TRABAJO FIN DE MÁSTER

MÁSTER EN CIENCIAS DE DATOS E INGENIERÍA DE COMPUTADORES

Clasificación de propaganda en noticias con modelos de Deep Learning

Autor
Alberto Argente del Castillo Garrido

Directores
Eugenio Martínez Cámara
María Victoria Luzón García

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, 10 de septiembre de 2020.

Clasificación de propaganda en noticias con modelos de Deep Learning

Autor
Alberto Argente del Castillo Garrido

Directores
Eugenio Martínez Cámara
María Victoria Luzón García

Clasificación de propaganda en noticias con modelos de Deep Learning

Alberto Argente del Castillo Garrido

Palabras clave: *Fake News*, Propaganda, Deep Learning, Atención, LSTM, BERT

Resumen

Hoy en día el flujo de información a través de Internet es muy grande, lo que facilita mucho la comunicación entre las personas y que se pueda estar al tanto de todo lo que ocurre en el mundo en cuanto ocurra. Pese a esto, muchos medios tratan de dar noticias con tal de lograr más visitas, lograr dar la novedad, etc., y generan noticias no del todo veraces.

También hay quienes aprovechan las velocidad a la que se transmite la información, para generar noticias que estén sesgadas a una ideología, político o no, o que tergiversen una situación para culpar a personas con las que no se coincide ideológicamente, condicionando el pensamiento de las personas.

En este trabajo se va a realizar un sistema que sea capaz de detectar este tipo de noticias de propaganda para que los lectores no tengan que preocuparse de si el contenido contiene propaganda o no, y así asegurar una información veraz.

Por ello se ha estudiado el análisis de propaganda y cómo realizar la clasificación binaria de documentos propagandísticos para poder detectar aquellas noticias que tengan propaganda.

El sistema que se presenta es un modelo de *Deep Learning*,[1] y consiste en una modelo basado en BiLSTM con mecanismos de atención local, que permite no solo la clasificación de los documentos sino explicar aquellas palabras que son más importantes para clasificar a un documento como propagandístico.

Este modelo consigue buenos resultados para identificar aquellas noticias propagandísticas, indicando por qué lo son y así previendo de la difusión, o indicando qué tipo de noticias son.

[1] Código disponible en: https://github.com/AlArgente/TFM

Classification of propaganda news with Deep Learning models

Alberto Argente del Castillo Garrido

Keywords: *Fake News*, Propaganda, Deep Learning, Attention, LSTM, BERT

Abstract

Nowdays the flow of information on the Internet is very large, which facilitates the communication among people, and allow that they can be aware of everything that happens in the world as soon as it happens. On the other hand, some media seek to publish manipulated news for achieving more web visits, generiting novelty and so on.

There are also media, which take advantage of the speed at which the information is transmitted to generate skewed news to an ideology, political or not, or that misrepresents a situation to blame people with whom it does not coincide ideologically, conditioning the way of thinking of the people to their favour.

In this master thesis, I propose a system with the capacity of detecting propaganda news, which will allow readers to not worry about whether the content of news contains propaganda, and thus ensure accurate information.

For this reason, the propaganda analysis and how to carry out the binary classification of propaganda documents have been studied in order to detect those news that have propaganda.

The presented model is built using Deep Learning techniques,[2] and it is based on a BiLSTM layer with local attention mechanisms, which allows not only the classification of documents, but also explains those words that are most important to classify a document as propaganda.

This model achieves good results identifying those propaganda news, indicating why they are and thus anticipating the diffusion, or indicating what type of news they are.

[2]Code available in: https://github.com/AlArgente/TFM

Yo, **Alberto Argente del Castillo Garrido**, alumno de la titulación Máster en Ciencia de Datos e Ingeniería de Computadores de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 76654048Q, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Alberto Argente del Castillo Garrido

Granada, 10 de septiembre de 2020.

D. **Eugenio Martínez Cámara**, Profesor del Área de Ciencias de la Computación e Inteligencia Artificial del Departamento Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

D. **María Victoria Luzón García**, Profesor del Área de Lenguajes y Sistemas Informáticos del Departamento Lenguajes y Sistemas Informáticos de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Clasificación de propaganda en noticias con modelos de Deep Learning***, ha sido realizado bajo su supervisión por **Alberto Argente del Castillo Garrido**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada, 10 de septiembre de 2020

Los directores:

Eugenio Martínez Cámara **María Victoria Luzón García**

Agradecimientos

En primer lugar quiero agradecer a mis tutores Eugenio y Vicky la oportunidad de haber trabajado con ellos en este proyecto, por haberme guiado de la manera correcta para la realización del mismo. Además quiero agradecer el apoyo que han tenido durante este curso tan complicado para todos, que han hecho que este trabajo haya podido salir adelante, aunque haya sido más tarde de lo esperado. También quiero agradecerles por todos los conocimientos que me han aportado, sin los cuales no podría haber terminado.

También quiero agradecer especialmente a mis padres, familiares, pareja y amigos, por la ayuda y por el apoyo durante el curso para poder afrontar los retos presentados en el máster.

Por último, pero no menos importante, quiero agradecer a los profesores por las enseñanzas en clase y por su comprensión.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este capítulo se realizará la introducción al trabajo realizado. En la Sección 1.2 se explicará por qué es importante abordar la tarea del análisis de propaganda, y qué se ha hecho en el trabajo para ello. Por último en la Sección 1.3 se mostrarán los objetivos de este trabajo. Además se ha añadido la Sección 1.4 para facilitar el seguimiento del lector por la presente memoria.

En la Sección 1.1 se realizará una introducción breve al trabajo realizado. En la Sección 1.2 se explicará por qué es importante abordar la tarea del análisis de propaganda y qué se ha hecho en el trabajo para ello. Por último en la Sección 1.3 se mostrarán los objetivos de este trabajo. Además se ha añadido la Sección 1.4 para facilitar el seguimiento del lector por la presente memoria.

1.1. Introducción

Gracias a Internet cualquier usuario pueda crear su propio blog, hacerse un perfil en una red social, crear su propia página web, etc., lo que supone un gran avance hacia la libertad de expresión puesto que puede opinar sobre cualquier tema y debatir de ello con otros usuarios.

Entre ese gran flujo de información se esconden muchas noticias falsas, *fake news*, que son generadas muchas veces sin pensar en las consecuencias que estas pueden llevar. Hay medios de comunicación que también difunden este tipo de noticias, lo que acaba provocando un sesgo en la opinión de sus lectores o seguidores.

Por ello, aunque la conectividad que tenemos hoy en día sea un avance, es importante no descuidar aquellas noticias que pueden llevar a la confusión, ya que muchas de ellas suelen estar manipuladas por sus propios autores, buscando que los lectores acaben pensando exactamente lo mismo que ellos

sin contrastar la información.

Algunos ejemplos del uso de las *fake news* para la manipulación de la opinón de la gente lo encontramos en la campaña a la presidencia de los Estados Unidos de 2016, en la que Donald Trump siguió una estrategia de desinformación en grupos pequeños de todo el país. Con este método logró convencer a la gente de que le votasen a él y no a la oposición, obteniendo así la presidencia [6].

También se han dado todo tipo de *fake news* con la Covid19, donde se han utilizado estrategias de difusión a través de redes sociales, medios que están sesgados a una ideología política, etc., con tal de llegar al mayor público posible y así utilizar la pandemia como críticas a los diferentes gobiernos de cara a unas futuras elecciones, o por parte de los propios gobiernos con tal seguir ganándose la confianza de la gente. Algunos ejemplos de este tipo de noticias durante la pandemia son las manifestaciones contra las mascarillas en Madrid,[1] o la del cantante español Miguel Bosé argumentando que la Covid19 es una mentira creada por todos los gobiernos.[2]

Estos ejemplos nos hacen ver que en cualquier momento podemos recibir este tipo de noticias que, sin darnos cuenta, pueden condicionar nuestra opinión, ya sea respecto a un político, una vacuna, o una medicación entre otras cosas. Por lo que los usuarios deben estar siempre alertas a las noticias que leen y tratar de contrastar para poder hacer frente a cualquier intento de desinformación.

1.2. Motivación

El hecho de que haya tal cantidad de noticias que puedan ser falsas, suscita la creación de un sistema que sea capaz de adelantarse a su difusión, ya que para una persona resulta casi imposible poder detectar este tipo de noticias debido a que no podrá contrastar toda la información, y tampoco podrá estar siempre alerta de que una noticia sea falsa o no.

Por ello este trabajo busca crear un sistema que sea capaz de detectar este tipo de noticias, más aún dentro del ámbito de la política, donde cada vez se está utilizando más con tal de poder obtener más votos de cara a unas elecciones.

[1] https://www.elmundo.es/espana/2020/08/16/5f396a9c21efa0fd5a8b45fc.html

[2] https://www.abc.es/estilo/gente/abci-miguel-bose-dice-coronavirus-gran-mentira-gobiernos-202006051022_noticia.html

El tipo de noticias dentro del ámbito político suele utilizar técnicas propagandísticas para lograr una mayor difusión, consiguiendo una mayor confusión por parte de los lectores, y finalmente sesgar su opinión.

Dado que una persona no puede contrastar toda esta información, necesita de un sistema automático que sea capaz de procesarla y avisarle en caso una noticia pudiera ser falsa. El área de la IA que trata el texto como dato es el Procesamiento del Lenguaje Natural (PLN), por lo que el procesamiento de estas noticias se tiene que realizar desde este área.

Por tanto este trabajo se enfocará desde el PLN, y se analizarán las noticias para determinar sin son noticias propagandísticas o no, ya que son este tipo de noticias las más utilizadas en la difusión de *fake news* en política. Para ello se propone un modelo de *Deep Learning* para la clasificación de noticias propagandísticas, que permita la identificación de estas noticias para impedir su difusión en la red.

1.3. Objetivos

Los objetivos que se persiguen en este trabajo son:

1. **Estudio de la tarea de detección de noticias propagandísticas**: Se deberá estudiar cómo abordar la tarea de detección de noticias propagandísticas.
2. **Estudio del estado del arte en la detección de noticias de propaganda**: Se estudiará el estado del arte para la detección de noticias propagandísticas.
3. **Estudio de los conjuntos de datos disponibles**: Se valorarán los diferentes conjuntos de datos disponibles y se seleccionará uno sobre el que abordar la tarea.
4. **Creación de un modelo de *Deep Learning* que resuelva el problema**: Se desarrollará un modelo basado en *Deep Learning* que resuelva esta tarea.

1.4. Estructura de la memoria

Tratando de facilitar al lector su seguimiento por la presente memoria se va a exponer brevemente el contenido de los capítulos restantes.

- **Capítulo 2**: Muestra en qué consiste en análisis de propaganda, y se muestran las bases de *Deep Learning* utilizadas en este trabajo.

- **Capítulo 3**: Se muestran las diferentes colecciones de datos para la detección de *fake news* y el marco experimental, que contiene el caso base y el modelo propuesto para la detección de noticias de propaganda.
- **Capítulo 4**: Recoge las conclusiones del proyecto y las vías futuras para abordar este problema.

Capítulo 2

Contexto

En este capítulo se procede a asentar las bases que se utilizarán en este proyecto. En la Sección 2.1 se realizará una introducción a la propaganda y a su análisis. En la Sección 2.2 se explicarán los conceptos de *Deep Learning* utilizados para el desarrollo de este trabajo.

2.1. Propaganda

La difusión de las noticias falsas se da en noticias de cualquier ámbito, ya sea política, medicinao o informática, y en función de las intenciones del autor el contenido de estas variará con tal de convencer al lector o espectador. Es por ello que las *fake news* se pueden identificar por diferentes tipos como los dados por "US News & World Report":[1]

- **Sátira**: Busca ridiculizar o burlarse de personas o situaciones, pero no intenta que se le tomen enserio.
- **Bulo**: Trata de convencer al lector o espectador de la veracidad de una historia inventada.
- **Propaganda**: Engaña al lector o espectador para que crea determinadas ideas políticas y sociales.

La sátira no trata de engañar al usuario, sino que pretende que el usuario sepa que lo que está viendo o leyendo no es real, sino que se trata de la caricaturización de personas o de situaciones. Además la sátira y los bulos buscan inventarse una historia, mientras que la propaganda utiliza una mezcla verdades y mentiras para confundir a los usuarios [30].

[1] https://www.usnews.com/news/national-news/articles/2016-11-14/avoid-these-fake-news-sites-at-all-costs

La propaganda puede realizarse mediante diferentes técnicas, según el autor [7], y se puede llegar hasta las 69 técnicas.[2] A la hora de analizar si una noticia es propagandística o no se pueden utilizar todas estas técnicas o se puede utilizar un número reducido, ya que muchas de ellas pueden ser muy similares entre sí.

Antes de ver algunas de las posibles técnicas de propaganda que se utilizan hoy en día, es importante saber qué es la propaganda. La definición fue dada por el Instituto para el Análisis de Propaganda [10], y la definieron como sigue:

Definición 1 *"La propaganda es la expresión de una opinión o acción por individuos o grupos, deliberadamente diseñada para influir en las opiniones o acciones de otros individuos o grupos con un determinado fin."*

Las técnicas de propaganda difieren según el autor que esté trabajando con las noticias, con 7 técnicas como [24], con 24 técnicas como [35], las 69 mencionadas en Wikipedia, o 18 como en [7]. Tal variación se da porque algunos autores o bien ignoran algunas técnicas, porque no las utilicen en su trabajo, o porque utilicen definiciones para algunas técnicas que incluyan a otras. Algunas de las técnicas de propaganda que se utilizan son las siguientes:

- **Idioma cargado (*Loaded Langueage*)**: El lenguaje utilizado tiene una carga sentimental fuerte. Por ejemplo: "In a glaring sing of just how *stupid and petty* things have become."
- **Insultos (*Name calling or labeling*)**: Utilizar como objeto de propaganda algo que la gente tema u odie. Por ejemplo: "Manchin says Democrats acted like *babies* at the SOTU".
- **Exageración o minimización (*Exaggeration or minimization*)**: Repetir una idea exagerándola para hacerla parecer peor, o quitarle importancia a algo que la tenga. Por ejemplo: *"they can't even stomach being in the same room as the president"*
- **Apelar al miedo o a los prejuicios (*Appeal to fear/prejuice*)**: Se busca el apoyo de una idea inculcando pánico o ansiedad en la gente basándose en prejuicios. Por ejemplo: ' *"A dark imprenetable and irreversible winter of persecution of the faithful by their own shepherds will fall*"

[2] https://en.wikipedia.org/wiki/Propaganda_techniques

- **Ondear la bandera (*Flag-waving*)**: Utilizar el sentimiento nacionalista para justificar una idea o promoverla. Por ejemplo: "*to stop the will of We the People!!!* It's time to jail Mueller"

- ***Whataboutism***: Tachar de hipócrita a otro sin motivos para descartar su razonamiento. Por ejemplo: "President Trump–*who himself avoided national military service* in the 1960's– keeps beating the war drums over North Korea"

- **Testimonial**: Se da por hecho que un testimonio es verdad por la persona que lo ha dicho pese a que esta persona no es experta en la materia, y se asume su razonamiento sin preguntarse si es o no es cierto. En [7] forma parte de la técnica de **apelar a la autoridad (*Appeal to authority*)**, ya que esta se utiliza bajo el mismo pretexto que la anterior siempre que la persona que de el testimonio sea experta en la materia. Por ejemplo: "*Monsignor Jean-Franois Lantheaume, who served as first Counsellor of the Nunciature in Washington, confirmed that Vigan said the truth. That's all*""

La mejor manera de hacer efectiva la propaganda es que pase inadvertida. De forma que cuando una persona lea un artículo periodístico en un medio de comunicación, ésta no pueda identificarlo como propagandístico. Así podría comenzar a compartirlo con su círculo cercano y comenzar a propagar esta noticia.

Las noticias propagandísticas pueden ser utilizadas por los diferentes políticos, entidades públicas u organizaciones, y por ello surgen organizaciones que tratan de dar con estas noticias y dar una explicación del por qué son falsas. Algunas de estas organizaciones son PolitiFact.com,[3] Newtral[4] o Maldito Bulo.[5] Aún así, el definir qué noticias son propagandísticas o no resulta un reto, puesto que estas mismas organizaciones que luchan por detectarlas pueden sesgar, directa o indirectamente, la anotación de las noticias.

No hay que confundir la propaganda con la desinformación pese a que la propaganda se pueda utilizar con estos fines. La propaganda puede mezclar hechos que no son verdad pero tampoco mentira, y sus intenciones no tienen por qué ser necesariamente maliciosas [6]. Sin embargo en la práctica tanto la desinformación como la propaganda son usadas simultáneamente para llevar las *fake news* al mayor número de personas posibles.

[3] https://www.politifact.com/
[4] https://www.newtral.es/
[5] https://maldita.es/malditobulo/

Por tanto son necesarios sistemas que sean capaces de detectar estas noticias, y es por ello que este trabajo se centra en el desarrollo de modelos de *Deep Learning* para la identificación de noticias de propaganda, y no en la búsqueda de las técnicas de propaganda en estas noticias.

2.2. Deep Learning

Es necesario la creación de un sistema que procese de forma automática todas estas noticias para poder avisar a los lectores o espectadores de ello. Para procesar todas las noticias de forma automática se debe realizar usando técnicas propias del PLN, que permite entender la información subyacente en un texto.

Durante mucho tiempo las técnicas que más se han utilizado para resolver los problemas de PLN son enfoques de *Machine Learning* que utilizan modelos lineales como las Máquinas de Soporte Vectorial (SVM)[12] o modelos de Regresión Logística [11].

Sin embargo, en los últimos años ha ganado cada vez más fuerza los modelos basados en redes neuronales, pues que han logrado ser el estado del arte en múltiples tareas del PLN como son el análisis de sentimientos, la categorización de noticias o de tópicos, *question answering*, etc. Es por esto que se ha decidido abordar la tarea del análisis de propaganda mediante un modelo de *Deep Learning*.

Dado que la tarea a abordar se centrará en la clasificación de texto, el sistema recibirá un documento y deberá decir si es o no propaganda, y por ello se debe estudiar en la literatura los diferentes enfoques utilizados.

En clasificación de texto se han utilizado diferentes arquitecturas neuronales [25], algunas de ellas son:

- Modelos basados en redes *feed-forward*: Son las redes más básicas, y están directamente inspiradas en el cerebro humano. Cada neurona es una unidad de computación que tiene una entrada y una salida. Las entradas tienen un peso asociado que lo multiplican por cada elemento de entrada a la red, y los suman para aplicar posteriormente una función no lineal sobre el resultado y generar una salida. Esta función no lineal es la función de activación de la neurona. En este tipo de redes todas las neuronas están conectadas entre sí, simulando un cerebro humano [11].

- Modelos basados en Redes Neuronales Recurrentes (RNN): Estos modelos procesan el texto como una secuencia a palabras, combinando

las independencias entre palabras. Se verán más adelante en la Sección 2.2.2. [11].

- Modelos basados en Redes Convolucionales (CNN): Estos modelos buscan reconocer patrones en el texto, como pueden ser frases clave que expresan sentimientos o para reconocer tópicos. Este tipo de arquitecturas es muy utilizada dado que ha logrado ser estado del arte en diversas tareas de PLN [11].

- Modelos con mecanismos de atención: Son útiles para obtener las palabras correladas en el texto. Se verán en la Sección 2.2.3.

- Modelos Híbridos: Son aquellos modelos que combinan las redes propuestas anteriormente, como por ejemplo, mecanismos de atención con redes RNN o CNN, o la misma combinación de redes CNN y RNN.

- Modelos basados en transformadores (Transformers): Aplican mecanismos de *Self-Attention* en paralelo a todas las palabras de una oración para así procesar la influencia que hay entre ellas. Estos modelos permiten una ejecución en paralelo superior a las CNN y también a las RNN. Un ejemplo de este tipo de modelos se verá en la Sección 2.2.4.

Para poder realizar este tipo de redes es muy importante la entrada, es decir, las características que se le proporcionarán a la red. En el caso de las redes neuronales la representación de las características, es decir, las palabras y signos ortográficos, serán los *word embeddings* [38] y se verán en la Sección 2.2.1.

2.2.1. *Word Embeddings*

En los modelos clásicos de *Machine Learning*, cada una de las características tiene su propia dimensión, y estas son las características dispersas. Para poder utilizar las diferentes características a la vez, hay que combinarlas ya sea mediante una concatenación, una suma o una combinación, para generar la entrada al modelo. Además estas características dispersas tienen una dimensionalidad alta ya cada dimensión equivale a una característica. Con los *word-embeddings* se logra una representación densa de las características, en la que cada característica está mapeada a un vector, logrando así una dimensionalidad baja. Con esto se logra que la dimensionalidad de las redes neuronales sea menor y que por tanto no tenga tanta carga computacional [38].

Una representación densa es un vector d-dimensional donde, características similares corresponden a vectores similares.

Además al tener una menor dimensionalidad se logra una mejor computación de los datos, ya que las redes tienen problemas tratando datos de alta dimensionalidad. Pero no solo se obtiene una ventana computacional, sino que además se logra una gran capacidad de generalización, ya que si características similares tienen vectores similares, aunque una característica se haya dado mucho y otra menos, el contexto en el que se han utilizado será muy parecido y por tanto se podrán generar relaciones estadísticas entre ellas.

Es por ello que los *embeddings* son un componente para la representación del conocimiento y muy importantes para el uso de las redes neuronales, ya que se usan para proporcionar la entrada de las diferentes características que se quieran añadir al problema. La obtención de los *word-embeddings* puede hacerse de diferentes métodos [11].

- **Inicialización aleatoria**: Se inicializan los vectores para todas las características disponibles pasando cada palabra a un vector de dimensión d y que obtiene una representación de números generados aleatoriamente en un intervalo puesto. En la práctica este enfoque se usa para características que se repiten mucho como los *Pos-Tags* (características morfológicas).
- **Pre-Entrenamiento supervisado**: Este enfoque se aplica cuando se tienen dos tareas, siendo la segunda auxiliar, y teniendo muchos más datos de la segunda tarea que de la primera. Así se entrenarían los *embeddings* de la segunda tarea, la auxiliar, para que ayude a resolver la primera tarea correctamente.
- **Pre-Entrenamiento no supervisado**: Es el caso más común, cuando sólo se tiene una tarea. La idea con este tipo de entrenamiento es que las palabras con significados similares tengan *embeddings* similares, Figura 2.1. Algunos algoritmos utilizados son *word2vec*[23], *Glove* [28] y *Fasttext* [17].

2.2.2. Redes Neuronales Recurrentes (RNN)

Las RNN procesan el texto como una secuencia de palabras y tratan de capturar las dependencias que existen en la estructura del texto. Las primeras RNNs, Figura 2.2, resultaron ser muy básicas y no conseguir realmente mejoras frente a redes como las *feed-forward*. Ante esto aparecieron variaciones de como las redes *Long-Short-Term Memory* [13], o LSTM (arquitectura de RNN más popular), que fue diseñada para poder capturar las dependencias en textos largos [25].

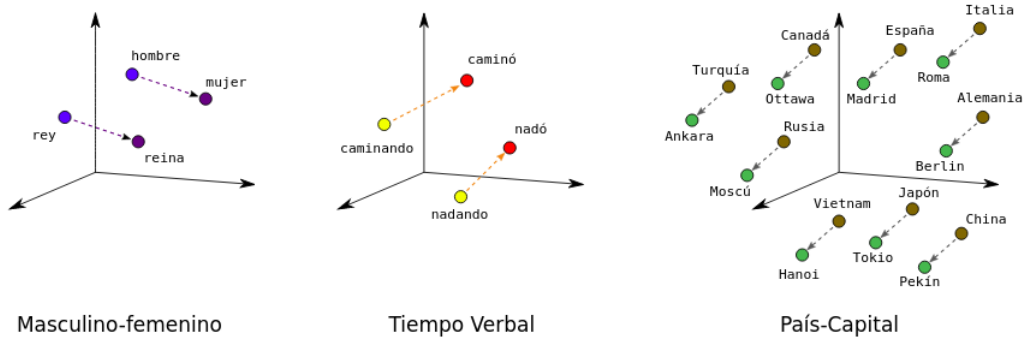

Figura 2.1: Ejemplo de *Embeddings*.
Fuente: https://developers.google.com/machine-learning/crash-course/embeddings/translating-to-a-lower-dimensional-space

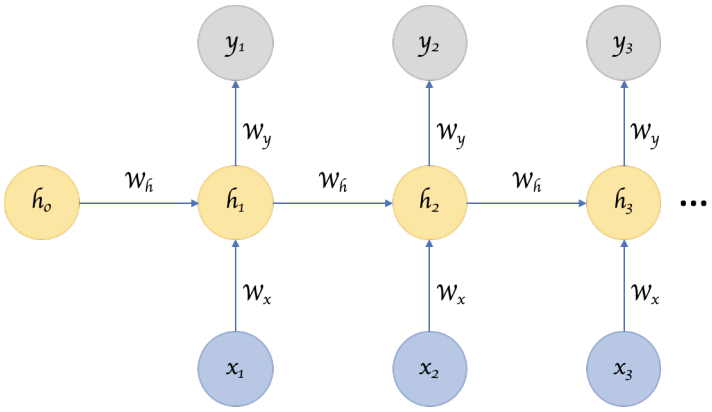

Figura 2.2: Ejemplo de RNN clásica.
Fuente: https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce

La idea principal tras las LSTM, Figura 2.3, es que introducen unos vectores, celdas, que pueden almacenar el gradiente a lo largo del tiempo, es decir, que "tienen memoria". Además se puede acceder a esta memoria mediante puertas, funciones matemáticas que simulan puertas lógicas. En las LSTM cada entrada recibida se utiliza para saber cuánto de esta se debe escribir en la celda, y cuánto "olvidar".

Las LSTM son el tipo de RNN más utilizado, seguido de las *Gated Recurrent Unit* (GRU). [5] Esta red surge por la complejidad de las LSTMs, y al igual que estas funcionan mediante un mecanismo de puertas pero no tienen un componente de memoria separado y son más eficientes computacionalmente.

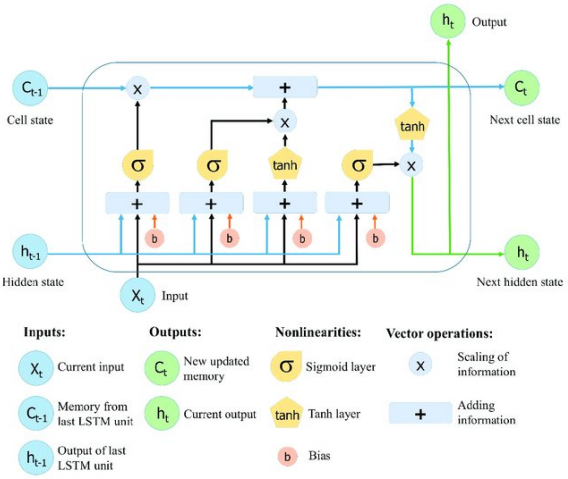

Figura 2.3: Ejemplo de LSTM [20].

Sin embargo las redes neuronales con GRU no han logrado superar a las RNN con LSTM, por lo que han surgido variables de la propia LSTM como pueden ser las *Tree-LSTM* [31] o las *chain-structured LSTM* [39].

Además hay variaciones que afectan a todas las RNN, y son las redes bidireccionales (Bi-RNN). Mientras que una RNN permite analizar la palabra de izquierda a derecha o de derecha a izquierda, una Bi-RNN permite el procesamiento de izquierda a derecha y viceversa.

Una Bi-RNN trabaja con dos estados diferentes, *forward* (al igual que una RNN básica), y *backward* para cada una de las posiciones de la entrada. De esta forma se consigue que pese a que las dos RNN se ejecuten de forma independiente, el gradiente se aplique a en ambas a la vez y por tanto puede acceder tanto *al futuro* como *al pasado*. Una comparativa entre una LSTM y una BiLSTM se puede encontrar en la Figura 2.4.

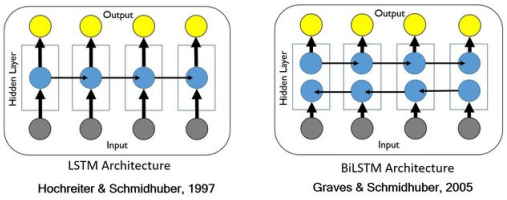

Figura 2.4: LSTM (izquierda) - BiLSTM (derecha) [26].

2.2.3. Mecanismos de atención

Los mecanismos de atención se basan en la atención visual de las personas a la hora de observar una imagen, en las regiones en las que se fija al mirarla, o en las palabras correladas que se observan al leer un documento [37]. Es por ello que la atención se puede interpretar como un vector de pesos que dará mayor importancia a las regiones más importantes en la clasificación del documento, o en la predicción de la siguiente palabra entre otras tareas. En caso de predecir la siguiente palabra a la escrita o a la traducida previamente, el vector de pesos actuaría ayudando al modelo, dando la correlación entre las palabras predichas hasta ese momento con las posibles opciones a predecir para elegir la más próxima posible.

Utilizando los mecanismos de atención, que en PLN se diseñaron para traducción de texto [21], se lograron modelos que se mantuvieron como estado del arte, además de ser un mecanismo que permite dar más explicabilidad a los modelos.

Algunos de los mecanismos son los presentados en [21], en los que se presentan los mecanismos de atención local y atención global, que se utilizarán en este proyecto, y que toman como entrada la salida de una LSTM (h_i), y los estados ocultos de esta (s_t).

Con los mecanismos de atención se trata de obtener el contexto (c_t) de la sentencia en tiempo t, para así poder realizar una mejor predicción. Para calcular el contexto hay que, primero, comprobar el alineamiento entre las palabras de entrada en las posiciones i y t (y_t, x_i), tal y como se muestra en la Ecuación 2.1.

$$\begin{aligned} c_t &= \sum_{i=1}^{n} a_{t,i} h_i \\ a_t(s) &= align(y_t, x_i) \\ &= \frac{exp(score(s_{t-1}, h_i))}{\sum_{s'} exp(score(s_{t-1}, h_{i'}))} \end{aligned} \tag{2.1}$$

Según la atención que se esté aplicando, el tipo de función de *score* varía. En el caso de la atención global, Figura 2.5, se presentan tres alternativas que se muestran en la Ecuación 2.2. La función de score *scaled_dot* se ha añadido a las tres presentadas en el trabajo original, puesto que esta permite que el gradiente no converja rápidamente [33]. Para lograr esto se añade la dimensión de la entrada d_k:

$$score(s_t, h_i) = \begin{cases} s_t^T h_i & dot \\ s_t^T W_a h_i & general \\ W_a[s_t; h_i] & concat \\ \frac{s_t^T h_i}{\sqrt{d_k}} & scaled\_dot \end{cases} \tag{2.2}$$

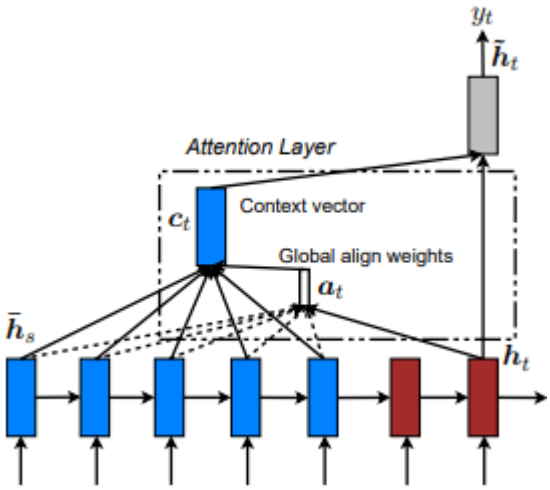

Figura 2.5: Atención global [21] .

Mientras que los mecanismos de atención global utilizan toda la frase o documento para conseguir las palabras clave, estos pueden tener problemas con textos largos debido al número de operaciones a realizar. Para evitar este problema se crean los mecanismos de atención local.

La atención local, Figura 2.6, fija una ventana pequeña de la entrada para así evitar el alto coste computacional que tiene la global. Para el cálculo de la atención local, el modelo primero genera una posición *alineada* p_t para la última palabra en el tiempo t. Después genera una ventana alrededor de esta palabra de tamaño $[p_t - D, p_t + D]$, siendo D el tamaño de la ventana elegido antes de ejecutar el modelo. Además se presentan dos opciones para calcular p_t [21]:

- Alineamiento *monotónico* (**local-m)**: Se fija $p_t = t$, asumiendo que la entrada y la salida están monotónicamente alineadas.

- Alineamiento *predictivo* (**local-p)**: El modelo predice la posición como se indica en la Ecuación 2.3.

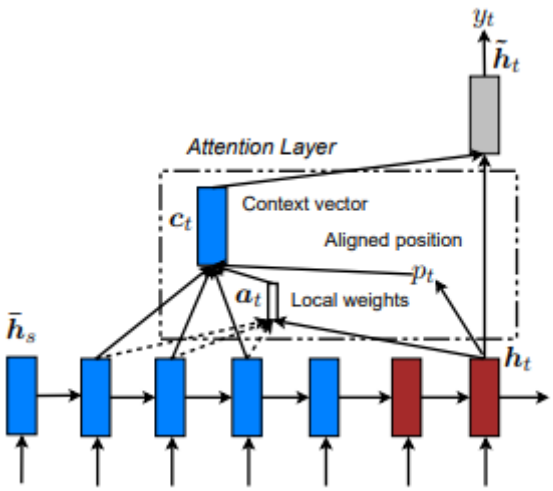

Figura 2.6: Atención local [21] .

$$p_t = S * sigmoid(v_p^T tanh(W_p s_t)) \tag{2.3}$$

Los parámetros W_p y h_t son parámetros del modelo que se aprenderán para predecir la posición, y S es el tamaño de la entrada. Además para favorecer el alineamiento de p_t se aplica una distribución gausiana sobre ésta. De forma que el alineamiento de los pesos quedaría como se define en la Ecuación 2.4.

$$a_t(s) = align(s_t, h_i) exp\left(-\frac{(s_t - p_t)^2}{2\alpha^2}\right) \tag{2.4}$$

Además de los mecanismos de atención local y global hay otros mecanismos de atención como los mecanismos de *self-attention* [33], o los de atención jerárquica.

Los mecanismos de *self-attention* se introdujeron en [4] y permiten que la red pueda captar las dependencias entre la palabra que se está analizando en el instante t con las anteriores hasta el momento. Un ejemplo del funcionamiento de esta red se ve en la Figura 2.7. Este mecanismo ha obtenido buenos resultados en tareas como *machine reading* o generación de descripciones de imágenes.

El mecanismo de *self-attention*, al igual que los anteriores mecanismos de atención, se presenta mediante la arquitectura *encoder-decoder*, muy utilizada en traducción de texto. Para utilizar este tipo de atención basta con aplicar las mismas operaciones de la Ecuación 2.1, solo que estas operaciones sólo se realizan con la salida de la capa LSTM.

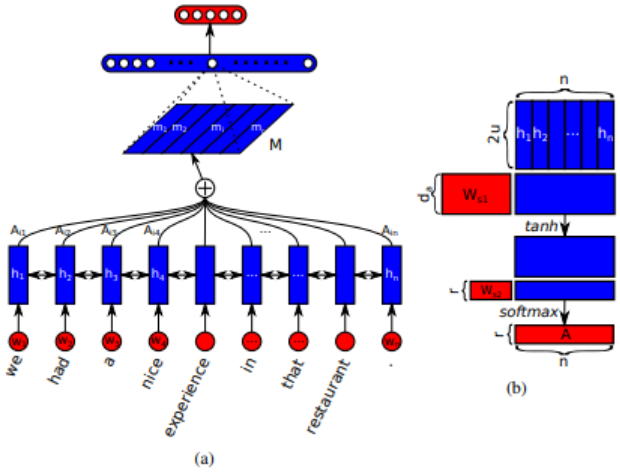

Figura 2.7: *Self-Attention*[4].

2.2.4. Modelos pre-entrenados

Otro enfoque que se da en PLN es el enfoque mediante el uso de modelos ya pre-entrenados, aplicando de esta forma *fine-tuning*. Con esto se logra partir de modelos que logran el estado del arte en diferentes áreas de PLN y entrenarlos para la tarea que se esté abordando. Algunos ejemplos de estos modelos son BERT [9], ALBERT [19] o ELMO [29].

En el caso de ELMO se trata de partir de unos *embeddings* pre-entrenados de una alta dimensión, y aplicarlos sobre una red, como puede ser una BiLSTM. Mientras que BERT parte de una estructura basada en *transformers* [33].

La arquitectura de BERT, Figura 2.8, es la de un *transformer* bidireccional multicapa basado en un encoder presentado en [33]. Por ello BERT utiliza *self-attention* bidireccional como mecanismo de atención, para que así cada uno de los *tokens* de entrada pueda tener el contexto de todo el documento y no sólo de la parte previa al mismo. Con BERT se puede hacer *fine-tuning*, de forma que se utiliza este modelo a la tarea que se está realizando, en este caso la clasificación binaria de propaganda, y para ello solo hay que añadirle la capa de salida deseada, la cual se verá en la Sección 3.2.3

Dado que en este proyecto sólo se ha utilizado BERT, y no se ha utilizado ELMO, ALBERT u otras variantes de BERT, sólo se ha explicado brevemente el funcionamiento de este y cómo es su arquitectura, y se ha decidido no entrar a explicar los otros modelos.

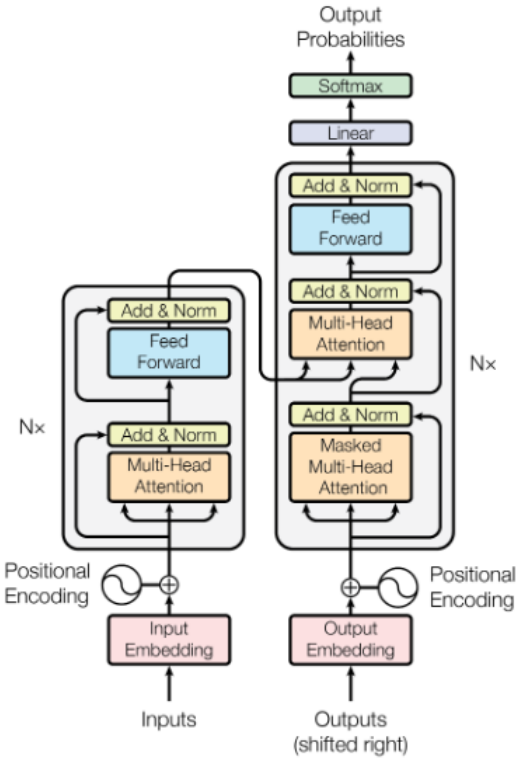

Figura 2.8: Arquitectura de BERT [33].

Capítulo 3

Clasificación Binaria de Propaganda

En este capítulo se presentará la colección de datos utilizada en este proyecto, el modelo que mejor resultado obtiene para ella, la propuesta realizada para resolver la tarea y un análisis de los resultados obtenidos. En la Sección 3.1 se mostrarán las diferentes colecciones de datos disponibles además de la utilizada. En la Sección 3.2 se presentará el marco experimental realizado sobre la colección de datos seleccionada, presentando la propuesta realizada. En la Sección 3.3 se realizará un análisis de los resultados obtenidos.

3.1. Colecciones de datos disponibles

La detección de *fake news* y en concreto el análisis de propaganda se puede realizar de diferentes formas [7, 3, 30]. Para poder realizar esta tarea es necesario disponer de una colección de datos anotada que indique si un documento es *fake* o no. Por ello se realizó una búsqueda de las diferentes colecciones de datos disponibles en la literatura para poder analizarlas, y elegir aquella que sea más conveniente para este trabajo. Inicialmente se realizó uno búsqueda de las colecciones de datos disponibles para *fake news* y no solo dedicadas a propaganda, por lo que se mostrarán las colecciones más relevantes, y son las siguientes.

TSHP-17[30] Se busca la detección de *fake news* en noticias sobre política. Estas noticias están evaluadas por la organización PolitiFact.com y se determina si la declaración es sátira, bulo, propaganda o es una noticia real.

PTC[7] Esta colección de datos está compuesta por declaraciones de políticos estadounidenses, y también proceden de la organización PolitiFact.com,

solo que esta colección busca, no solo saber qué noticias son propagandísticas y cuáles no, sino el por qué lo son, basándose en las diferentes técnicas de propaganda que se dan [7].

***Liar Liar Pants On Fire* [34]** Colección de datos compuesta por estamentos realizados por políticos y figuras públicas estadounidenses, y se busca saber el grado de veracidad o falsedad según 6 puntos proporcionados, *pants-fire*, falso, apenas cierto, medio cierto, muy cierto y verdadero. Esta colección es similar a la colección TSHP-17, solo que aquí no se busca diferenciar entre los diferentes tipos de *fake news* que hay, sino que sólo se busca saber si es o no falsa una noticia.

***Profiling Fake News Spreaders On Twitter.*[1][2]** Los usuarios juegan un papel clave en la distribución de *fake news* en la red, por lo que con esta colección de datos se busca, a través de tuits de diferentes usuarios, saber localizar aquellos usuarios que más distribuyen noticias falsas, para así lograr detener la difusión de este tipo de noticias.

***Fake News Detection Challenge.*[3]** Dado que muchas noticias buscan solamente lograr difusión a través de un titular novedoso, ya que mucho usuarios no leen las noticias y se quedan en éste, se ha creado esta colección de datos para comprobar que las noticias publicadas sí relacionan el título de la noticia con el contenido de la misma, y que no sólo buscan la difusión rápida del medio para difundir información falsa.

***HealthNewsReview* [8]** Las *fake news* también llegan a las noticias sobre salud, sobre todo tras la pandemia de la covid19. Dado que este tipo de noticias se daban mucho antes, por ejemplo en noticias relacionadas con la homeopatía, se ha creado una colección de datos en la que el objetivo no es sólo decir si una declaración a nivel médico es falsa, sino además se debe dar una respuesta del por qué es falsa.

QProp [3] parte de TSHP-17 3.1 y aumenta la cantidad de datos para poder entrenar mejor un modelo. Además centra la tarea en una clasificación binaria de propaganda y no propaganda. Al ser una colección que permite realizar una clasificación binaria y que además permite la iniciación en la detección de propaganda y *fake news*, se ha decidido que se trabajará con esta colección para así hacer un primer acercamiento a esta tarea que es cada vez más importante.

[1]Competición: https://pan.webis.de/clef20/pan20-web/author-profiling.html
[2]Disponible en: https://zenodo.org/record/3692319
[3]http://www.fakenewschallenge.org/

3.2. Marco Experimental

En esta sección se va a mostrar cómo se ha abordado la tarea del análisis de propaganda sobre la colección de datos seleccionada, **QProp** [3].

En la Tabla 3.1 se pueden ver los datos disponibles para esta colección. Además se tienen 3 conjuntos de datos, *train-test-dev*(validación), por lo que los resultados se mostrarán para los conjuntos de *test* y de *dev*.

Etiqueta	**Fuentes**	**Artículos**	**Train**	**Dev**	**Test**	**Longitud (Tokens)**
Propaganda	10	5.737	4.021	575	1.141	$1084,46 \pm 890,59$
No Propaganda	94	45.557	31.972	4.564	9.021	$620,31 \pm 518.92$
Total	**104**	**51.294**	**35.993**	**5.139**	**10.162**	$672,22 \pm 590,98$

Tabla 3.1: Estadísticas de QProp para el primer experimento [3].

Por la naturaleza de esta colección de datos, puesto que incluye la fuente de la que provienen las noticias, se pueden realizar dos experimentos:

- **Experimento 1**: Se hace una clasificación binaria de propaganda/no-propaganda sin tener en cuenta la fuente de las noticias, con los datos mostrados en la Tabla 3.1.
- **Experimento 2**: Se divide la colección de datos en dos conjuntos, con diferentes fuentes en cada cada conjunto, que ayudan a comprobar si el modelo distingue la propaganda ante nuevas fuentes o si por el contrario está aprendiendo el estilo de redacción de estas. Esta nueva división de datos se muestra en la Tabla 3.2.

Class	**Train**		**Test**	
	Fuentes	**Artículos**	**Fuentes**	**Artículos**
Propaganda	5	2.802	5	2.935
No Propaganda	47	22.776	47	22.781
Total	**52**	**25.578**	**52**	**25.716**

Tabla 3.2: Estadísticas de QProp para el segundo experimento [3].

Para esta colección de datos se parte de un caso base presentado en [3], y que está basado en máxima entropía y se presentará en la Subsección 3.2.1.

Además se presentarán las propuestas realizadas en este trabajo para mejorar los resultados del caso base. La primera propuesta es un clasificador sin *fine-tuning* que se presentará en la Subsección 3.2.2, y la segunda propuesta es un clasificador con *fine-tuning* que se presentará en la Subsección 3.2.3.

Para poder realizar la experimentación necesaria que se va a presentar en esta sección, se usará el *framework* TensorFlow [1], concretamente la versión 2.2.0 de este. Además se realizará bajo Python 3.8.2 y en un ordenador con Ubuntu 20.04 LTS, con 16GB de RAM y una gráfica Nvidia GTX 1050 4GB y un procesador Intel i7-7700H. El código realizado en para este trabajo está disponible en Github.[4]

3.2.1. Clasificador de Máxima Entropía

Para **QProp** se tiene como mejor modelo un clasificador de máxima entropía con los parámetros por defecto y diferentes características que aportan información sobre el estilo de escritura, vocabulario específico de palabras utilizadas en artículos periodísticos, vocabulario que se utiliza en textos sesgados por extrema izquierda y extrema derecha, etc. [3]. Dado que este es el modelo que mejores resultados tiene para la colección de datos seleccionada será tomado como un caso base.

Como se ha visto en la Sección 3.1, la colección **QProp** permite realizar dos experimentos. El primer experimento sirve para clasificar si un documento es o no propagandístico dentro de un dominio cerrado, *In-domain*.

El segundo experimento surge de la hipótesis de que los modelos aprenden el estilo de escritura de las fuentes que ven en entrenamiento y no a distinguir realmente si es propaganda o no una noticia, por lo que ante noticias de nuevas fuentes no tendría un correcto funcionamiento, *Out-of-domain*, por lo que surge el segundo particionado de los datos.

En el primer experimento los tres conjuntos de datos mostrados en la Tabla 3.1 tienen noticias de las mismas fuentes, es decir, si en *train* hay noticias de periódicos como "El País" y "El Mundo", también habrá noticias de estas fuentes en los conjuntos de *test* y *dev*, ya que se está ante un dominio cerrado.

El segundo experimento consta de dos conjuntos de datos, como se muestra en la Tabla 3.2, y aquí las fuentes de *train* son totalmente diferentes a las de *test*, para así poder comprobar si está aprendiendo a diferenciar propaganda de no propaganda o si está aprendiendo la fuente, por lo que sería

[4] https://github.com/AlArgente/TFM

Out-of-domain.

Además de ambos experimentos se tiene que la clase propaganda es minoritaria, de un 11 % de esta respecto a un 89 % de la clase no propaganda. Al estar ante un desbalanceo de clases como este no se utilizará como métrica el *accuracy*, sino que se utilizará la *F1* para la clase positiva a la hora de mostrar los resultados.

Para el clasificador de máxima entropía se han utilizado combinaciones de diferentes características. Estas son las siguientes:

- word n-grams: Se utilizaron word n-gramas para n=[1,3] con pesos por tf-idf.
- lexicon: Dado que las técnicas de propaganda pueden tener cierto vocabulario específico. Por ello se usan los lexicon de Wiktionary, LIWC [27], los subjetivos de Wilson [36], *Hyland hedges* [16], *Hooper's assertives* [14], formando así un total de 18 lexicons entre todos los anteriores [3].
- voc. richness y readability: Al ser muchos artículos sesgados por la influencia política, ya sea extrema derecha o extrema izquierda, se tiene un estilo de escritura diferente al de artículos neutros.
- char n-grams: Se utilizaron 3-gramas, con pesos por tf-idf.
- nela: NEws LAndscape features (NELA) [15] son 130 características basadas en el contenido, en las que se extraen características como el sesgo, el sentimiento, etc.

Para el conjunto de datos de *test* obtiene *82.89 %* de F1, cuando el clasificador utiliza las características char n-grams + lexicon + voc. richness + nela, mientras que para el conjunto de *dev* obtiene **83.21 %** de F1 si recibe como entrada las características de char n-grams + nela.

Para el segundo experimento se obtiene como mejor resultado el clasificador que recibe como entrada todas las características mostradas al inicio de de esta misma sección salvo los word n-grams, y se obtiene como resultado **65,61**.

3.2.2. Clasificador sin *fine-tuning*

En esta subsección se procede a mostrar la propuesta realizada. Para ello se mostrará el primer modelo realizado y los diferentes cambios aplicados a este hasta llegar al modelo final.

Lo primero que hay que definir es la entrada a la red. Para definir la entrada es importante elegir la longitud máxima (*padding*), que tendrá cada documento y los *word-embeddings* pre-entrenados que se utilizarán, en caso de querer usar unos ya pre-entrenados.

Los *word-embeddings* que se probaron fueron tanto los de *Glove* [28] como los de *FastText* [17], pero finalmente se decidió utilizar únicamente los de *FastText*, ya que disponen de un *token* para representar palabras desconocidas, y ayuda al modelo a lidiar con palabras que no estén en el vocabulario utilizado para crear sus vectores.

Para poder hacer uso de los *words embddings* se debe establecer primero el *padding* que deben tener todos los documentos, y aquellos con una mayor longitud se recortarán, mientras que a los que tengan una longitud menor se les añadirán tantos '0' como se necesiten hasta llegar a esa longitud.

Modelo inicial basado en CNN y BiLSTM La primera red que se realizó fue un modelo que utilizaba tanto una capa convolucional como una capa biLSTM. La idea tras este modelo era que a través de la capa convolucional se lograsen destacar aquellas palabras que resultaban más importantes para la detección de propaganda, y que mediante la biLSTM se lograsen captar las dependencias en los documentos de las diferentes técnicas de propaganda. Tras la biLSTM se añadió una capa densa, una de dropout y una capa *GlobalMaxPool* antes de la capa *softmax*.

Con este primer modelo además se realizaron las pruebas para definir el *padding* de la entrada, seleccionando entre la media, la mediana, y la moda de la longitud de los documentos. No se utilizó el máximo porque el tamaño máximo de los documentos era mucho mayor al de la media, mediana y moda, y generaba ruido en los datos, además del incremento del coste computacional que suponía. Si se aplicaba la media se tenía un tamaño máximo de 583, con la mediana un máximo de 463 y con la moda un máximo de 282.

Esta selección no sólo se vió influenciada por el tamaño mostrado en la Tabla 3.1, sino que además se tuvo en cuenta el histograma Figura del tamaño de los tokens, Figura 3.1, que confirma esta elección.

Los resultados de estos experimentos se muestran en la Tabla 3.3.

Con esta primera arquitectura, Figura 3.2, se obtuvo una mejor puntuación de 74.40 de F1 sobre la clase positiva en test,[5] lo cual está bastante lejos

[5]Todos los resultados representan la media de una muestra de 10 ejecuciones.

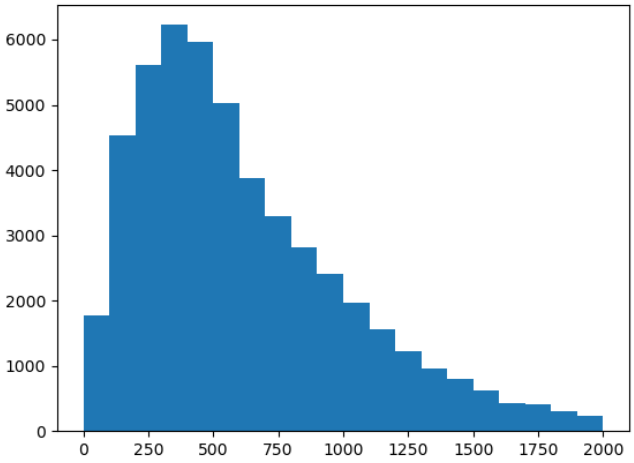

Figura 3.1: Histograma para la selección del padding.

Padding	**F1_Test(%)**
Mediana	**74.40**
Media	72.37
Moda	66.76

Tabla 3.3: Tabla de resultados del modelo CNN-biLSTM para la obtención del mejor valor máximo de entrada a la red.

del resultado obtenido por el modelo de máxima entropía, Sección 3.2.1.

Modelo basado en biLSTM Al tener un modelo complejo con resultados muy lejanos al modelo objetivo, se procedió a realizar una red más simple, Figura 3.3. Esta red tiene tan solo una capa biLSTM, y, al igual que el modelo anterior, se le añadió una capa densa tras esta seguida de una de *pooling*. Por último se añadía la capa *softmax* como capa de salida.

Los resultados de la esta red se muestran en la Tabla 3.4, y se observa que con esta red se tienen resultados cercanos al clasificador de máxima entropía, incluso superando el resultado obtenido por este para el conjunto de *dev*, pero aún no se lograba superarlo. Con *Deep Learning* se tiene la ventaja de que no es necesario generar las características lingüísticas, lo que supone una ventaja a nivel computacional.

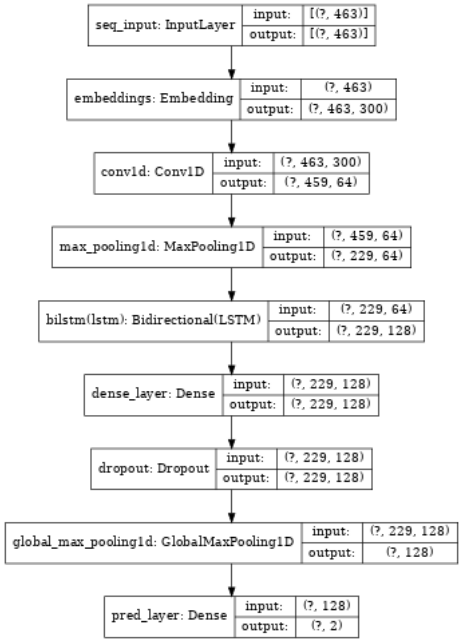

Figura 3.2: Arquitectura del modelo basado en CNN y BiLSTM.

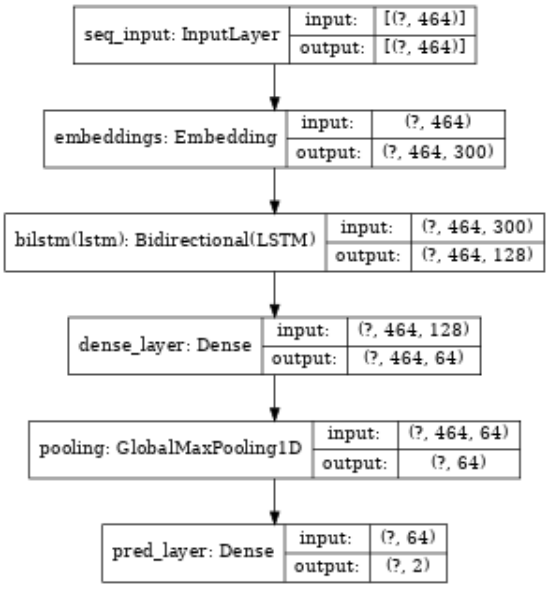

Figura 3.3: Arquitectura del modelo basado en biLSTM.

Modelo basado en biLSTM con Self-Attention Dado que se había logrado alcanzar los resultados que obtenía el clasificador de máxima entropía se buscó otro enfoque que ayudase a mejorar la red. Para poder mejorar la red se decidió aplicar mecanismos de atención, el primero que se aplicó fue

	F1_Test(%)	**F1_Dev(%)**
Máxima Entropía	**82.89**	83.21
BiLSTM Model	82.61	**83.35**

Tabla 3.4: Tabla de resultados del modelo basado en biLSTM para el experimento 1.

el mecanismo de *self-attention* ya que trata de buscar las relaciones que hay entre las palabras, y se buscaba poder localizar así las técnicas de propaganda en el texto. La arquitectura de la red, Figura 3.4, consta de una capa de *self-attention* entre las capas biLSTM y densa.

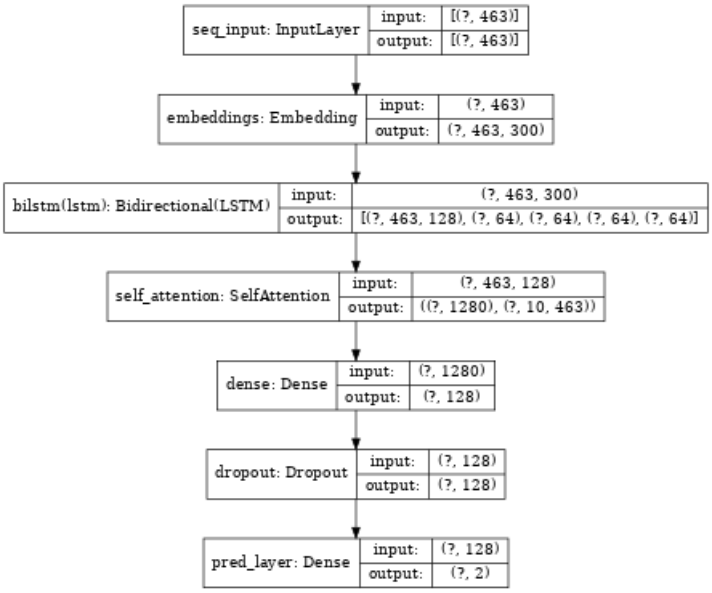

Figura 3.4: Arquitectura del modelo basado en biLSTM con *self-attention* para el experimento 1.

En la Tabla 3.5 se muestran los resultados obtenidos con este modelo, y se observa que con este modelo se supera al clasificador de máxima entropía en los conjuntos de *test* y *dev*.

Modelos basados en biLSTM y biLSTM con *self-attention* con *Spatial Dropout* Una vez se había logrado superar al clasificador de máxima entropía queda mejorar la red. Se decidió añadir tras la capa de *embeddings* una capa de *spatial dropout*(SpDo) [32] con un valor de 0.2 y se

procedió a ejecutar los dos anteriores modelos. El modelo de *self-attention* con esta capa extra se ve reflejado en la Figura 3.5.

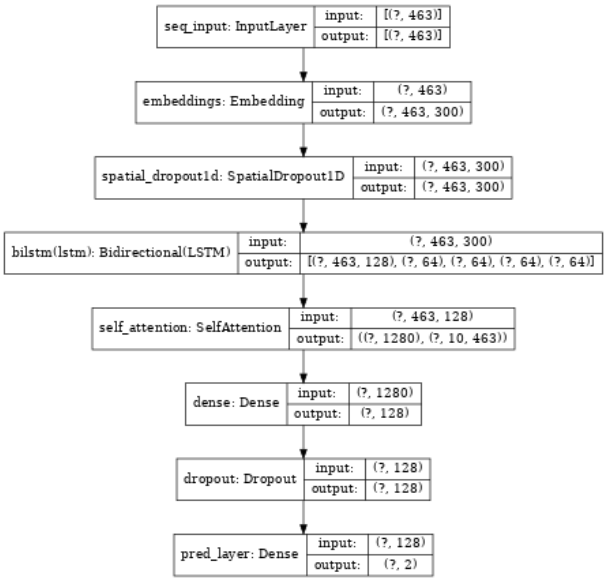

Figura 3.5: Arquitectura del modelo basado en biLSTM con *self-attention* con Spatial Dropout para el experimento 1.

En la Tabla 3.5 se muestran los resultados obtenidos. Aquí ambos modelos logran mejorar los resultados obtenidos previamente, pero el modelo simple es el que mejores resultados obtiene.

	F1_Test(%)	**F1_Dev(%)**
Máxima Entropía	82.89	83.21
BiLSTM + SpDo	**84.03**	**85.246**
BiLSTM+SelfAtt	83.11	84.05
BiLSTM+SelfAtt + SpDo	83.284	85.021

Tabla 3.5: Tabla de resultados del modelo BiLSTM y BiLSTM + SelfAttention para el experimento 1.

A causa de estos resultados, se decidió que los siguientes modelos incluirían esta capa para obtener mejores resultados. Igualmente se hicieron pruebas para comprobar que con esta capa se obtienen mejores resultados.

Modelo final basado en atención Dado que el mecanismo de *self-attention* no funcionó correctamente, se decidió aplicar los mecanismos de atención local y global, con el fin de superar al modelo basado en biLSTM inicial. Mientras que para el modelo de atención global sólo hay que elegir qué función de *score* se va a utilizar. Para el modelo de atención local también hay que definir el tamaño de ventana que se va a utilizar. La arquitectura de este modelo se puede ver en la Figura 3.6 y varía respecto a las anteriores, ya que se añade la capa de atención tras la capa biLSTM. Tras la capa de atención se añade la capa de *max pooling*, seguida de la capa *softmax* final.

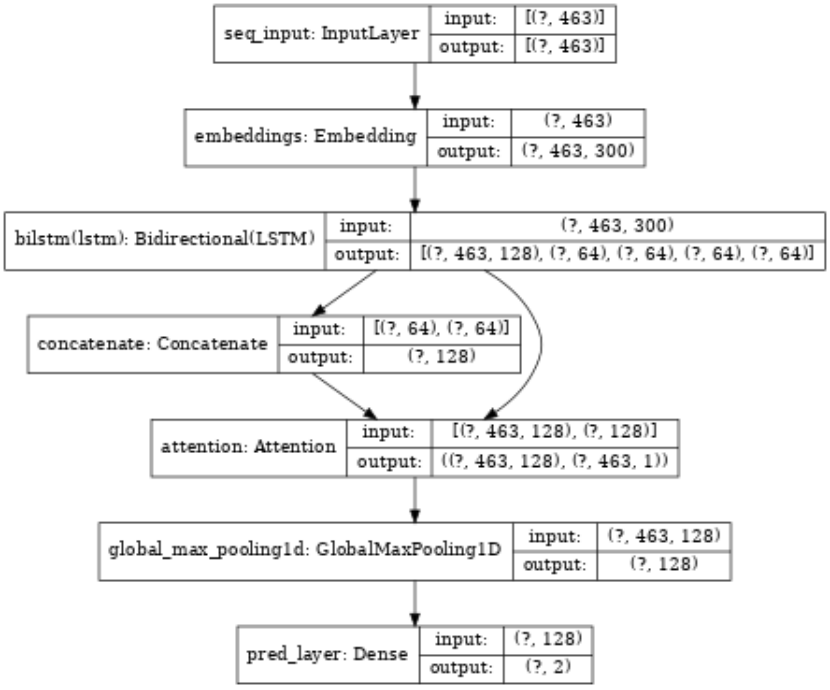

Figura 3.6: Arquitectura del modelo de atención local y global.

Las pruebas se hicieron con las diferentes operaciones comentadas en la Sección 2.2.3, tanto para el modelo de atención global como el local. Para el modelo de atención local se decidió elegir un tamaño de ventana con los siguientes valores [64, 100, 128].

Los resultados mostrados en la Tabla 3.6 muestra que el modelo con atención local con una ventana de 100 con utilizando *scaled-dot* como función para el *score*, obtiene el mejor resultado tanto en los conjuntos de test como de dev.

	F1_Test(%)	F1_Dev(%)
Máxima Entropía	82.89	83.21
BiLSTM + SpDo	84.03	85.246
BiLSTM + SpDo + Local (w=64, scaled-dot)	82.70	83.75
BiLSTM + SpDo + Local (w=100, scaled-dot)	**86.382**	**86.828**
BiLSTM + SpDo + Local (w=100, concat)	83.12	84.20
BiLSTM + SpDo + Local (w=128, scaled-dot)	85.35	86.17
BiLSTM + SpDo + Local (w=128, concat)	84.34	85.62
BiLSTM + SpDo + Global (scaled-dot)	85.41	86.638
BiLSTM + SpDo + Global (concat)	85.29	86.30

Tabla 3.6: Tabla de resultados de los modelos con atención local y global para el experimento 1.

Llegados al modelo final se procede a explicar con más detalle cada uno de los componentes de su arquitectura, la cual se muestra en la Ecuación 3.1:

- Representación: Capa de entrada a la red, representada por s palabras, $\{w_1, ...w_s\}$, y la representación cada palabra w_i como *word-embedding* tiene dimensión d, siendo la salida para cada documento cada vector de las palabras que lo forman $we_{s\times d}$.

- Codificación: Para codificar cada uno de los documentos se aplicó una capa biLSTM bidireccional, para así captar las dependencias de las palabras en ambos sentidos del documento. La capa biLSTM genera la salida $\mathbf{y}^1_{s\times 2h_{lstm}}$, y los estados ocultos de esta, $\hat{\mathbf{y}}^1_{s\times h_{lstm}}, \hat{\mathbf{y}}^2_{s\times h_{lstm}}$. Ambos estados ocultos se concatenan para así poder proporcionárselos a la capa de atención como entrada. Por último la capa de atención recibe tanto la salida de la biLSTM como sus estados ocultos, generando la capa de atención la salida $\mathbf{y}^3_{s\times 2h_{lstm}}$.

- Clasificación: Finalmente se encuentran las capas de clasificación, donde se incluyen la capa de *pooling* y la capa *softmax* que genera la salida. La capa de *pooling* recibe la salida de la capa de atención, y genera la salida $\mathbf{y}^4_{2h_{lstm}}$. Con la capa de *pooling* se busca reducir la dimensionalidad de la capa de atención, y así permitir que la capa *softmax* pueda procesar la salida de la capa de atención, indicando si el documento recibido es propaganda o no.

$$
\begin{aligned}
\mathbf{pred} &= softmax(y^4_{2h_{lstm}}) \\
\mathbf{y}^4_{2h_{lstm}} &= GlobalMaxPooling(y^3_{s\times 2h_{lstm}}) \\
\mathbf{y}^3_{s\times 2h_{lstm}} &= attention(y^2_{s\times 2h_{lstm}}, y^1_{s\times 2h_{lstm}}) \\
\mathbf{y}^2_{s\times 2h_{lstm}} &= concatenate(\hat{\mathbf{y}}^1_{s\times h_{lstm}}, \hat{\mathbf{y}}^2_{s\times h_{lstm}}) \\
\mathbf{y}^1_{s\times 2h_{lstm}}, \hat{\mathbf{y}}^1_{s\times h_{lstm}}, \hat{\mathbf{y}}^2_{s\times h_{lstm}} &= biLSTM(we_{s\times d})
\end{aligned}
\tag{3.1}
$$

Para todos estos modelos se utilizaron los mismos hiperparámetros, ya que fueron comprobados con los experimentos realizados para la obtención del tamaño de entrada, Tabla 3.3, y estos parámetros se muestran en la Tabla 3.7.

Hiperparámetro	**Valor**
Tamaño de entrada(s)	mediana
Dimensión de los *word-embeddings* (d)	300
Unidades LSTM (h_{lstm})	64
Batch size	16
Épocas	15
Tasa de aprendizaje	0.001
Tasa de regularización	0.00005
Función de pérdida	*Binary Crossentropy*
Optimizador	*Adam*

Tabla 3.7: Tabla de hiperparámetros utilizada en los modelos presentados en la Sección 3.2.2.

Se decidió utilizar *Adam* [18] frente a otros optimizadores como *RMSProp* o el gradiente estocástico porque con este se obtenían mejores resultados. Además pese a que la tasa de aprendizaje mostrada en la Tabla 3.7 sea de 0.001, se probaron valores del intervalo [0.1, 0.00005]. Finalmente los parámetros de *Adam*, *beta1* y *beta2*, tomaron los valores 0.9 y 0.98 respectivamente, y un *epsilon* con valor 1e-9.

Tras haber comprobado qué modelo era el mejor, se procedió a probar con este modelo un tamaño de entrada s diferente al de toda la experimentación con tal de comprobar si añadiendo más información el modelo de atención era capaz de mejorar. Para ello se tomó como referencia el intervalo

de valores de la Tabla 3.1, y el histograma de la Figura 3.1, y se seleccionó como valor de entrada el 650.

	F1_Test(%)	**F1_dev(%)**
Máxima Entropía	82.89	83.21
BiLSTM(s=463) + SpDo + Local (w=100, scaled-dot)	86.382	86.828
BiLSTM(s=650) + SpDo + Local (w=100, scaled-dot)	**87.81**	**88.67**

Tabla 3.8: Tabla de resultados de los modelos con atención local con diferente tamaño de entrada para el experimento 1.

Como se muestra en la Tabla 3.8 se vuelve a mejorar los resultados obtenidos para el modelo de atención local con ventana de tamaño 100 y con *scaled-dot* como *score*.

En la Tabla 3.9 se muestra una comparativa de los resultados obtenidos por los diferentes modelos sin *fine-tuning* utilizados para abordar esta tarea, desde el modelo inicial hasta el modelo de atención local que mejor resultado ha obtenido.

Una vez se tuvo el mejor modelo para el experimento 1, se procedió a ejecutar este modelo para el segundo experimento. De esta forma se comprobará si el modelo propuesto aprende la fuente o aprende a distinguir realmente la propaganda.

Para esta segunda experimentación se decidió utilizar los dos modelos con mayor resultado que tenían como tamaño de entrada la mediana, y el último modelo con tamaño de entrada 650. Los modelos de atención local tienen una ventana de 100 y como función de *score* la *scaled-dot*. El modelo de atención global también tenía como función de *score* la *scaled-dot*. En la Tabla 3.10 se muestran los resultados obtenidos por estos modelos.

No se ha logrado superar el mejor resultado que obtiene el clasificador de máxima entropía, dado que este modelo está aprendiendo aún más la forma de escribir de las fuentes y no distinguir correctamente si una noticia es propaganda.

	F1_Test(%)	**F1_dev(%)**
Máxima Entropía	82.89	83.21
CNN BiLSTM Model	74.40	-
BiLSTM Model	82.61	83.35
BiLSTM + SpDo	84.03	85.246
BiLSTM+SelfAtt	83.11	84.05
BiLSTM+SelfAtt + SpDo	83.284	85.021
BiLSTM(s=463) + SpDo + Local (w=100, scaled-dot)	86.382	86.828
BiLSTM(s=650) + SpDo + Local (w=100, scaled-dot)	**87.81**	**88.67**

Tabla 3.9: Tabla resumen de todos los modelos creados para abordar la tarea del análisis de propaganda para el experimento 1.

	F1_Test(%)	**F1_macro**(%)	**Accuracy**(%)
Máxima Entropía	**65.51**	-	**92.64**
BiLSTM (s=650) + Local (w=100, scaled-dot)	61.41	**78.09**	91.50
BiLSTM (s=463) + Local (w=100, scaled-dot)	60.804	77.56	90.81
BiLSTM (s=463) + Global (scaled-dot)	58.243	77.425	90.34

Tabla 3.10: Tabla de resultados de los modelos con atención local y global para el experimento 2.

3.2.3. Clasificador con *fine-tuning*

Para poder comprobar la efectividad de nuestra propuesta, se decidió comparar el modelo creado con un modelo ya pre-entrenado que haya conseguido buenos resultados en las diferentes tareas del campo del PLN. Como se indicó en la Sección 2.2.4, el modelo que se utilizará será BERT [9].

Para poder utilizar BERT primero se tuvo que adaptar la entrada a la red. Dado que BERT necesita dos *tokens* especiales para reconocer el inicio, [CLS], y el final de cada sentencia, [SEP], se ajustó el tamaño de todos los documentos al mismo que el modelo anterior, la mediana, y se le añadieron estos *tokens*. Tras esto se aplicó el tokenizador de BERT para aprovechar los *embeddings* pre-entrenados que trae el modelo. Una vez se preparan las 3 entradas de BERT, se conectan a este y se añade la capa de salida, una capa *softmax* al igual que en las redes de la sección anterior. La arquitectura del modelo utilizado en esta sección se muestra en la Figura 3.7.

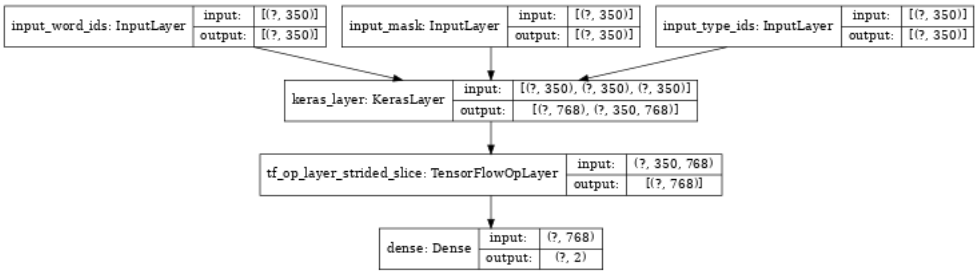

Figura 3.7: Arquitectura de BERT.

Además se utilizaron los mismos hiperparámetros que en los de la Sección 3.2.2, pero sólo cambiando la tasa de aprendizaje, puesto que se busca adaptar los pesos del modelo a los de nuestro problema, pero aprovechando los que ya tiene entrenados. Estos parámetros se pueden ver en la Tabla 3.11.

Hiperparámetro	**Valor**
Tamaño de entrada(s)	mediana
Batch size	16
Épocas	3
Tasa de aprendizaje	0.00005
Función de pérdida	*Binary Crossentropy*
Optimizador	*Adam*

Tabla 3.11: Tabla de hiperparámetros utilizada en los modelos presentados en la Sección 3.2.2.

El número de épocas también se disminuye, en este caso a 3, ya que, al igual que ocurre con la tasa de aprendizaje, no se busca que el modelo tenga

sobre ajuste, sino que se adapte a la tarea que se está realizando.

Los resultados obtenidos por BERT para el experimento 1 se muestran en la Tabla 3.12, mientras que los resultados del experimento 2 se muestran en la Tabla 3.13. Los resultados obtenidos para el primer experimento logran superar al estado del arte en los resultados de dev, mientras que se queda por detrás en test. En cuando al experimento 2 BERT se queda por detrás del estado del arte, al igual que con los modelos de atención local y global presentados en la Sección 3.2.2.

	F1_Test(%)	**F1_Dev(%)**
Máxima Entropía	**82.89**	83.21
BERT	82.66	**83.67**

Tabla 3.12: Tabla de resultados de BERT en los experimentos 1.

	F1_Test(%)	**F1_macro**(%)	**Accuracy**(%)
Máxima Entropía	**65.51**	-	**92.64**
BERT	56.96	75.19	88.57

Tabla 3.13: Tabla de resultados de BERT en los experimentos 2.

El hecho de que sea un modelo pre-entrenado que obtiene resultados en muchas otras tareas del PLN no basta para obtener buenos resultados en tareas de una complejidad alta como es la de este proyecto, y por eso ha resultado ser mejor el modelo basado en atención local que el presentado en esta subsección.

3.3. Análisis

En esta sección se realizará un análisis de los resultados obtenidos por el modelo de atención local comparado con el caso base, el clasificador de máxima entropía. Por ello en la Subsección 3.3.1 se analizará si hay diferencias diferencias significativas entre los modelos. En la Subsección 3.3.2 se analizará qué está aprendiendo el modelo, si aprende a diferenciar las técnicas de propaganda o no.

3.3.1. Significacia estadística

El modelo de atención local logra superar al estado del arte en el experimento 1, pero no en el experimento 2. Para comprobar qué modelo es mejor, se debe realizar un test estadístico comparando los resultados obtenidos y así comprobar si hay diferencias significativas entre el modelo de atención local y el clasificador de máxima entropía. Este test se debe aplicar en ambos experimentos, en el de *test* y en el de *dev*, además de aplicarse a ambos experimentos.

Dado que no se dispone de la salida de los modelos de máxima entropía, se ha decidido comparar el mejor y el peor resultado del modelo basado en atención local en *test* y en *dev*, con el resultado obtenido por el modelo de máxima entropía en *test* y *dev*. Por ello, para el conjunto de *test* se tomarán [88.47, 86.11] como valores del modelo de atención local y [82.89, 82.89] para el clasificador de máxima entropía. Para el conjunto de *dev* los resultados que se utilizarán para el modelo de atención local son [89.53, 87.31] y para el de máxima entropía [83.21, 83.21].

Dado que se está ante una muestra pequeña, tanto por parte del modelo de atención presentado, pero sobre todo por falta de valores del modelo de máxima entropía, se ha optado por realizar un test no-paramétrico, puesto que no podemos asegurar que los datos sigan una distribución normal. Con este test se busca comprobar si hay o no diferencias significativas entre el modelo de atención local propuesto y el clasificador de máxima entropía.

El test no-paramétrico que se ha decidido utilizar, teniendo en cuentas las condiciones de los datos, es el test de Wilcoxon. La hipótesis que se plantea es la de que no hay diferencia significativa entre los modelos de atención local y máxima entropía, por lo que si al aplicar el test se obtiene un p-valor inferior a 0.05, se cumpliría esta hipótesis con una confianza del 95 %. Tras aplicar el test para ambos conjuntos, *test* y *dev*, el p-valor obtenido en ambos casos es superior a 0.05, por lo que se puede afirmar que hay diferencias significativas entre ambos modelos en el experimento 1, siendo el modelo de

atención local el mejor modelos de los dos.

Se aplicó el mismo test con la misma hipótesis para el conjunto de test en el experimento 2. Para este se tomaron los valores [63.55, 59.82] para el modelo de atención local, y [65.61, 65.61] para el clasificador de máxima entropía. En ambos se obtuvo p-valor mayor a 0.05, por lo que en este experimento también se obtienen diferencias significativas entre ambos modelos, siendo el clasificador de máxima entropía el mejor modelo de ambos para el experimento 2.

Se quiso aplicar el test de McNemar [22], pero debido a la falta de una salida por parte del clasificador de máxima entropía no se ha podido realizar esta comparación entre los modelos para así comprobar que hay diferencias significativas a través de la predicción y no de las métricas utilizadas para comprobar la eficacia de los modelos.

Dado que los resultados de BERT en el experimento 1 son muy similares a los del clasificador de máxima entropía tanto en *test* como en *dev*, no se ha realizado ningún test puesto que se puede ver a simple vista que no hay diferencias significativas en cuanto a los resultados.

Tampoco se ha realizado el test para los resultados del experimento 2, puesto que con BERT se obtienen resultados más bajos que para el modelo de atención local, por lo que las diferencias con el clasificador de máxima entropía son aún mayores.

3.3.2. Cómo aprende el modelo

Los modelos de *Deep Learning* son catalogados como modelos de caja negra porque puesto que, tradicionalmente, no se podía dar una explicación a los resultados obtenidos ni analizarlos. Es por ello que, para aprovechar los resultados obtenidos por estos modelos, se busca que además de obtener los mejores resultados puedan explicar cómo han aprendido y qué les lleva a dar los resultados que obtienen, haciéndolos más transparentes [2] a las personas.

El modelo de atención local que se ha realizado en este trabajo, es un modelo que permite visualizar aquellas palabras a las que da mayor importancia a la hora de clasificar una noticia como propaganda, gracias al vector de pesos que se genera con cada noticia.

En las Figuras 3.8 y 3.9 se puede observar este vector de pesos sobre cada documento, mostrando qué es lo que el modelo considera más relevante a la hora de clasificar si un documento es propagandístico o no , siendo

la escala de colores de blanco a rojo, a mayor intensidad mayor importancia.

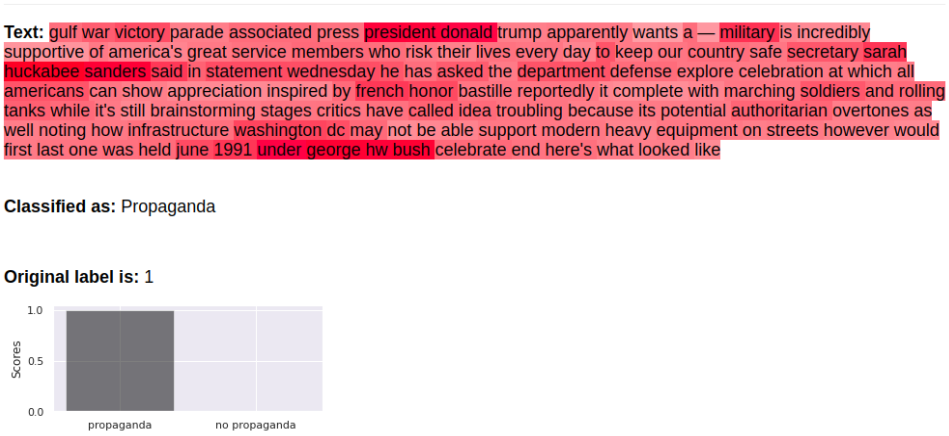

Figura 3.8: Modelo de atención local sobre un texto de propaganda.

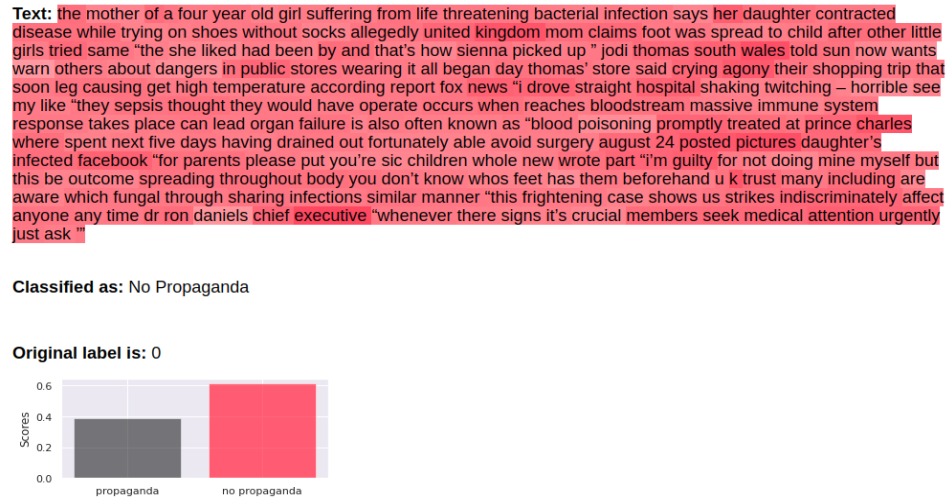

Figura 3.9: Modelo de atención local sobre un texto de no propaganda.

En la Figura 3.9 se observa cómo hay noticias que a pesar de no ser de propaganda el modelo duda de ello. Esto se debe a que el modelo puede no estar detectando bien las técnicas de propaganda pero sí está teniendo en cuenta el estilo de la redacción del documento a la hora de clasificarlo, ya que este modelo consiguió muy buenos resultados en el experimento 1, pero en el 2 se vio que el modelo parece estar aprendiendo el estilo de escritura en vez de las posibles técnicas de propaganda que pueda haber en cada documento.

Para observar en qué se fija correctamente el modelo y tratar de ver si se pueden detectar algunas de las técnicas de propaganda en la atención, se procede a mostrar más noticias propagandísticas que son clasificadas como tal por el modelo de atención local.

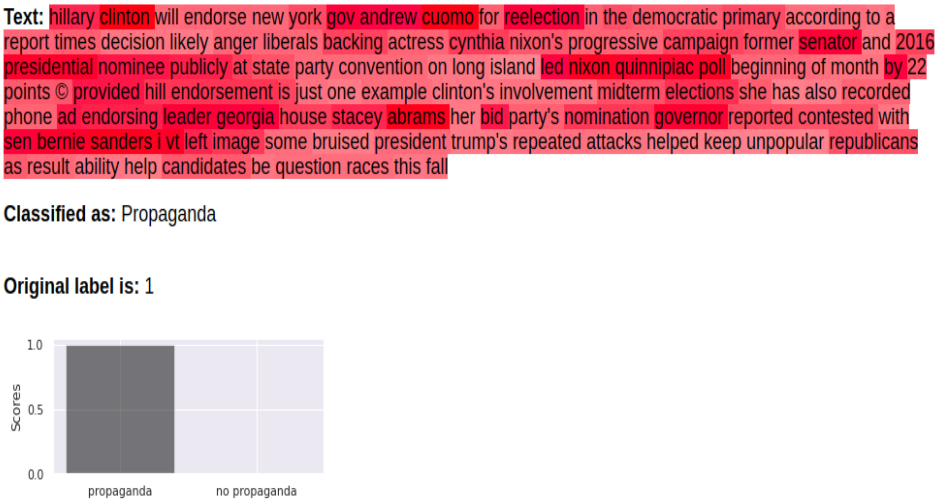

Figura 3.10: Proyección de la atención del modelo de atención local sobre noticias propagandísticas 1.

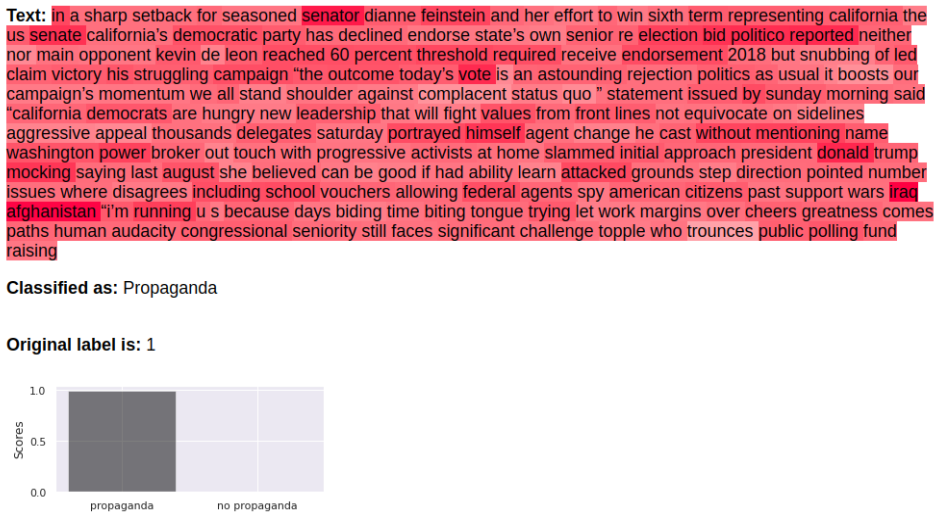

Figura 3.11: Proyección de la atención del modelo de atención local sobre noticias propagandísticas 2.

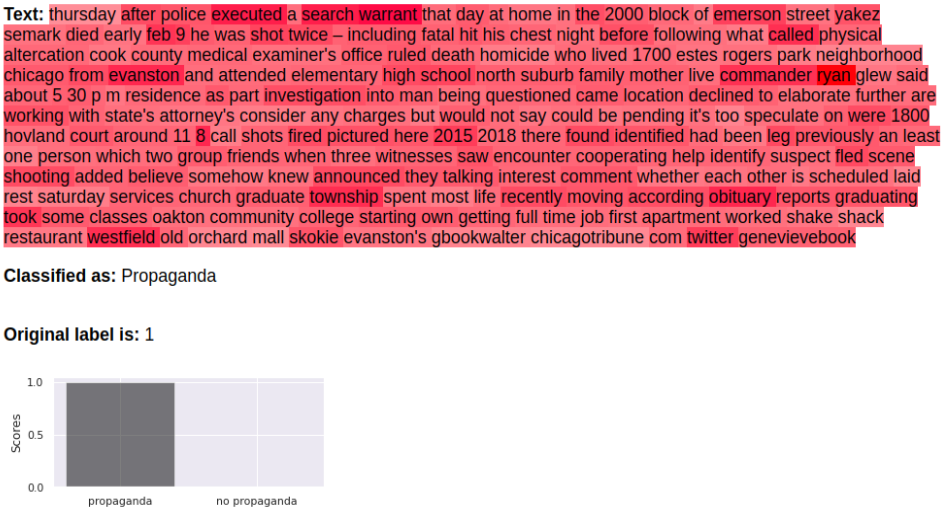

Figura 3.12: Proyección de la atención del modelo de atención local sobre noticias propagandísticas 3.

En las Figuras 3.10, 3.11, 3.12 se muestran diferentes noticias propagandísticas en las que el modelo las clasifica como tal, además de las palabras en las que más atención se fija, a través de la intensidad del color rojo sobre ellas.

En la Figura 3.10 se pueden observar algunas de las técnicas de propaganda que se utilizan actualmente en la política como son las técnicas de duda [7] y *Whataboutism*. Esto es porque está poniendo en duda la decisión tomada por un político y cuestionando su decisión, mientras que está tratando de desacreditar tanto al político seleccionado para ocupar una posición.

En la Figura 3.11 se puede observar una noticia en la que el redactor utiliza un tono en el que trata de ridiculizar a un político opuesto a su pensamiento por las palabras que utiliza sobre tu campaña electoral, y el cómo está está aplicando la técnica *Whataboutism* explicada en el trabajo, puesto que está tratando de desacreditar la postura del rival político, o la técnica *Reductio ad Hitlerum* explicada en [7], puesto que está tratando de persuadir al lector a rechazar la postura del rival político . Esto dice que el modelo sí es capaz de detectar técnicas de propaganda en las noticias que analiza.

En la Figura 3.12 se pueden observas las técnicas de propaganda de exageración, explicada en este trabajo, la técnica de lenguaje cargado, explicada en este trabajo, y la técnica de *repetición* [7], puesto que el autor trata de remarcar varias veces la situación del asesinato con tal de que la audiencia

acabe aceptando su postura.

Además de este análisis se ha realizado una nube de palabras, Figura 3.13, para comprobar a través de ella las palabras clave para que el modelo diga que una noticia es propagandística. De la nube de palabras se han decidido eliminar las *stopwords*, para tratar de descubrir aquellas entidades o verbos que puedan explicar más las técnicas de propaganda.

Figura 3.13: Wordcloud con palabras más importantes para el modelo de atención con noticias propagandísticas.

De la nube de palabras de la Figura 3.13 no se pueden extraer claramente las técnicas propaganda que se están utilizando en las noticias como sí se podía observar analizando el texto, pero sí se pueden observar algunas de las entidades que más en cuenta tiene, como por ejemplo Trump, el cual está muy relacionado con noticias de propaganda y *fake news* por su campaña electoral en el año 2016. También se observan las palabras *president* y *government*, las cuales se utilizan mucho en los diferentes medios que utilizan la propaganda para ganar votos, con tal de atacar y desacreditar al gobierno actual y así ganar votos para el partido que se esté apoyando.

Otros temas que se obtienen de esta nube de palabras son *school*, y cómo se utiliza la educación con fines electorales aunque quizás sí se puedan estar aplicando diferentes políticas sobre ello. También se ve la palabra *country*, y cómo de esta forma se puede ver la técnica de propaganda *Flag Waving*, explicada en el trabajo, apelando al nacionalismo para justificar sus ideas. Esto se puede ver también puesto que *Irán* también está entre las palabras clave, y cómo pueden aprovechar en conflicto Estados Unidos - Irán para así continuar apelando a la gente a que vote a la opción que se quiera por su postura respecto a este tema.

Como se puede observar de este análisis, el modelo es capaz de detectar algunas técnicas de propaganda en las noticias, pero es capaz de detectarlas según la fuente de la que provengan, ya que este modelo también confirma la hipótesis planteada en [3], ya que está aprendiendo las diferencias más el estilo de redacción que las propias técnicas, puesto que en el experimento 2 no está obteniendo buenos resultados.

Aunque el modelo no es adecuado para noticias de nuevos medios, ha logrado buenos resultados para el experimento 1, lo que quiere decir que sería capaz de identificar correctamente en un dominio cerrado más noticias propagandísticas, ya que permite saber cuando un medio, del cual sabe el la credibilidad que tiene, intenta publicar una noticia propagandística detectándola rápidamente.

Un caso de esto sería si, por ejemplo, un periódico como El Mundo, que presuponemos que no publica noticias de propaganda, comienza a publicar noticias de este tipo. Automáticamente el modelo presentado detectaría este cambio y sería capaz de avisar a los lectores de este medio.

Capítulo 4

Conclusiones y trabajo futuro

En este capítulo se procede a mostrar las conclusiones y el trabajo futuro. Por ello en la Sección 4.1 se mostrarán las conclusiones del trabajo realizado, y en la Sección 4.2 se mostrará el trabajo a realizar próximamente.

4.1. Conclusiones

La detección de noticias propagandísticas no es una tarea sencilla puesto que el número de técnicas que se pueden utilizar para encubrirlas es muy alto. Además el gran flujo de información que se genera día a día hace que dificulte más a esta tarea.

En este trabajo se ha realizado un modelo competitivo de *Deep Learning* que es capaz de detectar este tipo de noticias propagandísticas para intentar acabar con la difusión de estas noticias. Este modelo supera al modelo con *fine-tuning* presentado, ya que la dificultad de esta tarea implica que se debe realizar un modelo propio que entrene en el mismo dominio del problema.

El modelo realizado utiliza técnicas avanzadas de *Deep Learning* como son los mecanimos de atención, obtienendo buenos resultados en la detección de noticias propagandísticas en el dominio cerrado. Estos resultados dicen que se tiene un modelo que ante periódicos y noticiarios que no acostumbran a publicar propaganda, sean rápidamente detectados por el modelo si tratan de hacerlo, ya que ha aprendido el estilo de redacción que les caracteriza, y que puede ser imperceptible por las personas.

Gracias a los mecanismos de atención que se han implementado, se logra comprender mejor el aprendizaje del modelo, puesto que permiten obtener aquellas palabras que son más importantes para el modelo a la hora de decir

cuándo una noticia es propagandística.

También se ha logrado una recopilación de diversas colecciones de datos relacionadas con las *fake news* y no solo con las noticias propagandísticas, como *fake news* relacionadas con la salud [8], que proporcionan un razonamiento del por qué la noticia es falsa y que se debe dar como respuesta para añadir más explicabilidad al modelo.

4.2. Trabajo futuro

Aunque se han completado los objetivos, el trabajo realizado plantea diversas opciones para trabajar de cara al futuro:

- **Incorporación de conocimiento externo para mejorar el aprendizaje de qué es o no propaganda**: Dado que no se ha logrado mejorar el resultado obtenido por el modelo del estado del arte en el segundo experimento, uno de los puntos a mejorar en el futuro es este, añadiendo diferentes características que puedan proporcionar más información a la red como lo son nela, o los lexicon. De esta forma se buscaría que el modelo pudiera identificar fácilmente las técnicas de propaganda en los documentos.

- **Trabajar en una propuesta basada en transformers**: Otro trabajo independiente sería usar un modelo basado en una arquitectura de *transformers* [33], puesto que es un modelo que se está utilizando mucho en la literatura y obtiene buenos resultados en diferentes tareas de PLN. Pese a que BERT no ha obtenido buenos resultados, la creación de un modelo basado en esta arquitectura adaptado al problema puede hacer que se saque partido a esta arquitectura.

- **Continuar trabajando en modelos basados en *fine-tuning***: Pese a que el *fine-tuning* no ha funcionado bien, se estudiará cómo mejorar este enfoque para tratar de obtener mejores resultados de los obtenidos, ya sea aplicando cambios en la estructura presentada o utilizando otro nuevo modelo, ya que con *fine-tuning* se han obtenido buenos resultados en escenarios que tenían incluso pocos datos.

- **Clasificación de técnicas propagandísticas**: El siguiente paso a la clasificación de documentos propagandísticos, es clasificar las técnicas de propaganda, para ello se tomarían unas técnicas ya definidas como en [7], puesto que presentan una base de datos anotada y permiten comenzar en esta tarea que mezcla la clasificación de documentos y la identificación de *spans*.

- **Explicabilidad**: Las redes neuronales son modelos muy potentes pero que tienen margen de mejora para explicar los resultados. Mediante la tarea anterior se lograría poder explicar por qué un modelo es propagandístico, ya que si se detectan estas técnicas en un documento se podrá, no sólo decir que una noticia es propagandística, sino por qué lo es.

Bibliografía

[1] Martín Abadi, Ashish Agarwal, Paul Barham, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82 – 115, 2020.

[3] Alberto Barrón-Cedeño, Giovanni Martino, Israa Jaradat, and Preslav Nakov. Proppy: A system to unmask propaganda in online news. 12 2019.

[4] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *ArXiv*, abs/1601.06733, 2016.

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[6] Giovanni Da San Martino, Stefano Cresci, Alberto Barrón-Cedeno, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. A survey on computational propaganda detection. *arXiv e-prints*, pages arXiv–2007, 2020.

[7] Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. Fine-grained analysis of propaganda in news articles. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, EMNLP-IJCNLP 2019, Hong Kong, China, November 2019.

[8] Enyan Dai, Yiwei Sun, and Suhang Wang. Ginger cannot cure cancer: Battling fake health news with a comprehensive data repository. In *ICWSM*, 2020.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[10] Institute for Propaganda Analysis. *How to Detect Propaganda*, volume Volume I of the Publications of the Institute for Propaganda. 1938.

[11] Yoav Goldberg. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726, 2015.

[12] Marti A. Hearst. Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28, July 1998.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[14] J.B. Hooper. *On Assertive Predicates*. Indiana University Linguistics Club. Indiana University Linguistics Club, 1974.

[15] Benjamin Horne, Sara Khedr, and Sibel Adali. Sampling the news producers: A large news and feature data set for the study of the complex media landscape, 2018.

[16] Ken Hyland. *Metadiscourse*, pages 1–11. American Cancer Society, 2015.

[17] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[19] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite BERT for self-supervised learning of language representations, 2019.

[20] Xuan Hien Le, Hung Ho, Giha Lee, and Sungho Jung. Application of long short-term memory (lstm) neural network for flood forecasting. *Water*, 11:1387, 07 2019.

[21] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings*

of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[22] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, June 1947.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[24] C.R. Miller. *How to Detect and Analyze Propaganda ...: An Address Delivered at Town Hall, Monday, February 20, 1939.* A Town Hall pamphlet. Town Hall, Incorporated, 1939.

[25] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Asgari Chenaghlu, and Jianfeng Gao. Deep learning based text classification: A comprehensive review. *ArXiv*, abs/2004.03705, 2020.

[26] Arvind Mohan and Datta Gaitonde. A deep learning based approach to reduced order modeling for turbulent flow control using lstm neural networks. 04 2018.

[27] James W. Pennebaker, Martha E. Francis, and Roger J. Booth. *Linguistic Inquiry and Word Count.* Lawerence Erlbaum Associates, Mahwah, NJ, 2001.

[28] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[29] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[30] Hannah Rashkin, Eunsol Choi, Jin Jang, Svitlana Volkova, and Yejin Choi. Truth of varying shades: Analyzing language in fake news and political fact-checking. pages 2931–2937, 01 2017.

[31] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China, July 2015. Association for Computational Linguistics.

[32] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[34] William Yang Wang. "Liar, Liar Pants on Fire": A new benchmark dataset for fake news detection. *CoRR*, abs/1705.00648, 2017.

[35] A. Weston. *A Rulebook for Arguments*. Hackett Publishing Company, Incorporated, 2018.

[36] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, page 347–354, USA, 2005. Association for Computational Linguistics.

[37] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. volume 37 of *Proceedings of Machine Learning Research*, pages 2048–2057, Lille, France, 07–09 Jul 2015. PMLR.

[38] José Camacho - Collados y Mohammad Taher Pilehvar. From word to sense embeddings: A encuesta sobre representaciones vectoriales de significado. *CoRR*.

[39] Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. Long short-term memory over recursive structures. volume 37 of *Proceedings of Machine Learning Research*, pages 1604–1612, Lille, France, 07–09 Jul 2015. PMLR.