras.

Entre las técnicas más utilizadas se encuentran: *one-hot encoding*, n-gramas, bolsa de palabras, *TfIdf* o *Word2Vec*. A continuación, se describen los métodos mencionados anteriormente.

2.2.1. Modelo *one-hot encoding*

En este modelo se crea un vector de dimensión n donde n es la cardinalidad del conjunto que representa el vocabulario del texto y se asocia cada palabra única con un índice en el vector. Para la representación de la palabra única se establece el componente del vector en 1 y el resto de los componentes se establece en 0. Entre las desventajas de este modelo está que no se tienen en cuenta las relaciones entre las palabras, ni información sobre su contexto, es decir, todas las palabras se consideran independientes. Además, cuando la cardinalidad del vocabulario es muy grande se torna muy ineficiente su implementación debido a que los vectores se vuelven demasiado grandes Liu et al. [2020].

Para representar documentos de forma vectorial, una vez que tengamos a la representación *one-hot* de las palabras, podemos sumar todas las representaciones vectoriales de las palabras que componen el documento:

$$d = \sum_{i=1}^{N} w_i \tag{2.1}$$

En la ecuación anterior tenemos que N representa el número de palabras presentes en el documento. Por otra parte, w_i es la representación vectorial de la palabra i. El resultado obtenido d representa un vector con dimensionalidad V.

2.2.2. Modelo bolsa de palabras

Este modelo convierte el texto en un vector de características contando la frecuencia de aparición de palabras en un texto o documento determinado Feldman et al. [2007]. Una de las desventajas de este modelo es que no se considera la importancia de las palabras en un documento.

2.2.3. Modelo de n-gramas

Los modelos de n-gramas son modelos probabilísticos para el trabajo con textos que utilizan una cantidad limitada de historial o dependencias de palabras, donde n se refiere al número de palabras que participan en la relación de dependencia Liu and Özsu [2009]. Es decir, en este modelo de lenguaje se estima la probabilidad de la siguiente palabra dadas una secuencia de palabras anteriores.

2.2.4. Frecuencia de término - Frecuencia inversa de documento (*TfIdf*)

Tf es una medida de la frecuencia de una palabra (w) en un documento (d). La misma se define como la relación entre la aparición de una palabra en un documento y el número total de palabras en un documento, esta medida no considera la importancia de las palabras.

$$Tf(w,d) = \frac{f(w,d)}{max\{f(w,d) : w \in d\}} \tag{2.2}$$

Donde,

- $f(w,d)$ es la frecuencia del término w en el documento d.
- $max\{f(w,d) : w \in d$ es la frecuencia máxima de algún término en el documento.

Por otra parte Idf es una medida de la importancia de una palabra. Por ejemplo en un texto pueden aparecer algunas palabras como es el caso de "la", 'y' repetidas muchas veces, pero estas palabras no son de gran importancia. Es por esto que esta medida proporciona una ponderación a cada palabra en función de su frecuencia en el corpus D.

$$Idf(w,D) = log\frac{|D|}{|\{d \in D : w \in d\}|} \tag{2.3}$$

Donde,

- $|D|$ es la cardinalidad de D, o número de documentos en la colección.
- $|\{d \in D : w \in d\}|$ es el número de documentos donde aparece el término w. Si el término no está en la colección se producirá una división por cero. Por lo tanto, es común ajustar esta fórmula a $1 + |\{d \in D : w \in d\}|$

De lo anterior se tiene que la medida *TfIdf* ofrece mayor importancia a las palabras que son más frecuentes en el documento. La misma se computa de la siguiente forma Feldman et al. [2007]:

$$TfIdf(w,d,D) = Tf(w,d) \times Idf(w,D) \tag{2.4}$$

2.2.5. Modelo *Word2Vec*

Es una de las técnicas más populares para determinar *word embeddings* de un corpus dado utilizando redes neuronales. La misma fue desarrollada por Tomas Mikolov en 2013 Mikolov et al. [2013]. En esta técnica se distinguen dos modelos *Continuous Bag of Words (CBOW)* y el modelo *Skip-Gram*.

En el modelo *CBOW* la palabra objetivo se predice teniendo en cuenta las palabras circundantes de la misma. Formalmente tenemos que dada una secuencia de palabras $w_1, w_2, w_3, \ldots w_T$ el objetivo del modelo *CBOW* es maximizar la siguiente función:

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} \log\left(p(w_t|w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c})\right) \tag{2.5}$$

Donde

- θ son las representaciones vectoriales de cada palabra
- c representa una ventana de tamaño fijo dada la palabra central
- t representa las diferentes posiciones en una secuencia de 1...T

En la ecuación anterior tenemos que c representa el tamaño del contexto y T representa el tamaño del corpus en base a la cantidad de palabras.

A diferencia del modelo anterior, en el modelo *Skip-Gram* la palabra objetivo es utilizada para la predicción de las palabras del contexto. Formalmente tenemos que dada una secuencia de palabras $w_1, w_2, w_3, \ldots w_T$ el objetivo del modelo *Skip-Gram* es maximizar la siguiente probabilidad.

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} \sum_{-c\leq j\leq c, j\neq 0} \log\left(p(w_{t+j}|w_t)\right) \tag{2.6}$$

En la ecuación anterior tenemos que T representa el tamaño del corpus en base a la cantidad de palabras, w_i se refiere a la palabra en la posición t. Por otra parte, tenemos que la definición estándar de $w_{t+j}|w_t$ utiliza la función *softmax* como se muestra a continuación:

$$p(w_O|w_I) = \frac{exp(V(w_O^T V(w_I)))}{\sum_{w=1}^{W} exp(W(w)^T W(w_I))} \tag{2.7}$$

Una vez obtenidos los *embeddings* de todas las palabras del vocabulario, un enfoque que se puede utilizar para la representación de los documentos puede ser realizar la suma de todos los *embeddings* como se presentó en la ecuación 2.1, otra alternativa puede ser tomar el promedio de los *embeddings* computados.

A modo general los *embeddings* obtenidos mediante *Word2Vec* son independientes del contexto, es decir, existe una única representación vectorial para una misma palabra. En caso de aparecer la misma palabra en múltiples contextos esta información es combinada en un mismo vector. Además, estos *embeddings* permiten explorar relaciones matemáticas interesantes entre palabras. Por ejemplo, al restar el vector de la palabra "hombre" del vector de la palabra "rey" y luego agregar al vector resultante el vector en representación de la palabra "mujer", se obtiene el vector asociado a la palabra "reina".

2.2.6. Modelo *GloVe (Global Vectors)*

Este modelo fue presentado por los autores Pennington et al. [2014]. En esencia es un modelo de regresión de mínimos cuadrados ponderados. La idea principal detrás de este modelo es que se pueden derivar relaciones semánticas entre palabras de la matriz de co-ocurrencia.

Formalmente el modelo de regresión de mínimos cuadrados ponderados se define de la siguiente forma:

$$J_\theta = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2 \tag{2.8}$$

Donde,

- θ son las representaciones vectoriales de cada palabra
- w_i representa las palabras en contexto
- $\tilde{w}_j$ representa las palabras fuera de contexto
- $f(X_{ij})$ es una función de ponderación, la cual pondera las palabras según el contenido de información disponible.
- b_i representa el sesgo referente a w_i
- $\tilde{b}_j$ representa el sesgo referente a $\tilde{w}_j$

Intuición sobre la función de ponderación $f(X_{ij})$:

- $f(0) = 0$
- Si f se ve como una función continua, debería desaparecer cuando $x \to 0$ suficientemente rápido como para que el valor sea finito.
- $f(x)$ no debe ser decreciente para que las co-ocurrencias raras no estén sobre ponderadas
- De manera similar, $f(x)$ debe ser relativamente pequeño para valores grandes de x, de modo que las co-ocurrencias frecuentes no estén sobre ponderadas.

Existen varias funciones que cumplen con las reglas anteriores. Sin embargo los autores de *GloVe* utilizan como función f una extensión de la función Ley Potencial la cual es bastante utilizada en modelado estadístico.

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{si} \quad x < x_{max} \\ \\ 1 & \text{en} \quad \textit{otro caso} \end{cases}$$

A diferencia del modelo *Word2Vec* este modelo se basa en un enfoque diferente. En lugar de extraer los *embeddings* de una red neuronal o una regresión logística diseñada para realizar una tarea de clasificación, en particular, predecir palabras vecinas. *GloVe* captura estadísticas globales y estadísticas locales de un corpus, para generar los vectores de palabras. Además optimiza los *embeddings* directamente para que el producto escalar de dos vectores de palabras sea igual al logaritmo de la cantidad de veces que estas dos palabras aparecerán cerca una de la otra.

2.2.7. *Bidirectional Encoder Representation From Transformers (BERT)*

Este modelo fue lanzado a finales de 2018 Devlin et al. [2018]. *BERT* es un método preentrenado para la representación del lenguaje. Puede ser utilizado para extraer características del lenguaje y además es posible ajustar este modelo en tareas específicas como son la clasificación de textos, el reconocimiento de entidades nombradas o predicción de partes de la oración (POS).

El modelo *BERT* utiliza una arquitectura *Transformer*, el cual es un mecanismo de atención que aprende las relaciones contextuales entre palabras o subpalabras en un texto. Los *Transformers* incluyen dos mecanismos separados, un codificador que lee la entrada de texto y un decodificador que produce una predicción para la tarea. Dado que el objetivo de *BERT* es generar un modelo de lenguaje, solo es necesario el mecanismo del codificador. existen dos versiones del modelo *BERT*, el modelo *BERT base* y el modelo *BERT large*. El modelo *BERT base* consta de 12 capas *transformers*, cada bloque *transformer* tiene un tamaño de valor 12 y 110 millones de parámetros. Por la otra parte el modelo *BERT large* consta de 24 capas *transformers*, cada bloque *transformer* tiene un tamaño de valor 16 y 340 millones de parámetros. Cada *transformer* toma una lista de *tokens embeddings* y produce la misma cantidad de *embeddings* en la salida (con los valores de características cambiados).

Una vez aplicado el procesamiento del corpus, el texto de las noticias debe ser dividido en *tokens* haciendo uso del tokenizador que viene incluido con el modelo *BERT*, luego estos tokens deben asignarse a su índice en el vocabulario del tokenizador. Una vez tokenizado es necesario agregar tokens especiales ([SEP], [CLS]). El token [SEP] debe ser agregado al final de cada oración y el token [CLS] debe ser utilizado al inicio de cada oración para las tareas de clasificación. A su vez las oraciones deben ser truncadas a una sola longitud constante, una vez truncado se debe utilizar el token [PAD] para el rellenado. Por otra parte, la "máscara de atención" es una matriz de 1 y 0 que indica qué tokens se están rellenando y cuáles no. Esta máscara le dice al mecanismo de "Atención" en *BERT* que no incorpore estos tokens [PAD] en su interpretación de la oración.

Utilizar *BERT* proporciona numerosas ventajas en comparación con entrenar un modelo de aprendizaje profundo como *CNN*, *LSTM* o *BiLSTM* en una tarea específica del procesamiento del lenguaje natural. Entre estas ventajas se encuentran el desarrollo más rápido, debido a que es un modelo preentrenado los pesos precalculados codifican mucha información sobre nuestro idioma. Como resultado, se necesita mucho menos tiempo para entrenar nuestro modelo ajustado; debido a que es como si hubiéramos entrenado las capas inferiores de la red y solo necesitáramos ajustarlas suavemente mientras usamos su salida como características para nuestra tarea en particular. En el artículo donde fue presentado los autores recomiendan utilizar solo entre 2 y 4 épocas de entrenamiento para ajustar *BERT* en una tarea específica de procesamiento del lenguaje natural. Otra de las ventajas es que, debido a que los pesos son previamente entrenados, es posible entrenar el modelo con buen rendimiento utilizando menos datos. A diferencia de un modelo que se construye desde cero, por tanto, mediante este modelo se puede dedicar menos tiempo a la recopilación del corpus.

Durante esta investigación se utilizaron dos versiones de este modelo *BERT pretrained multilingual language model* [1] el cual está preentrenado en 104 idiomas entre ellos el español con artículos extraídos de Wikipedia. La otra versión de *BERT* utilizada durante esta investigación tiene el nombre de *BETO* [2], este modelo fue propuesto por Cañeyte et al. Canete et al. [2020] específicamente para el lenguaje español.

[1] `https://huggingface.co/bert-base-multilingual-cased`
[2] `https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased`

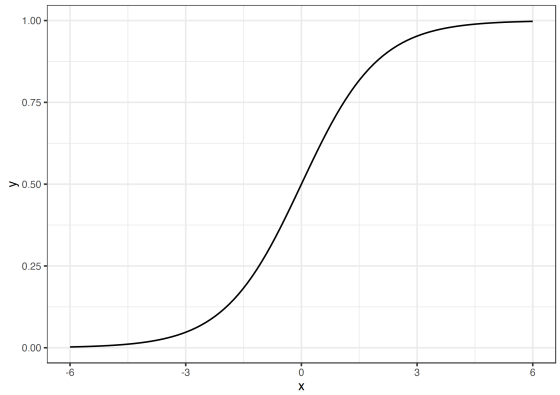


Figura 2.1: Representación de la función logística

2.3. Clasificación de textos

Para la clasificación de textos en el estado del arte se han desarrollado enfoques basados en aprendizaje de máquina y aprendizaje profundo. Entre los diferentes algoritmos basados en aprendizaje de máquina se encuentran las máquinas de vectores de soporte, la regresión logística, el clasificador bayesiano ingenuo, entre otros. Por otra parte, entre los diferentes modelos utilizados de aprendizaje profundo se encuentran las redes neuronales densas, las redes neuronales recurrentes, las redes neuronales convolucionales y las arquitecturas basadas en *transformers.*

2.3.1. Máquinas de vectores de soporte

Las máquinas de vectores de soporte (*Support Vector Machine o SVM por sus siglas en inglés*) son un algoritmo de aprendizaje de máquina que puede ser utilizado para tareas de clasificación y de regresión. Este algoritmo fue introducido por Cortes and Vapnik [1995]. A través de este algoritmo se puede determinar el mejor límite de decisión entre los vectores que pertenecen a un grupo o categoría y los vectores que no pertenecen a dicho grupo Smola and Vishwanathan [2008]. En el caso de la clasificación de textos los vectores constituyen la representación de los textos. Una vez que el algoritmo determina el límite de decisión mencionado anteriormente, *SVM* decide dónde dibujar la mejor línea o el mejor hiperplano que divide el espacio en dos subespacios. El primer subespacio es para los vectores que pertenecen a la categoría dada y el segundo para los vectores que no pertenecen.

2.3.2. Regresión logística

La regresión logística modela las probabilidades de problemas de clasificación que poseen dos posibles resultados. Este algoritmo es una extensión del modelo de regresión lineal para problemas de clasificación. El modelo de regresión logística, en lugar de ajustar una línea recta o un hiperplano, utiliza la función logística para comprimir la salida de una ecuación lineal entre 0 y 1.

La función logística se define de la siguiente forma:

$$logistic(\eta) = \frac{1}{1 + \exp(-\eta)} \tag{2.9}$$

En la regresión logística la relación entre el resultado y las características es modelada mediante el uso de la ecuación 2.9 y está dada por un valor de probabilidad, lo que fuerza a la salida a tomar solo valores entre 0 y 1 Smola and Vishwanathan [2008].

$$P(y^{(i)} = 1) = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_1^{(i)} + \cdots + \beta_p x_p^{(i)}))} \tag{2.10}$$

Dado que las redes neuronales han mostrado buenos resultados en la clasificación de textos, durante la experimentación del presente trabajo de tesis se utilizaron varias arquitecturas de estas redes, las cuales se describen a continuación.

2.3.3. Redes neuronales densas

En este tipo de red, todas las neuronas de una capa están conectadas a todas las neuronas de la siguiente capa. Esto permite que sean eficaces en cuanto al tratamiento de información debido a que cada neurona dispone de más datos para tratar. No obstante, este tipo de red es más lenta a la hora de procesar los datos, debido a la gran cantidad de información que mueven de una capa a otra.

Una red neuronal densa básica con una capa de entrada, una única capa oculta y una capa de salida, puede ser representada mediante la siguiente función:

$$f(x) = g(b^{(2)} + W^{(2)}(s(b^{(2)} + W^{(2)}))) \tag{2.11}$$

En este caso $b^{(1)}$ y $b^{(2)}$ son los vectores de sesgo, $W^{(1)}$ y $W^{(2)}$ son las matrices de pesos y g y s representan las funciones de activación. En el caso de redes más complejas, es decir, con más capas ocultas, para el entrenamiento se utiliza el algoritmo *back-propagation* en combinación con alguna variación de la optimización de descenso por gradiente Goldberg [2016]. Para el procedimiento de entrenamiento de las redes neuronales se ha utilizado frecuentemente el optimizador *Adaptative Moment Estimation (Adam)*. Este optimizador calcula las actualizaciones de parámetros aprovechando un promedio exponencialmente decreciente de gradientes anteriores, junto con tasas de aprendizaje adaptables para cada parámetro Da [2014]. Es decir, en la práctica, realiza actualizaciones más grandes para parámetros poco frecuentes y actualizaciones más pequeñas para parámetros frecuentes.

Estas redes completamente conectadas permiten recibir como entrada secuencias de palabras mediante representación vectorial. Las mismas constan de una capa de entrada, una capa de salida y puede constar de una serie de capas ocultas en dependencia de la profundidad con la que queramos experimentar en una tarea en particular.

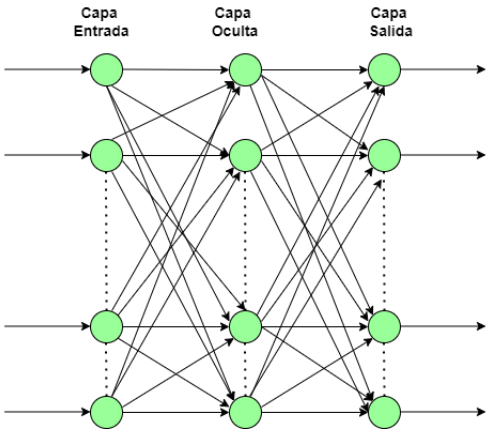


Figura 2.2: Arquitectura de una red neuronal densa

En la Figura 2.2 se muestra la arquitectura de una red neuronal densa.

Otra de las arquitecturas generalmente utilizadas en el área del procesamiento del lenguaje natural son las redes neuronales convolucionales (*CNN*) y las redes neuronales recurrentes (*RNN*).

2.3.4. Redes neuronales recurrentes

Los modelos basados en redes neuronales recurrentes analizan el texto como una secuencia de palabras, este modelo permite capturar dependencias entre las palabras y estructuras del texto que ayudan a la clasificación. A modo general estas redes están conformadas por una capa de entrada, una capa de salida y pueden tener varias unidades recurrentes ocultas que tienen puertas de memoria.

Entre las variantes existentes de las redes neuronales recurrentes se encuentra *Long Short Term Memory (LSTM)* la cual fue introducida por Hochreiter and Schmidhuber [1997] y *Gated Recurrent Unit (GRU)* propuesta por Chung et al. [2014].

El modelo *LSTM* permite capturar mejor las dependencias a largo plazo. Este modelo enfrenta el problema del desvanecimiento del gradiente que afrontan las redes neuronales recurrentes clásicas. Esto es a partir de que en este modelo se introduce una celda de memoria para recordar valores en intervalos de tiempo arbitrarios y tres puertas (puerta de entrada, puerta de salida, puerta de olvido) para regular el flujo de información dentro y fuera de la celda.

Las arquitecturas *LSTM* funcionan de la siguiente forma. El primer paso consiste en decidir qué información se va a desechar del estado de la celda, esta decisión es tomada a través de la capa sigmoide llamada *forget gate layer*. Revisa el valor de entrada h_{t-1} y x_t y devuelve un valor entre 0 y 1 para cada número en el estado de la celda C_{t-1}. El valor 1 representa mantener la información mientras que un valor 0 representa desecharla por completo.

El siguiente paso es decidir cual información se almacenará en el estado de la celda. Primeramente, una capa sigmoide llamada *input gate layer* decide que valores se actualizarán. Seguidamente una capa *tanh* genera un vector de nuevos valore candidatos nombrado $\tilde{C}_t$, el cual puede agregarse al estado. En el siguiente paso se combinan los dos resultados anteriores para generar una actualización del estado. Para obtener la actualización final del antiguo estado de la celda C_{t-1} a la celda C_t se multiplica el estado antiguo por f_t, teniendo en cuenta la información que se decidió desechar previamente, luego se le añade $i_t * \tilde{C}_t$.

Por último, se decide la salida que se va a generar, la misma consiste en el estado de la celda. Primero, se ejecuta una capa sigmoide que decide qué partes del estado de la celda se generarán. Luego, se pasa el estado de la celda a través de *tanh* para obtener valores estén entre -1 y 1 y se multiplica por la salida de la puerta sigmoide, de modo que solo se seleccionan las partes que decidimos.

A continuación, se resume formalmente mediante un conjunto de ecuaciones lo explicado anteriormente:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2.12}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2.13}$$

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C) \tag{2.14}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.15}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.16}$$

$$h_t = o_t * \tanh C_t \tag{2.17}$$

En la Figura 2.3 se puede visualizar la arquitectura de una red neuronal *LSTM*.

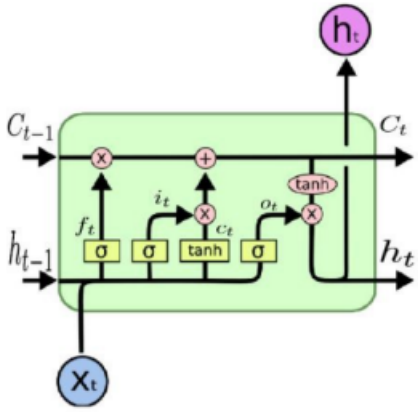


Figura 2.3: Arquitectura de una red neuronal *LSTM*

2.3.5. Redes neuronales convolucionales

Las redes neuronales convolucionales fueron introducidas por Fukushima and Miyake [1982]. Estas redes son muy similares a las redes neuronales densas, estas están formadas de igual manera por una capa de entrada y una capa de salida, así como de varias capas ocultas. Las capas ocultas de estas redes generalmente son capas convolucionales, capas de agrupación, capas completamente conectadas y capas de normalización. En particular en la clasificación de textos otra de las capas presentes es conocida como capa de *embedding*.

A diferencia de las redes neuronales recurrentes que están entrenadas para reconocer patrones en el tiempo, estas aprenden a reconocer patrones en el espacio. Este tipo de arquitecturas

funcionan bien cuando es importante detectar patrones locales y de posición invariable, estos patrones pueden ser frases clave que expresen un sentimiento particular o un tema.

A continuación, ofrezco una breve descripción de las principales características de las capas que componen esta arquitectura.

Capa de *Embedding*:

Esta capa es de gran importancia en las arquitecturas basadas en aprendizaje profundo en particular en las tareas de clasificación de textos debido a que permite convertir cada palabra en un vector de longitud fijo. El vector resultante es denso y tiene valores reales en lugar de solo ceros y unos. De esta forma, la capa de *embedding* funciona como una tabla de búsqueda. Las palabras son las claves en esta tabla, mientras que los vectores de palabras densos son los valores.

Capa convolucional:

Esta capa es la primera capa presente en la arquitectura utilizada para extraer las características de la entrada. En el caso de la clasificación de textos estas características son extraídas de la representación de *embeddings* del texto. En esta capa se realiza la operación de convolución entre la matriz de entrada y un filtro de tamaño $K \times K$. Al deslizar el filtro sobre la matriz de entrada, se toma el producto escalar entre el filtro y las partes de la matriz de entrada con respecto al tamaño del filtro. La salida se denomina mapa de características que nos brinda información sobre la representación del texto de entrada. Seguidamente, este mapa de características se alimenta a otras capas para aprender otras características de la entrada.

Los valores subsiguientes del mapa de características se calculan de acuerdo con la fórmula 2.18, donde la matriz de entrada se denota por f y nuestro filtro o *kernel* se denota por h. Los índices de filas y columnas de la matriz de resultados están marcados con m y n respectivamente.

$$G[m,n] = (f * h)[m,n] = \sum_j \sum_k h[j,k] f[m-j, n-k] \tag{2.18}$$

Capa de agrupación:

Generalmente la capa de convolución en las redes convolucionales se encuentra seguida por capas de agrupación (*pooling layers*). El objetivo de esta capa es disminuir el tamaño del mapa de características convolucionado para reducir los costos computacionales. Según el método utilizado, existen varios tipos de operaciones de agrupación. En el método Agrupación Máxima (*Max Pooling*), el elemento más grande se selecciona del mapa de características. En el método de Agrupación Promedio (Average Pooling) se calcula el promedio de los elementos en una sección de la matriz de tamaño predefinido. La suma total de los elementos en la sección predefinida se calcula meiante el método de Agrupación Sum *(Sum Pooling)*. Esta capa de agrupación generalmente sirve como puente entre la capa convolucional y la capa completamente conectada.

Capa completamente conectada:

La capa completamente conectada (*Fully Connected Layer*) consta de los pesos y sesgos junto con las neuronas y se utiliza para conectar las neuronas entre dos capas diferentes. Estas capas generalmente se colocan antes de la capa de salida y forman las últimas capas de una arquitectura de red neuronal convolucional.

En la Figura 2.4 se puede visualizar una arquitectura de una red neuronal convolucional.

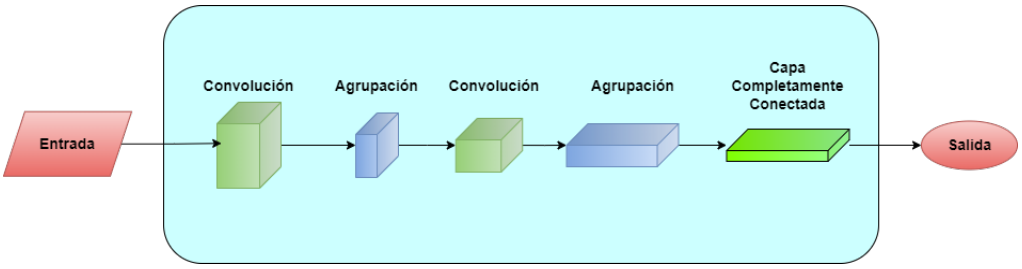


Figura 2.4: Arquitectura de redes neuronales convolucionales

2.4. Análisis estilométrico en textos.

La estilometría es una rama de la lingüística computacional que estudia el análisis estadístico de las características lingüísticas en los textos Zheng et al. [2006]. Los métodos basados en características estilométricas se aplican a múltiples tareas del procesamiento del lenguaje natural, incluida la atribución de autoría, la verificación de autoría, la creación de perfiles de autor, la detección de cambios de estilo y la clasificación de textos escritos Lagutina et al. [2019].

Las características estilométricas se pueden clasificar en características basadas en el léxico, características basadas en la sintaxis, características estructurales y características específicas del contenido del texto.

2.4.1. Características basadas en el léxico:

Estas características son analizadas tanto a nivel de caracteres como a nivel de palabras. En el caso de nivel de caracteres los rasgos evaluados indican las preferencias del redactor del texto por el uso de ciertos caracteres especiales o símbolos. Por ejemplo, en este caso son computados el número total de caracteres presentes en el texto, el número total de caracteres alfabéticos, el número total de letras mayúsculas, el número total de dígitos y el número total de espacios en blanco, el número total de espacios de tabulación, la frecuencia de caracteres y la frecuencia de caracteres especiales.

En el caso de nivel de palabras son computados el número total de palabras, número total de palabras cortas, es decir, con una cantidad máxima de caracteres especificados, número total de caracteres por palabra, promedio de longitud de palabra, longitud media de las frases en términos de caracteres, promedio de longitud de oración en términos de palabra, número total de palabras diferentes en el texto, distribución de la longitud de las palabras, análisis de las palabras funcionales y la riqueza del vocabulario.

2.4.2. Características sintácticas

En este caso son analizadas características que capturar el estilo de escritura de un autor al nivel de las oraciones. Por ejemplo, la frecuencia de los signos de puntuación y la frecuencia de las palabras funcionales empleadas en el texto, las diferentes partes del discurso (*Part Of Speech*).

2.4.3. Características estructurales:

En particular estas características analizan la forma en que el autor organiza el diseño de un escrito. Por ejemplo, se tiene en cuenta el número total de líneas en el documento, número total de oraciones en el documento, número total de párrafos en el documento, número de oraciones por párrafos, número de caracteres por párrafos, número de palabras por párrafos, longitud promedio de los párrafos, número de párrafos por documento.

2.4.4. Características basadas en el contenido:

En este caso se analizan palabras especiales o caracteres relacionados con el tema específico abordado por el autor en el texto.

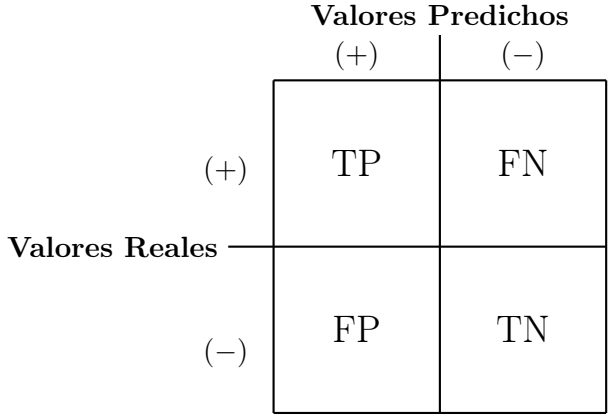
2.5. Métricas de evaluación

Existen varias métricas para evaluar los clasificadores de textos, en particular para evaluar las clasificaciones de las noticias durante la fase de experimentación se utilizarán las métricas *precisión, exhaustividad o recobrado, exactitud y* F_1.

Primeramente, antes de pasar a la definición de las métricas es necesario describir que es una matriz de confusión, la cual permite representar de forma matricial los resultados de las predicciones obtenidas mediante un clasificador binario.

La matriz de confusión es una representación tabular del rendimiento de un modelo de clasificación en el conjunto de prueba, que consta de cuatro parámetros: verdadero positivo, falso positivo, verdadero negativo y falso negativo. Por tanto, una vez obtenidas las predicciones cada uno puede coincidir en algunas de dichas categorías, basado en como coincide con el valor real:

- Verdadero Positivo (TP): Predicho Verdadero y Verdadero en realidad.
- Verdadero Negativo (TN): Predicho Falso y Falso en realidad.
- Falso Positivo (FP): Predicción de verdadero y falso en la realidad.
- Falso Negativo (FN): Predicción de falso y verdadero en la realidad.



2.5.1. Exactitud

Esta métrica es de las más utilizadas, representa el porcentaje de observaciones predichas correctamente, ya sean verdaderas o falsas. Dicha métrica se define de la siguiente manera:

$$\text{Exactitud} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.19}$$

2.5.2. Precisión

La precisión mide la proporción de verdaderos positivos con respecto a todos las observaciones pronosticados como verdaderas. Formalmente, la métrica de precisión se define de la siguiente manera:

$$\text{Precisión} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.20}$$

Los posibles valores para tomar por esta métrica se encuentran en el intervalo de $[0, 1]$. Por tanto, la precisión máxima posible se alcanza con valor 1 y representa que todas las instancias clasificadas como clase positiva fueron correctamente predichas. Por el contrario, en el caso que el valor de la precisión sea más cercano s 0 mayor será el número de instancias clasificadas incorrectamente.

2.5.3. Exhaustividad

La exhaustividad o recobrado, representa qué tan sensible es el modelo para identificar la clase positiva. Representa el número total de clasificaciones positivas fuera de la clase verdadera. Formalmente, la métrica de exhaustividad se define de la siguiente manera:

$$\text{Exhaustividad} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.21}$$

Los posibles valores para tomar por esta métrica se encuentran en el intervalo de $[0, 1]$. Por tanto, la exhaustividad máxima posible se alcanza con valor 1 y representa que se ha encontrado todo documento relevante que residía en el conjunto de datos. Por el contrario, en el caso que el valor de la exhaustividad sea igual a 0, se tiene que los documentos obtenidos no poseen relevancia alguna.

2.5.4. F1

La métrica F_1 es la media armónica entre la precisión y la exhaustividad, esta métrica se encuentra en el intervalo $[0, 1]$, donde 1 representa su mejor valor, es decir, los valores de las métricas de precisión y exhaustividad obtuvieron resultados perfectos y 0 en el caso contrario. Formalmente, la métrica F_1 se define de la siguiente manera:

$$F_1 = 2 \times \frac{\text{Precisión} \times \text{Exhaustividad}}{\text{Precisión} + \text{Exhaustividad}} \tag{2.22}$$

Estas métricas son comúnmente usadas en la evaluación de los modelos de detección de noticias falsas. En esta tarea la métrica exactitud mide la similitud entre las noticias que se predijeron como falsas y las noticias que son falsas realmente. Por otra parte, la métrica de exhaustividad representa el número de artículos predichos como verdaderos del total de artículos verdaderos. En el caso de la precisión se analiza el número de artículos que están marcados como verdaderos de todos los artículos predichos positivamente (verdaderos).

2.6. Librerías para el PLN

En la actualidad existe una gran variedad de librerías empleadas para resolver tareas en el área del procesamiento del lenguaje natural, específicamente el procesamiento de los textos. A modo general estas librerías disponen de distintos *tokenizadores*, lematizadores, vectorizadores, analizadores de las diferentes partes de la oración *POS(part-of-speech)*, vecorizadores, entre otros. Entre las más utilizadas se encuentran *NLTK* Bird [2006], *SpaCy* Montani et al. [2022], *Stanza* Qi et al. [2020], *FreeLing* Padró and Stanilovsky [2012].

- ***Natural Language Toolkit (NLTK)*** [3]**:** Es una plataforma creada especialmente para el lenguaje de programación *Python*, la cual contiene herramientas para una gran variedad de tareas del procesamiento del lenguaje natural. En particular esta herramienta se encuentra centrada en el lenguaje español. Entre las tareas para las cuales puede ser usado se encuentran la tokenización, lematización, *stemming*, parseo, etiquetado *POS*, entre otras.

- ***SpaCy*** [4]**:** Es una librería de *software* creada especialmente para el lenguaje de programación *Python*. Posee soporte para más de 66 idiomas, entre ellos el español. Entre las características principales de esta herramienta se encuentran, la existencia de vectores de palabras pre-entrenados, posee componentes para el reconocimiento de entidades nombradas, etiquetado de partes del discurso, análisis de dependencia, segmentación de oraciones, clasificación de texto, tokenización, lematización, análisis morfológico, vinculación de entidades, entre otrs.

- ***FreeLing*** [5]**:** Es un kit de herramientas para el procesamiento del lenguaje natural creado especialmente para el lenguaje *C++*. Posee soporte para múltiples idiomas, pero el español es uno de los idiomas principales. Esta herramienta cubre una gran cantidad de tareas de PLN. Entre ellas se encuentra la tokenización de los textos, análisis morfológico, reconocimiento de palabras compuestas, segmentación de contracciones, reconocimiento de entidades nombradas, anotación y desambiguación del sentido basadas en *WordNet*, análisis de dependencia basado en reglas, análisis de dependencia estadística, resolución de correferencias, extracción de gráficos semánticos, entre otras.

- ***Stanza*** [6]**:** Es una herramienta construida sobre la biblioteca *PyTorch* perteneciente al lenguaje de programación *Python*. Posee modelos neuronales pre-entrenados que admiten 70 idiomas. Entre las principales tareas que se pueden abordar con esta herramienta se encuentran la tokenización, lematización, segmentación de oraciones, análisis de las diferentes partes de la oración *POS (Part-Of-Speech)*, reconocimiento de entidades nombradas, análisis de sentimiento, entre otros.

Por otra parte, para la creación de los modelos de clasificación de textos basados en redes neuronales existen dos *frameworks* implementados para el lenguaje de programación *Python*. Estos son *PyTorch* Stevens et al. [2020] y *Tensorflow* Zaccone et al. [2017].

[3] http://www.nltk.org/
[4] https://spacy.io/
[5] http://nlp.lsi.upc.edu/freeling/
[6] https://stanfordnlp.github.io/stanza/

- ***PyTorch*** [7]**:** Es un *framework* de de código abierto utilizado para el *machine learning*. Fue desarrollado por *Facebook* en el 2016 y su implementación está basada en el lenguaje de programación *Python*. Este *framework* se especializa en cálculos de tensores, diferenciación automática y aceleración de *GPU*.

- ***Tensorflow*** [8]**:** Al igual que *PyTorch*, es un *framework* de código abierto utilizado para el desarrollo de inteligencia artificial y *machine learning*. Fue desarrollado por *Google* y lanzado en 2015. *Tensorflow* permite construir y entrenar redes neuronales para detectar patrones y razonamientos usados por los humanos. Además de trabajar con redes neuronales, *TensorFlow* es multiplataforma, por lo que permite trabajar con *GPUs* y *CPUs* e incluso con las unidades de procesamiento de tensores (*TPUs*).

2.7. Resumen

En este capítulo se presentó la terminología y conceptos relacionados con la tarea de detección de noticias falsas la cual es una subtarea dentro del área de la clasificación de textos. A modo general se describieron las principales técnicas de procesamiento de textos que fueron utilizadas durante la fase de experimentación del presente trabajo de tesis. Además, se presentaron las diferentes formas de representar un texto en forma numérica para poder ser utilizada mediante un modelo de aprendizaje automático o profundo. De igual forma se describieron las diferentes arquitecturas basadas en redes neuronales utilizadas para la clasificación de textos y las diferentes métricas empleadas para la evaluación de dichas arquitecturas. También en este capítulo fueron descritas las diferentes características estilométricas que pueden ser extraídas de los textos.

[7] https://pytorch.org/
[8] https://tensorflow.org/

Capítulo 3

Estado del arte

"Imagination is the discovering faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science."

Ada Lovelace

En el presente capítulo se realiza una descripción general sobre la tarea de detección de noticias falsas. En la Sección 3.2 se presenta la definición formal de la tarea de detección de noticias falsas. Además, en esta sección se describen los diferentes enfoques abordados en el estado del arte para la resolución de dicha tarea.

Seguidamente la Sección 3.3 se divide en dos partes, en la primera parte se presenta una descripción general de las diferentes arquitecturas basadas en aprendizaje de máquina y en la segunda se presentan algunos de los trabajos presentes en el estado del arte que combinan características estilísticas con técnicas de aprendizaje de máquina para la detección de noticias falsas.

Por último, la Sección 3.4 se divide en dos partes, de igual forma a la sección anterior. En la primera parte se presenta una descripción general de las diferentes arquitecturas basadas en aprendizaje de profundo y en la segunda se presentan algunos de los trabajos presentes en el estado del arte que combinan características estilísticas con técnicas de aprendizaje profundo para la detección de noticias falsas.

3.1. Introducción

En el estado del arte se han presentado múltiples investigaciones relacionadas con el tema de la detección de noticias falsas. Algunos estudios plantean que hay al menos tres tipos de noticias falsas Rubin et al. [2015]. El primer tipo de noticias falsas son (i) sátiras o parodias donde sitios como *Onion4* [1] o *El Deforma* [2] publican noticias falsas para satirizar de forma humorística a los medios. El segundo tipo de noticias falsas engloba aquellas noticias falsas que son verdaderas a medias, pero se utilizan en contextos erróneos, noticias que constituyen engaños, rumores y noticias engañosas que no se basan en hechos verificados. El último grupo

[1] https://www.theonion.com/
[2] https://eldeforma.com/

involucra noticias creadas intencionalmente con información falsa, es decir, estas noticias se fabrican y se difunden deliberadamente en plataformas digitales.

Entre las investigaciones existente se han propuesto múltiples definiciones sobre el término de *noticias falsas (fake news)*. Una de las definiciones fue propuesta por Shu et al. [2017] la cual plantea que las noticias falsas "como artículos de noticias intencionalmente y de manera verificable falsas". En particular de esta definición se desprenden dos características claves; la primera es que las noticias falsas incluyen información falsa que se puede verificar como tal y la segunda característica es que estas noticias son creadas con la intención deshonesta de engañar a los consumidores. Por otra parte, Sharma et al. [2019] define las noticias falsas como "un artículo de noticia o mensaje publicado y propagado a través de los medios de comunicación, que lleva información falsa sin importar los medios y motivos detrás la misma".

Otra de las definiciones fue dada por los autores Golbeck et al. [2018], los cuales plantean que las noticias falsas son "información presentada como una noticia que es objetivamente incorrecta y diseñada para engañar al consumidor haciéndole creer que es verdad".

3.2. Tarea de detección de noticias falsas

Como se mencionó en el capítulo anterior entre los problemas de clasificación de textos se encuentra la tarea de detección de noticias falsas, la cual puede ser abordada como un problema de clasificación binaria. Generalmente en el estado del arte se define la tarea de detección de noticias falsas como un problema de clasificación binaria, es decir, se clasifican las noticias en falsas y verdaderas. No obstante, debido a la dificultad que representa realizar la categorización en solo dos clases, en múltiples ocasiones se formula esta tarea como un problema de clasificación multiclases, donde se pueden categorizar las noticias como falsas, verdaderas, parcialmente falsas u otra categoría en la cual se agrupan aquellas noticias que no pueden ser categorizadas como verdaderas, falsa o parcialmente falsa debido a la falta de evidencia sobre sus afirmaciones.

La tarea de detección de noticias falsas fue definida formalmente por Shu et al. [2017], de la siguiente forma:

Dado un conjunto de noticias $\mathcal{A} = \{a_1, a_2, \ldots, a_N\}$, la tarea de la detección de noticias falsas es predecir si el artículo de noticias es falso o no, $\mathcal{F} :\rightarrow \{0, 1\}$ de tal forma que:

$$\mathcal{F}(a) = \begin{cases} 1 & \text{si} \quad a\ es\ una\ noticia \quad falsa \\ \\ 0 & \text{en} \quad otro\ caso \end{cases}$$

donde $\mathcal{F}$ es la función de predicción.

En este capítulo se presenta un panorama global del estado del arte en el área de la detección de noticias falsas. En el estado del arte se describe esta tarea desde cuatro perspectivas, estas son:

- ***Métodos basados en el conocimiento:*** En este enfoque se verifica si el conocimiento proveniente del contenido de la noticia es consistente con hechos verdaderos que se conocen al respecto. Generalmente para la detección de noticias falsas mediante el uso de este enfoque se utiliza el proceso conocido como verificación de hechos (*fact checking*). En este proceso se distingue la verificación manual de hechos y la verificación automática de hechos.

  La verificación manual de hechos se utiliza mediante el trabajo de expertos en el dominio, para la verificación de los contenidos de las noticias. Dado que es realizada por

personas expertas se obtienen muy buenos resultados, pero es un proceso muy costoso en dependencia del volumen de noticias a verificar. En la actualidad existen varios sitios *webs* para la verificación de hechos, en el cual se brinda información sobre la clasificación de las noticias Duke Reporter's Lab [2021]. Por ejemplo, algunos utilizan etiquetas como Verdadero; Mayormente Verdadero; Verdadero a medias; Mayormente falso; Falso. Entre estos sitios se encuentran *FactCheck*[3] , *PoliFact*[4], *The Washington Post Fact Checker* [5], Chequeado [6], VerificadoMX [7], Escenario Tlaxcala [8], ElToque [9].

Debido al gran volumen de información en las redes sociales, han surgido los verificadores de hechos automáticos, los cuales están basados en técnicas de recuperación de información, procesamiento de lenguaje natural, aprendizaje automático y en teoría de grafos. En el estado del arte estos verificadores son construidos dividiendo el proceso en dos etapas: la extracción de los hechos o construcción de la base de conocimientos y en la verificación de datos o comparación de conocimientos.

- ***Métodos basados en el origen:*** En este caso la detección se realiza basándose en la credibilidad de la fuente de dónde surgió la noticia, dónde fue publicada y también se tiene en cuenta la divulgación de la noticia en las redes sociales. Este análisis se sustenta en los múltiples estudios que han evidenciado que la mayoría de las noticias falsas provienen de sitios *webs* que acostumbran a publicar noticias falsas o engañosas. En este enfoque el concepto "fuente" hace alusión tanto a los redactores que crean la noticia, como a los medios que la publican, los usuarios o medios que las publican en las redes sociales.

- ***Métodos basados en la propagación de la noticia:*** En este caso la detección se realiza evaluando el alcance de la noticia en *Internet* y además se analiza como son difundidas estas noticias por los usuarios. Al igual que en los otros enfoques, estos métodos se formulan como problemas de clasificación binaria, la diferencia radica en las entradas de los clasificadores que en este caso es una cascada de las noticias o una representación gráfica donde se captura información sobre la propagación de la noticia.

- ***Métodos basados en el estilo:*** En este enfoque, al igual que en los métodos basado en el conocimiento, se analiza el contenido de las noticias. La diferencia radica en que en los métodos basados en el estilo se tiene en cuenta la forma de la escritura del contenido de la noticia, con el objetivo de hacer una evaluación de la intención de la noticia, si muestran intención de engañar o no al lector. La intuición detrás de estos métodos radica en que los autores de noticias falsas tienden a escribir con ciertos estilos o características, con el propósito de convencer al lector. Las características de los textos se pueden dividir en características generales y características latentes. Las características generales describen las noticias desde varios niveles: el léxico, la sintaxis, el discurso y la semántica Conroy et al. [2015]. Por ejemplo, entre las características o estilos generales se pueden verificar las estadísticas de frecuencia de los léxicos, las frecuencias de las etiquetas *POS*, frecuencia de signos de exclamación o frecuencia de mayúsculas. A modo

[3] https://www.factcheck.org/
[4] https://www.politifact.com/
[5] https://www.washingtonpost.com/news/fact-checker/
[6] https://chequeado.com/
[7] https://verificado.mx/
[8] https://escenariotlx.com/ficcionesinformativas/
[9] https://eltoque.com/proyectos/eltoque-defacto-verificacion-datos

general la exactitud de estos métodos depende de cuan bien se capturan y representan el contenido de las noticias.

A modo general en el estado del arte se han presentado varias investigaciones en base a los patrones que presentan las noticias falsas que permiten distinguirlas de las verdaderas Zhou et al. [2019]. En particular se ha mostrado que estas noticias presentan mayor informalidad, es decir, son escritas mediante un lenguaje coloquial, donde en ocasiones hay presencias de palabras obscenas, mayor frecuencia de palabras emocionales, verbos que indican que se va a reportar algo ("anunciar") o el uso de verbos subjetivos como es el caso de "sentir" o "creer".

En particular entre las propuestas presentes en el estado del arte de algoritmos de detección de noticias falsas mediante métodos basados en el estilo se dividen en modelos de aprendizaje automático tradicional y modelos de aprendizaje profundo. A continuación, se mencionan algunas de las diferentes propuestas presentes en el estado del arte.

3.3. Métodos basados en aprendizaje automático

En el estado del arte principalmente se muestran el uso de clasificadores supervisados para la detección de noticias falsas basadas en el estilo. En particular se presentan máquinas de vectores de soporte, *random forest*, el clasificador bayesiano ingenuo, regresión logística y *XGBoost*. En estos modelos el contenido de las noticias es representado mediante características sintácticas, léxicas y semánticas que son extraídas de los textos de dichas noticias.

Por ejemplo, en Jain and Kasbe [2018] los autores proponen un método para la detección de noticias falsas basado en aprendizaje automático. Además, presentan formas de aplicar dicho método en *Facebook*, una de las plataformas de redes sociales en línea más populares. El método propuesto por los autores utiliza el modelo de clasificación *Naive Bayes* para predecir si una publicación en *Facebook* se etiquetará como real o falsa.

En particular en Castillo et al. [2011] se utiliza un clasificador supervisado combinado con un método basado en selección de características para evaluar la credibilidad de un corpus de *tweets*. En este trabajo los autores identifican cuatro tipo de características, estas son características basadas en los mensajes (tamaño de los mensajes, si es un *re-tweet* o no, número de palabras de sentimiento positivo o negativo contenido en el mensaje y aparición de *hashtags* o no), características basadas en los usuarios (edad de registro, cantidad de seguidores y la cantidad de *tweets* que el usuario ha escrito en su cuenta), características basadas en los temas(proporción de *tweets* que contienen *urls*, proporción de *tweets* que contienen *hashtags*) y características basadas en la propagación de los *tweets* (profundidad del grafo construido en base a los *re-tweets* y el número de *tweets* iniciales del tema).

Siguiendo esta línea en Shu et al. [2017] los autores propusieron un enfoque que consta de dos etapas, la primera para la selección de características relevantes y la segunda para la construcción del clasificador lineal. En particular el autor utiliza características estilométricas como el total de palabras del texto, cantidad de caracteres por palabra, frecuencias de palabras grandes, frecuencias de n-gramas, enfoques de bolsa de palabras y etiquetado de partes del discurso (*POS*) para la detección de noticias falsas.

Otro de los trabajos presentes en el estado del arte donde se analiza el estilo de escritura y es combinado con algoritmos de aprendizaje automático fue presentado por Potthast et al. [2017]. En esta investigación los autores utilizan el conjunto de datos Buzzfeed el cual está compuesto por la producción de 9 editores en una semana cercana a las elecciones estadounidenses en

el 2016. Entre las características estilométricas empleadas por los autores se encuentran n-gramas de caracteres y palabras funcionales, índices de legibilidad, así como características como enlaces externos y el número promedio de palabras por párrafo.

Otro de los trabajos donde se realiza una propuesta de detección de noticias falsas mediante aprendizaje de máquinas y características linguísticas fue presentado por Pérez-Rosas et al. [2017], en este trabajo los autores presentan el *dataset FakeNewsAMT* y además proponen un modelo para la clasificación de noticias en dicho *dataset*. Para la clasificación realizan varios experimentos con diferentes conjuntos de características combinado con un clasificador *SVM* con *kernel* lineal.

Entre las propuestas más recientes se encuentra la presentada por Jain et al. [2020] en la cual mediante el uso de características estilométricas o linguísticas y modelos de aprendizaje automático como son *Naive Bayes, Random Forest* y *Logistic Regression*, mejoraron los resultados existentes en el estado del arte para la detección de noticias falsas específicamente en el *dataset FakeNewsNet* Shu et al. [2018]. En el sistema propuesto por los autores utilizaron tres conjuntos de características estilométricos más destacadas de los textos de las noticias del conjunto de datos. El primer conjunto consiste en características utilizadas en el estado del arte para la atribución de autoría, este conjunto esta subdividido en las categorías: cantidad (número de palabras, número de caracteres, número de oraciones), vocabulario (número de palabras únicas, promedio de sílabas por palabras, promedio de tamaño de oraciones) y legibilidad (facilidad de lectura de *Flesch*, puntaje de *Kincaid* de *Flesch*, Índice *Smog* , Índice *Coleman liau*, Índice de legibilidad automatizada, puntaje de legibilidad Dale Chall, diversidad del léxico). El segundo conjunto de características son utilizadas para la detección de mentiras, de igual forma esta subdividida en las categorías: cantidad (número de palabras, número de caracteres, número de oraciones), vocabulario (número de palabras con más de 6 caracteres, promedio de sílabas por palabras), gramática (número de oraciones con menos de 10 palabras, número de oraciones con más de 15 palabras, número de palabras por oraciones, número de conjunciones, complejidad de las oraciones), incertidumbre (número de palabras de certeza, número de palabras tentativas, número de verbos modales), especificidad (número de adjetivos, número de adverbios, número de términos afectivos), inmediatez verbal (número de referencias a si mismo, número de referencias a grupos, número de pronombres en primera persona, número de pronombres en segunda persona, número de pronombres en tercera persona). El último conjunto de características basado en rasgos utilizados para la atribución de autoría en documentos cortos. Este conjunto también se encuentra subdividido en categorías: caracteres (número de caracteres, número de dígitos, número de letras, número de mayúsculas, número de minúsculas, número de espacios en blanco, frecuencia de caracteres especiales), palabras (número de palabras, número de palabras con menos de 4 caracteres, número de caracteres en palabras, promedio de tamaño de oraciones en base a cantidad de caracteres, promedio de tamaño de oraciones en base a cantidad de palabras, número de palabras diferentes, frecuencia de palabras que solo ocurren una vez, frecuencia de palabras que solo ocurren dos veces, frecuencia de palabras de diferente longitud), sintáctica (puntuación y frecuencia de palabras funcionales), estructural (número de líneas, número de oraciones, número de párrafos, número de oraciones por párrafos, número de caracteres por párrafos, número de palabras por párrafos, número de palabras de saludo, número de contenido citado, número de *urls*) y contenido (frecuencia de las palabras de contenido).

3.4. Métodos basados en aprendizaje profundo

Varios trabajos de investigación presentes en el estado del arte utilizan modelos de aprendizaje profundo para la detección de noticias falsas. En estos modelos el contenido de las noticias es primeramente incrustado a nivel de palabras y luego esta incrustación es procesada por una red neuronal. Por ejemplo redes neuronales convolucionales (*CNN*), redes neuronales recurrentes como *Long short term memory (LSTM)*, *Bidirectional long short term memory (BI-LSTM)* o el uso de arquitecturas *transformers* como *BERT*. La principal ventaja de utilizar modelos de aprendizaje profundo sobre los enfoques clásicos existentes basados en características es que estos modelos son capaces de identificar el mejor conjunto de características que describen los textos por sí solos.

En Wang [2017] se propuso un modelo híbrido basado en redes neuronales convolucionales que supera a otros modelos tradicionales de aprendizaje automático, también comparó el rendimiento de modelos *SVM*, *LR*, *Bi-LSTM* y Modelos de *CNN* sobre su conjunto de datos propuesto llamado *"LIAR"*. Por otra parte, en Rashkin et al. [2017] se realizó un análisis de las características estilométricas de un texto poco confiable y se presentó un modelo *LSTM* que obtuvo buenos resultados.

Otro de los trabajos que utilizan técnicas de aprendizaje profundo fue presentado por Kumar et al. [2020] en el cual realizan una comparación entre múltiples arquitecturas basadas en redes neuronales profundas. Entre las que utilizaron se encuentran las redes neuronales convolucionales (*CNN*) y las redes neuronales recurrentes (*RNN*) en particular la *LSTM*.

En la actualidad, los modelos de lenguaje pre-entrenados como *BERT* y *ELMo* están recibiendo gran atención en diferentes tareas del procesamiento del lenguaje natural referentes a la clasificación de textos. Por ejemplo en Adhikari et al. [2019] y González-Carvajal and Garrido-Merchán [2020] se presentan comparaciones de *BERT* respecto a los métodos de aprendizaje de máquina tradicionales. En Kaliyar et al. [2021] el autor propone el modelo *FakeBERT* el cual es una combinación de *BERT* y tres bloques paralelos de *1d-CNN* que posee diferentes capas convolucionales de diferentes tamaño de *kernel* con filtros para un mejor aprendizaje.

En el estado del arte se han presentado algunos trabajos referentes a tareas de clasificación de textos en el cual los autores incluyen características estilométricas para mejorar la clasificación. Por ejemplo en Majumder et al. [2017] los autores presentan un método para extraer rasgos de personalidad de los escritores de ensayos de flujo de conciencia, para esto los autores utilizan una red neuronal convolucional (*CNN*). Para la representación vectorial de los ensayos, es concatenado al vector resultante de la operación de agregación aplicada a todos los vectores de las oraciones que representan el ensayo, con las características de *Mairesse*, las cuales están conformadas por el conteo de palabras y el número promedio de palabras por oración, así como el número total de pronombres, verbos en tiempo pasado, verbos en tiempo presente, verbos en tiempo futuro, letras, fonemas, sílabas, preguntas y afirmaciones en el documento, entre otras. Todas estas características son extraídas de los textos directamente en la etapa de preprocesamiento. En particular esta técnica de extensión del vector les permitió mejorar el rendimiento del método de detección de personalidad.

En este trabajo nos enfocaremos en los métodos basados en el estilo utilizando aprendizaje profundo.

3.5. Resumen

En este capítulo se describe la tarea de detección de noticias falsas y se resumen las técnicas y métodos que han sido desarrollados hasta el momento para abordar dicha tarea, la cual

constituye una subtarea en el área de la clasificación de textos.

En la Sección 3.2 se describen las diferentes perspectivas presentadas en el estado del arte para la resolución de la tarea de detección de noticias falsas. Entre las diferentes perspectivas abordadas se encuentran los métodos basados en el conocimiento, métodos basados en el origen, métodos basados en la propagación de la noticia y por último los métodos basados en el estilo.

Seguidamente se profundizó en los métodos basados en el estilo, el cual es la base de la metodología propuesta en el presente trabajo de investigación. En las secciones 3.3 y 3.4 se presentaron múltiples ejemplos de métodos basados en aprendizaje automático y métodos basados en aprendizaje profundo presentes en el estado del arte para la detección de noticias falsas basadas en el estilo.

Capítulo 4

Metodología propuesta

"Imagination is the discovering faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science."

Ada Lovelace

4.1. Introducción

En este capítulo se describe el marco metodológico de este trabajo de tesis. En la Sección 4.2 se describe la arquitectura basada en aprendizaje de máquina combinada con características estilométricas que se utilizarán a modo de *baselines* para comparar con los resultados de la arquitectura propuesta. En la Sección 4.3 se describe la arquitectura basada en aprendizaje profundo propuesta en este trabajo de tesis.

4.2. Arquitectura *baseline*

Como *baselines* en este proyecto se desarrollaron arquitecturas basadas en aprendizaje supervisado combinado con características estilométricas para la detección de noticias falsas. En particular estas arquitecturas están conformadas por algoritmos de aprendizaje de máquina como son el clasificador bayesiano ingenuo multinomial, máquinas de vectores de soporte y regresión logística.

A modo general la arquitectura *baseline* está compuesta por varios módulos y múltiples capas. Estos módulos son: módulo de procesamiento de noticias, módulo de representación vectorial, módulo de selección de características lingüísticas, módulo de combinación, módulo de aprendizaje automático y por último el módulo de predicción.

En la Figura 4.1 se ilustra la arquitectura utilizada como *baseline*.

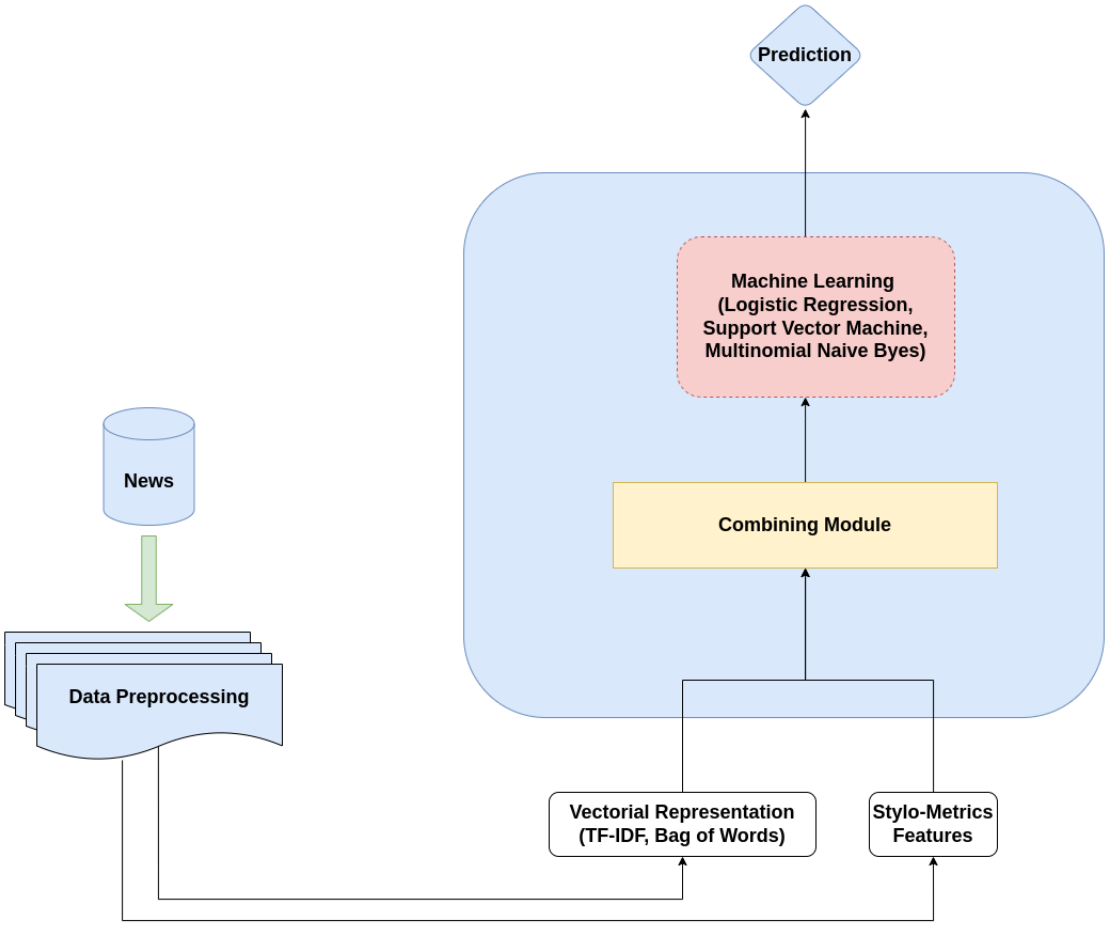


Figura 4.1: Arquitectura ML-Style Fake.

Luego del procesamiento de los textos de las noticias presentes en el corpus, procedimos a la obtención de la representación vectorial de dichos textos, en particular realizamos experimentos con el método de representación de textos bolsa de palabras y el esquema de pesado *TfIdf*. Una vez obtenida la representación vectorial de los textos procedimos a la obtención de los modelos de aprendizaje supervisado. En particular fueron utilizados los algoritmos *MultinomialNB, Support Vector Classifier y Logistic Regression*.

Para la selección de cada uno de los algoritmos mencionados anteriormente, *MultinomialNB, Support Vector Classifier y Logistic Regression* fue necesario realizar un proceso de ajuste de hiper parámetros, en particular se utilizó *GridSearchCV (GridSearch Cross Validation)* para el ajuste de nuestros algoritmos. Esta técnica consiste en una búsqueda exhaustiva de los valores de los parámetros de un algoritmo. A modo general *GridSearchCV* proporciona una forma de probar varios valores para los hiper-parámetros. Permite realizar una validación cruzada de muchas combinaciones diferentes de hiper-parámetros para encontrar el conjunto de hiper-parámetros que produce la mejor puntuación en base a una métrica de evaluación establecida. En este trabajo como métrica de puntuación fue seleccionada F_1.

4.3. Descripción de la arquitectura *Deep Style Fake*

En este trabajo de tesis se propone la arquitectura *Deep Style Fake*, la cual combina el poder de las arquitecturas basadas en aprendizaje profundo como son las redes neuronales convolucionales y las redes neuronales recurrentes como es el caso de *LSTM* junto con características estilométricas que se encuentran presentes en las noticias. La arquitectura *Deep Style Fake* está conformada por varios módulos y múltiples capas. Entre los módulos se encuentra el módulo de preprocesamiento de los textos de las noticias, el módulo de generación de los *embeddings* ya sean de palabras o de oraciones, otro de los módulos está asociado al cómputo y normalización de las características estilométricas presentes en los textos de las noticias. Por último, la arquitectura contiene un módulo de combinación, en el cual son añadidas las características estilométricas. Además, la arquitectura está conformada por una capa lineal, una capa *dropout* y una capa *softmax*.

El diagrama asociado a la arquitectura propuesta se puede visualizar en la Figura 4.2.

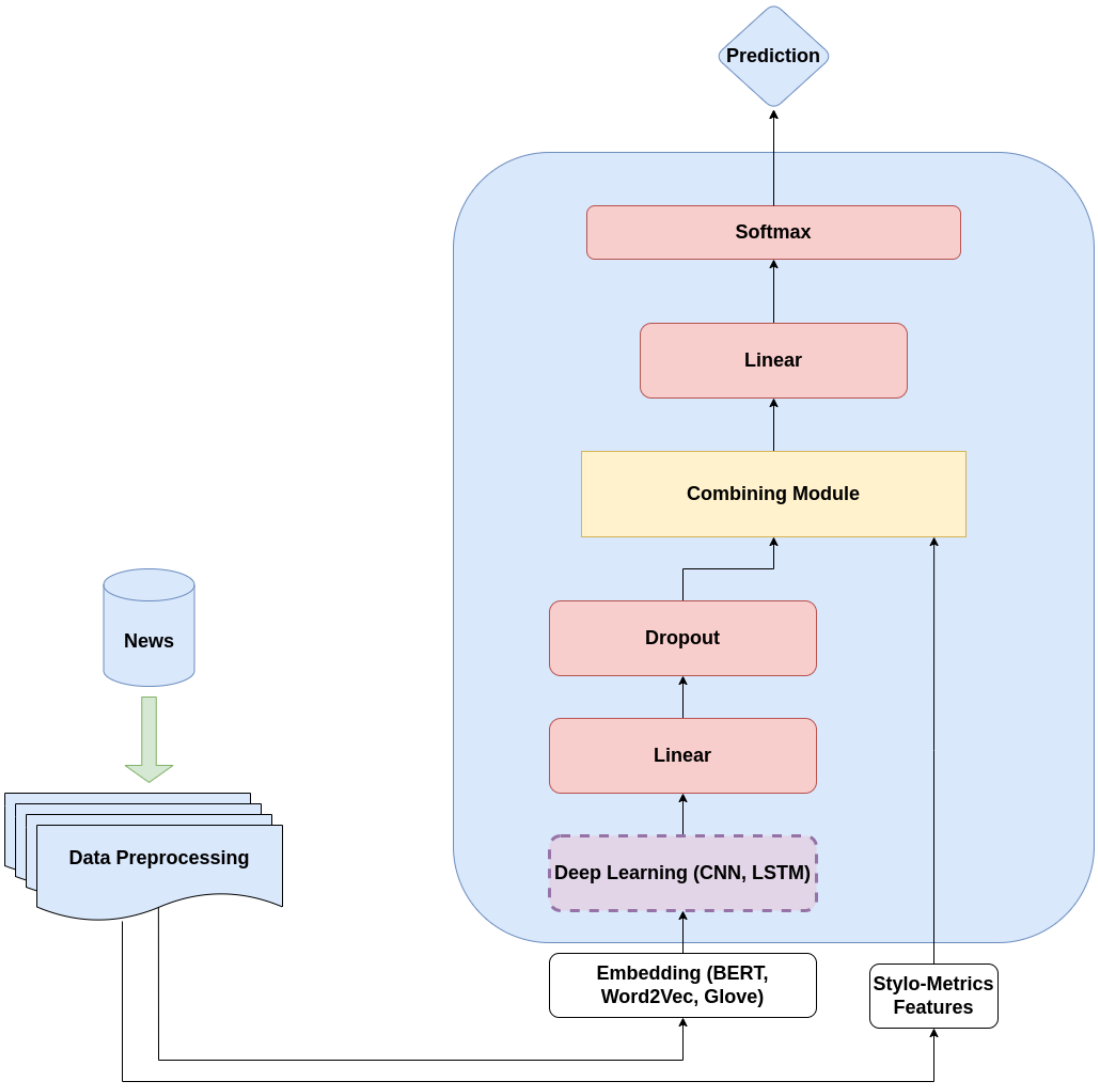


Figura 4.2: Arquitectura Deep Style Fake

4.3.1. Módulo de preprocesamiento

La primera fase de la arquitectura consiste en el módulo de preprocesamiento de las noticias. Durante esta fase se siguieron algunos pasos, los mismos consisten en la tokenización, conversión de caracteres a minúsculas, la eliminación de *stopwords*, la eliminación de signos de puntuación, eliminación de palabras que no pertenecen al lenguaje español y la corrección ortográfica.

El primer paso que se llevó a cabo consistió en la tokenización de los textos. Para esto se utilizó la biblioteca de *Python* llamada *Spacy* [1] la cual permite la construcción de aplicaciones basadas en procesamiento de lenguajes natural. El principal motivo de la selección de esta biblioteca estuvo basado en que la misma a diferencia de otras bibliotecas de *Python* presenta soporte para el lenguaje español. En particular esta biblioteca contiene modelos de lenguaje pre-entrenados en el lenguaje español con corpus de noticias en el lenguaje español. Los modelos que contienen son los siguiente: *es_core_news_sm*, *es_core_news_md*, *es_core_news_lg* y *es_dep_noticias_trf* [2].

Una vez tokenizados los textos de las noticias, como segundo paso, se procedió a la conversión de mayúsculas a minúsculas. La principal motivación de la realización de este preprocesamiento consistió en disminuir el tamaño del vocabulario. Debido a que, al tener dos palabras, por ejemplo, *"Covid"* y *"covid"* al ser representadas mediante vectores ambas tendrían dos representaciones completamente diferentes, a pesar de poseer el mismo significado.

Como tercer paso, se procedió a la eliminación de palabras funcionales para la cual se utilizó la lista de *stopwords* que contiene la biblioteca de *Spacy* para el lenguaje español (*spacy.lang.es.stop_words.STOP_WORDS*). En esta lista incluimos algunas palabras funcionales que detectamos manualmente su existencia en los textos. Estas son: 'a', 'o', 'versus', 'vía', 'acá', 'allá', 'atrás', 'cuan', 'etc', 'sr', 'sra', 'sres', 'y', '*number*'. En particular el último *token* añadido a la lista de *stopwords* "*number*" representa la presencia de un valor numérico en los textos de las noticias. A modo general este *token* en el corpus utilizado se encuentra entre las diez palabras con más repeticiones.

El cuarto paso consistió en la eliminación de los signos de puntuación excepto los signos de exclamación ("!,¡"), signos de interrogación ("¿?"), símbolo de número ("#") y símbolo de arroba ("@"). Estos signos no fueron eliminados debido a que como se menciona en la Sección 4.3.4, entre las características identificadas en las noticias falsas se encuentra el uso de la exageración en la información por medio del uso de mayúsculas y signos de exclamación por parte de los autores y el uso excesivo de etiquetas (#, @).

El quinto paso consistió en la eliminación de palabras que no pertenecían al idioma español. Durante una revisión del corpus se detectaron algunas palabras en el cuerpo de las noticias que no pertenecían al lenguaje español como se muestra en la Tabla 4.1, una noticia con caracteres propios del mandarín.

El sexto y último paso de la fase de preprocesamiento de las noticias consistió en la corrección de errores ortográficos, para la cual se utilizó la biblioteca de *Python Pyspellchecker* [3] la cual admite varios idiomas, dentro de los que figura el lenguaje español.

[1] https://spacy.io/
[2] https://spacy.io/models/es
[3] https://pyspellchecker.readthedocs.io/en/latest/#pyspellchecker

Id	Categoría	Tópico	Fuente	Texto
421	Fake	Politics	El Ruinaversal	EXTRANJEROS QUE TENGAN MÁS DE ... un ciudadano chino, comentó: "我知道墨西哥存在這種情況,這就是為什麼我會投票給奧布拉多,他是最好"

Tabla 4.1: Muestra de una noticia con palabras que no pertenecen al lenguaje español

4.3.2. Módulo de generación de *embeddings*

La segunda fase de la arquitectura consiste en la obtención de los *word embeddings*. A modo general, para este trabajo nos decantamos por la selección de *word embeddings* como técnica de representación de las palabras de las noticias. A diferencia de otras técnicas existentes en el estado del arte, como son *one-hot encoding*, n-gramas o bolsa de palabras. Entre las diferentes ventajas del uso de *word embeddings* respecto a técnicas tradicionales, se encuentra que los mismos representan vectores multidimensionales que intentan capturar las relaciones entre las palabras y el significado de las palabras Incitti et al. [2023].

En esta fase se evaluaron distintas técnicas de obtención de *word embeddings* como son: *BERT*, *Word2Vec* y *GloVe*. Como se mencionó en el Capítulo 2 los *word embeddings* representan una técnica del procesamiento del lenguaje natural que consiste, básicamente, en asignar un vector a cada palabra. Este vector guarda información semántica, lo que permite que pueda ser asociado o no a otros vectores, que a su vez representan otras palabras, según distintos contextos gramaticales. En este sentido los *word embeddings* representan una solución efectiva para codificar tanto la semántica como la relación de las palabras entre sí.

Entre las ventajas que presenta *Word2Vec* en comparación con técnicas como bolsa de palabras se encuentra que mediante *Word2Vec* se conserva el significado semántico de las diferentes palabras en un documento. Es decir, la información del contexto no se pierde. Otra de las ventajas más representativas de *Word2Vec* consiste en que el tamaño del vector generado es pequeño. En este modelo no se necesitan vectores dispersos, como es el caso de la representación de bolsa de palabras. Además, cada dimensión del vector incrustado contiene información sobre un aspecto de la palabra.

Por otra parte, el modelo *GloVe* a diferencia de *Word2Vec* agrega un significado más práctico a los vectores de palabras, al considerar las relaciones existentes entre frases, en lugar de relaciones entre palabra y palabra.

A pesar, de que los dos modelos anteriores permiten obtener una mejor representación de las palabras contenidas en los textos, los mismos presentan algunas desventajas. Tanto *Word2Vec* como *GloVe* no resuelven problemas como el aprendizaje de la representación de palabras que se encuentran fuera del vocabulario en el cual fueron entrenados. Además, otras de las desventajas radica en como separar algunos pares de palabras opuestas. Por ejemplo, “bueno” y “malo” generalmente se ubican muy cerca uno del otro en el espacio vectorial, lo que puede limitar el rendimiento de los vectores de palabras en algunas tareas de procesamiento de lenguaje natural como es el caso del análisis de sentimientos.

A diferencia de estas dos técnicas anteriores, los *BERT embeddings* ofrecen una ventaja sobre modelos como *Word2Vec*. A pesar de que en el modelo *Word2Vec* cada palabra tiene una representación fija, independientemente del contexto en el que aparece la palabra, *BERT* produce representaciones de palabras que están dinámicamente informadas por las palabras que las rodean, es decir, toma en cuenta información del contexto de las palabras.

4.3.2.1. *BERT*

Para la obtención de los *BERT embeddings* es necesario, una vez preprocesados los textos, transformarlos al formato que *BERT* espera, para esto es necesario agregar los *tokens* [*CLS*] y [*SEP*]. Para obtener esta transformación se puede utilizar la clase *BertTokenizer* que provee *Huggin Face*[4]. El *token* [*CLS*] representa el primer *token* que debe ser añadido al principio a cada secuencia, que significa *token* de clasificación. El *token* [*SEP*] es el encargado de notificarle a *BERT* qué *token* pertenece a qué secuencia.

En el modelo *BERT* solo se pueden introducir como tamaño máximo 512 *tokens*. Si los *tokens* en una secuencia son inferiores a 512, se puede utilizar el *token* de relleno [*PAD*] para llenar los espacios de los *tokens* no utilizados. En caso de que la cantidad de *tokens* en una secuencia (texto, párrafo, oración, etc.) sea superior a 512, existen varias estrategias a seguir para evitar tener que truncar los textos a solo 512 *tokens* y con esto perder información relevante. La primera alternativa que tuvimos en cuenta fue dividir los textos en fragmentos que no contuvieran más de 512 *tokens* cada uno. En la Figura 4.3 se puede visualizar un ejemplo, en el cual se tiene un texto que al ser procesado que contiene 1361 *tokens* se puede dividir en tres tensores más pequeños. Los dos primeros compuestos por 512 *tokens* cada uno, y el tensor final compuesto por los 337 *tokens* restantes. Una vez que se tienen los fragmentos y se encuentran transformados al formato de entrada de *BERT*, utilizamos como entrada del modelo los fragmentos anteriores.

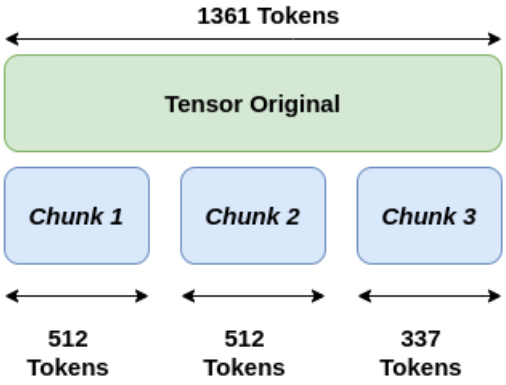


Figura 4.3: Partición de un texto en múltiples tensores

El modelo *BERT* luego genera un vector de *embedding* de tamaño 768 en cada uno de los *tokens*. Estos vectores pueden ser utilizados como entrada para diferentes tipos de aplicaciones de NLP, ya sea clasificación de texto, predicción de la siguiente oración, reconocimiento de entidades nombradas (*NER*) o respuesta a preguntas. En particular en este trabajo de tesis se utiliza para la tarea de clasificación de textos, específicamente la clasificación de noticias falsas.

A modo general para la tarea de clasificación de textos, se puede utilizar la salida del vector *embedding* del *token* especial [*CLS*], el cual tiene tamaño 768 como entrada a algún clasificador específicamente este vector es una representación *embedding* de la oración. En caso de desear trabajar utilizando *word embeddings*, existen algunas alternativas para su generación. Por ejemplo, en concatenar las últimas cuatro capas, resultando en un vector de palabra por cada *token*. En este caso cada vector posee una longitud de $4 \times 768 = 3072$. Otra alternativa consiste

[4] https://huggingface.co/docs/transformers/main/en/model_doc/bert#transformers.BertTokenizer

en la suma de las últimas cuatro capas. Debido a que las diferentes capas que componen la arquitectura de *BERT* codifican diferentes tipos de información, es necesario probar diferentes estrategias de agrupación para la generación de los *embeddings* en dependencia de la tarea a realizar.

4.3.2.2. *Word2Vec*

El algoritmo *Word to Vector (Word2Vec) word embeddings* puede ser utilizado mediante un mapeo al estilo llave-valor entre las palabras y los vectores correspondientes. Es decir, la entrada es una palabra. Los vectores generados por este modelo están compuesto por 300 valores flotantes en el rango $[-1, 1]$para cada palabra. Dado que este modelo aprende los *embeddings* a nivel de palabras, en caso de que una palabra no se encuentre en el vocabulario con el cual fue entrenado el modelo, no es posible obtener el *embedding* correspondiente a la misma.

Durante la fase de experimentación utilizamos *Spanish Billion Word Corpus and Embeddings*[5] para la obtención de los *embeddings* de las palabras.

Spanish Billion Word Corpus and Embeddings está compuesto por un corpus no anotado de la lengua española de aproximadamente 1.500 millones de palabras, compilado a partir de diferentes corpus y recursos de la web; y un conjunto de vectores de palabras, creados a partir de este corpus utilizando el algoritmo *Word2Vec*.

4.3.2.3. *GloVe*

Como se mencionó en el Capítulo 2, *GloVe* (Global Vectors) es una técnica *word embeddings* captura estadísticas globales y estadísticas locales de un corpus, para generar vectores de palabras. Al igual que *Word2Vec* puede ser utilizado mediante un mapeo al estilo llave-valor entre las palabras y los vectores correspondientes. Además, en caso de que una palabra no se encuentre en el vocabulario con el cual fue entrenado el modelo, no es posible obtener el *embedding* correspondiente a la misma. En este trabajo de investigación fue utilizado *GloVe Spanish Billion Words Corpora* [6], estos vectores están compuestos por 300 valores.

4.3.3. Módulo de aprendizaje profundo

Otro de los módulos que componen la arquitectura, es el módulo de aprendizaje profundo. Este módulo puede ser implementado mediante el uso de redes neuronales convolucionales (*CNN*) o mediante en uso de redes neuronales recurrentes como es el caso de *LSTM*.

4.3.3.1. Redes neuronales convolucionales *(CNN)*

Como se mencionó en la Sección 2.3.5 las redes neuronales convolucionales, son redes neuronales artificiales profundas donde las conexiones entre los nodos no forman ciclos. Estas redes se usan generalmente en el área referente a visión por computadora. Sin embargo, también han mostrado resultados prometedores cuando se aplican a varias tareas del área del procesamiento del lenguaje natural Jacovi et al. [2018].

A modo general, cuando se utilizan estas redes con datos textuales, el resultado de cada convolución se dispara cuando se detecta un patrón especial entre las palabras que conforman el texto. Al variar el tamaño de los núcleos y concatenar sus resultados, se permite detectar

[5] https://crscardellino.ar/SBWCE/
[6] https://github.com/dccuchile/spanish-word-embeddings

patrones de múltiples tamaños, es decir, 2, 3 o varias palabras adyacentes. Estos patrones pueden ser expresiones como n-gramas de palabras, los cuales pueden ser identificados por la red independientemente de la posición en el texto. Como se mencionó en la Sección 2.3.5, en estas redes se realizan una serie de operaciones específicas de este tipo de arquitecturas, estas son operaciones de convolución y agrupación. Las operaciones de convoluciones y agrupación pierden información sobre el orden local de las palabras, a su vez la operación de agrupación reduce la dimensionalidad de salida, pero mantiene la información más relevante. En estas arquitecturas se puede asociar que cada filtro detecta una característica específica, como detectar si la oración contiene, por ejemplo, una negación. Si esta frase aparece en algún lugar de la oración, el resultado de aplicar el filtro a esa región se obtiene un valor grande, en cambio en otras regiones, se obtiene un valor pequeño. Al realizar la operación de agrupación *Max-Pooling*, se conserva información sobre si la característica apareció o no en la oración, pero se pierde información sobre dónde apareció exactamente.

Para el uso de la red neuronal convolucional primeramente obtuvimos los *words embeddings*, para esto se seleccionó entre las tres siguientes técnicas (*Word2Vec*, *GloVe* y *BERT*). Una vez obtenidos los *embeddings* en representación de los textos de las noticias, estos son utilizados como entrada a la red neuronal en la cual son deslizados los filtros o *kernels* sobre estos *embeddings* para encontrar las convoluciones, y estas a su vez son reducidas dimensionalmente de forma que se reduce la complejidad y los cálculos por la capa de *Max Pooling*

Para este trabajo de investigación implementamos los modelos de redes neuronales convolucionales propuestos en el artículo *Convolutional Neural Networks for Sentence Classification* por Chen [2015]. Los cuatros modelos para la clasificación de oraciones son los siguientes:

- ***CNN-static***: Los *embeddings* referentes a todas las palabras se inicializan con vectores pre-entrenados, en nuestro caso los vectores pueden ser *Word2Vec embeddings*, *GloVe embeddings* o *BERT embeddings*. En este modelo los *embeddings* se mantienen estáticos y solo se aprenden los demás parámetros del modelo.

- ***CNN-non-static***: Los *embeddings* referentes a todas las palabras se inicializan con vectores pre-entrenados, en nuestro caso los vectores pueden ser *Word2Vec embeddings*, *GloVe embeddings* o *bert embeddings*. En este modelo los *embeddings* son ajustados (*fine-tuned*)

- ***CNN-multichannel***: En este modelo se tienen dos conjuntos de vectores de palabras. Cada conjunto de vectores se trata como un "canal" y cada filtro se aplica a ambos canales, en este caso los gradientes se propagan hacia atrás solo a través de uno de los canales. Por lo tanto, el modelo es capaz de ajustar un conjunto de vectores manteniendo el otro estático. Ambos canales se inicializan con *Word2Vec embeddings*, *GloVe embeddings* o *BERT embeddings*.

En la Figura 4.4 se puede visualizar la arquitectura de la red neuronal convolucional incluída en el módulo de aprendizaje profundo, basada en el modelo *CNN-multichannel*.

4.3.3.2. *Long Short Term Memory (LSTM)*

Como se mencionó en la Sección 2.3.4 las redes neuronales *LSTM (Long Short Term Memory)* como su nombre lo indica son redes con memoria a largo y corto plazo. Estas son redes

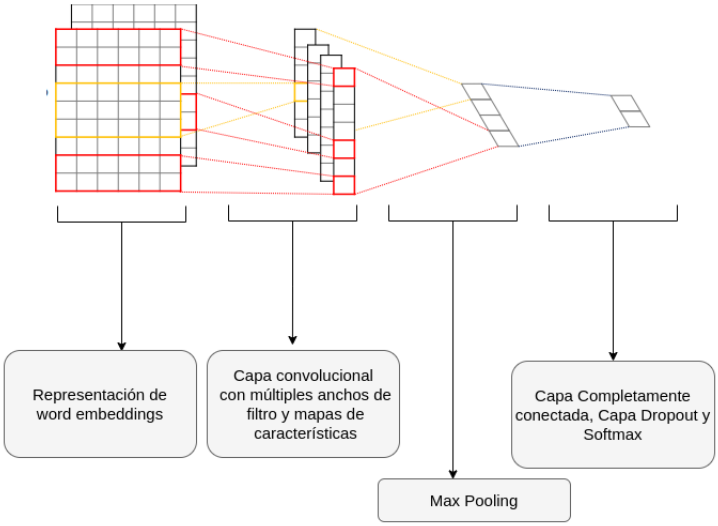


Figura 4.4: Arquitectura de una red neuronal convolucional basada en el modelo *CNN-multichannel*

neuronales recurrentes con mejor funcionamiento en términos de memoria en comparación con redes neuronales recurrentes tradicionales.

A modo general, la principal razón para la selección de esta arquitectura en la tarea de clasificación de textos consiste en su capacidad de memorizar información. En este tipo de arquitecturas se puede tener una cadena conformada por múltiples palabras para determinar la clase a la que pertenece. A diferencia de arquitecturas no neuronales que son entrenadas tomando como entrada múltiples palabras separadas que son solo palabras que no tienen un significado real como oración, y mientras predicen la clase, darán la salida según las estadísticas y no según el significado Tarwani and Edem [2017]. Eso significa que cada palabra se clasifica en una de las categorías.

La arquitectura basada en *LSTM* incluida en el módulo de aprendizaje profundo está conformada de la siguiente manera. La primera capa representa la capa *embedding*, en esta capa las palabras están representadas mediante una representación vectorial densa. Para esta representación se seleccionó entre las tres siguientes técnicas (*Word2Vec*, *GloVe* y *BERT*). La siguiente capa es una capa *LSTM*. Seguidamente tenemos una capa *dropout* para regular la red y evitar cualquier tipo de sesgo. Por último, como capa final tenemos una función lineal como clasificador que tiene n celdas de salida, donde n representa la cantidad de clases, en este caso $n = 2$ representa como salida la clase falsa y la clase verdadera de una noticia.

4.3.4. Módulo de obtención de características estilométricas

Otro de los módulos que conforman la arquitectura es el módulo de obtención de características estilométricas. Al realizar un análisis manual de los textos de las noticias tanto falsas como verdades presentes en ambas versiones de los corpus por medio de una perspectiva lingüística, se detectó que existen varios patrones lingüísticos en las mismas. En las noticias verdaderas se identificó que la información suele presentarse de forma concisa y mediante un lenguaje fácil de entender para el público. Además, la información se expone al lector en tercera persona, por lo que el autor no se incluye en el público ni ofrece su opinión sobre la noticia.

En estas noticias los signos de puntuación son utilizados correctamente a modo general y fue difícil encontrar errores ortográficos o errores de redacción en los textos. Esto puede ser debido a que estas noticias son sometidas a una ardua revisión antes de ser publicadas.

Por otra parte, en las noticias falsas se identificó que predomina el uso de la exageración en la información por medio del uso de mayúsculas y signos de exclamación por parte de los autores. También se identificó que los autores de las noticias tienen tendencia a utilizar escritura en primera persona del singular y en primera persona del plural. Es decir, los autores se incluyen dentro del público y no proporcionan la información en tercera persona. Otra de las características identificadas durante la revisión es la presencia de errores ortográficos y errores gramaticales, el uso de etiquetas (#, @) dentro de los textos de las noticias. Además, los textos de las noticias presentan repetición de la información, tanto como párrafos y oraciones completas.

Basado en la información anterior, una vez preprocesados los textos de las noticias se procedió a contar la cantidad de caracteres en mayúsculas por cada una de las noticias, la cantidad de palabras en mayúsculas, la cantidad de símbolos (?, ¿,¡#, @) presentes en la noticia y la cantidad de palabras con errores ortográficos. Luego de contar dichos valores se procedió al escalado de las nuevas características mediante la técnica de normalización *Min-Max*. Mediante esta técnica los valores son escalados al rango $[0, 1]$.

En la Ecuación 4.1 se puede visualizar la fórmula aplicada para el escalado de las nuevas características.

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4.1}$$

4.3.5. Módulo de combinación

El módulo de combinación consiste en la concatenación del vector resultante luego de ser aplicado a una capa de aprendizaje profundo basada en modelos *CNN* o *LSTM*, luego aplicado a una capa lineal y por último aplicado a una capa *dropout*, este vector resultante es concatenado con los valores normalizados referentes a las características estilométricas computadas de los textos de las noticias. Estas características estilométricas consisten en la cantidad de caracteres en mayúsculas, cantidad de palabras en mayúscula, cantidad de símbolos (?, ¿,¡#, @) y cantidad de palabras con errores ortográficos.

4.3.6. Módulo de clasificación

La arquitectura propuesta en este trabajo de tesis tiene como último módulo, el módulo de clasificación. Una vez detectados todos los patrones relevantes presentes en las noticias en el módulo de aprendizaje profundo y seguidamente el resultado obtenido en dicho módulo combinarlo con las características lingüísticas presentes en los textos de dichas noticias, se procedió al módulo de clasificación en el cual se determina si las noticias pertenecen a la clase falsa o verdadera.

Este módulo está compuesto por tres capas, la primera una capa lineal, la segunda es una capa *dropout* y por último una capa *softmax*. En las siguientes secciones se presentan las características relevantes de las capas anteriormente mencionadas.

4.3.6.1. Capa lineal

Esta capa es utilizada para aplicar una transformación lineal a las instancias de entrada. Dicha transformación lineal se describe en la ecuación 4.2.

$$y = xA^T + b \tag{4.2}$$

En la Ecuación 4.2, la variable x representa el vector de entrada al cual se le desea aplicar la transformación lineal, la variable A representa la matriz de pesos asociados a la variable de entrada x y la variable b representa el sesgo asociado a la variable de entrada.

4.3.6.2. Capa *dropout*

Una de las problemáticas que presentan las redes neuronales durante el entrenamiento utilizando conjuntos de datos relativamente pequeños, consiste en el sobreajuste a los datos de entrenamiento. Esto tiene el efecto de que el modelo aprende el ruido estadístico en los datos de entrenamiento, lo que da como resultado un rendimiento deficiente cuando el modelo se evalúa en datos nuevos, por ejemplo, un conjunto de datos de prueba. El error de generalización aumenta debido al sobreajuste.

El *dropout* es un método de regularización que aproxima el entrenamiento de una gran cantidad de redes neuronales con diferentes arquitecturas en paralelo. Durante el entrenamiento, una cierta cantidad de salidas de una capa en particular se ignoran aleatoriamente o se "eliminan". Esto tiene el efecto de hacer que la capa parezca y sea tratada como una capa con un número diferente de nodos y conectividad a la capa anterior.

El *dropout* se puede usar con la mayoría de los tipos de capas, como capas densas totalmente conectadas, capas convolucionales y capas recurrentes. En nuestra arquitectura la capa *dropout* se encuentra posicionada luego de la salida del módulo de aprendizaje profundo. Para el desarrollo de la arquitectura fue utilizada la implementación de *dropout* incluida en la biblioteca *PyTorch*. En la implementación presente en *PyTorch* durante el entrenamiento, se establece a cero aleatoriamente algunos de los elementos del tensor de entrada con probabilidad p utilizando muestras de una distribución de Bernoulli [7]. En este caso las salidas de la capa son escaladas por el factor $\frac{1}{1-p}$.

4.3.6.3. Capa *softmax*

La arquitectura propuesta en la fase final contiene una capa compuesta por la función de activación *softmax*. La función de activación *softmax* devuelve las probabilidades de pertenecer una instancia a cada clase. A continuación, la ecuación mediante la cual puede ser calculada:

$$softmax(x_i) = \frac{\exp x_i}{\sum_j \exp x_j} \tag{4.3}$$

En la ecuación anterior la x representa los valores de las neuronas de la capa de salida. La función exponencial actúa como la función no lineal. Posteriormente estos valores se dividen por la suma de valores exponenciales con el fin de normalizar y luego convertirlos en probabilidades. Dado que las salidas son valores de probabilidades, los mismos se encuentran en el rango $[0, 1]$ y la suma de los mismos es 1.

[7] https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html

4.4. Resumen

En este capítulo se describió la metodología propuesta para el desarrollo del trabajo de tesis. En la Sección 4.2, se presentaron los métodos *baselines* utilizados para comparar con la arquitectura propuesta, los cuales están basados en algoritmos de aprendizaje de máquina como son el clasificador bayesiano ingenuo multinomial, máquinas de vectores de soporte y regresión logística. Seguidamente en la Sección 4.3 se describió en profundidad los módulos que componen la arquitectura *Deep Style Fake*. Estos módulos son: módulo de preprocesamiento, módulo de generación de *embeddings*, módulo de aprendizaje profundo, módulo de combinación y el módulo de clasificación.

Capítulo 5

Corpus

"Imagination is the discovering Faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science."

Ada Lovelace

En este capítulo se describirá a modo general el corpus que fue utilizado durante la fase de experimentación de este trabajo de investigación. En la primera sección se presenta la composición del corpus y en la segunda sección se presenta un análisis exploratorio de los datos que conforman el corpus.

5.1. *Spanish Fake News Corpus*

"The Spanish Fake News Corpus" Posadas-Durán et al. [2019], se encuentra disponible en GitHub [1]. Este corpus está compuesto por dos versiones, la Versión 1.0 (@MEXLEF 20) y la Versión 2.0 (FakeDeS Task @ Iberlef 2021).

La primera versión de este corpus es una colección de noticias compiladas a partir de varias fuentes *web*: sitios *web* de periódicos establecidos, sitios *web* de empresas de medios, sitios *web* especiales dedicados a validar noticias falsas, sitios *web* designados por diferentes periodistas como sitios que publican regularmente noticias falsas. Estas noticias fueron recopiladas entre enero y julio de 2018 y todas fueron redactadas en español mexicano, llegando a un total de 491 noticias verdaderas y 480 noticias falsas. La clasificación de las noticias se realizó siguiendo un proceso manual, en el cual se consideraba una noticia como verdadera de existir evidencia de que ha sido publicada en sitios confiables. Por otra parte, una noticia se consideró falsa de existir noticias en sitios confiables o sitio *web* especializados en la detección de contenido engañoso que la contradiga o no se encontró otra evidencia sobre la noticia además de la fuente. Para evitar sesgos temáticos las noticias se agrupan en los nueve tópicos: Ciencia, Deporte, Economía, Educación, Entretenimiento, Política, Salud, Seguridad y Sociedad. En la Tabla 5.1 se puede visualizar una muestra del contenido del corpus.

[1] http://github.com/jpposadas/FakeNewsCorpusSpanish

Id	1	34
Categoría	Fake	True
Tópico	Education	Society
Fuente	El Ruinaversal	CNN en español
Titular	RAE INCLUIRÁ LA PALABRA “LADY” EN EL DICCIONARIO DEL IDIOMA ESPAÑOL COMO DEFINICIÓN DE “MUJER PROBLEMÁTICA”	WhatsApp desde Alepo: “No hay escapatoria’
Texto	RAE INCLUIRÁ LA PALABRA “LADY” EN EL DICCIONARIO DEL IDIOMA ESPAÑOL COMO DEFINICIÓN DE “MUJER PROBLEMÁTICA” España .- El presidente de la Real Academia Española (RAE) ...	WhatsApp desde Alepo: “No hay escapatoria” El pasado miércoles *NUMBER* de septiembre aterrado por el diluvio de bombas que se abatía sobre Alepo, Karam al-Masri envió por WhatsApp varios mensajes a sus colegas de la agencia de noticias AFP en Beirut.
Link	http://www.elruinaversal.com/2017/06/10/rae-incluira-la-palabra-lady-en-el-diccionario-del-idioma-espanol-como-definicion-de-mujer-problematica/	https://www.proceso.com.mx/456982/whatsapp-alepo-escapatoria

Tabla 5.1: Muestra del contenido del corpus versión 1.0 (@MEXLEF 20)

Por otra parte, la segunda versión de este corpus está compuesto por pares de publicaciones falsas y verdaderas sobre diferentes eventos. Estas publicaciones fueron recopiladas en el periodo de noviembre de 2020 a marzo de 2021 y todas fueron recopiladas en español, llegando a un total de 572 publicaciones. Las publicaciones fueron extraídas de diferentes fuentes de la *web*: sitios *web* de periódicos y empresas de medios, y sitios *web* de verificación de datos. De igual forma a la primera versión del corpus para evitar sesgos temáticos las noticias se agruparon en los siete tópicos: Ciencia, Deporte, Política, Sociedad, COVID-19, Medio ambiente e Internacional. En la Tabla 5.2 se puede visualizar una muestra del contenido de la segunda versión del corpus.

Id	1	35
Categoría	True	Fake
Tópico	Covid-19	Sociedad
Fuente	El Economista	Alerta Digital
Titular	Covid-19: mentiras que matan	El siguiente paso de la ONU se llama canibalismo: Deberemos comer cadáveres, placentas y fetos para combatir el «cambio climático»
Texto	El control de la Covid-19 no es sólo un tema de médicos y el resto del personal sanitario y científico. desgracia o por fortuna, es un asunto esencialmente político que se decide por hombres y mujeres que se dedican a la política. De las creencias y opiniones	Es completamente real, e incluimos enlaces y documentación gráfica de los hechos que les contamos. Busquen ustedes mismos. La verdad está ahí fuera. Ahora ya sabemos cual será la degradación Comernos a nosotros mismos y a nuestros muertos. Un sistema perfecto. Nada se pierde, todo se recicla
Link	https://www.eleconomista.com.mx/opinion/Covid-19-mentiras-que-matan-20210212-0029.html	https://www.alertadigital.com/2019/09/08/el-siguiente-paso-de-la-onu-se-llama-canibalismo-deberemos-comer-cadaveres-placentas-y-fetos-para-combatir-el-cambio-climatico/

Tabla 5.2: Muestra del contenido del corpus Versión 2.0 (FakeDeS Task @ Iberlef 2021)

Ambas versiones de los corpus contienen la siguiente información:

- Categoría (Falso/Verdadero)
- Fuente (nombre del medio donde fue publicada)
- Titular de la noticia
- Texto de la noticia

- Enlace (*url* donde se publicó la noticia)
- Tópicos

5.2. Análisis exploratorio de datos

En esta sección presentamos un análisis exploratorio de los datos que conforman el corpus *"Spanish Fake News Corpus"* Aragón et al. [2020], Gómez-Adorno et al. [2021] el cual como se mencionó en la sección anterior está compuesto por dos versiones, la Versión 1.0 (@MEXLEF 20) y la Versión 2.0 (FakeDeS Task @ Iberlef 2021).

Un análisis exploratorio de datos es un proceso mediante el cual podemos familiarizarnos con los datos para de esta forma comenzar a formular hipótesis que posteriormente podamos comprobar al obtener resultados. En este proceso generalmente nos apoyamos de estadística descriptiva y visualizaciones. A continuación, se presentará una análisis a nivel de oraciones, un análisis a nivel de palabras y un análisis a nivel de caracteres de los textos que conforman las noticias del corpus. En el análisis a nivel de oraciones se evaluará la distribución de la longitud de las oraciones, es decir, la longitud mínima, la longitud máxima y la longitud promedio.

5.2.1. Versión 1.0 (@MEXLEF 20)

Como se mencionó en la sección anterior el corpus está conformado por 5 variables referentes a (Categoría, Tema, Fuente, Titular y Texto), 1 variable numérica el cual es el identificador de la noticia y una variable referente a la *url* que representa el enlace de la fuente de origen de la publicación de la noticia en la *web*. Además, las noticias se encuentran agrupadas en nueve tópicos: *Ciencia, Deporte, Economía, Educación, Entretenimiento, Política, Salud, Seguridad y Sociedad*. En la Tabla 5.3, se evidencia la distribución del corpus en base a los temas de las noticias.

Tópicos	Verdadero	Falso	Cantidad	Frecuencia
Ciencia	46	43	89	9.2 %
Deportes	66	58	124	12.8 %
Economía	24	19	43	4.4 %
Educación	10	12	22	2.3 %
Entretenimiento	70	78	148	15.2 %
Política	175	148	323	33.3 %
Salud	23	23	46	4.7 %
Seguridad	17	25	42	4.3 %
Sociedad	60	74	134	13.8 %
Total	491	480	971	-

Tabla 5.3: Distribución de las noticias respecto a los tópicos del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

En el artículo Posadas-Durán et al. [2019] donde se propone el corpus que nos encontramos analizando, realizan un análisis respecto a la superposición del vocabulario entre las noticias verdaderas y las falsas. Para dicho análisis consideraron los lemas y descartaron los *stopwords* y

calcularon la superposición dividiendo la intersección del vocabulario de las noticias verdaderas y falsas entre el vocabulario de todas las noticias.

En la Tabla 5.4 se puede visualizar el vocabulario perteneciente a las noticias falsas y verdaderas y la superposición de vocabulario [2]. Como se puede observar el mayor porcentaje de superposición del vocabulario entre noticias verdaderas y falsas se obtuvo para la categoría "Política" con un valor de 29.95 %. En promedio se tiene que la superposición del vocabulario entre noticias verdaderas y falsas a modo general es del 23.63 %.

Tópicos	Verdadera	Falsa	Superposición del Vocabulario
Ciencia	2543	1837	24.96 %
Deportes	2261	1804	25.23 %
Economía	1774	1011	17.60 %
Educación	917	800	18.41 %
Entretenimiento	3164	2523	27.68 %
Política	**5383**	**3844**	**29.95 %**
Salud	1858	1318	22.95 %
Seguridad	917	994	20.94 %
Sociedad	3255	2778	25.01 %
Promedio de Superposición	-	-	23.63 %

Tabla 5.4: Superposición del vocabulario respecto a los diferentes tópicos del *Spanish Fake News Corpus* Versión 1.0 (@MEX-LEF 20)

[2]Tabla tomada del artículo *"Spanish Fake News Corpus"*

.

Por otra parte, en la Tabla 5.5 se puede visualizar la distribución de las noticias respecto a la categoría (Verdadera, Falsa). Como se puede observar el corpus se encuentra balanceado, con un total de 491 noticias verdaderas que representan el 50.6 % del total de noticias y 480 noticias falsas que representan el 49.4 % del total de noticias.

Valor	Cantidad	Frecuencia (%)
Verdadero	491	50.6 %
Falso	480	49.4 %

Tabla 5.5: Distribución de noticias respecto a la clase falsa y verdadera del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

En la Tabla 5.6 se puede visualizar la distribución de las noticias respecto a los sitios *webs* donde fueron publicadas. Como se puede ver el sitio del cual se extrajo mayor cantidad de noticias fue "El Dizque". De este sitio fueron extraídas un total de 135 noticias lo que representan el 13.9 % del total de noticias que conforman el corpus.

Valor	Cantidad	Frecuencia (%)
El Dizque	**135**	**13.9 %**
El Ruinaversal	94	9.7 %
El País	80	8.2 %
El Universal	69	7.1 %
Excelsior	45	4.6 %
Hay noticia	39	4.0 %
Censura 0	36	3.7 %
Forbes	34	3.5 %
Huffpost	26	2.7 %
Proceso	25	2.6 %
Otros valores(132)	388	39.9 %

Tabla 5.6: Distribución de las noticias en dependencia del origen de la publicación del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

5.2.1.1. Análisis a nivel de caracteres de los textos:

A continuación, en la Figura 5.1 mediante un histograma se muestra el número de caracteres presentes en cada noticia. Esto nos proporciona una idea aproximada de la longitud de los textos que conforman las noticias. Dicho histograma muestra que los textos que representan las noticias oscilan entre 100 y 16000 caracteres y en general, entre 100 y 11000.

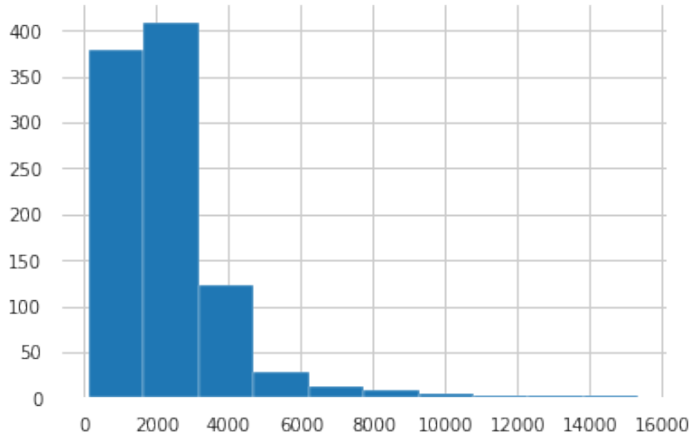


Figura 5.1: Longitud de los textos de las noticias del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

Análisis a nivel de palabras de los textos: Ahora, pasaré a la exploración de datos a nivel de palabra. En la Figura 5.2 se muestra la cantidad de palabras que aparecen en cada texto de las noticias. Como se puede visualizar el número de palabras en los textos de las noticias varía de 20 a 2500 y en su mayoría se encuentran entre 20 y 500.

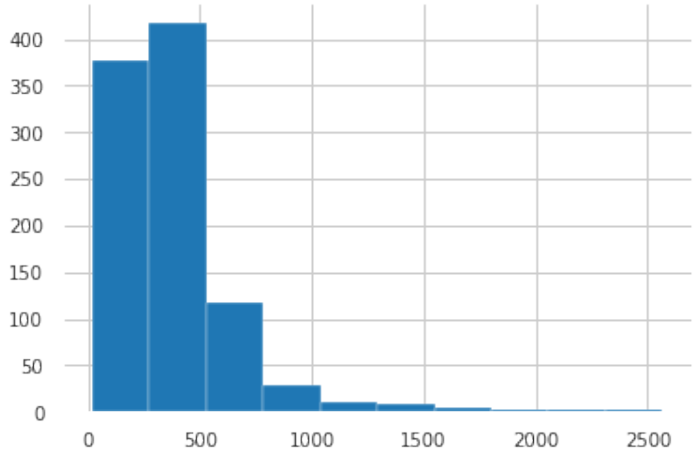


Figura 5.2: Distribución de cantidad de palabras por noticias del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20).

A continuación, verifiquemos la longitud promedio de las palabras en cada oración, para esto primeramente se realizó un procesamiento eliminando los *stopwords* de los textos de las noticias, este procesamiento tiene el objetivo de evitar sesgos en el gráfico, dado que los *stopwords* son palabras generalmente pequeñas y son las más frecuentes en los textos.

En la Figura 5.3 se muestra que la longitud promedio de las palabras por oración varía entre 6 y 9, siendo 7 la longitud más común.

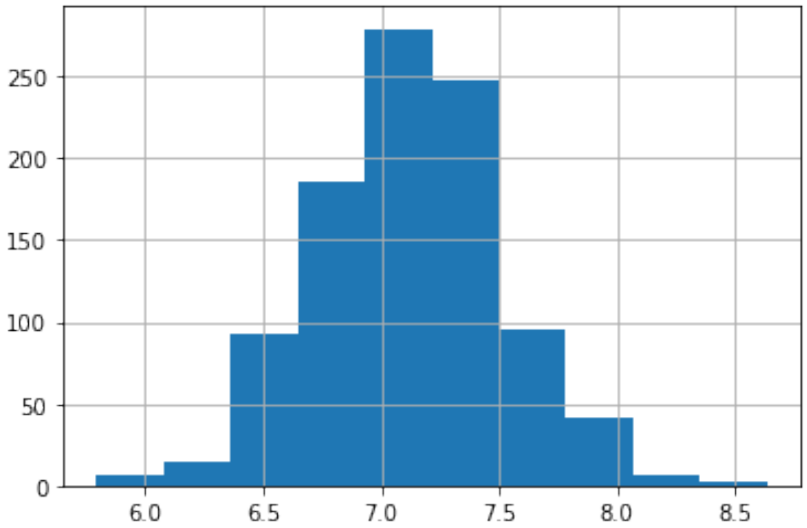


Figura 5.3: Longitud promedio de las palabras por oración del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

A continuación, en la Figura 5.4 se muestra las palabras con mayor frecuencia en las noticias verdaderas. Para la confección de las nubes de palabras realicé un preprocesamiento eliminando los stopwords dado que generalmente estas son las palabras más frecuentes en los textos. Como se puede apreciar en particular las palabras más frecuentes son "país, año, méxico, gobierno, caso, lópez, obrador, presidente, empresa, partido, mujeres, persona".

Figura 5.4: Nube de palabras de las noticias verdaderas del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

Por otra parte, en la Figura 5.5 se presenta la nube de palabras correspondientes a las

noticias falsas. Como se puede ver aparecen como palabras más frecuentes "país, año, méxico, persona, presidente, partido, mexicano, gente, gobierno", que justamente en gran parte coinciden con las más frecuentes en los textos de las noticias verdaderas. Esto es una muestra de la superposición del vocabulario mencionado anteriormente.

Figura 5.5: Nube de palabras de las noticias falsas

5.2.2. Versión 2.0 (FakeDeS Task @ Iberlef 2021)

Como se mencionó en la primera sección del presente capítulo, la segunda versión del corpus está conformado por la información referente a la Categoría, Fuente, Titular, Texto y enlace de la noticia. A modo general esta versión del corpus está conformada por 572 observaciones y no presenta datos duplicados.

A continuación, en la Tabla 5.7 se puede visualizar la distribución de las noticias respecto a la Categoría (Verdadera, Falsa). Como se puede observar el corpus se encuentra perfectamente balanceado, existen 286 noticias verdadera y 286 noticias falsas.

Valor	Cantidad	Frecuencia (%)
Verdadero	286	50 %
Falso	286	50 %

Tabla 5.7: Distribución de noticias respecto a la clase falsa y verdadera del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021)

En la Tabla 5.8 se puede visualizar la distribución de las noticias respecto a los tópicos del corpus (Covid-19, Política, Sociedad, Internacional, Ciencia, Deporte y Ambiental). Se puede

observar que el tópico con presencia de noticias es en "Covid 19", el cual hay 237 noticias lo que representa un 41.4 % del total de noticias del corpus.

Tópicos	Verdadero	Falso	Cantidad	Frecuencia
Covid-19	118	119	237	41.4 %
Política	53	54	107	18.7 %
Sociedad	99	96	195	34.1 %
Internacional	7	7	14	2.4 %
Ciencia	6	7	13	2.3 %
Deporte	1	1	2	0.3 %
Ambiental	2	2	4	0.7 %
Total	286	286	572	-

Tabla 5.8: Distribución de las noticias respecto a los tópicos del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @ Iberlef 2021)

En la Tabla 5.9 se puede visualizar el vocabulario perteneciente a las noticias falsas y verdaderas y la superposición de vocabulario [3]. De igual forma a la primera versión del corpus, para obtener la superposición del vocabulario se consideraron los lemas y se descartaron los *stopwords* y se calculó la superposición dividiendo la intersección del vocabulario de las noticias verdaderas y falsas entre el vocabulario de todas las noticias.

Tópicos	Verdadera	Falsa	Superposición del Vocabulario
Covid-19	4653	4480	22.1 %
Política	4095	3205	15.45 %
Sociedad	5327	4017	21.3 %
Internacional	1021	685	2.66 %
Ciencia	574	658	1.93 %
Deporte	106	81	0.06 %
Ambiental	318	428	0.83 %
Promedio de Superposición	-	-	9.19 %

Tabla 5.9: Superposición del vocabulario respecto a los diferentes tópicos del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021)

De la Tabla 5.9 tenemos que la mayor superposición del vocabulario está presente en las noticias pertenecientes al tópico "Covid-19" con un valor de 22.1 %.

[3] Tabla tomada del artículo *"The Spanish Fake News Corpus"*

En la Tabla 5.10 se puede visualizar la distribución de las noticias respecto a las fuentes donde fueron publicadas. Como se puede ver el sitio del cual se extrajo mayor cantidad de noticias fue "El Dizque". De este sitio fueron extraídas un total de 135 noticias lo que representan el 13.9 % del total de noticias que conforman el corpus.

Valor	Cantidad	Frecuencia (%)
AFP Factual	**90**	**15.73 %**
El Universal	29	5.1 %
Milenio	27	4.7 %
El País	21	3.7 %
Mediterráneo Digital	16	2.8 %
El Financiero	15	2.6 %
BBC	26	4.5 %
Agencia Efe	10	1.7 %
Maldita	10	1.7 %
CNN	9	1.6 %
Otros valores (190)	317	55.4 %

Tabla 5.10: Distribución de las noticias en dependencia del origen de la publicación del *Spanish Fake News Corpus* Versión 1.0 (@MEXLEF 20)

Análisis a nivel de caracteres de los textos:

A continuación, en la Figura 5.6 mediante un histograma se muestra el número de caracteres presentes en cada noticia. Esto nos proporciona una idea aproximada de la longitud de los textos que conforman las noticias. Dicho histograma muestra que los textos que representan las noticias oscilan entre 100 y 16000 caracteres y en general, entre 200 y 25000.

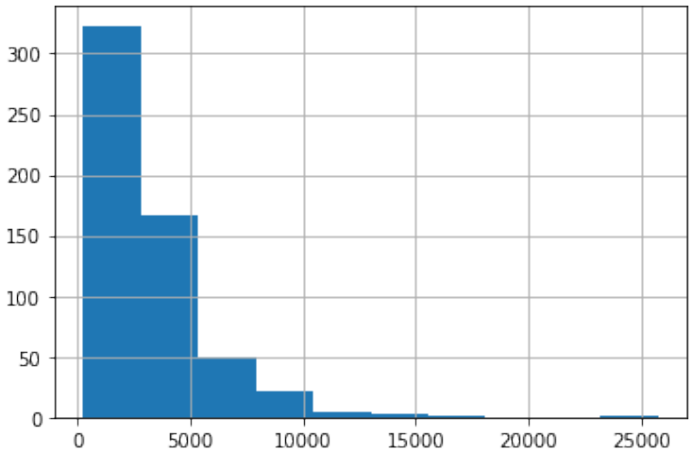


Figura 5.6: Longitud de los textos de las noticias del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021)

Análisis a nivel de palabras de los textos:

Ahora, pasaré a la exploración de datos a nivel de palabra. En la Figura 5.7 se muestra la cantidad de palabras que aparecen en cada texto de las noticias. Como se puede visualizar el número de palabras en los textos de las noticias varía de 40 a 4000 y en su mayoría se encuentran entre 40 y 500.

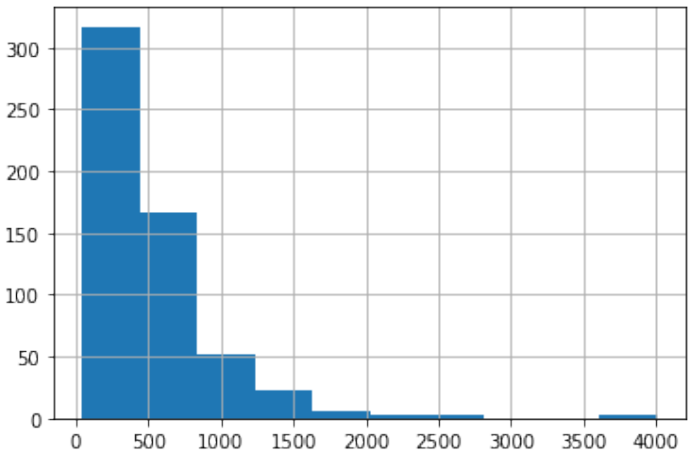


Figura 5.7: Distribución de cantidad de palabras por noticias del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021).

A continuación, verifiquemos la longitud promedio de las palabras en cada oración, para esto primeramente se realizó un procesamiento eliminando los *stopwords* de los textos de las noticias, este procesamiento tiene el objetivo de evitar sesgos en el gráfico, dado que los *stopwords* son palabras generalmente pequeñas y son las más frecuentes en los textos.

En la Figura 5.8 se muestra que la longitud promedio de las palabras por oración varía entre 6 y 9, siendo 7 la longitud más común.

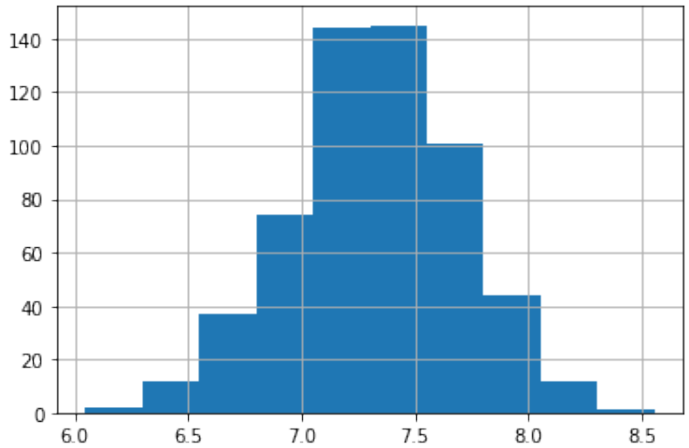


Figura 5.8: Longitud promedio de las palabras por oración del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021).

A continuación, en la Figura 5.9 se muestra las palabras con mayor frecuencia en las noticias verdaderas. Para la confección de las nubes de palabras realicé un preprocesamiento eliminando los *stopwords* dado que generalmente estas son las palabras más frecuentes en los textos. Como se puede apreciar en particular las palabras más frecuentes son "año, vacuna, virus, persona, covid, gobierno, coronavirus, pandemia".

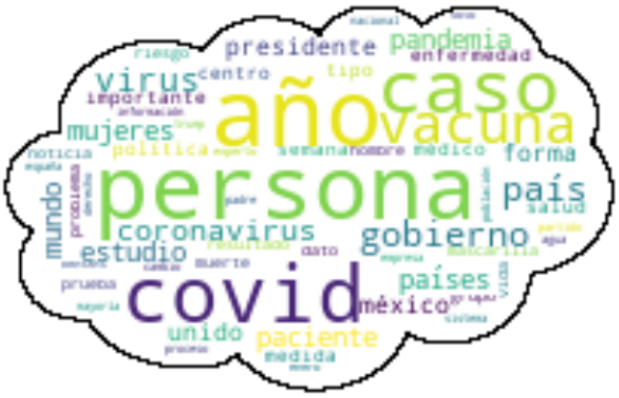


Figura 5.9: Nube de palabras de las noticias verdaderas del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021)

Por otra parte, en la Figura 5.10 se presenta la nube de palabras correspondientes a las noticias falsas. Como se puede ver aparecen como palabras más frecuentes "año, vacuna, virus, persona, covid, caso, gobierno, coronavirus, mundo", que justamente en gran parte coinciden con las más frecuentes en los textos de las noticias verdaderas. Esto es una muestra de la superposición del vocabulario mencionado anteriormente.

Figura 5.10: Nube de palabras de las noticias falsas del *Spanish Fake News Corpus* Versión 2.0 (FakeDeS Task @Iberlef 2021)

Capítulo 6

Resultados experimentales

"Imagination is the discovering faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science."

Ada Lovelace

En este capítulo se mostrarán y discutirán los resultados obtenidos durante la fase de experimentación del presente trabajo de investigación. En la Sección 6.1 se presentan los resultados obtenidos al evaluar la arquitectura *baseline* descrita en la Sección 4.2. Esta arquitectura *baseline* como se mencionó en la Sección 4.2 está basada en aprendizaje de máquina combinada con características estilométricas, la misma será entrenada y evaluada utilizando la primera y segunda versión respectivamente del conjunto de datos *"Spanish Fake News Corpus"*. Seguidamente en la Sección 6.2 se muestran los resultados obtenidos al evaluar la arquitectura *Deep Style Fake*, la cual fue propuesta y descrita en la Sección 4.3 en el presente trabajo de investigación, en el conjunto de datos *"Spanish Fake News Corpus"*.

Para el entrenamiento de los métodos *baselines* y la arquitectura *Deep Style Fake* se utilizó como corpus de entrenamiento la primera versión del corpus *"Spanish Fake News Corpus"* 5.2.1 el cual está conformado por noticias agrupadas en nueve tópicos diferentes (Ciencia, Deportes, Economía, Educación, Entretenimiento, Política, Salud, Seguridad y Sociedad). Luego los modelos entrenados fueron evaluados en la segunda versión del corpus 5.2.2 el cual está conformado por noticias agrupadas en siete tópicos (Covid-19, Política, Sociedad, Internacional, Ciencia, Deporte y Ambiental).

Durante la etapa de experimentación se entrenaron los modelos utilizando solamente los textos de las noticias y también utilizando la unión de los títulos con los textos de las noticias. Obteniendo mejores resultados utilizando solo el cuerpo de las noticias.

6.1. Métodos *baselines*

Como se mencionó en la Sección 4.2 la arquitectura utilizada a modo de *baseline* durante la investigación está conformada por varios módulos, entre ellos se encuentra el módulo de aprendizaje de máquina el cual puede ser implementado mediante varios algoritmos. Estos algoritmos pueden ser regresión logística, máquinas de vectores de soporte o clasificador bayesiano ingenuo multinomial. Además, otro de los módulos que compone la arquitectura es el

módulo de representación vectorial el cual puede ser implementado mediante *TfIdf* o bolsa de palabras.

Durante la fase de experimentación se utilizó *GridSearchCV* para el proceso de ajuste de hiper parámetros del esquema de pesado, para el cual se utilizó *TfIdf* y para el ajuste de hiper parámetros de los algoritmos de aprendizaje automático, en particular experimentamos con regresión logística, clasificador bayesiano ingenuo multinomial y con máquinas de vectores de soporte. En la Tabla 6.1 se encuentran las combinaciones de parámetros del vectorizador *TfIdf* que mejores resultados alcanzaron al ser combinados con los algoritmos de aprendizaje automático.

Para el caso del vectorizador *TfIdf* se ajustaron los parámetros *binary*, *max_df*, *ngram_range* y *norm*. En este caso el parámetro *binary* denota que el término *Tf* en *TfIdf* es binario, el parámetro *max_df* representa que al construir el vocabulario se deben ignorar los términos que tienen una frecuencia de documento estrictamente superior al umbral. Además, tenemos que el parámetro *ngram_range* denota el límite inferior y el límite superior del rango de valores n para los diferentes n-gramas que se extraerán y el parámetro *norm* representa el tipo de norma a utilizar.

max_df	***binary***	***ngram_range***	***norm***
0.30	*True*	(1, 1)	l_2
0.20	*True*	(1, 2)	l_1
0.40	*True*	(1, 2)	l_1

Tabla 6.1: Combinación de parámetros para el modelo de extracción de características *TfIdf*.

Por otra parte, en la Tabla 6.2 se muestra las combinaciones de parámetros que alcanzaron mejores resultados para los algoritmos de aprendizaje automático.

Para cada uno de los algoritmos de aprendizaje automático se ajustaron algunos hiper parámetros. En el caso del algoritmo de regresión logística se ajustó el parámetro *C*, *penalty* y *max_iter*. El parámetro *C*

Algoritmo	***C***	***penalty***	***max_iter***	***alpha***	***fit_prior***	***gamma***	***kernel***
Logistic Regression	1000	l_2	100	–	–	–	–
Multinomial NB	–	–	–	0.10	*False*	–	–
SVC	100	–	–	–	–	– 1	rbf

Tabla 6.2: Combinación de parámetros para los algoritmos de aprendizaje de máquina.

6.1.1. Regresión logística

Durante el proceso de experimentación mediante *GridSearchCV* obtuvimos que al utilizar como esquema de pesado *TfIdf* y como modelo de clasificación de noticias Regresión Logística los mejores resultados se alcanzaron al utilizar como parámetros del clasificador los siguientes valores:

- $C = 1000$
- $penalty = l_2$
- $max\_iter = 100$

Como parámetros del vectorizador se utilizaron los siguientes valores:

- $max\_df = 0.30$
- $binary = True$
- $ngram\_range = (1, 1)$
- $norm = l_2$

Mediante los parámetros anteriores obtuvimos los siguientes los resultados presentados en la Tabla 6.3:

Clase	*Precision*	*Recall*	*F1-Score*
True	0.66	0.88	0.75
Fake	0.82	0.53	0.65
Accuracy	-	-	0.72

Tabla 6.3: Resultados obtenidos con regresión logística utilizando *TfIdf*.

Como se puede observar en la Tabla 6.3 se obtuvo como valor de la métrica F_1 un valor de 0.65 para la clase *fake*. Luego al utilizar la misma combinación de parámetros para el modelo de Regresión Logística mediante el esquema de pesado *TfIdf* y además incluir características estilométricas como el número de palabras en mayúsculas (PM), el número de caracteres (CM), el número de símbolos (S ?¿¡#@) y el número de errores ortográficos (EO) se visualizó una mejora en los resultados. Estos resultados se presentan en la Tabla 6.4. En esta tabla "X" indica la exclusión de la función en el experimento y "O" indica la inclusión de la función.

Como se puede observar en la Tabla 6.4 mediante el uso de combinaciones de características estilométricas fue posible mejorar el valor obtenido para la métrica F_1. En particular el mejor valor se obtuvo al combinar el vector resultante de la extracción de características mediante el modelo *TfIdf* junto con el número de palabras en mayúsculas que aparecen en los textos de las noticias.

6.1.2. Clasificador bayesiano ingenuo multinomial

Para adaptar este modelo a nuestro corpus fue necesario ajustar algunos parámetros. Durante la fase de implementación se utilizó el modelo *MultinomialNB* y para la extracción de características mediante *TfIdf* se utilizó el módulo *TfIdfVectorizer*, ambos de la biblioteca *sklearn* de *Python*. Entre los parámetros ajustados se encuentran el parámetro *alpha* y

#PM	#CM	#S(?¿¡#@)	# EO	*Accuracy*	*Precission(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.71	0.82	0.53	0.65
O	**X**	**X**	**X**	**0.73**	**0.73**	**0.72**	**0.73**
X	O	X	X	0.68	0.65	0.79	0.71
X	X	O	X	0.72	0.75	0.65	0.70
X	X	X	O	0.72	0.75	0.65	0.70
O	O	X	X	0.69	0.67	0.77	0.71
O	X	O	X	0.72	0.73	0.71	0.72
O	X	X	O	0.72	0.73	0.69	0.71
X	X	O	O	0.68	0.65	0.78	0.71
X	O	X	O	0.69	0.66	0.78	0.71
X	X	O	O	0.72	0.76	0.64	0.7
O	O	O	X	0.69	0.67	0.77	0.71
O	O	X	O	0.69	0.67	0.76	0.71
X	O	O	O	0.68	0.66	0.76	0.7
O	O	O	O	0.69	0.67	0.76	0.71

Tabla 6.4: Resultados obtenidos con regresión logística, *TfIdf* y combinación de características estilométricas.

el parámetro *fit_prior* ambos referentes al modelo de clasificación. Junto con los parámetros anteriores también fueron ajustados los parámetros referentes al modelo de extracción de características. Para el caso del modelo *TfIdfVectorizer* se ajustaron los parámetros *binary, max_df, ngram_range y norm.*

Durante el proceso de experimentación mediante *GridSearchCV* se obtuvo que al utilizar como modelo de extracción de características *TfIdf* con los siguientes parámetros:

- $max\_df = 0.2$
- $binary = True$
- $ngram\_range = (1, 2)$
- $norm = l_1$

Como modelo de clasificación de noticias se utilizó el clasificador bayesiano ingenuo multinomial. Se tiene que los mejores resultados se alcanzaban con los siguientes parámetros:

- $alpha = 0.1$
- $fit\_prior = False$

En este caso utilizando los parámetros mencionados anteriormente para el modelo de extracción de características *TfIdf* y el modelo de clasificación *MultinomialNB* se obtuvo para las métricas *Accuracy, Precision, Recall, F1-Score* los siguientes resultados:

Seguidamente al incluir características estilométricas como el número de palabras en mayúsculas (PM), el número de caracteres (CM), el número de símbolos (S ?¿¡#@) y el número de errores ortográficos (EO). Utilizando la misma combinación de parámetros para el modelo bayesiano ingenuo multinomial mediante el esquema de pesado *TfIdf* obtuvimos los resultados presentados en la Tabla 6.6:

Clase	*Precision*	*Recall*	*F1-Score*
True	0.65	0.74	0.69
Fake	0.70	0.61	0.65
Accuracy	-	-	0.67

Tabla 6.5: Resultados obtenidos con el clasificador bayesiano ingenuo multinomial y *TfIdf*.

#PM	#CM	#S(?¿¡#@)	# EO	*Accuracy*	*Precission(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.67	0.70	0.61	0.65
O	X	X	X	0.67	0.69	0.63	0.66
X	O	X	X	0.68	0.79	0.49	0.61
X	**X**	**O**	**X**	**0.69**	**0.67**	**0.74**	**0.70**
X	X	X	O	0.64	0.87	0.32	0.47
O	O	X	X	0.67	0.79	0.46	0.58
O	X	O	X	0.68	0.69	0.67	0.68
O	X	X	O	0.62	0.88	0.28	0.42
X	O	O	X	0.68	0.77	0.51	0.62
X	O	X	O	0.59	0.88	0.22	0.35
X	X	O	O	0.65	0.87	0.34	0.49
O	O	O	X	0.67	0.78	0.48	0.59
O	O	X	O	0.6	0.90	0.22	0.36
X	O	O	O	0.6	0.88	0.23	0.37
O	O	O	O	0.61	0.91	0.24	0.38

Tabla 6.6: Resultados obtenidos con el clasificador bayesiano ingenuo multinomial, *TfIdf* y combinación de caracteristicas estilométricas.

Como se puede observar en la Tabla 6.6 mediante el uso de características estilométricas fue posible mejorar el valor obtenido para la métrica F_1. En particular se mejora en un 5 % utilizando como combinación de características estilométricas solo la cantidad de símbolos (?¿¡#@) presentes en los textos. En este caso el valor obtenido para la métrica F_1 de la clase $fake$ fue de 0.70.

6.1.3. Máquinas de vectores de soporte

Para adaptar este modelo a nuestro corpus es necesario ajustar algunos parámetros, entre esos se encuentran los parámetros *C, gamma y kernel*. De igual forma al procedimiento realizado con el modelo *MultinomialNB* ajustaremos los parámetros referentes al modelo de extracción de características.

Durante el proceso de experimentación mediante *GridSearchCV* obtuvimos que se alcanzaron los mejores resultados al utilizar como modelo de extracción de características *TfIdf* con los siguientes parámetros:

- $max\_df = 0.4$
- $binary = True$
- $ngram\_range = (1, 2)$
- $norm = l_1$

y como modelo de clasificación de noticias *Support Vector Classifier(SVC)* con los siguientes parámetros:

- $C = 100$
- $gamma = 1$
- $kernel = rbf$

Mediante los parámetros mencionados anteriormente se obtuvieron los resultados presentados en la Tabla 6.7:

Clase	*Precision*	*Recall*	*F1-Score*
True	0.75	0.60	0.67
Fake	0.65	0.79	0.71
Accuracy	-	-	0.74

Tabla 6.7: Resultados obtenidos con máquinas de vectores de soporte y *TfIdf*.

Al combinar el modelo anterior con características estilométricas se obtienen los resultados presentados en la Tabla 6.8:

#PM	#CM	#S(?¿¡#@)	# EO	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.71	0.79	0.56	0.66
O	**X**	**X**	**X**	**0.67**	**0.63**	**0.83**	**0.72**
X	O	X	X	0.63	0.59	0.85	0.7
X	X	O	X	0.72	0.74	0.67	0.71
X	X	X	O	0.69	0.76	0.57	0.65
X	O	O	X	0.6	0.56	0.88	0.69
O	X	O	X	0.67	0.64	0.78	0.71
O	X	X	O	0.69	0.71	0.65	0.68
X	O	O	X	0.66	0.62	0.81	0.7
X	O	X	O	0.69	0.67	0.76	0.71
X	X	O	O	0.68	0.78	0.51	0.62
O	O	O	X	0.63	0.59	0.87	0.7
O	O	X	O	0.61	0.57	0.91	0.7
X	O	O	O	0.69	0.69	0.69	0.69
O	O	O	O	0.63	0.59	0.86	0.7

Tabla 6.8: Resultados obtenidos con máquinas de vectores de soporte, *TfIdf* y combinación de características estilométricas.

Como se puede observar en la Tabla 6.8 mediante el uso de características estilométricas fue posible mejorar el valor obtenido para la métrica F_1. En particular se mejora en un 6 % utilizando como combinación de características estilométricas solo la cantidad de palabras en mayúsculas (#PM) presentes en los textos.

A modo resumen, tenemos que la arquitectura *baseline* que mejores resultados obtuvo se encuentra compuesta por un modelo de regresión logística en el cual se combina el vector resultante de la extracción de características mediante *TfIdf* junto con el número de palabras en mayúsculas que aparecen en los textos de las noticias. Esta arquitectura al ser evaluada en el corpus *"Spanish Fake News Corpus"* alcanzó en la métrica F_1 un valor de 0.73.

A continuación, se presentan los resultados obtenidos al evaluar la arquitectura propuesta *Deep Style Fake* en el corpus *"Spanish Fake News Corpus"*.

6.2. Arquitectura *Deep Style Fake*

Como se mencionó en la Sección 4.3 referente a la descripción de la arquitectura *Deep Style Fake* propuesta en este trabajo de investigación, la arquitectura está compuesto por múltiples módulos, entre ellos se encuentran el módulo de preprocesamiento, módulo de generación de *embeddings*, módulo de aprendizaje profundo, módulo de combinación y el módulo de clasificación.

En el módulo de aprendizaje profundo se puede incluir una red neuronal convolucional (*CNN*) o una red neuronal recurrente específicamente una red *Long Short Term Memory(LSTM)*. En las siguientes secciones 6.2.1 y 6.2.2 presentaremos los diferentes resultados obtenidos al utilizar ambos tipos de redes neuronales en el módulo de aprendizaje profundo de la arquitectura *Deep Style Fake*.

La Tabla 6.9 muestra las diferentes combinaciones estilométricas que permitieron obtener los mejores resultados con la arquitectura *Deep Style Fake*. Las combinaciones se componen del número de caracteres en mayúsculas, número de frases repetidas, número de símbolos y número de errores ortográficos. En la Tabla 6.9, "X" indica la exclusión de la función en el experimento y "O" indica la inclusión de la función.

Combinación	#Palabras en mayúsculas	# Caracteres en mayúsculas	# Símbolos (?!#@)	# Errores ortográficos
C1	X	O	X	X
C2	X	X	X	O

Tabla 6.9: Combinaciones estilométricas.

Durante el entrenamiento de la arquitectura se realizaron experimentos utilizando varios modelos de lenguaje para la generación de los *word embeddings*. Por ejemplo, se utilizó *BERT* en particular su versión multilingual [1] la cual contiene el lenguaje español y *BETO* [2] el cual fue creado por la Universidad de Chile específicamente para el idioma Español. Además, se realizaron experimentos utilizando el algoritmo de generación de *embeddings Wod2Vec* y *GloVe*.

6.2.1. *LSTM*

Para el entrenamiento de la arquitectura se utilizaron distintas configuraciones. La primera configuración utiliza los *embeddings* generados por *Word2Vec*, específicamente se utilizó *Word2Vec embeddings from Spanish Billion Words Corpora* [3]. La segunda configuración utiliza los *embeddings* generados por *GloVe*, en este caso se utilizó *GloVe embeddings from Spanish Billion Words Corpora* [4].

[1] https://huggingface.co/bert-base-multilingual-uncased
[2] https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased
[3] https://github.com/dccuchile/spanish-word-embeddings#word2vec-embeddings-from-sbwc
[4] https://github.com/dccuchile/spanish-word-embeddings#glove-embeddings-from-sbwc

Para las dos configuraciones anteriores se utilizó la siguiente configuración de parámetros:

- $lr = 0.05$
- $dropout = 0.2$
- $n\_epochs = 100$

En la tercera configuración se utilizó los *BERT embeddings* generados por el modelo pre-entrenado *BETO*, con la siguiente configuración de parámetros:

- $max\_len = 512$
- $batch\_size = 32$
- $n\_epochs = 20$
- optimizador: Adam
- $lr = 3.00E - 05$
- $embedding\_size = 768$

En la Figura 6.10 se presentan los diferentes resultados obtenidos con tres diferentes modelos de *embeddings* (*Word2Vec*, *GloVe*, *BERT*) y utilizando *LSTM* en la capa de aprendizaje profundo de la arquitectura.

Embeddings	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
Word2Vec	0.848	0.865	0.848	0.8259	0.6657	0.6931	0.6657	0.6543
GloVe	0.8382	0.8509	0.8382	0.8439	0.664	0.6904	0.664	0.6565
BERT Embeddings	**0.8529**	**0.8736**	**0.8529**	**0.8452**	**0.6579**	**0.6872**	**0.6579**	**0.6685**

Tabla 6.10: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *LSTM* en la capa de aprendizaje profundo y diferentes tipos de *embeddings*.

Como se pudo observar en la Figura 6.10 los mejores resultados de la arquitectura *Deep Style Fake* utilizando en la capa de aprendizaje profundo un modelo *LSTM* se obtuvo utilizando como representación de los textos *BERT embeddings*. Utilizando esta representación se alcanzó en la métrica F_1 para la clase *fake* un valor de 0.6685.

A continuación, en la Figura 6.11 se presentan los resultados obtenidos al combinar características estilométricas con la arquitectura *Deep Style Fake*. Utilizando en la capa de aprendizaje profundo un modelo *LSTM* y como representación de los textos *BERT embeddings*.

Como se puede observar en la Figura 6.11 los mejores resultados se obtuvieron utilizando como características estilométricas el número de caracteres en mayúsculas y el número de errores ortográficos presentes en los textos de las noticias, con un valor de 0.6898. Con estas características se alcanzó a superar en un 2 % los valores obtenidos anteriormente en la métrica F_1 sin incluir características estilométricas.

#PM	#CM	#S (?¿¡#@)	# EO	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.8529	0.8736	0.8529	0.8452	0.6579	0.6872	0.6579	0.6685
X	**O**	**X**	**X**	**0.8382**	**0.8824**	**0.8382**	**0.8494**	**0.651**	**0.7311**	**0.651**	**0.6898**
X	X	X	O	0.8382	0.8824	0.8382	0.8494	0.651	0.7311	0.651	0.6898
O	O	X	X	0.8725	0.894	0.8725	0.8666	0.6527	0.6916	0.6527	0.6634
X	O	X	O	0.8382	0.8772	0.8382	0.8423	0.6527	0.6946	0.6527	0.673
X	X	O	O	0.7598	0.8194	0.7598	0.7108	0.6588	0.7078	0.6588	0.6853

Tabla 6.11: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *LSTM* en la capa de aprendizaje profundo.

6.2.2. *CNN*

Como se mencionó al inicio de la Sección 6.2, durante la fase de experimentación con la arquitectura *Deep Style Fake*, en la capa de aprendizaje profundo se utilizaron las redes neuronales recurrentes, en particular *LSTM* y las redes neuronales convolucionales (*CNN*). A continuación, se presentan los resultados obtenidos al utilizar en la capa de aprendiza profundo una red neuronal convolucional.

Para el entrenamiento de la arquitectura se utilizó la siguiente combinación de parámetros:

- $max\_len = 512$
- $batch\_size = 32$
- $n\_epochs = 20$
- optimizador: *Adam*
- $lr = 3.00E-05$
- $embedding\_size = 768$
- $n\_conv\_layers = 3$
- $n\_kernels = 16$.

Embeddings	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
Word2Vec	0.767	0.809	0.767	0.77	0.689	0.755	0.689	0.69
GloVe	0.825	0.857	0.825	0.826	0.696	0.793	0.696	0.707
BERT Embeddings	**0.8497**	**0.8277**	**0.8497**	**0.8533**	**0.7098**	**0.7268**	**0.7098**	**0.7117**

Tabla 6.12: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *CNN* en la capa de aprendizaje profundo y diferentes tipos de *embeddings*.

Como se pudo observar en la Tabla 6.12 los mejores resultados de la arquitectura *Deep Style Fake* utilizando en la capa de de aprendizaje profundo un modelo *CNN* se obtuvo utilizando como representación de los textos *BERT embeddings*. A continuación, en la Figura 6.13 se presentan los resultados obtenidos utilizando al vincular la mejor combinación anterior con características estilométricas.

Como se puede observar en la Tabla 6.13 el mejor resultado referente a la métrica F_1 tuvo un valor de 0.744 y fue obtenido utilizando como combinación de características lingüísticas

#PM	#CM	#S (?¿¡#@)	# EO	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.8497	0.8277	0.8497	0.8533	0.7098	0.7268	0.7098	0.7117
X	**O**	**X**	**X**	**0.892**	**0.928**	**0.892**	**0.894**	**0.755**	**0.781**	**0.71**	**0.744**
X	X	O	X	0.8229	0.8065	0.8229	0.8206	0.71	0.7326	0.71	0.7252
X	X	X	O	0.8229	0.8065	0.8229	0.8206	0.7132	0.7309	0.7132	0.7236
X	X	X	O	0.8318	0.8125	0.8318	0.8518	0.7065	0.7383	0.7065	0.7273
O	O	X	X	0.8273	0.8078	0.8273	0.8435	0.7118	0.7415	0.7118	0.7321
O	X	O	X	0.8318	0.8164	0.8318	0.8531	0.7137	0.7473	0.7137	0.7346
O	X	X	O	0.8273	0.8078	0.8273	0.8435	0.7118	0.7415	0.7118	0.7321
X	O	O	X	0.8318	0.8164	0.8318	0.8531	0.7137	0.7473	0.7137	0.7346
X	O	X	O	0.8229	0.8078	0.8229	0.8449	0.7083	0.7434	0.7083	0.7306
O	O	X	X	0.82	0.8664	0.82	0.7925	0.7534	0.7828	0.7534	0.7244

Tabla 6.13: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *CNN* en la capa de aprendizaje profundo y combinaciones de características estilométricas.

el número de caracteres en mayúsculas presente en los textos de las noticias. A modo general, el uso de características estilométricas permitió mejorar los resultados en un 3 %.

6.3. Resumen

La tarea de detección de noticias falsas es un problema difícil en el área del procesamiento del lenguaje natural. El desarrollo de técnicas especialmente diseñadas para generar desinformación es un desafío constante para el área que tiene como objetivo identificar dicha desinformación. A pesar de esto, a partir de los resultados presentados en este capítulo podemos resumir que la metodología presentada en este trabajo de tesis basada en la combinación de características estilométricas con arquitecturas de aprendizaje profundo se evidencia que es posible mejorar los resultados al evaluarla en la tarea de detección de noticias falsas en textos en español. En particular la metodología fue evaluada en el corpus *"Spanish Fake News Corpus"* [5]. Los mejores resultados fueron alcanzados con la arquitectura que contiene en el Módulo de Aprendizaje Profundo una red neuronal convolucional y utilizando como características estilométricas el número de caracteres en mayúsculas y el número de símbolos (?¿¡#@) logrando un valor de 0.744 para la métrica F_1.

Estos resultados obtenidos coinciden con el análisis manual de las noticias que se presentó en la Sección 4.3.4 en la cual se detectó como patrones lingüísticos en las noticias falsas el uso de mayúsculas, signos de exclamación por parte de los autores, además del uso de etiquetas (#, @) dentro de los textos de las noticias.

Me parece interesante el hecho que las palabras al ser representadas mediante *word embeddings* previamente entrenados como es el caso de *Word2Vec* o *GloVe* arrojaron valores en base a la métrica F_1 muy similares o en ocasiones más bajos en comparación con representaciones más simples como es el caso de *TfIdf*. Esta situación considero que puede estar dada por varias razones, entre ellas el tamaño de los artículos de las noticias. Es posible que las representaciones de espacio latente como *Word2Vec* o *GloVe* no sean capaces de capturar la importancia de las palabras si la longitud de los textos de las noticias es muy grande.

Por otra parte, tenemos que, aunque los mejores resultados fueron alcanzados con la arquitectura propuesta *Deep Style Fake*. Utilizando una arquitectura de aprendizaje automático tradicional combinado con características estilométricas también es posible alcanzar resultados similares. Como es el caso, de utilizar una arquitectura basada en máquina de vectores de soporte con la cual se alcanzó para la métrica F_1 un valor de 0.72. Esto parece indicar la

[5] https://github.com/jpposadas/FakeNewsCorpusSpanish

complejidad de la tarea de detección de noticias falsas en el idioma español, la cual debe ser abordada por sistemas que consideren múltiples variables.

Capítulo 7

Conclusiones y trabajo futuro

"Imagination is the discovering faculty, pre-eminently. It is that which penetrates into the unseen worlds around us, the worlds of science."

Ada Lovelace

En este capítulo se presentan las conclusiones del presente trabajo de tesis. Como resultado de esta investigación se propuso una arquitectura para la detección de noticias falsas basada en la combinación de aprendizaje profundo y características estilométricas.

Como se pudo visualizar en el Capítulo 6 los resultados ilustran que la tarea de detección de noticias falsas constituye una tarea difícil en el área del procesamiento del lenguaje natural, en particular en el lenguaje español. Mediante la arquitectura propuesta al evaluarlo en la segunda versión del corpus *"Spanish Fake News Corpus"* se obtuvo como resultado de la métrica F_1 un valor 0.744. Este valor se alcanzó al incluir en el módulo de aprendizaje profundo una red neuronal convolucional con las características descritas en la Sección 4.3.3 y utilizando como características estilométricas el número de caracteres en mayúsculas presente en los textos de las noticias.

La segunda versión del corpus *"Spanish Fake News Corpus"* la cual fue descrita en el Capítulo 5, contiene tres nuevos tópicos. Estos tópicos son, *Covid-19, Ambiental e Internacional* los cuales no se encuentran incluidos en la primera versión del corpus la cual fue empleada para entrenar la arquitectura *Deep Style Fake*.

7.1. Conclusiones finales

A partir de los resultados presentados en el Capítulo 6, podemos darle respuesta a las preguntas de investigación que se plantearon al inicio de este trabajo de tesis, en el Capítulo 1.

A continuación, respondemos brevemente dichas preguntas:

1. **¿Qué efecto tienen las técnicas de preprocesamiento de textos en el rendimiento de los modelos de clasificación obtenidos?**

   El preprocesamiento de textos es tradicionalmente un paso importante para las tareas de procesamiento de lenguaje natural. El objetivo de este paso es transformar el texto en un forma en la cual los algoritmos de aprendizaje automático o aprendizaje profundo

puedan funcionar mejor. Durante este trabajo de investigación se realizaron múltiples experimentos en los cuales se utilizaron diferentes técnicas de preprocesamiento de textos como son la tokenización, eliminación de plabras funcionales, transformación de todos los caracteres a minúsculas, eliminación de signos de puntuación, corrección ortográfica, *stemming* y lematización.

Al evaluar los diferentes modelos utilizando combinaciones de las diferentes técnicas mencionadas anteriormente se obtuvo que los modelos presentaron mejor rendimiento utilizando solamente tokenización, eliminación de palabras funcionales, transformación de todos los caracteres a minúsculas, eliminación de signos de puntuación y corrección de errores ortográficos.

2. **¿Cuáles de los modelos de aprendizaje profundo obtenidos permiten alcanzar mejores resultados en base a la métrica F_1 ?**

   Los mejores resultados obtenidos en base a la métrica F_1 fueron obtenidos con la arquitectura *Deep Style Fake* utilizando *BERT embeddings* para la representación vectorial de los textos de las noticias y en la capa de aprendizaje profundo una red neuronal convolucional *CNN* con las características descritas en la Sección 4.3.3. Además, se utilizó como características estilométricas el número de caracteres en mayúsculas presente en los textos de las noticias.

   En la Tabla 7.1 se presentan los valores alcanzados referentes a las métricas *Accuracy*, *Precission*, *Recall* y F_1 para la clase *fake* tanto en la fase de entrenamiento como en la fase de evaluación.

Embeddings	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
BERT Embeddings	**0.892**	**0.928**	**0.892**	**0.894**	**0.755**	**0.781**	**0.71**	**0.744**

Tabla 7.1: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *CNN* en la capa de aprendizaje profundo.

3. **Con base en el tópico de las noticias ¿Qué categorías obtuvieron mayor porcentaje de noticias clasificadas correctamente con el modelo que mostró mejor desempeño?**

   De los resultados presentados en las subsecciones 6.2.1 y 6.2.2 obtuvimos que la arquitectura *Deep Style Fake*, la cual fue propuesta en el presente trabajo de tesis alcanzó los mejores resultados en base a la métrica F_1 con un valor igual a 0.744 al utilizar *BERT embeddings* en el módulo de generación de *embeddings* y una red neuronal convolucional con las características presentadas en la Sección 4.3.3, en el módulo de aprendizaje profundo de dicha arquitectura.

   Como se mencionó en la Sección 5.2.2 en la segunda versión del corpus *"Spanish Fake News Corpus"* las noticias se encuentran agrupadas en siete tópicos. Estos son Covid-19, Política, Sociedad, Internacional, Ciencia, Deporte y Ambiental. En base a esta agrupación de las noticias se realizó un análisis referente al porciento de noticias clasificadas correctamente.

   A continuación, en la Tabla 7.2 se presentarán los resultados obtenidos referentes a la métrica F_1 en base al tópico de las noticia, al evaluar la arquitectura *Deep Style Fake* en el corpus *"Spanish Fake News Corpus"*.

Tópico	F_1	*Accuracy*
Covid-19	0.74	0.74
Política	0.78	0.79
Sociedad	0.73	0.75
Internacional	0.93	0.93
Ciencia	0.50	0.54
Deporte	1.00	1.00
Ambiental	0.80	0.75

Tabla 7.2: Resultados en base al tópico de las noticias

Como se puede observar en la Tabla 7.2 presentada anteriormente, la arquitectura presentó un mejor comportamiento en base a las noticias pertenecientes al tópico *Deporte* en este caso tanto la métrica *accuracy* como la métrica F_1 alcanzaron un valor igual a 1.0. En cambio, el tópico en el cual la arquitectura presentó un peor desempeño fue en *Ciencia* con valores 0.50 y 0.54 respectivamente referente a las métricas F_1 y *accuracy*. También cabe resaltar que el desempeño de la arquitectura no está totalmente relacionada con la cantidad de noticias presentes en los tópicos del conjunto de entrenamiento. Dado que, si bien no hubo noticias relacionadas con el tópico de Covid-19 y el de Ambiental en el conjunto de entrenamiento, los resultados alcanzado en este tópico son comparables a los de tópicos disponibles en el conjunto de entrenamiento, como Política y Sociedad.

4. **¿Qué efecto tiene el uso de las características estilométricas en los resultados de la clasificación en conjunto con las arquitecturas de aprendizaje profundo implementadas?**

   En base a los resultados presentados en el Capítulo 6 se evidenció que fue posible aumentar el valor de la métrica F_1 hasta en un 3 % al combinar con características lingüísticas. En particular la arquitectura propuesta presentó un mejor desempeño al utilizar una red neuronal convolucional en el módulo de aprendizaje profundo, esta arquitectura primeramente se evaluó sin tomar en cuenta el uso de características lingüísticas y los resultados obtenidos se resumen en la primera fila de la Tabla 7.3. Por otra parte en la segunda fila de la Tabla 7.3 se presentan los resultados obtenidos al combinar con el número de caracteres en mayúsculas presente en las noticias.

#PM	#CM	#S(?¿¡#@)	# EO	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.8497	0.8277	0.8497	0.8533	0.7098	0.7268	0.7098	0.7117
X	**O**	**X**	**X**	**0.892**	**0.928**	**0.892**	**0.894**	**0.755**	**0.781**	**0.71**	**0.744**

Tabla 7.3: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *CNN* en la capa de aprendizaje profundo.

7.2. Contribuciones del trabajo

Los resultados obtenidos en este trabajo de investigación poseen gran trascendencia científica, debido a que mediante la arquitectura propuesta como resultado de esta investigación, la cual está basada en la combinación de técnicas de aprendizaje profundo y características

estilométricas, es posible clasificar noticias publicadas en la *web*, en falsas y verdaderas principalmente en el idioma español. Las principales contribuciones de este trabajo se pueden enumerar de la siguiente forma:

1. Se propuso una arquitectura novedosa la cual combina características estilométricas y técnicas de aprendizaje profundo para la clasificación de noticias en falsas y verdaderas.

2. Los resultados obtenidos mediante la evaluación de la arquitectura en varios corpus demuestran que la arquitectura propuesta obtiene resultados comparables a los del estado del arte.

7.3. Trabajo futuro

En el presente trabajo de tesis se abordó la tarea de detección de noticias falsas en el idioma español, como se mencionó en el Capítulo 3 referente al análisis del estado del arte. Para abordar esta tarea existen varias perspectivas, estas son mediante métodos basados en el conocimiento, métodos basados en la propagación de la noticia, métodos basados en el origen y métodos basados en el estilo. En este trabajo de investigación, para la resolución de la tarea nos enfocamos en la perspectiva asociada con métodos basados en el estilo. En particular combinando técnicas de aprendizaje profundo con características estilométricas.

A lo largo de este trabajo de investigación se ha podido observar que existen múltiples vías a seguir para continuar expandiendo la presente investigación referente a la detección de noticias falsas. Como trabajo futuro, propongo seguir las siguientes dos líneas para continuar profundizando y mejorando el presente trabajo.

Una primera línea de investigación a seguir puede estar basada en realizar una mayor exploración manual de las características estilométricas presentes en los textos de las noticias. Para de esta forma realizar experimentos con diferentes combinaciones de características estilométricas.

Una segunda línea de investigación puede estar basada en explorar la posibilidad de automatizar con la ayuda de algoritmos la detección de características estilométricas presente tanto en las noticias verdaderas como las noticias falsas.

En resumen, en el campo del procesamiento del lenguaje natural constantemente se experimentan grandes avances debido a la gran cantidad de recursos disponibles. A pesar de esto en el idioma español existen muy pocas investigaciones donde se realicen experimentos evaluando arquitecturas de aprendizaje automático o aprendizaje profundo para la detección de noticias falsas en español. Esto debido a que no existen en el estado del arte una gran cantidad de corpus conformados con noticias etiquetadas en español.

Apéndice A

Resultados al aplicar la arquitectura *Deep Style Fake* en un corpus en inglés

El presente apéndice muestra los resultados de aplicar la arquitectura *Deep Style Fake* a un corpus en el idioma inglés. El objetivo de estos experimentos es comprobar la generalidad de la metodología propuesto a otros idiomas. Para probar la arquitectura se utilizó el corpus *BuzzFeed-Webis Fake News Corpus*.

BuzzFeed-Webis Fake News Corpus

Este conjunto de datos fue introducido por Potthast et al. [2017], se encuentra disponible en *GitHub* [1]. Está compuesto por 1627 noticias publicadas en *Facebook* por 9 agencias de noticias durante una semana cercana a las elecciones estadounidenses del 2016. En este caso todos los editores obtuvieron la marca de verificación de *Facebook*, lo que indica autenticidad y un estatus elevado dentro de la red. Estas noticias fueron recopiladas entre el 19 y 23 de septiembre y entre el 26 y 27 de septiembre. Cada publicación y su correspondiente artículo vinculado fueron verificados de manera manual por 5 periodistas profesionales pertenecientes a la empresa de medios de comunicación estadounidense *BuzzFeed*.

En este corpus las noticias fueron clasificadas en las 4 categorías siguientes:

- **Mayormente verdaderas:** La noticia está basada en información fáctica. Esta calificación no permite especulaciones no respaldadas.
- **Mayormente falsas:** La mayor parte o la totalidad de la información en la noticia es inexacta.
- **Mezcla de contenido verdadero y falso:** En estas noticias, algunos elementos son precisos en cuanto a los hechos, pero algunos elementos o afirmaciones no lo son. Ejemplo de estas noticias es cuando el título contiene afirmaciones falsas pero el texto de la noticia es en gran medida verdadero. También pertenecen a esta clasificación aquellas noticias las cuales están basadas en información no confirmada, o en las cuales la información no respaldada o falsa es aproximadamente igual a la información que contiene verdadera.

[1] https://github.com/BuzzFeedNews/2016-10-facebook-fact-check

- **Sin contenido fáctico:** Esta calificación se usa para publicaciones que son pura opinión, *cómics*, sátira o cualquier otra publicación que no haga una afirmación fáctica.

El corpus está compuesto por la siguiente información:

- Id de Cuenta
- Id de Publicación
- Categoría (editoriales Principales, editoriales de izquierda, editoriales de derecha)
- Página
- *Url* de publicación
- Fecha de Publicación
- Clasificación (mayormente verdaderas, mayormente falsas, sin contenido fáctico y por último mezcla de contenido verdadero y falso)
- Cantidad de Veces compartida
- Título
- Texto de la Noticia

Para los experimentos consideré como categoría de noticia falsas la unión de las noticias mayormente falsas y las noticias que son clasificadas como mezcla de contenido verdadero y falso.

Resultados

Para la obtención de los resultados presentados en esta sección se utilizó la arquitectura *Deep Style Fake* compuesta en la capa de aprendizaje profundo de una red neuronal convolucional.

Para el entrenamiento de la arquitectura se utilizó la siguiente combinación de parámetros:

- max_len = 402
- $batch\_size = 32$
- $n\_epochs = 50$
- optimizador: *Adam*
- $lr = 3.00E-05$
- $embedding\_size = 768$
- $n\_conv\_layers = 3$
- $n\_kernels = 16$.

Seguidamente en la Tabla 7.4 se presentan los resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *CNN* en la capa de aprendizaje profundo y combinaciones de características estilométricas en el corpus *BuzzFeed-Webis Fake News*

#PM	#CM	#S(?¿¡#@)	# EO	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*	*Accuracy*	*Precision(Fake)*	*Recall(Fake)*	*F1(Fake)*
X	X	X	X	0.854	0.858	0.854	0.838	0.792	0.836	0.792	0.790
X	X	X	X	0.854	0.858	0.854	0.838	0.854	0.883	0.854	0.852
O	O	X	X	0.854	0.858	0.854	0.838	0.833	0.847	0.833	0.820
O	X	X	X	0.767	0.848	0.767	0.763	0.833	0.908	0.833	0.850
O	**X**	**O**	**X**	**0.875**	**0.888**	**0.875**	**0.856**	**0.875**	**0.891**	**0.875**	**0.868**
O	**X**	**X**	**O**	**0.875**	**0.888**	**0.875**	**0.856**	**0.875**	**0.891**	**0.875**	**0.868**
O	O	O	X	0.767	0.792	0.767	0.777	0.854	0.862	0.854	0.846

Tabla 7.4: Resultados obtenidos con la arquitectura *Deep Style Fake* utilizando *CNN* en la capa de aprendizaje profundo y combinaciones de características estilo métricas en el corpus *BuzzFeed-Webis Fake News*.

Como se puede observar en la Tabla 7.4 al utilizar la arquitectura sin características estilométricas se obtuvo un valor de 0.790 referente a la métrica F_1. En cambio, al utilizar características estilométricas se pudo mejorar los resultados en un 7 %, logrando alcanzar para la métrica F_1 en la clase *fake* un valor de 0.868. En particular, este resultado fue alcanzado mediante dos combinaciones de características estilométricas, la primera consiste en el uso del número de palabras en mayúsculas y el número de símbolos. Por otra parte, la segunda combinación consiste en el número de palabras en mayúsculas y el número de errores ortográficos. A modo general estos resultados pueden estar dados debido a que los textos de las noticias son extraídas de publicaciones de *facebook*, donde en múltiples ocasiones los títulos de las noticias se escriben en mayúsculas para atraer la atención de los usuarios.

Abreviaturas

Tf: *Term Frequency* (Frecuencia de término)

Idf: *Inverse Document Frecuency* (Frecuencia Inversa de Documento)

SVM: *Support Vector Machine* (Máquina de Vectores de Soporte)

BOW: *Bag of Words* (Bolsa de Palabras)

RF: *Random Forest* (Bosques Aleatorios)

LR: *Logistic Regression* (Regresión Logística)

MNB: *Multinomial Naive Bayes* (Clasificador Bayesiano Ingenuo Multinomial)

F1: *F1-Score*

PLN: *Natural Language Processing* (Procesamiento del Lenguaje Natural)

POS: *Part of Speech* (Categoría Gramatical)

TP: *True Positive* (Verdaderos Positivos)

TN: *True Negative* (verdaderos Negativos)

FP: *False Positive* (Falsos Positivos)

FN: *False Negative* (Falsos Negativos)

CNN: *Convolutional Neural Networks* (Redes Neuronales Convolucionales)

LSTM: *Long Short Term Memory* (Redes neuronales recurrentes con memoria a corto y largo plazo)

BERT: *Bidirectional Encoder Representation Transformer* (Representación de Codificador Bidireccional de Transformadores)

Bibliografía

Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*, 2019.

Mario Ezra Aragón, Horacio Jesús Jarquín-Vásquez, Manuel Montes-y Gómez, Hugo Jair Escalante, Luis Villaseñor Pineda, Helena Gómez-Adorno, Juan Pablo Posadas-Durán, and Gemma Bel-Enguix. Overview of mex-a3t at iberlef 2020: Fake news and aggressiveness analysis in mexican spanish. In *IberLEF@ SEPLN*, pages 222–235, 2020.

Steven Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72, 2006.

José Canete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. Spanish pre-trained bert model and evaluation data. *Pml4dc at iclr*, 2020: 2020, 2020.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 675–684, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450306324. doi: 10.1145/1963405.1963500. URL https://doi.org/10.1145/1963405.1963500.

Yahui Chen. Convolutional neural network for sentence classification. Master's thesis, University of Waterloo, 2015.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Nadia K Conroy, Victoria L Rubin, and Yimin Chen. Automatic deception detection: Methods for finding fake news. *Proceedings of the association for information science and technology*, 52(1):1–4, 2015.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20 (3):273–297, 1995.

Kingma Da. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Duke Reporter's Lab. Global fact-checking sites. https://reporterslab.org/fact-checking/, 2021. Accessed: 2021-10-05.

Europa Press. ¿cuántas personas en el mundo usan redes sociales? https://www.excelsior.com.mx/hacker/cuantas-personas-en-el-mundo-usan-redes-sociales/1465500, 2021. Accessed: 2021-10-02.

Ronen Feldman, James Sanger, et al. *The text mining handbook: advanced approaches in analyzing unstructured data*. Cambridge university press, 2007.

Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

Jennifer Golbeck, Matthew Mauriello, Brooke Auxier, Keval H Bhanushali, Christopher Bonk, Mohamed Amine Bouzaghrane, Cody Buntain, Riya Chanduka, Paul Cheakalos, Jennine B Everett, et al. Fake news vs satire: A dataset and analysis. In *Proceedings of the 10th ACM Conference on Web Science*, pages 17–21, 2018.

Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

Helena Gómez-Adorno, Juan Pablo Posadas-Durán, Gemma Bel Enguix, and Claudia Porto Capetillo. Overview of fakedes at iberlef 2021: Fake news detection in spanish shared task. *Procesamiento del Lenguaje Natural*, 67:223–231, 2021.

Santiago González-Carvajal and Eduardo C Garrido-Merchán. Comparing bert against traditional machine learning text classification. *arXiv preprint arXiv:2005.13012*, 2020.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Francesca Incitti, Federico Urli, and Lauro Snidaro. Beyond word embeddings: A survey. *Information Fusion*, 89:418–436, 2023.

Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. Understanding convolutional neural networks for text classification. *arXiv preprint arXiv:1809.08037*, 2018.

Akshay Jain and Amey Kasbe. Fake news detection. In *2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pages 1–5. IEEE, 2018.

Mayank Kumar Jain, Dinesh Gopalani, Yogesh Kumar Meena, and Rajesh Kumar. Machine learning based fake news detection using linguistic features and word vector features. In *2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, pages 1–6. IEEE, 2020.

Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. Fakebert: Fake news detection in social media with a bert-based deep learning approach. *Multimedia Tools and Applications*, 80(8):11765–11788, 2021.

Sachin Kumar, Rohan Asthana, Shashwat Upadhyay, Nidhi Upreti, and Mohammad Akbar. Fake news detection using deep learning models: A novel approach. *Transactions on Emerging Telecommunications Technologies*, 31(2):e3767, 2020.

Ksenia Lagutina, Nadezhda Lagutina, Elena Boychuk, Inna Vorontsova, Elena Shliakhtina, Olga Belyaeva, Ilya Paramonov, and PG Demidov. A survey on stylometric text features. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pages 184–195. IEEE, 2019.

Ling Liu and M Tamer Özsu. *Encyclopedia of database systems*, volume 6. Springer, 2009.

Zhiyuan Liu, Yankai Lin, and Maosong Sun. Representation learning and nlp. In *Representation Learning for Natural Language Processing*, pages 1–11. Springer, 2020.

Navonil Majumder, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. Deep learning-based document modeling for personality detection from text. *IEEE Intelligent Systems*, 32(2):74–79, 2017.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

Ines Montani, Matthew Honnibal, Matthew Honnibal, Sofie Van Landeghem, Adriane Boyd, Henning Peters, Paul O'Leary McCann, Jim Geovedi, Jim O'Regan, Maxim Samsonov, Duygu Altinok, György Orosz, Daniël de Kok, Søren Lind Kristiansen, Lj Miranda, Explosion Bot, Roman, Peter Baumgartner, Leander Fiedler, Richard Hudson, Madeesh Kannan, Edward, Grégory Howard, Wannaphong Phatthiyaphaibun, Yohei Tamura, Sam Bozek, murat, Ryn Daniels, and Flusskind. explosion/spaCy: v3.4.1: Fix compatibility with CuPy v9.x, July 2022. URL https://doi.org/10.5281/zenodo.6907665.

Lluís Padró and Evgeny Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *LREC2012*, 2012.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. Automatic detection of fake news. *arXiv preprint arXiv:1708.07104*, 2017.

Juan-Pablo Posadas-Durán, Helena Gómez-Adorno, Grigori Sidorov, and Jesús Jaime Moreno Escobar. Detection of fake news in a new corpus for the spanish language. *Journal of Intelligent & Fuzzy Systems*, 36(5):4869–4876, 2019.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*, 2017.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*, 2020.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937, Copenhagen, Denmark, sep 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1317. URL https://www.aclweb.org/anthology/D17-1317.

Victoria L Rubin, Yimin Chen, and Nadia K Conroy. Deception detection for news: three types of fakes. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015.

Santander Universidades. Los 6 idiomas más hablados en el mundo? https://www.becas-santander.com/es/blog/idiomas-mas-hablados.html, 2021. Accessed: 2021-10-02.

Karishma Sharma, Feng Qian, He Jiang, Natali Ruchansky, Ming Zhang, and Yan Liu. Combating fake news: A survey on identification and mitigation techniques. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(3):1–42, 2019.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *SIGKDD Explor. Newsl.*, 19(1):22–36, September 2017. ISSN 1931-0145. doi: 10.1145/3137597.3137600. URL https://doi.org/10.1145/3137597.3137600.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. Fakenewsnet: A data repository with news content, social context and spatialtemporal information for studying fake news on social media. *arXiv preprint arXiv:1809.01286*, 2018.

Alex Smola and SVN Vishwanathan. Introduction to machine learning. *Cambridge University, UK*, 32(34):2008, 2008.

Eli Stevens, Luca Antiga, and Thomas Viehmann. *Deep learning with PyTorch*. Manning Publications, 2020.

Kanchan M Tarwani and Swathi Edem. Survey on recurrent neural network in natural language processing. *Int. J. Eng. Trends Technol*, 48(6):301–304, 2017.

Sander van Der Linden, Jon Roozenbeek, and Josh Compton. Inoculating against fake news about covid-19. *Frontiers in psychology*, 11:2928, 2020.

Andreas Vlachos and Sebastian Riedel. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 workshop on language technologies and computational social science*, pages 18–22, 2014.

William Yang Wang. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*, 2017.

Giancarlo Zaccone, Md Rezaul Karim, and Ahmed Menshawy. *Deep learning with TensorFlow*. Packt Publishing Ltd, 2017.

Rong Zheng, Jiexun Li, Hsinchun Chen, and Zan Huang. A framework for authorship identification of online messages: Writing-style features and classification techniques. *Journal of the American society for information science and technology*, 57(3):378–393, 2006.

Xinyi Zhou, Reza Zafarani, Kai Shu, and Huan Liu. Fake news: Fundamental theories, detection strategies and challenges. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 836–837, 2019.