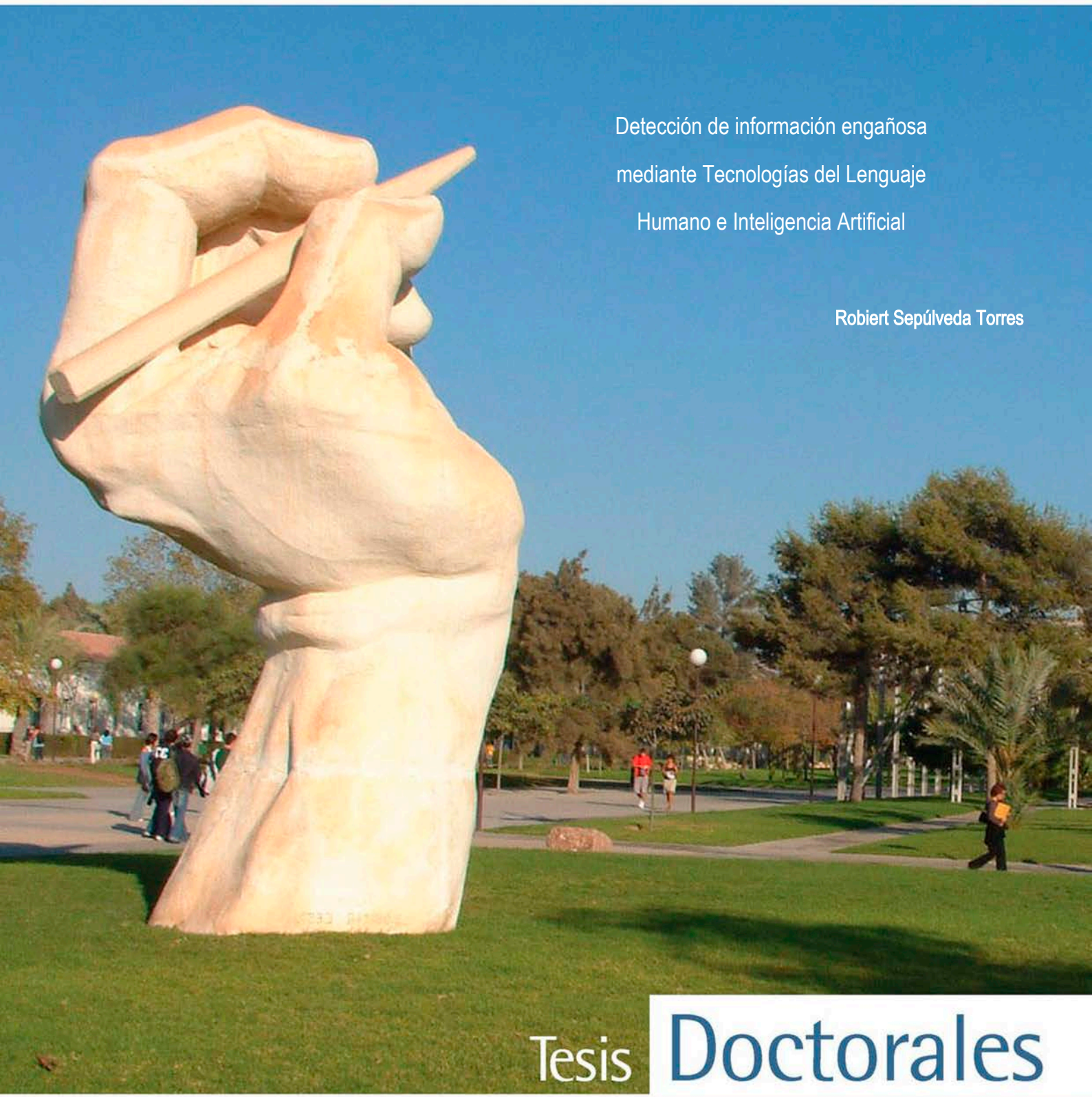
Universitat d'Alacant
Universidad de Alicante

Detección de información engañosa mediante Tecnologías del Lenguaje Humano e Inteligencia Artificial

Robiert Sepúlveda Torres



Tesis Doctorales
UNIVERSIDAD de ALICANTE

Unitat de Digitalització UA
Unidad de Digitalización UA

Universitat d'Alacant
Universidad de Alicante

Departamento de Lenguajes y Sistemas Informáticos
Escuela Politécnica Superior

Detección de información engañosa mediante Tecnologías del Lenguaje Humano e Inteligencia Artificial

Robiert Sepúlveda Torres

Tesis presentada para aspirar al grado de

DOCTOR POR LA UNIVERSIDAD DE ALICANTE

DOCTORADO EN INFORMÁTICA

Dirigida por

Dr. Estela Saquete Boró

Esta tesis ha sido financiada por la Generalitat Valenciana a través del proyecto PROMETEU/2018/089.

Agradecimientos

Miras hacía atrás y parece que no ha pasado el tiempo desde que empezaste esta aventura, que el camino ha sido lineal: nada más lejos de la realidad. Han pasado *n* cosas, con *n* que tiende al infinito; fines de semanas, vacaciones y noches intentando terminar un experimento para enviar un paper. Cuando te acuestas a dormir se te ocurre una idea que tiene mucho sentido y por qué no, te levantas a probarla. Luego los meses que pasan sin que te acepten ese trabajo que tanto ha costado, te acusan las dudas, las incertidumbres "*estaré por buen camino*", "*esto lo debería haber hecho de otro modo*"; pero lo cierto es que si estás escribiendo estos agradecimientos es que estás llegando al final.

Entonces es imposible no agradecer expresamente, espero haberlo hecho de otras muchas maneras, a todos lo que de una forma u otra han viajado durante todo este tiempo conmigo. En este camino siempre he tenido la guía de mi tutora Estela, "profe" como yo le llamo. La verdad que ha sabido conducir esta investigación, siempre una idea, un plan, disponible a todas horas, además de estar pendiente de trámites administrativos bien complicados, es infinita la ayuda que me ha brindado. Mi padre no es tutor oficial pero sus criterios y motivaciones han influenciado mucho esta investigación y sobre todo la forma en la intento abordar mi vida, desde "*descansa que mañana se te ocurrirá algo*", hasta darte un punto de vista externo, en muchas ocasiones trivial pero no lo habías visto, él siempre ha estado ahí.

Mi madre que mucho tiempo ha estado dedicada a las tareas del hogar y esperándome con la comida o llevándome una leche cuando no sabía que la necesitaba, ella siempre pendiente de mis avances y ayudando en todo lo que puede; tengo la suerte de haber compartido momentos muy especiales contigo.

A mi esposa Adlyn que ha sabido comprender en cada momento lo que necesitaba, ha intentado facilitar que me centrará en el trabajo pero también ha sabido llevarme a caminar, a ver una película para despejarnos y por supuesto ha estado pendiente en cada momento de los progresos y de lo que podía hacer para ayudar. Mi hermana, que a cada momento me daba consejos investigativos y diversos que me han sido de mucha ayuda para caminar por este laberinto, a Lian con el que he compartido muchas cosas, entre ellas las frustraciones propias de este camino.

A Bea y Paula que desde el principio me han hecho sentir como en mi hogar, en Narnia hemos compartido cafés, problemas y muchas risas que nunca olvi-

daré. Bea me ha acogido como si fuera su hermano cubano y eso que no la he llevado a bailar aún, gracias por ser tan emotiva y detallista, entre ella, Gloria y su madre me han alimentado y motivado mucho. Tania que siempre ha dado la claridad necesaria a los trabajos realizados, tú no solo editas el inglés, tus aportes mejoran sustancialmente nuestros trabajos, además de lo que he aprendido de tu inglesa forma de ver la vida. A José que siempre tiene tiempo para explicarte cualquier cosa durante el tiempo que sea necesario, darte una opinión precisa, además de compartir disímiles momentos juntos. Elena y Marta con las que he colaborado en numerosos trabajos y se han dejado la piel para alcanzar los objetivos. Marta, tú que siempre tienes respuesta para todo lo que te pregunto, me has sido de infinita ayuda, todo lo que se de overleaf te lo debo a tí, además de tus indicaciones sobre las cosas que tenía que ir haciendo.

A Yoan que me ha enseñado a organizar mi trabajo y a enfocarme en cosas concretas para ver sus frutos en un corto plazo. Suilan y Ale que me hacían comer a la 1 de la tarde como si estuviera en Cuba, me han ayudado mucho con dudas sobre modelos, implementaciones, son unas máquinas. Mario, María, Javi y Javi el técnico también han puesto su granito de arena para que este trabajo avance. Alba con la que también he compartido realización de trabajos y varias conferencias juntos, siempre está dispuesta a echarte una mano en lo que sea, hemos hecho un buen *team*.

Mis amigos no podían faltar, Frank con el que comparto y discuto de cualquier cosa, gracias por estar ahí todos los días a las 2 de la tarde diciendo "£Qué vuelta míster?", sus motivaciones son muy importantes. Karen y Tito con los que he disfrutado estos dos últimos años, siempre estaban disponibles para hacer cualquier cosa, no se me olvidan las comiditas creadas por tito en 5 minutos.

A mi familia que desde la distancia ha estado pendiente de los progresos que se iban haciendo. Mis tías, mis primos y primas, y por supuesto mi abuelo siempre estaban preguntado como iban las cosas; además de estar locos porque vaya a verlos. La lista es infinita porque tengo de la suerte de tener una familia que aunque geográficamente este separada mantiene las relaciones como si viviéramos muy cerca. A mis primos Elizabeth y Ale que han venido a verme tres veces en esta etapa y me han hecho reír y disfrutar muchísimo. A Ana, Alicia y Zoila que igualmente hicieron un viaje para ver a la familia del viejo continente. A mi familia por parte de Adlyn que también han estado pendiente y motivándome a cada momento, los de la Habana y los tíos y primos de Barcelona.

La lista podría ser interminable porque si de algo no me puedo quejar son de personas cercanas o incluso desconocidos que son capaces de echarte una mano cuando más lo necesitas, incluso unas simples palabras de cariño y motivación son el combustible necesario para seguir esforzándote todo lo que puedas.

Muchas gracias a todos.

Esta tesis ha sido financiada por la Generalitat Valenciana a través del proyecto "SIIA: Tecnologías del lenguaje humano para una sociedad inclusiva, igualitaria, y accesible" (PROMETEU/2018/089); y por FEDER/Ministerio de Ciencia e Innovación - Agencia Estatal de Investigación a través del proyecto "LIVING-LANG: Modelado del comportamiento de entidades digitales mediante tecnologías del lenguaje humano" (RTI2018-094653-B-C21 / C22).

Resumen

En los últimos años, el consumo de noticias en medios impresos ha sido sustituido en gran medida por el acceso a estas en variados formatos a través de medios digitales y redes sociales. Los bajos costes de acceso a la información y la profusión de las plataformas de comunicación y dispositivos móviles han producido un cambio en los hábitos de consumo de información, la que es recibida desde múltiples fuentes y replicada con inmediatez en un ambiente global.

En este contexto, se ha incrementado la desinformación, un problema originado en los albores de la prensa tradicional. En la última década, la desinformación ha alcanzado una escala inmanejable debido al gran volumen de información al que un ciudadano común está expuesto cada día. A esto se suma que la mayoría de estos medios digitales no son arbitrados, y permiten publicar y compartir cualquier tipo de información.

En este ambiente es muy probable la proliferación de información engañosa que, en la mayoría de los casos, pretende influir en la opinión pública para perseguir un objetivo económico, social o político subyacente. Esto puede perjudicar a las organizaciones, a las marcas y a las personas, entre otros, derivando en muchas ocasiones en conclusiones precipitadas por parte de los usuarios que la consumen. En este contexto surge el término de la posverdad como una tendencia a priorizar la subjetividad de una interpretación a la verificación de hechos reales.

El titular de una noticia está diseñado para resumir sucintamente su contenido, proporcionando al lector una comprensión clara de la misma. Desafortunadamente, en la era de la posverdad, los titulares están más enfocados en atraer la atención del lector que en presentar con precisión el contenido de la noticia. Esto abre una enorme oportunidad para difundir desinformación con la construcción de titulares falsos o distorsionados.

Las técnicas tradicionales de verificación de hechos realizadas por humanos son definitivamente impracticables y obsoletas ante la cantidad de textos informativos que se generan incluso cada hora. En este trabajo se abordan soluciones novedosas utilizando Tecnologías de Lenguaje Humano (TLH) y técnicas de Inteligencia Artificial (IA).

Esta investigación se ha desarrollado en un área donde se intersecan confusamente diferentes conceptos, herramientas y aproximaciones. Se parte de una ubicación en el estado del arte acerca de las principales soluciones relaciona-

das con la detección de titulares engañosos, detección de posturas, detección de contradicciones, interrelación entre estos elementos y verificación automática de hechos.

A partir del problema enunciado y sus conceptos, se profundiza en diferentes estrategias de solución con la aspiración de proponer una aproximación que permita, con un enfoque suficientemente práctico, aportar a la detección de información engañosa en medios digitales lo que puede convertirse en una herramienta de alerta en el complejo ambiente antes descrito.

Entre los elementos considerados, se valora la utilización de ML y de DL como técnicas tradicionales de trabajo en el espacio de estas soluciones, así como sus alcances y limitaciones. Además, se introduce la idea de sustituir el contenido de una noticia por un resumen suficientemente esencial y obtenido de manera automática.

La memoria presenta de manera lógica el curso de la investigación que parte de lo conceptual y utiliza el pensamiento deductivo y experimental para alcanzar generalizaciones y aplicarlas deductivamente a la solución de problemas específicos. Con ello, se abordan determinadas tareas que pueden contribuir parcialmente a la solución de parte del problema planteado, se diseñan experimentos y se especifica la solución en el ámbito del idioma español donde no se reportan aportaciones similares.

Se propone una arquitectura flexible para la detección de titulares engañosos que ha permitido implementar sobre ella dos prototipos cuyos resultados experimentales y documentados suponen un paso de avance hacia la automatización de esta tarea. Esta arquitectura alcanza resultados notables al ser aplicada sobre dos conjuntos de datos en idioma inglés y español.

Siguiendo los principios y las experiencias adquiridas se presenta una aplicación de una arquitectura similar para la detección de noticias falsas, lo que hace presumir su posible generalidad.

Abstract

Recently, news consumption via traditional print media has been by and large replaced by access to news through digital media and social networks. Low cost access to information as well as falling prices of mobile devices have produced a change in information consumption habits. Information is now received from multiple sources and replicated with immediacy and globally.

In this context, disinformation, whose origins can be traced from some of the very first newspapers published, has increased. In the last decade, disinformation has reached an unmanageable scale due to the sheer volume of information that the average citizen is exposed to on a daily basis. Added to this is the fact that most of these digital media are uncensored, allowing the publication and sharing of any type of information.

In this environment the proliferation of misleading information is highly likely, and especially that which, in many cases, seeks to influence public opinion so as to pursue an underlying economic, social or political goal. This potentially causes damage to organizations, brands and people, often resulting in hasty conclusions by consumers. In this context, the term post-truth arises to describe a tendency to prioritize the subjectivity of an interpretation over the verification of real facts.

The headline of a news story is designed to succinctly summarize its content, providing the reader with a clear understanding of the story. Unfortunately, in the post-truth era, headlines are more focused on attracting the reader's attention rather than accurately presenting the news content of the story. This opens up a huge opportunity for spreading disinformation with the construction of false or distorted headlines.

Traditional techniques of using human fact checkers for news texts on digital channels are clearly impractical and obsolete given the sheer volume of these generated on an hourly basis. In this work, novel solutions are addressed using Human Language Technologies (HLT) and Artificial Intelligence (IA) techniques.

This research has been developed in an area where different concepts, tools and approaches intersect. It starts with the state of the art about the main solutions related to misleading headline detection, stance detection, contradiction detection, the interrelation between these elements, and automatic fact checking.

Based on the stated problem and its concepts, different solution strategies

were studied in depth with the aim of proposing a practical approach that enables a contribution to the detection of misleading information in digital media. This would facilitate the development of a warning tool in the previously described complex environment.

Among the elements considered, the use of ML and DL as traditional working techniques in the space of these solutions, as well as their scopes and limitations, are evaluated. In addition, the idea of replacing the content of a news item by a sufficiently essential and automatically obtained summary is introduced.

This PhD thesis presents the research trajectory, starting from the conceptual and using deductive and experimental thinking to reach generalizations and apply them deductively to the solution of specific problems. With this, certain tasks are approached that can partially contribute to the solution of part of the problem posed, experiments are designed and the solution is specified in the field of the Spanish language where similar contributions have not been reported.

A flexible architecture for the detection of misleading headlines is proposed, which has allowed the implementation of two prototypes whose experimental and documented results represent a step forward towards the automation of this task. This architecture achieves remarkable results when applied to two datasets in English and Spanish.

Índice general

Índice de figuras

Índice de tablas

Acrónimos

ANN	Redes Neuronales Artificiales
BERT	*Bidirectional Encoder Representations from Transformers*
CNN	Red Neuronal Convolucional
CWE	*contradiction-specific word embedding*
DAG	Grafo Acíclico Dirigido
DL	Aprendizaje Profundo
ESIM	*Enhanced Sequential Inference Model*
FEVER	*Fact Extraction and VERification*
FLM	*Factored Language Model*
FNC-1	*Fake News Challenge*
GloVe	*Global Vectors for Word Representation*
GLUE	*General Language Understanding Evaluation*
GNN	*Graph Neural Network*

GPU	Unidad de Procesamiento Gráfico
IA	Inteligencia Artificial
IR	recuperación de información
K-NN	*K Nearest Neighbor*
LCL	Clasificador Local por Nivel
LCN	Clasificador Local por Nodo
LCPN	Clasificador Local por Nodo Padre
LIWC	*Linguistic Inquiry and Word Count*
LR	*Logistic Regression*
LSTM	*Long short-term memory*
MAP	*Mean Average Precision*
ML	Aprendizaje Automático
MLP	*Multilayer Perceptron*
NLI	*Natural Language Inference*
PLM	*Positional Language Model*
PLN	Procesamiento de Lenguaje Natural
PPDB	*Paraphrase Database*

QA	*question answering*
RACE	*Reading Comprehension Dataset From Examinations*
RoBERTa	*Robustly Optimized BERT Pretraining Approach*
RL	Aprendizaje Reforzado
RNN	Red Neuronal Recurrente
RTE	*Recognising Textual Entailment*
SQuAD	*Stanford Question Answering Dataset*
SRL	*semantic role labeling*
SVM	Máquinas de Soporte Vectorial
TF-IDF	*Term Frequency-Inverse Document Frequency*
TLH	Tecnologías del Lenguaje Humano
TPU	Unidad de Procesamiento Tensorial

CAPÍTULO 1

Introducción

1.1 Motivación y contexto

En la actualidad, las personas tienen acceso a un gran volumen de información, propiciado en gran medida por el aumento de su consumo a través de medios digitales (Conroy, Rubin, y Chen, 2015). Los medios digitales y principalmente las redes sociales tienen bajo coste y rápido acceso (Shu, Sliva, Wang, Tang, y Liu, 2017) que unido a la estructura de red de contactos y su horizontalidad las convierte en un medio propicio para la transmisión de información (Estrada-Cuzcano, Alfaro-Mendives, y Saavedra-Vásquez, 2020; Allcott y Gentzkow, 2017). Los hechos e ideas que se plantean en una parte significativa de estas informaciones no han sido verificados (Ciampaglia y cols., 2015), de ahí que su consumo está teniendo un impacto negativo en la gestión eficaz de la información para la sociedad en general (Narwal, 2018). Esta sobrecarga de información y, en particular, de noticias, puede derivar en infoxicación (Dias, 2014; Rodríguez y Barrio, 2015)

La infoxicación se define como la intoxicación por el exceso de información. Se traduce en recibir centenares de informaciones cada día a las que no se les puede dedicar tiempo (Dias, 2014). Una de las peores consecuencias de la infoxicación es no poder profundizar en nada y saltar de una cosa a la otra (Doval Avendaño y Domínguez Quintas, 2016). Este fenómeno ocasiona fatiga, parálisis del análisis y la creación de mecanismos de evasión, todo ello en el contexto de las noticias (C. S. Park, 2019).

En este contexto surge el problema de la desinformación que aumenta a gran velocidad a la par del crecimiento exponencial de la información en la web (Rubin, 2019). Aunque en español se utiliza únicamente el término desinformación, en idioma inglés se distinguen como *misinformation* y *disinformation*. Ambos aluden a la inexactitud y falta de veracidad de cierta información, sin embargo, mientras *misinformation* se relaciona con un engaño que puede ser causado

involuntariamente principalmente por desconocimiento o ignorancia (Wardle y Derakhshan, 2017), *disinformation* se refiere a engañar o desviar deliberadamente (Tudjmanand y Mikelic Preradovic, 2003). En ambos casos representan un tipo de fenómeno que, en el dominio de las noticias digitales, puede provocar una confusión masiva sobre los hechos reales, con la problemática añadida de que se extienda a escala viral (B. Huang y Carley, 2020).

Una de las formas de expresar desinformación son las noticias falsas. Las noticias falsas están altamente extendidas en medios digitales y especialmente en redes sociales, que parecen ser su hábitat de transmisión natural (Di Domenico, Sit, Ishizaka, y Nunan, 2021). Este tipo de noticias ha existido durante mucho tiempo (Allcott y Gentzkow, 2017), pero el término *fake news* en inglés es relativamente nuevo. The New York Times lo definió como una "historia inventada con la intención de engañar, a menudo con una ganancia monetaria como motivo" (Tavernisen, 2019). Este fenómeno ha experimentado un auge significativo desde las elecciones estadounidenses de 2016 (Bovet y Makse, 2019) y el referéndum del Brexit de 2016 (Bastos y Mercea, 2019). Con el auge en los últimos años de la desinformación, existen organizaciones que, basadas en intereses comerciales, políticos y empresariales, fabrican noticias falsas para generar ingresos a través de *clickbait* e información engañosa. Por ejemplo, National Report[1] y Disinformedia (Hooper, 2018) son ejemplos de un sitio web y de una empresa que producen y/o difunden noticias falsas.

La actual pandemia provocada por el COVID-19 está corroborando el grado de desinformación al que están sometidas las sociedades modernas. Esta desinformación se propaga más rápido que el propio virus y se publican una gran cantidad de recomendaciones contradictorias y sin sentido, que carecen de base científica (Colomina, s.f.; Moscadelli y cols., 2020). Por ejemplo, el titular “Una niña de 13 años muere en Portugal por una parada cardiorrespiratoria asociada al uso de mascarilla”, se demostró que era falso poco después de su publicación[2]. No hay duda de que estas declaraciones pueden tener implicaciones peligrosas para la salud pública global, como las que podrían resultar de titulares engañosos como éstos: “Datos oficiales del Reino Unido muestran un aumento del 5.400% en el número de mujeres que han perdido a su bebé después de recibir vacunas COVID"[3].

Investigaciones recientes han estudiado los factores que conllevan a los usuarios de redes sociales a compartir noticias. En (Apuke y Omar, 2021) se analiza la influencia de algunos factores (altruismo, entretenimiento, socialización e intercambio de información) para determinar el grado de influencia en compartir noticias, obteniendo que al menos en el caso de la pandemia de COVID-19 el altruismo es un factor determinante. Por otra parte, en (Talwar, Dhir, Kaur, Zafar, y Alrasheedy, 2019) obtienen como hallazgo que la confianza en línea, el

[1] `https://nationalreport.net/` (consultado el 17 de septiembre de 2021).
[2] `https://bit.ly/3LaA0xM` (consultado el 17 de septiembre de 2021).
[3] `https://bit.ly/3rsJ1dF` (consultado el 17 de septiembre de 2021).

miedo a perderse algo y la fatiga de las redes sociales propician que una noticia sea compartida; y que la confianza en línea influye en que estas se repliquen sin la debida verificación de autenticidad. Estos hallazgos unidos al deseo de obtener clics o compartir contenido inmediato tiene el inconveniente de transmitir información de menor calidad porque las historias que se publican primero suelen ser menos precisas dado que no se consumió tiempo en verificar sus fuentes (Estrada-Cuzcano y cols., 2020). Además de estos factores que influyen en el intercambio de noticias, en (Vosoughi, Roy, y Aral, 2018) se plantea que la información falsa tiene un 70% más de probabilidad de ser compartida que la información verdadera. Todo ello convierte a la desinformación en un complejo problema difícil de abordar desde una única perspectiva.

Existen intentos de grandes empresas y organizaciones para detectar este tipo de información. Por ejemplo, The New York Times, BBC y Google han creado soluciones que permiten comprobar si una noticia ha sido detectada como falsa, entre ellas:

- Reality check[4]: es una página web de la BBC que verifica afirmaciones, pero no es una tecnología de detección de noticias falsas.
- Fact Check Explorer[5]: es un explorador proporcionado por Google que permite la búsqueda de afirmaciones verificados, pero no verifica afirmaciones ingresadas en línea.
- Newtral[6]: es una empresa española con el mismo nombre que se enfoca en la verificación de hechos por parte de un equipo de verificadores de hechos (*fact checker*).
- Fact Checks[7]: proporcionado por The New York Times realiza la búsqueda de noticias falsas verificadas, pero no permite ingresar una historia para verificar los hechos.

La mayoría de estas empresas están formadas por periodistas e investigadores (comúnmente conocidos como verificadores humanos de hechos) que se encargan de verificar afirmaciones usando métodos tradicionales, sin utilizar prácticamente tecnologías que apoyen al proceso de verificación. Estas soluciones y otras tantas que existen en este momento no son capaces de evaluar la veracidad de noticias en un tiempo razonable debido al gran volumen que circula por redes sociales y medios digitales (Tsipursky, Votta, y Roose, 2018).

Por otra parte, la comunidad científica ha comenzado a aunar esfuerzos para atacar el problema desde la perspectiva de la detección automática. La Inteligencia Artificial (IA), específicamente, el Aprendizaje Automático (ML) y

[4] `https://www.bbc.com/news/reality_check` (consultado el 10 de enero de 2022).
[5] `https://toolbox.google.com/factcheck/explorer` (consultado el 10 de enero de 2022).
[6] `https://www.newtral.es/` (consultado el 10 de enero de 2022).
[7] `https://www.nytimes.com/spotlight/fact-checks` (consultado el 10 de enero de 2022).

el Procesamiento de Lenguaje Natural (PLN), así como otros enfoques de tecnologías, han sido usados para abordar la creciente desinformación (Boró, Tomás, Moreda, Martínez-Barco, y Palomar, 2020; Bondielli y Marcelloni, 2019; X. Zhang y Ghorbani, 2020).

En (Boró y cols., 2020) se realizó una revisión sistemática del fenómeno de las noticias falsas. Este estudio identifica las principales tareas que pueden intervenir en el proceso de detección de noticias falsas de forma automática, tales como: detección de engaños, detección de posturas, controversia y polarización, verificación automática de hechos, detección de ciberanzuelos y credibilidad. Además, se discute acerca de los recursos y sistemas existentes hasta la fecha en cada una de las tareas antes mencionadas.

El problema de la información engañosa es particularmente importante en el dominio periodístico, siendo este el escenario donde se inscribe esta investigación. Por la complejidad de este problema sus soluciones exigen la integración de diferentes enfoques (Boró y cols., 2020). En específico, este trabajo de tesis se centra en desarrollar soluciones basadas en Tecnologías del Lenguaje Humano (TLH) para las tareas de verificación automática de hechos y la detección de titulares engañosos. Estas tareas comúnmente implican resolver problemas de clasificación, de aquí que se profundice en esta área.

La verificación automática de hechos es una de las tareas que más avance ha tenido dentro de la detección de noticias falsas (Boró y cols., 2020). En ella se intenta automatizar las principales acciones que realizan los verificadores humanos de hechos para analizar una noticia, con el objetivo de verificar automáticamente la veracidad de una afirmación pública con todos los datos disponibles y clasificarla en valores de veracidad (Dale, 2017). Para cumplir este objetivo se emplean, comúnmente, etiquetas cualitativas tales como verdadero, mayormente verdadero, medio verdadero, mayormente falso y falso. El proceso de verificación automática de hechos se divide en las mismas etapas que la verificación manual de hechos realizada por periodistas o investigadores (FullFact.org, 2016).

Del mismo modo, se ha avanzado significativamente en los últimos años en la tarea de detección de posturas (Küçük y Fazli, 2020). En el caso de las noticias, esta tarea se relaciona con la detección de titulares engañosos, lo que implica estimar la perspectiva o postura relativa de dos textos relacionados con un tema, afirmación o cuestión (ALDayel y Magdy, 2021). Específicamente, la tarea implica clasificar la postura del contenido de una noticia con su titular, a fin de detectar posibles contradicciones (Babakar y cols., 2016). Siguiendo este enfoque, se ha centrado una gran atención y esfuerzo en el análisis y estudio de uno de los elementos más esenciales de una noticia, su titular, en algunos casos centrándose en la relación entre el contenido del artículo y el titular, y en otros considerando la constitución del titular en sí mismo (Wei y Wan, 2017).

Dado que el dominio de este trabajo se centra en el análisis de artículos de noticias, se define la estructura esperada de una noticia en: título, subtítulo, entradilla, contenido y conclusiones (Lajusticia, 2000), aunque el subtítulo y

las conclusiones no siempre están presente (Bonet-Jover, Piad-Morffis, Saquete, Martínez-Barco, y Ángel García-Cumbreras, 2020). Sin embargo, en esta investigación se simplifica la estructura en titular y contenido, representando el contenido al resto de partes antes mencionadas. El titular es parte esencial de una noticia, dado que resume el contenido y le brinda al lector una idea preliminar del artículo (van Dijk, 1988; Reis y cols., 2015). Este actúa como preludio de la noticia y debe redactarse como una invitación para que el lector descubra la pieza completa (Conroy y cols., 2015). Por lo tanto, se espera que su redacción sea lo más efectiva posible, sin perder precisión ni resultar engañoso, para respaldar la veracidad de todo el artículo (Kuiken, Schuth, Spitters, y Marx, 2017).

En el escenario que se ha delineado, donde el flujo de información crece permanentemente y el filtrado de contenidos puede ser abrumador, el papel de los titulares es crucial, dado que un análisis exhaustivo de la noticia es prácticamente imposible (K. Park, Kim, Yoon, Cha, y Jung, 2020). Por un lado, un titular adecuado puede ayudar a identificar correctamente el contenido de mayor interés, pero debido a la avalancha de datos antes mencionada, puede resultar tentador leer solo los titulares y compartir la noticia sin leer la historia completa (Gabielkov, Ramachandran, Chaintreau, y Legout, 2016). En consecuencia, una noticia se puede convertir en una historia viral debido a un titular atractivo en ausencia de la veracidad de la información en su propio contenido. Este fenómeno manipula la opinión pública y afecta la credibilidad de las redes sociales, entre otras consecuencias (Lutz, Adam, Feuerriegel, Pröllochs, y Neumann, 2020). En particular, la investigación realizada en (Gabielkov y cols., 2016) encontró que el 59% de las URLs mencionadas en Twitter no recibieron ningún clic. Esto demuestra una mayor disposición de los usuarios a compartir noticias a partir de la interpretación del titular que a acceder al contenido para determinar su validez. Incluso un estudio plantea que una primera impresión obtenida del titular es persistente de tal manera que su postura permanece incluso después de leer todo el contenido de la noticia (Ecker, Lewandowsky, Chang, y Pillai, 2014).

Desafortunadamente, en la práctica, los titulares en los medios digitales tienden a estar más enfocados en atraer la atención del lector, con poca consideración por la precisión, lo que lleva a errores o desinformación a través de hechos falsos o contradicción entre titulares y contenidos (Y. Chen, Conroy, y Rubin, 2015). En este contexto, los titulares se pueden clasificar en dos clases (Wei y Wan, 2017):

- **Titulares ciberanzuelos (*Clickbait headlines*)**: Un ciberanzuelo se refiere a un titular cuyo propósito principal es llamar la atención y alentar a los visitantes a hacer clic en un enlace a una página web (Chesney, Liakata, Poesio, y Purver, 2017), con el propósito de monetizar las visitas a través de los ingresos publicitarios (cuantos más clics, más dinero se gana) (Y. Chen y cols., 2015). Este tipo de titular es a menudo ambiguo y exhibe un estilo de escritura particular para explotar directamente la curiosidad humana,

utilizando redacciones exclamatorias o interrogativas que insten al público a hacer clic en el enlace para descubrir la información que falta (Blom y Hansen, 2015). Por lo general, los titulares ciberanzuelos se difunden en las redes sociales en forma de breves mensajes que pueden leerse como los siguientes ejemplos citados:

- *"La nueva vida de Iker Casillas tras su divorcio de Sara Carbonero: esto es lo que se ha comprado"* [8]
- *"El fantástico hilo de Twitter que habla de LA NUEVA SEPA y que debería ser leído por todo aterrado tragacionista"* [9]

Los métodos existentes para detectar automáticamente los titulares con ciberanzuelos suelen tratar la tarea como un problema de clasificación (*clickbait*/*no clickbait*) y se centran exclusivamente en el titular, su estilo o estructura de redacción, en lugar de considerar el contenido de las noticias en sí mismas (Kuiken y cols., 2017).

- **Titulares engañosos (*Misleading headlines*)**: Este tipo de titular tergiversa significativamente los hallazgos reportados en la noticia, exagerando o distorsionando los hechos descritos (Chesney y cols., 2017). El lector solo puede descubrir las contradicciones después de leer completamente la noticia (Wei y Wan, 2017). Aunque en la literatura estos titulares a veces se denominan titulares incongruentes (*incongruent headlines*), en este trabajo nos referiremos a ellos como titulares engañosos ya que el término representa un concepto con mayor afinidad a nuestros objetivos. En este tipo de titular, algunos matices importantes que forman parte del contenido de la noticia se excluyen deliberadamente, lo que hace que el lector llegue a una conclusión errónea. A continuación se muestran ejemplos de titulares engañosos:

  - *"Selena Gomez se atreve y les enseña: la foto más íntima"* [10]
  - *"La amenaza de De Gea que lo cambia todo en el Real Madrid (y con Florentino Pérez)"* [11]

  Para detectar automáticamente titulares engañosos, se debe analizar el contenido con el objeto de extraer la evidencia de la que supuestamente se ha derivado el titular, detectando así las posibles contradicciones entre el titular y el contenido. Para el análisis de este tipo de titular es insuficiente un tratamiento estructural, siendo necesario optar por un enfoque semántico (Chesney y cols., 2017).

[8] `https://bit.ly/3Hx6Vds` (consultado el 17 de septiembre de 2021).
[9] `https://bit.ly/3GvpJJ1` (consultado el 17 de septiembre de 2021).
[10] `https://bit.ly/3sg222k` (consultado el 20 de septiembre de 2021).
[11] `https://bit.ly/3Lf5ClG` (consultado el 20 de septiembre de 2021).

A partir del análisis antes presentado sobre titulares ciberanzuelos y titulares engañosos se aprecia que las diferencias esenciales en su detección se centran en que en el caso de titulares ciberanzuelos puede ser suficiente un análisis estructural, mientras que los titulares engañosos exigen procesamientos semánticos en la relación titular-contenido. Hace unos años la mayoría de las investigaciones se centraban en la detección de titulares ciberanzuelos, quedando algo subestimada la detección de titulares engañosos (Wei y Wan, 2017). Sin embargo, recientes investigaciones indican que en la actualidad es más común encontrar trabajos en la detección de titulares engañosos o titulares incongruentes (K. Park y cols., 2020; Wei y Wan, 2017; Yoon y cols., 2021) e incluso tratado como un problema de detección de posturas entre titulares y contenidos de noticias (Hanselowski, PVS, y cols., 2018; W. Ferreira y Vlachos, 2016; Q. Zhang, Liang, Lipani, Ren, y Yilmaz, 2019).

Resulta importante explicar que las definiciones no son mutuamente excluyentes, un titular ciberanzuelo también puede ser contradictorio o incongruente con su contenido. Además, los titulares de ciberanzuelo pueden ser aceptables si representan con precisión los hechos del contenido correspondientes; sin embargo, las consecuencias pueden ser más graves si los titulares atractivos engañan a las personas con información incorrecta (K. Park y cols., 2020).

La principal línea de investigación de este trabajo es la detección de titulares engañosos. En este sentido, existen investigaciones que sugieren abordar la detección de titulares engañosos como un problema de relación semántica y tratarlo como detección de contradicciones o reconocimiento de vinculación textual (Chesney y cols., 2017).

Los avances alcanzados en las investigaciones sobre titulares engañosos han incluido una serie de recursos para avanzar en la creación de modelos y sistemas de detección. La mayoría de estos recursos se encuentran en idioma inglés (Babakar y cols., 2016), aunque se han encontrado recursos en otros idiomas como el coreano (K. Park y cols., 2020) y el chino (Wei y Wan, 2017). Sin embargo, a pesar de que el español es uno de los idiomas más hablados en el mundo, no existen recursos potentes para llevar a cabo la tarea de detectar titulares engañosos ni para la detección de contradicciones entre textos, desde la perspectiva directa de este idioma.

Partiendo de la necesidad de verificar la relación semántica entre un titular y el contenido correspondiente, es de vital importancia obtener las principales evidencias dentro del contenido de la noticia. Las técnicas actuales relacionadas con IA, incluyendo los modelos neuronales pueden tener un impacto negativo en la eficiencia al procesar textos largos, por lo que estudios anteriores han utilizado la primera oración del texto (Hayashi y Yanagimoto, 2018), un fragmento específico (Z. Huang, Ye, Li, y Pan, 2017; Yoon y cols., 2018) o la obtención automática de resúmenes para evadir este problema (ShimJae-Seung, WonHa-Ram, y AhnHyunchul, 2019). Esto sugiere experimentar sobre la utilización de resúmenes que extraigan la información clave de la noticia. El uso de resúmenes automáticos, hasta donde sabemos, no ha sido explotado previamente para la

tarea de detección de titulares engañosos.

Nuestra investigación se centra en detectar información engañosa automáticamente, primero, con la creación y optimización de modelos y el uso de técnicas conocidas de verificación automática de hechos para recuperar evidencias y comprobar las afirmaciones de una noticia, y segundo, creando recursos y modelos en la tarea de titulares engañosos usando enfoques relacionados con detección de posturas y contradicciones como las descritas en (De Marneffe, Rafferty, y Manning, 2008; Harabagiu, Hickl, y Lacatusu, 2006). Además, se evaluará la pertinencia de técnicas del estado del arte basadas en algoritmos tradicionales de ML y en modelos más modernos basados en Aprendizaje Profundo (DL)[12] como (Canete, Chaperon, Fuentes, y Pérez, 2020; Y. Liu, Ott, y cols., 2019; Devlin, Chang, Lee, y Toutanova, 2019).

La figura 1.1 muestra las principales tareas que se abordan en esta investigación para atacar el problema de la detección de información engañosa.

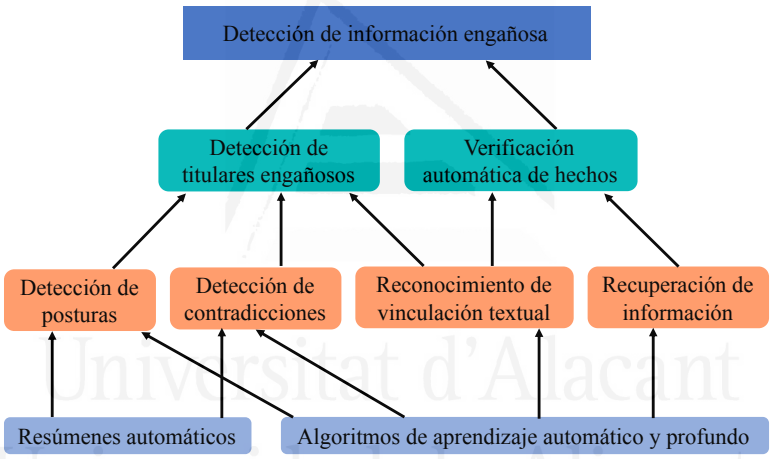


Figura 1.1: Tareas dentro de la detección de información engañosa.

Como se puede observar esta figura tiene cuatro (4) niveles representados por diferentes colores. La detección de información engañosa es el nivel más alto dado que es nuestro objetivo principal, en el próximo nivel se encuentran las tareas de alto nivel con las que se pretende detectar este tipo de información (detección de titulares engañosos y verificación automática de hechos). El nivel 3 serían tareas específicas (detección de posturas, detección de contradicciones, reconocimiento de vinculación textual y recuperación de información) que usualmente son utilizadas para solucionar las tareas de alto nivel, incluso algunas soluciones pueden ser comunes para las tareas más generales. En el último nivel se encuentran los recursos genéricos usados por las tareas específicas (resúmenes automáticos y algoritmos de aprendizaje automático y profundo).

[12] DL es un tipo específico de ML. Se utiliza la nomenclatura DL para indicar la diferencia entre los enfoques sin DL y el resto.

Esta figura representa la relación de las tareas específicas dentro del PLN con las tareas de alto nivel abordadas para detectar información engañosa.

En resumen, abordar el problema de la detección de información engañosa desde la perspectiva de la detección de titulares engañosos y la verificación automática de hechos, podría contribuir al diseño de sistemas que operen en entornos humano-máquina. Estos podrían apoyar a periodistas y otros usuarios en tareas de verificación, superando la imposibilidad de verificar el gran volumen que circula por plataformas digitales.

1.2 Objetivos de la investigación

En el contexto actual de desinformación y en correspondencia con los avances en algunas tareas específicas para abordar la detección de información engañosa, se ha trazado como objetivo general de esta investigación: **proponer recursos novedosos que aporten a la detección de información engañosa y que contribuyan al diseño de sistemas automatizados para este fin o similares**.

Para dar cumplimiento al objetivo general se trazaron los siguientes **objetivos específicos**:

- Identificar los fundamentos teóricos, prácticos y metodológicos relacionados con los problemas de detección de información engañosa.
- Analizar la influencia del uso de resúmenes automáticos en la tarea de detección de posturas entre titulares y contenidos de noticias.
- Diseñar una arquitectura escalable y configurable de detección de titulares engañosos que pueda ser aplicada a distintas tareas y en varios idiomas.
- Implementar prototipos de la arquitectura variando los módulos que la componen.
- Proponer recursos en idioma español para la detección de titulares engañosos aplicando detección de contradicciones entre textos.
- Avanzar en la solución de problemas abiertos dentro de la verificación automática de hechos (detección de noticias falsas y detección de frases que sean verificables).

1.3 Metodología

Con el objetivo de llevar a cabo esta investigación doctoral se siguieron métodos de investigación científica clásicos. De manera cíclica en el proceso de desarrollo de esta investigación se aplicó el método inductivo-deductivo con base experimental.

En una primera fase se realiza un estudio sistemático sobre las principales tareas abordadas, entre ellas:

- Detección de titulares engañosos: Se analizan los conceptos asociados con la tarea y las relaciones con otras tareas de interés. Se discute sobre las técnicas y recursos disponibles.
- Detección de posturas: Se estudia el campo de aplicación de la tarea y algunos problemas abiertos.
- Detección de contradicciones: Se analiza la posible aportación de esta tarea a la detección de titulares engañosos.
- Problemas de clasificación: Se consultan las principales formas de resolver problemas de clasificación complejos. Se analizan modelos avanzados en el área del PLN.

Como consecuencia de la utilización de problemas de clasificación se analizan los recursos disponibles en las tareas para los idiomas de interés. Esto principalmente se refiere a conjuntos de datos (corpus) que puedan ser utilizados para entrenar modelos de clasificación y abordar las tareas con estos.

En el ámbito de la investigación es habitual encontrar colaboración entre investigadores, planteando competiciones y concursos que pretenden instar a la comunidad científica a avanzar en una tarea específica. Se aprovechan estos entornos para experimentar en tareas abiertas en las áreas de interés y con el objetivo de discutir acerca de los resultados.

Basado principalmente en las deducciones y experiencias extraídas del estudio de la literatura, y con las experimentaciones realizadas se proponen soluciones que expongan cierto nivel de generalización para alguna tarea concreta.

Por último, se documenta y publican los resultados obtenidos para que puedan ser replicados y discutidos por la comunidad científica.

1.4 Estructura de la memoria

El documento de tesis se encuentra estructurado en siete capítulos. Un resumen del contenido de cada uno se expresa a continuación:

- **Capítulo 1 Introducción**: Se caracteriza el problema a resolver, así como los objetivos de la investigación y los aspectos metodológicos esenciales que se aplican.
- **Capítulo 2 Estado del arte**: Se documentan los conceptos, algoritmos y enfoques aplicados en la solución de problemas relacionados con la detección de titulares engañosos, la detección de posturas, las contradicciones entre textos y su reflejo en el caso particular de titulares y noticias. Se

estudian soluciones a problemas de clasificación de múltiples clases y se analizan modelos del estado del arte basados en DL.

- **Capítulo 3 Análisis de la aplicación de resúmenes a la detección de posturas en titulares**: Se presentan diferentes métodos de resúmenes automáticos de textos y de detección de posturas. Se documenta el diseño y los resultados de los experimentos que comparan la aplicación de resúmenes de texto en sustitución del contenido de una noticia.
- **Capítulo 4 Propuesta de arquitectura de detección de posturas en titulares**: Se presentan las bases de diseño de una arquitectura con capacidad para detectar posturas entre titulares y contenidos de noticias, la implementación de dos prototipos y los experimentos relacionados.
- **Capítulo 5 Titulares engañosos aplicando detección de contradicciones**: Se desarrolla un conjunto de datos en idioma español para la tarea de detección de titulares engañosos aplicando detección de contradicciones. Se documenta el proceso de creación y se realizan experimentos asociados con el corpus creado.
- **Capítulo 6 Verificación automática de hechos**: Se argumentan las experiencias adquiridas y los resultados obtenidos por el equipo GPLSI en las tareas *Check-Worthiness Estimation* y Detección de noticias falsas de la convocatoria promovida por el laboratorio CheckThat! 2021 y su impacto en la investigación.
- **Capítulo 7 Conclusiones y trabajos futuros**: En el capítulo se presentan las conclusiones generales de la investigación, las principales aportaciones, las publicaciones realizadas en el contexto del trabajo y los trabajos futuros derivados de problemas abiertos o resultados insatisfactorios.

La memoria cuenta además con Referencias bibliográficas y Apéndices.

CAPÍTULO 2

Estado del arte

2.1 Introducción

Las noticias falsas y específicamente los titulares engañosos son un fenómeno que ha existido por mucho tiempo (Allcott y Gentzkow, 2017), sin embargo, el cambio de hábito de los usuarios que han pasado a consumir noticias a través de medios digitales (específicamente en redes sociales) y la facilidad de viralización intrínseca a estas tecnologías (Conroy y cols., 2015) lo han convertido en un grave problema de la sociedad moderna. Este trabajo tiene como principal objetivo proponer soluciones novedosas a la detección de titulares engañosos, es por ello que a lo largo de este capítulo se presentará un estudio sobre las principales investigaciones relacionadas con las diferentes técnicas y aproximaciones de clasificación y detección de titulares en medios digitales.

La detección de posturas es la principal vía de detección de titulares engañosos. En esta sección se aborda el problema de la detección de posturas en un plano general, esto permitirá tener una referencia certera de las tipologías de los recursos y las soluciones en esta área. Posteriormente se introduce el problema de la detección de titulares engañosos, agrupándose las diferentes formas de solucionar el problema. Finalmente se sustenta la hipótesis acerca del posible aporte de la detección de contradicciones a la detección de titulares engañosos.

Además como los problemas abordados en esta tesis implican clasificaciones de textos, en las secciones posteriores se presentan algunas consideraciones para solucionar problemas de clasificación complejos, haciendo especial énfasis en las soluciones en el contexto de PLN. Se estudian en profundidad los clasificadores jerárquicos, las soluciones basadas en DL y el aprendizaje por transferencia.

2.2 Detección de posturas

Las redes sociales (como Twitter y Facebook) se han convertido en la principal fuente de información e interacción entre millones de personas en todo el mundo (ALDayel y Magdy, 2021). Estas, unidas a los foros y debates políticos, son espacios públicos en los que se expresan opiniones sobre diversos temas (Ghosh, Singhania, Singh, Rudra, y Ghosh, 2019). En este contexto gana importancia analizar la postura en determinados entornos para detectar los estados de opinión sobre un tema concreto (ALDayel y Magdy, 2021).

Desde una perspectiva general, la detección de posturas puede definirse como la tarea de identificar la perspectiva de un autor o texto frente a un tema, una afirmación, un titular o incluso una personalidad (Zarrella y Marsh, 2016; Ghosh y cols., 2019). Por lo tanto, tendríamos una tupla con la frase o discurso por un lado y el tema por otro lado. Mediante un proceso de clasificación habría que determinar cómo se sitúa el primero con respecto al segundo, con el objeto de conocer si ¿apoya el texto el tema? ¿está en desacuerdo con la afirmación? Las etiquetas que se utilizan para clasificar estas posturas (apoyo, en contra, a favor o neutral) dependen del problema en concreto. Esta tarea se diferencia del análisis de sentimientos en que una postura a favor o en contra puede medirse independientemente del estado emocional del autor (Zarrella y Marsh, 2016) aunque algunos trabajos consideran que es un subproblema dentro del análisis de sentimientos (Küçük y Fazli, 2020).

La detección de posturas afecta a una amplia gama de ámbitos, se estudia en áreas tan variadas como los debates políticos (Somasundaran y Wiebe, 2010; Konjengbam, Ghosh, Kumar, y Singh, 2018), los debates en foros en línea (C. Li, Porco, y Goldwasser, 2018), las redacciones de estudiantes (Faulkner, 2014) o incluso las discusiones internas de las empresas (Agrawal, Rajagopalan, Srikant, y Xu, 2003; Murakami y Raymond, 2010).

2.2.1 Conjuntos de datos para detección de posturas

Una gran cantidad de trabajos en esta tarea se ha dedicado a detectar la postura de los tuits u otros tipos de textos cortos como rumores o declaraciones de blogs con respecto a un tema determinado (Gorrell y cols., 2019). En los últimos años han surgido tareas compartidas que ofrecen conjuntos de datos en diferentes idiomas para fomentar la investigación en esta tarea. En SemEval-2016[1] se planteó la tarea de detección de posturas en tuits (tarea 6) (Mohammad, Kiritchenko, Sobhani, Zhu, y Cherry, 2016), proporcionando cerca de 5 mil tuits en inglés que cubrían cinco temas comúnmente conocidos ("Hillary Clinton" como personalidad, "Ateísmo", etc.). Un ejemplo de tuit en el tema "Hillary Clinton" es: "Hillary Clinton tiene algunos puntos fuertes y otros débiles".

En el SemEval-2017 (tarea 8) (Derczynski y cols., 2017) y SemEval-2019 (tarea 7) (Gorrell y cols., 2019) se propone clasificar la postura de rumores en forma de

[1] Taller internacional de evaluación semántica, con ediciones 2016, 2017 y 2019.

tuits con respecto a otros tuits que responden a este rumor. En el primer caso se desarrolla un corpus con 3685 ejemplos, los cuales se dividen 2907 para entrenar los modelos y 778 para realizar las pruebas. Por su parte, en la edición de 2019 se sigue el mismo principio pero está vez además se utilizan frases extraídas de Reddit[2], obteniéndose un total de 8574 ejemplos, de los cuales 6702 son para entrenar y 1872 para realizar pruebas.

Multi Perspective Consumer Health Query (MPCHI) (Sen, Sinha, Mannarswamy, y Roy, 2018) es otro conjunto de datos público dedicado a detectar la postura de las frases recogidas en artículos de calidad, hacia cinco afirmaciones diferentes (por ejemplo, "La exposición al sol provoca cáncer de piel").

En esta tarea existen abundantes trabajos que la abordan desde la perspectiva de idiomas distintos al inglés (Taulé y cols., 2017; Zotova, Agerri, y Rigau, 2021; Vychegzhanin y Kotelnikov, 2019). La carencia de conjuntos de datos en otros idiomas hizo que proliferaran tanto los esfuerzos de anotación como las tareas compartidas destinadas a avanzar en la investigación. Se han encontrado dos conjuntos de datos para la tarea de detección de posturas en idioma español y catalán sobre el tema de la independencia catalana: StanceCat (presentada en IberEval 2017 (Taulé y cols., 2017) y el conjunto de datos *Catalonia Independence* (Zotova y cols., 2021).

En (Lai y cols., 2020) se crea un conjunto de datos multilingüe, que contiene debates políticos en cinco idiomas (francés, italiano, catalán, español, inglés). Otro ejemplo de conjunto multilingüe es el creado en (Vamvas y Sennrich, 2020) sobre debates de políticos suizos en más de 150 temas. Este conjunto contiene preguntas y respuestas en tres idiomas (alemán, francés e italiano). Ambos conjuntos persiguen el objetivo de crear enfoques multilingües para la detección de posturas.

Por último, en (Baly y cols., 2018) se desarrolla un conjunto de datos que integra las tareas de detección de posturas con la verificación automática de hechos para el idioma árabe. Una de las fases más importante de la verificación automática de hechos es la recuperación de información (Lazarski, Al-Khassaweneh, y Howard, 2021), de aquí que el trabajo referido propone crear un conjuntos de datos que anote la relación de posturas entre una afirmación (*claim*) y los documentos recuperados para su evaluación. Los experimentos realizados en este trabajo muestran que los sistemas de verificación automática de hecho se pueden beneficiar de una recuperación de información con detección de posturas.

2.2.2 Enfoques aplicados en la detección de posturas

En cuanto a la tipología de técnicas utilizadas en la tarea, existe una gran variedad de enfoques. Se observan trabajos utilizando algoritmos tradicionales de ML como: Máquinas de Soporte Vectorial (SVM) (Patra, Das, y Bandyopadhyay, 2016), *K Nearest Neighbor* (K-NN) (Al-Ghadir, Azmi, y Hussain, 2021)

[2]Reddit es un sitio web de marcadores sociales y agregador de noticias donde los usuarios pueden añadir textos, imágenes, vídeos o enlaces. `https://www.redditinc.com/`.

y *Logistic Regression* (LR). Además de enfoques basados en redes neuronales como: *Long short-term memory* (LSTM) (Augenstein, Rocktäschel, Vlachos, y Bontcheva, 2016) y Red Neuronal Convolucional (CNN) (Zhou, Lin, Tan, y Liu, 2019) y algunos enfoques basados en conjuntos de clasificadores (*ensembles*) de los algoritmos mencionados (Sen y cols., 2018; Tutek y cols., 2016; L. Liu, Feng, Wang, y Zhang, 2016).

En (Al-Ghadir y cols., 2021) se propone una solución que utiliza K-NN ponderado (*weighted*) para crear un sistema de clasificación de posturas de tuits. Este sistema está dividido en tres etapas: preprocesamiento (eliminar stop-word, y otros procesos lexicográficos), extracción de características (vectores de tipo *Term Frequency-Inverse Document Frequency* (TF-IDF) e información de sentimientos) y un módulo de clasificación. La solución obtiene los mejores resultados en el SemEval-2016.

(Augenstein y cols., 2016) desarrollan una solución basada en codificadores LSTM condicional para representar los tuits y la relación con el tema. Esta solución es propuesta originalmente para la tarea de *Recognising Textual Entailment* (RTE), adaptándose a detección de postura, obteniéndose el segundo puesto en el contexto del SemEval-2016.

Por último, (Ghosh y cols., 2019) experimenta con varios enfoques de detección de posturas que han sido publicados en la bibliografía (basado en técnicas como LSTM, CNN, SVM, *Bidirectional Encoder Representations from Transformers* (BERT), entre otros) sobre dos conjuntos de datos en inglés (SemEval-2016 y el MPCHI), obteniéndose los mejores resultados con la utilización del modelo de lenguaje BERT.

2.2.3 Consideraciones generales sobre detección de posturas

A diferencia de los enfoques y conjuntos de datos mostrados en esta sección, un gran número de trabajos aplica las técnicas de detección de posturas a determinar la relación entre un titular y un contenido de noticia (Küçük y Fazli, 2020). En este caso, al involucrar documentos más largos, se enfrenta a diferentes retos. Tratar con el discurso como un conjunto coherente y cohesionado de oraciones añade una cierta complejidad que no está presente cuando se procesan enunciados más cortos o de similar longitud. Dentro del discurso, un argumento puede desarrollarse de tal manera que algunas frases pueden mostrar apoyo a la afirmación, mientras que otras pueden parecer negarlo. Únicamente considerando el documento como un todo se puede identificar efectivamente la postura, lo que tiende a la necesidad de aplicar un procesamiento más cercano al análisis semántico.

La tarea de la detección de posturas en titulares ha emergido con mucha fuerza en el contexto del análisis de noticias falsas (*fake news*) acentuado por demandas impuestas por las nuevas tecnologías para prevenir y combatir este fenómeno con un incremento en la disponibilidad de corpus anotados (Boró y cols., 2020). En la próxima sección se aborda el problema de la detección de

titulares engañosos como una tarea específica.

2.3 Titulares engañosos

El titular en una noticia es el elemento que proporciona la idea principal de la historia (Bonet-Jover y cols., 2020). Su objetivo es llamar la atención sobre la noticia de forma rápida y breve (Reis y cols., 2015), pero de forma efectiva y sin perder precisión ni resultar engañoso, para respaldar la veracidad de toda la noticia (Kuiken y cols., 2017). Este brinda al lector una idea preliminar de lo que leerá en el artículo (van Dijk, 1988), siendo el mecanismo natural de los lectores para elegir si una noticia es leída en su totalidad (Dor, 2003). Lamentablemente, en la práctica, los titulares están más enfocados a "enganchar" al lector que en la precisión de las ideas que plantean, lo que en muchas ocasiones deriva en la creación de titulares engañosos (Normala, Jamil, Ishak, y Lilly Suriani, 2021).

Estudios recientes plantean que la primera impresión obtenida en la lectura de un titular influye en las conclusiones obtenidas con la lectura del artículo completo (Reis y cols., 2015) e incluso puede persistir sobre estas aun cuando la información sea contradictoria entre el titular y el contenido de la noticia (Ecker y cols., 2014; Normala y cols., 2021). A ello se suma el comportamiento habitual de muchos usuarios que comparten noticias incluso sin leer el contenido completo de estas y sin verificar su veracidad (Gabielkov y cols., 2016).

Como si fuera poco, en los medios impresos tradicionales, el lector tiene disponible ante su vista el titular y el contenido de la noticia, aspecto que ha cambiado radicalmente en los medios digitales donde para constatar el contenido de la noticia, el lector debe dar clic sobre un enlace. Por ello, hay cierta tendencia a inferir el contenido de la información por medio de la interpretación del titular sin leer en profundidad la noticia, por lo que el riesgo de no detectar contradicciones aumenta considerablemente (Kuiken y cols., 2017). Estos hallazgos convierten a los titulares engañosos en un serio problema que afecta a la sociedad actual.

En (Ecker y cols., 2014) se hace un estudio detallado de los efectos que pueden causar en los lectores los titulares engañosos, concluyendo que estos deben considerarse un aspecto crítico de la alfabetización y la educación mediática. Además, se recomienda centrarse en monitorear profundamente la forma en que los editores de los periódicos y sitios web de noticias enmarcan los titulares y sus contenidos, exigiendo precisión fáctica.

En este contexto gana importancia la detección de titulares engañosos de forma automática debido al gran volumen de noticias que se generan y distribuyen principalmente por redes sociales (K. Park y cols., 2020) y a la velocidad con que un titular atractivo, y no necesariamente verificado, es compartido y se convierte en una componente viral (Lutz y cols., 2020). En los últimos años se han propuesto diferentes retos y competiciones de investigación para abordar este problema. Las competiciones, los conjuntos de datos y los enfoques más

importantes se explican en detalles a continuación. Se hace especial énfasis en el desafío *Fake News Challenge* (FNC-1) debido a la gran cantidad de propuestas que lo han utilizado.

2.3.1 Detección de titulares engañosos vs detección de posturas

Desde la óptica de la detección de posturas se creó el desafío FNC-1 que pone a disposición un conjunto de datos para la detección de titulares engañosos. Este tiene como objetivo crear soluciones explotando tecnologías de IA, especialmente ML y PLN para abordar la detección automática de noticias falsas. El FNC-1 (Babakar y cols., 2016) fue creado utilizando el conjunto de datos Emergent (W. Ferreira y Vlachos, 2016) como punto de partida.

El corpus FNC-1 fue liberado con alrededor de 75000 instancias que fueron clasificadas como *agree*, *disagree*, *discuss* o *unrelated*. Por ejemplo, dado el título *"Robert Plant ha roto un contrato de 800 millones de libras para la unión de Led Zeppelin"*, los siguientes fragmentos[3] ilustrarían las diferentes clases mencionadas para FNC-1:

- *agree*: el contenido de la noticia concuerda con el titular. Ejemplo de evidencia: *"[...] Robert Plant de Led Zeppelin rechazó 500 millones de libras para reformar el supergrupo."*
- *disagree*: el contenido de la noticia no concuerda con el titular. Ejemplo de evidencia: *" [...] No, Robert Plant no rompió un acuerdo de 800 millones de libras para que Led Zeppelin volviera a estar juntos."*
- *discuss*: el contenido de la noticia trata el mismo tema que el titular, pero no se presenta un posicionamiento en específico. Ejemplo de evidencia: *"[...] Robert Plant supuestamente rompió un acuerdo de unión de Led Zeppelin de 800 millones."*
- *unrelated*: el contenido de la noticia no está relacionado con el titular. Ejemplo de evidencia: *"[...] Virgin Galactic de Richard Branson está lista para lanzar SpaceShipTwo hoy."*

La competencia FNC-1 recibió un total de 200 presentaciones que lograron puntajes relativos[4] de alrededor de 82% en los equipos mejor punteados (o ranqueados). La organización propuso un sistema *baseline*[5] utilizando características externas y un clasificador de tipo *gradient boosting*. Este sistema se encuentra disponible en Github[6]. Los tres mejores sistemas en la competencia fueron

[3] Ejemplos extraídos del sitio web `http://www.fakenewschallenge.org/` (consultado el 15 de enero de 2022).

[4] Puntuación propuesta por los organizadores del desafío.

[5] Término en inglés que refiere un punto de partida o de referencia.

[6] `https://github.com/FakeNewsChallenge/fnc-1-baseline` (consultado el 15 de enero de 2022).

Talos (Baird, Sibley, y Pan, 2017), el sistema Athene (Andreas Hanselowski y Caspelherr, 2017) y UCLMR (Riedel, Augenstein, Spithourakis, y Riedel, 2017) en el orden que se presenta.

Talos (Baird y cols., 2017) propone un conjunto de clasificadores basado en un promedio ponderado de 50/50 entre un árbol de decisión de tipo *gradient boosting* y una red neuronal de tipo CNN. El árbol de decisión utiliza características externas obtenidas de la relación entre el titular y el contenido de noticia (cantidad de palabras superpuestas, medida de similitud entre bigramas y trigramas de cada parte y entre los vectores TF-IDF). Por su parte, la red neuronal CNN recibe el titular y el contenido de la noticia (representados como vectores a nivel de palabra utilizando vectores pre-entrenados de tipo word2vec (Mikolov, Chen, Corrado, y Dean, 2013)) y posteriormente se coloca una red neuronal *Multilayer Perceptron* (MLP) para llevar a cabo la clasificación en cuatro clases. Esta configuración obtiene los mejores resultados en el contexto del desafío. La mayoría de los equipos presentados en el contexto del desafío proponen sistemas basados en algoritmos de ML clásicos como SVM, LR, árboles de decisión etc.

Trabajos recientes utilizaron el FNC-1 para sus experimentos y mejoraron los resultados obtenidos en la competición. Por ejemplo, en (Q. Zhang y cols., 2019) enfocaron el problema proponiendo una representación jerárquica de las clases, la que combina las clases *agree*, *disagree* y *discuss* en una nueva clase *related*. Este trabajo diseña una red neuronal en dos capas, que es capaz de aprender esta representación jerárquica de clases, obteniendo un puntaje relativo de 89,30% en su propuesta.

Además, en (Dulhanty, Deglint, Daya, y Wong, 2019) construyeron un modelo de detección de posturas realizando un ajuste fino del modelo de lenguaje *Robustly Optimized BERT Pretraining Approach* (RoBERTa), aprovechando la atención cruzada bidireccional entre pares de titular-contenido del artículo mediante la codificación de pares con autoatención. Este trabajo reporta un puntaje relativo de 90,01%.

Por último, (Slovikovskaya, 2019) realiza una profunda experimentación con algunos modelos basados en aprendizaje por transferencia sobre este corpus, concluyendo que el modelo RoBERTa es el que mejores resultados obtiene. Además, valida la generalidad de los modelos entrenados sobre otro conjunto de datos, obteniéndose resultados prometedores.

El aspecto más interesante de resolver la detección de titulares engañosos como una tarea de detección de posturas es que no solo se enfoca en determinar si un titular es consistente o no con su contenido, sino que también brinda una clasificación detallada que determina el tipo de disonancia. Son abundantes las investigaciones que aplican enfoques de detección de posturas para detectar titulares engañosos, sin embargo, existen otros modos de abordar este problema que serán introducidos en secciones posteriores.

2.3.2 Enfoques fuera de la detección de posturas

En el contexto de la relación entre el titular y el contenido de noticia, existe una gran diferencia de tamaño y de niveles de complejidad lingüística. Algunas soluciones a estos problemas van desde extraer citas clave (Pouliquen, Steinberger, y Best, 2007), afirmaciones (Vlachos y Riedel, 2015) para facilitar la detección hasta evadir la diferencia de tamaño entre las partes (Chesney y cols., 2017).

Existen trabajos que abordan el problema de la extensión del contenido de las noticias proponiendo arquitecturas complejas (Yoon y cols., 2018, 2021). Estos trabajos dividen el contenido de la noticia en párrafos y analizan independientemente la relación con el titular de cada uno de ellos.

En (Yoon y cols., 2018) se codifica la relación de similitud entre cada párrafo de la noticia y el titular mediante el uso de una Red Neuronal Recurrente (RNN) con mecanismo de atención. Posteriormente la codificación de cada relación es pasada a través de otra red de tipo RNN para finalmente clasificar la relación entre titular y contenido. Se valida la generalidad de la arquitectura, probándola en el conjunto de datos FNC-1, obteniendo mejores resultados que el equipo de mejor puntaje dentro de la competición. Este trabajo además analiza la carencia de conjuntos de datos para la detección de titulares engañosos, proponiendo un conjunto de datos en idioma coreano, para lo que se extraen noticias de medios de comunicación de reconocido prestigio en Corea. El conjunto de datos se creó insertando partes del contenido de otro artículo del corpus en un artículo escogido y así hacerlo incongruente. El proceso de creación del corpus es automático, lo que permite tener casi 2 millones de ejemplos entre *congruent* e *incongruent*.

Una propuesta de arquitectura similar a la descrita anteriormente es desarrollada en (Yoon y cols., 2021), creando un modelo de detección basado en *Graph Neural Network* (GNN). Este modelo crea una representación en grafo de la relación entre el titular y los párrafos del contenido de noticia para procesar los extensos textos de un contenido de noticia. Este enfoque es probado sobre un conjunto de datos propio desarrollado en idioma inglés donde logra superar al enfoque propuesto en (Yoon y cols., 2018).

Otra solución posible al problema de la diferencia de longitud entre titular y contenido de noticia podría ser aplicar resúmenes automáticos para reducir la complejidad y longitud del contenido de noticia, elemento que podría beneficiar a los modelos de detección basados en redes neuronales, con los cuales se reportan problemas para procesar textos largos (vea análisis de la sección 2.6.2). El resumen como recurso de reducción de tamaño se ha utilizado en múltiples tareas dentro del PLN, tanto en la detección de noticias falsas (Esmaeilzadeh, Peh, y Xu, 2019; Kim y Ko, 2021), así como en el contexto de discusiones en línea y redes sociales para detectar si el autor de un comentario está a favor o contra, por ejemplo, de una entidad o tema (Krejzl, Hourová, y Steinberger, 2017; Krejzl, 2018). Sin embargo, no se han encontrado trabajos que aborden la detección de titulares engañosos utilizando resúmenes para reducir la longitud del contenido de la noticia.

En la bibliografía es común encontrar trabajos que mezclan los conceptos de titulares engañosos, titulares incongruentes y titulares ciberanzuelos (*click-baits*). Con el análisis realizado en esta sección se propone una relación entre términos basado en el objetivo y las técnicas utilizadas en las soluciones de los trabajos consultados, tal y como se muestra en la figura 2.1.

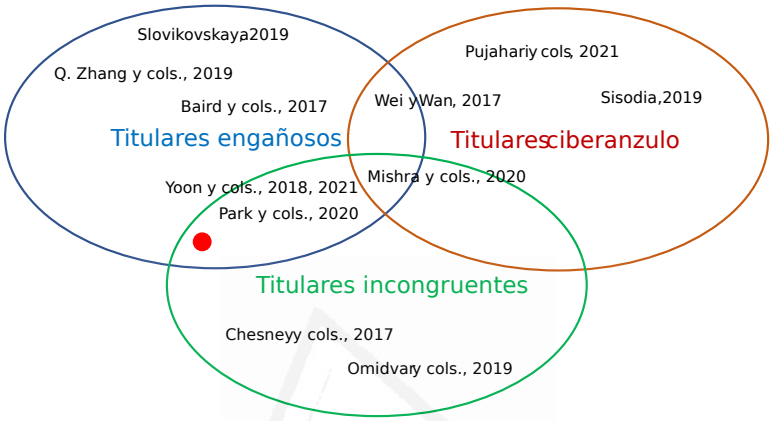


Figura 2.1: Relación entre términos y trabajos relacionados.

En las definiciones encontradas en diferentes trabajos existen diferentes interpretaciones de los términos. (Mishra, Yadav, Calizzano, y Leippold, 2020) plantea que los titulares de noticias que representan incorrectamente el contenido de la noticia se denominan incongruentes o ciberanzuelos. Otro trabajo plantea que no existe una única definición en la literatura para estos titulares, por lo que proponen nombrar titulares ciberanzuelos a los titulares cuyo objetivo principal es atraer la atención de los lectores para que hagan clic en la noticia (Kuiken y cols., 2017). Por su parte (Molina, Sundar, Le, y Lee, 2021) plantea que un titular ciberanzuelo no necesariamente tiene porque ser un titular engañoso. En el conjunto que representa esta definición han sido colocados trabajos que abordan la detección de titulares ciberanzuelos analizando solamente el titular, sin tener en cuenta el contenido de la noticia.

En (Pujahari y Sisodia, 2021) se propone un sistema automático de detección de titulares ciberanzuelos que clasifica en *clickbait* y *non-clickbait*. Este experimenta la validez de un total de once características extraídas del titular (ambigüedad, exageración, cebo, etc.) sobre tres algoritmos de ML (SVM, árboles de decisión y bosques aleatorios). Los resultados obtenidos con los tres algoritmos son muy similares, mejorando los resultados de sistemas anteriores sobre el corpus utilizado. (Sisodia, 2019) es otro ejemplo de investigación que representa este tipo de titulares, esta investigación propone experimentar sobre algunos métodos basados en conjuntos de clasificadores de tipo *bagging*, bosques aleatorios y *adaptive boosting* (adaboost). Se utilizan características extraídas únicamente del titular, obteniéndose que el método de ensamblador bosques aleatorio es el que mejor rendimiento obtiene.

Los titulares incongruentes son identificados en (Chesney y cols., 2017) como una exageración sutil o una tergiversación de los hechos, pero no es necesario que represente un punto de vista opuesto. Se encuentran trabajos que abordan la incongruencia como (Chesney y cols., 2017) que realiza un análisis exhaustivo de la tarea concluyendo que no se cuenta con conjuntos de datos acordes para la tarea. Además, (Omidvar, Poormodheji, An, y Edall, 2019) propone una solución que evalúa la calidad de los titulares de noticias antes de su publicación, basado en la relación de similitud entre titular y contenido. Es una solución que aunque no detecta titulares engañosos, resulta interesante su aplicación en el ámbito periodístico para evitar que se publiquen titulares de baja calidad.

Por su parte los titulares engañosos son definidos en (Wei y Wan, 2017) como "un titular cuyo significado difiere del contenido de la noticia". Solo se puede descubrir el engaño si se lee la noticia completa. En esta definición se encuentran los trabajos relacionados con el FNC-1 (Baird y cols., 2017; Q. Zhang y cols., 2019; Slovikovskaya, 2019).

En la frontera entre los titulares engañosos y titulares incongruentes se han colocado algunos trabajos presentados con anterioridad en esta sección (Yoon y cols., 2018, 2021). Estos abordan la relación semántica entre un titular y el contenido de la noticia, utilizando indistintamente el término incongruente y engañoso. Además en (K. Park y cols., 2020) se presenta BaitWatcher, un sistema que alerta a los usuarios de la existencia de incongruencias entre titulares y los contenidos de noticias antes de hacer clic en el enlace de la noticia. Este sistema está disponible para ser utilizado mediante una extensión del navegador web Chrome.

Entre titulares ciberanzuelos y titulares engañosos se ha encontrado una investigación que desarrolla un conjunto de datos para detectar titulares ambiguos y titulares engañosos en idioma chino (Wei y Wan, 2017). Para ambos conjuntos de datos se proponen sistemas *baseline* de clasificación utilizando el algoritmo SVM. En el primer caso se utilizan características relacionadas con titulares ciberanzuelos y en el segundo caso características de similitud, informalidad, análisis de sentimiento, entre otras variantes.

Por último, existe un trabajo que utiliza los tres términos indistintamente (Mishra y cols., 2020). Esta investigación utiliza dos conjuntos de datos, uno de la tarea de titulares ciberanzuelos y otro de la tarea de titulares incongruentes para desarrollar un sistema de detección basado en atención mutua para mejorar la tarea de similitud semántica. Este se diferencia de los trabajos típicos de titulares ciberanzuelos en que analiza la relación semántica entre el titular y el contenido acercándose al resto de definiciones.

En este trabajo se utiliza el término titulares engañosos debido al interés de detectar el tipo de titulares que tiene como objetivo engañar al lector. Esto no implica que el titular no pueda ser considerado incongruente o ciberanzuelo. De hecho, se han referido trabajos que se encuentran claramente en la frontera entre dos o más definiciones. La interpretación realizada con este estado del ar-

te encuentra una alta relación entre los titulares engañosos e incongruentes por lo que esta investigación se sitúa en la frontera entre estas dos definiciones[7], intentando detectar contradicciones claras entre titular y contenido, así como las incongruencias más sutiles. La mayoría de los trabajos presentados en esta sección están relacionados con la definición de titulares engañosos y algunos con titulares incongruentes, pero obviando los enfoques relacionados con titulares ciberanzuelos.

Conceptualmente la detección de titulares engañosos se puede reducir a determinar una relación semántica entre dos parejas de textos. Haciendo esta simplificación se podría intentar detectar titulares engañosos mediante el uso de tareas genéricas dentro del PLN. Por ejemplo, en (Chesney y cols., 2017) se menciona la posibilidad de beneficiarse de tareas tales como detección de contradicciones (De Marneffe y cols., 2008), contraste (Harabagiu y cols., 2006) y RTE (Levy, Zesch, Dagan, y Gurevych, 2013).

Todo lo anterior respalda una de las hipótesis de este trabajo que sugiere la detección de contradicciones como una tarea que beneficie la detección de titulares engañosos. En la sección siguiente se analizará la tarea de detección de contradicciones con el fin de comprender como se desarrolla y las posibles intersecciones que existe con la detección de titulares engañosos.

Con el análisis de esta sección se evidencia de manera clara la no existencia de conjuntos de datos en un idioma tan hablado en el mundo como el español, aspecto que justifica la atención a investigaciones en el ámbito de esta lengua.

2.4 Detección de contradicciones

La contradicción entre textos se define en (De Marneffe y cols., 2008) como: "Entre dos textos A y B no existe ninguna situación en la que A y B sean ambos verdaderos". Por lo tanto, en PLN, la tarea de identificación de contradicciones implica detectar declaraciones en lenguaje natural que transmiten información sobre eventos o acciones que no pueden ocurrir simultáneamente (Dragos, 2017). Esta tarea puede ser de interés en otros campos dentro del PLN que en parte impliquen detectar elementos contradictorios entre dos textos. Por ejemplo, para la verificación de hechos puede ser útil detectar contradicciones entre afirmaciones y evidencias de fuentes confiables, lo que puede ser extensible a la tarea de detección de titulares engañosos entre el titular y el contenido de la noticia.

En (De Marneffe y cols., 2008) se presenta una definición de diferentes tipos de contradicciones, donde los autores definen una tipología para la contradicción en inglés, encontrando dos categorías principales: (1) aquellas que ocurren a través de antonimia, negación y desajuste de fecha o número, que son relativamente simples de detectar, y (2) contradicciones que surgen del uso de palabras

[7] Pequeño círculo rojo representado en la figura 2.1

fácticas o modales, contrastes léxicos estructurales y sutiles, así como el conocimiento del mundo. Con ello, se introducen aspectos formales (estructurales) y contextuales (cierta aproximación semántica) a la definición de contradicciones.

En esta sección se presenta una revisión de los métodos de detección de contradicciones y se introducen algunas investigaciones en este campo para dominios específicos. Finalmente, se proporciona una revisión de los principales recursos existentes en diferentes idiomas para la detección de contradicciones, componentes indispensables para la realización de estas tareas.

2.4.1 Métodos de detección de contradicciones

Los métodos de detección de contradicciones siguen varias aproximaciones conceptuales, siendo las más destacadas aquellas que colocan las prioridades en el análisis lingüístico o estructural, las que profundizan en aspectos de carácter semántico y las que hibridan enfoques.

Las primeras investigaciones sobre la detección de contradicciones en el campo del PLN fueron reportadas por (Harabagiu y cols., 2006) cuyo trabajo abordó las contradicciones por medio de tres tipos de información lingüística: negación, antonimia e información semántica y pragmática asociada con las relaciones discursivas. Después de los experimentos de evaluación, se obtuvo un 62% de precisión para detectar contradicciones.

En (Lingam y cols., 2018) se propuso un enfoque para detectar tres tipos diferentes de contradicciones (negación, antónimos y desajuste numérico). Este enfoque implementa una red RNN de tipo LSTM y vectores para la representación de palabras *word embedding* de tipo *Global Vectors for Word Representation* (GloVe) e incluye cuatro características lingüísticas extraídas del texto: coeficiente de *jaccard* (Vijaymeena y Kavitha, 2016), indicador de negación, indicador de antónimo y coeficiente de superposición (Vijaymeena y Kavitha, 2016).

La utilización de evidencias lingüísticas como polaridad, números, fechas y horas, antonimia, estructura, información fáctica y características de modalidad para detectar contradicciones es característico del trabajo de (De Marneffe y cols., 2008).

Un sistema que utiliza métricas simples de similitud de texto (similitud de coseno (Passalis y Tefas, 2018), puntuación F_1 y alineación local (Smith y Waterman, 1981)) es creado en (Lendvai y Reichel, 2016), obteniendo buenos resultados para la clasificación de contradicciones. Este enfoque utilizó dos conjuntos de datos construidos con ejemplos de pares de tuits.

Como se puede apreciar, los trabajos (Lendvai y Reichel, 2016; De Marneffe y cols., 2008) son enfoques que solamente utilizan características lingüísticas. Por otra parte (Harabagiu y cols., 2006; Lingam y cols., 2018) son aproximaciones que hibridan enfoques sintácticos y semánticos.

Otros sistemas de detección de contradicciones se basan en enfoques puramente semánticos (Dragos, 2017; Pham, Nguyen, y Shimazu, 2013; Ritter, Soder-

land, Downey, y Etzioni, 2008; L. Li, Qin, y Liu, 2017).

Un modelo para detectar la contradicción y una arquitectura que permite su validación se presenta en (Dragos, 2017). El modelo define la extracción de relaciones semánticas entre un par de oraciones y verifica algunas reglas para detectar contradicciones. Además, este autor definió una medida de la contradicción considerando la estructura de relaciones extraídas de los textos y el nivel de incertidumbre que se les atribuye.

En (L. Li y cols., 2017) se crea un conjunto a gran escala de pares contrastantes y el modelo *contradiction-specific word embedding* (CWE) de detección. Este enfoque mejoró los resultados en la detección de contradicciones en un corpus de referencia en SemEval 2014 (Marelli y cols., 2014). Esta investigación concluyó que los algoritmos tradicionales de *word embedding* han tenido un gran éxito en el cumplimiento de las principales tareas de PLN, pero la mayoría de estos algoritmos no son lo suficientemente potentes para la tarea de detección de contradicciones (L. Li y cols., 2017).

Otros autores (Pham y cols., 2013) combinaron representaciones semánticas superficiales derivadas del *semantic role labeling* (SRL) con relaciones binarias extraídas de oraciones en un sistema basado en reglas.

2.4.2 Detección de contradicciones en dominios específicos

Se constatan investigaciones de dominio específico con respecto a la detección de contradicciones. En el ámbito médico, los autores de (Rosemblat, Fiszman, Shin, y Kilicoglu, 2019) detectaron contradicciones al comparar tuplas sujeto-relación-objeto de un par de textos en el dominio de la Medicina. Este trabajo detectó 2236 contradicciones automáticamente, pero estas contradicciones se comprobaron manualmente y solo 56 detecciones eran correctas.

En (Vosoughi y cols., 2018) se creó un sistema de clasificación basado en SVM, con algunas características (negación, antónimos y medidas de similitud) que ayudan a detectar contradicciones en textos médicos. Este sistema detectó antónimos y contradicción de negación, pero no contradicción numérica. Estos resultados mejoraron el estado del arte en un corpus de datos médicos.

En el dominio del Turismo, otra investigación proporciona un análisis del tipo de contradicciones presentes en las reseñas de hoteles en línea. Además, se propone un modelo para la detección de contradicciones numéricas para la industria del turismo (Azman, Ishak, Sharef, y Sidi, 2017).

2.4.3 Recursos para la detección de contradicciones

Actualmente, la disponibilidad de grandes conjuntos de datos anotados para la detección de contradicciones está principalmente presente para idioma inglés (Sifa y cols., 2019), como SNLI (Bowman, Angeli, Potts, y Manning, 2015), MultiNLI (A. Williams, Nangia, y Bowman, 2018), o incluso el conjunto de datos en varios idiomas XNLI (Conneau y cols., 2018). Estos tres conjuntos de datos

han permitido el entrenamiento de complejos sistemas de aprendizaje profundo, que requieren corpus muy grandes para obtener resultados exitosos.

Existen investigaciones que los utilizan para fusionar modelos de DL con conocimiento externo. Las relaciones de Wordnet se introdujeron en (Q. Chen, Zhu, Ling, Inkpen, y Wei, 2017) para enriquecer los enfoques de redes neuronales en *Natural Language Inference* (NLI), que es un paso previo en la detección de contradicciones. En otro sentido, la investigación desarrollada en (Z. Zhang y cols., 2020) introduce información derivada del SRL que permite mejorar modelos basados en aprendizaje por transferencia.

Según el conocimiento de los autores, existen pocos estudios que aborden la detección de contradicciones en idiomas distintos del inglés, como los de (Sifa y cols., 2019; Takabatake y cols., 2015; Rahimi y ShamsFard, 2021). La traducción automática de SNLI del inglés al alemán se realizó en (Sifa y cols., 2019). Estos autores construyeron un modelo usando la versión alemana de SNLI y los resultados de las predicciones son muy similares al mismo modelo entrenado en la versión original de SNLI en inglés. (Takabatake y cols., 2015) ha creado una base de datos a gran escala de pares de eventos contradictorios en japonés. Esta base de datos se utiliza para generar declaraciones coherentes para un sistema de diálogo. (Rahimi y ShamsFard, 2021) realizó la traducción automática al idioma persa de un subconjunto de ejemplos de los corpus SNLI y MNLI. Este corpus en idioma persa se utiliza para crear un sistema de detección de contradicciones.

Desde la perspectiva multilingüe, en (Conneau y cols., 2018) se ha creado el conjunto de datos XNLI en varios idiomas, que se divide en tres particiones: entrenamiento, desarrollado y prueba. El conjunto de entrenamiento se desarrolla en inglés, y los conjuntos de desarrollo y prueba están disponibles en 15 idiomas diferentes. El XNLI se ha utilizado para crear sistemas de detección de contradicciones para entrenar en inglés y predecir en otros idiomas, obteniendo buen rendimiento. Cada ejemplo en XNLI se clasifica como *contradiction, entailment* o *neutral.*

El conjunto de datos XNLI no distingue entre diferentes tipos de contradicciones en su anotación, y el idioma español solo está disponible en los conjuntos de desarrollo y prueba traducidos automáticamente del inglés. Ambos hechos pueden afectar el rendimiento de los modelos creados a partir del conjunto de datos XNLI para diferentes idiomas.

Según (De Marneffe y cols., 2008) para lidiar con las contradicciones es importante considerar su amplia gama y gran variedad de características. Partiendo de esta afirmación podría ser interesante diferenciar entre los diferentes tipos de contradicciones para ayudar a realizar un tratamiento más específico de las mismas, potenciando así la capacidad para detectarlas de forma más amplia sin disponer de muchos ejemplos previos de ellas.

Retomando la carencia de conjuntos de datos en idioma español tanto para la tarea de detección de titulares engañosos y detección de contradicciones sería interesante crear un conjunto de datos que permita desarrollar sistemas de detección de titulares engañosos y de contradicciones. Este corpus puede permitir

evaluar el beneficio de asociar estas dos tareas en esta lengua.

2.5 Problemas de clasificación

En ML, la mayoría de los trabajos relacionados con problemas de clasificación se abordan mediante clasificación flat o plana. La clasificación plana se refiere tanto a problemas de clasificación binaria como de múltiples clases o de múltiples etiquetas. En estos, cada instancia se clasifica en un conjunto de clases posibles y sin tener en cuenta posibles relaciones jerárquicas entre ellas (Freitas y Carvalho, 2007). La clasificación binaria solo diferencia dos clases; en los problemas de múltiples clases se discriminan más de dos clases, con una etiqueta por instancia. Por último, en los problemas de clasificación de múltiples etiquetas se busca una mayor especificidad en la clasificación, cada instancia puede estar asociada a una o más etiquetas (R. M. Pereira, Bertolini, Teixeira, Silla, y Costa, 2020).

Los modelos computacionales que abordan tareas de clasificación dentro de ML buscan fundamentalmente mejorar la precisión en sus clasificaciones. Esta precisión se ve afectada cuando la cantidad de clases aumenta (Lorena, De Carvalho, y Gama, 2008). Además, según (Kumar, Ghosh, y Crawford, 2002) con el aumento de clases en problemas de clasificación múltiple, la representatividad de cada clase individual se empieza a solapar con el resto de las clases, dificultando así la tarea de clasificación. Los problemas de clasificación abordados en esta investigación implican esencialmente problemas de clasificación de múltiples clases por lo que se analiza una técnica que puede influir positivamente en la precisión de la clasificación, la clasificación jerárquica.

2.5.1 Clasificaciones jerárquicas

Existen problemas de clasificación más complejos en los que las clases que se van a predecir están relacionadas jerárquicamente (Silla y Freitas, 2010). En los problemas de clasificación jerárquica, las clases están dispuestas en una estructura jerárquica, como un árbol o un Grafo Acíclico Dirigido (DAG). En estas estructuras se asume que cada nodo en la jerarquía es un tipo genérico de sus nodos secundarios y un tipo específico de su nodo principal (Naik y Rangwala, 2018; Freitas y Carvalho, 2007). La figura 2.2, reproducida de (Naik y Rangwala, 2018), muestra la estructura arbórea de una jerarquía de clases.

Se consultaron las definiciones formales de Naik y Rangwala y Ma y cols., prefiriéndose para los objetivos de este trabajo la última.

Definición de (Naik y Rangwala, 2018): “La estructura jerárquica (o jerarquía) se define como un conjunto de objetos V y un orden parcial sobre los pares de elementos de V. El orden parcial proviene de la relación padre-hijo entre los elementos de V”.

Definición de (Ma y cols., 2022): “La estructura jerárquica formaliza la relación entre categorías mediante la forma de un árbol donde una categoría tiene

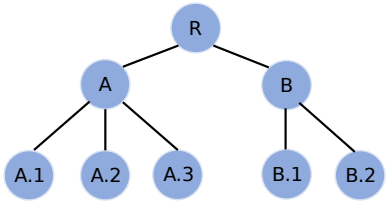


Figura 2.2: Representación de una jerarquía de clases en un árbol.

al menos una categoría principal, o de un grafo acíclico dirigido donde una categoría posiblemente tiene múltiples categorías principales, que se pueden introducir como conocimiento externo para mejorar el desempeño de clasificación".

La clasificación en los clasificadores jerárquicos se basa en decisiones tomadas durante la predicción. Estas decisiones equivalen a la ruta entre la raíz y la hoja predicha. Si en algún nodo intermedio se toma una decisión incorrecta aparece un error que se propaga entre ese nodo y la hoja, lo que afecta la precisión de la clasificación (Babbar, Partalas, Gaussier, y Amini, 2013). Algunos desafíos que se presentan en las clasificaciones jerárquicas (nodo hoja obligatorio versus predicción de nodo interno, categorías extrañas, selección de características, aprender la relación jerárquica, escalabilidad, estructura jerárquica inconsistente y la propia propagación del error) y sus posibles soluciones pueden consultarse en (Naik y Rangwala, 2018).

Una clasificación de los enfoques que se pueden aplicar para solucionar el problema de la clasificación jerárquica es propuesta por (Naik y Rangwala, 2018). La figura 2.3, reproducida de (Naik y Rangwala, 2018), muestra una taxonomía de enfoques para solucionar el problema de la clasificación jerárquica. En la figura se aprecian dos grandes grupos según el valor que le conceden a la relación jerárquica. La clasificación plana ignora la relación jerárquica, mientras que su consideración puede realizarse a nivel global o local (clasificador global o clasificador local).

Clasificación plana

En la mayoría de los problemas de clasificación, independientemente del área, esta es la solución más habitual, sencilla y rápida de alcanzar que no explota la relación jerárquica entre clases (Balyan, McCarthy, y McNamara, 2020). Se basa en una única decisión que incluye todas las clases. Por lo general, no alcanzan resultados de vanguardia en problemas de clasificación jerárquica, debido a que la decisión es difícil de tomar pues involucra muchas categorías, potencialmente desequilibradas (Babbar y cols., 2013).

Clasificador local

Este método explora la información de la estructura jerárquica local, como las relaciones entre padres e hijos y hermanos durante el aprendizaje del modelo.

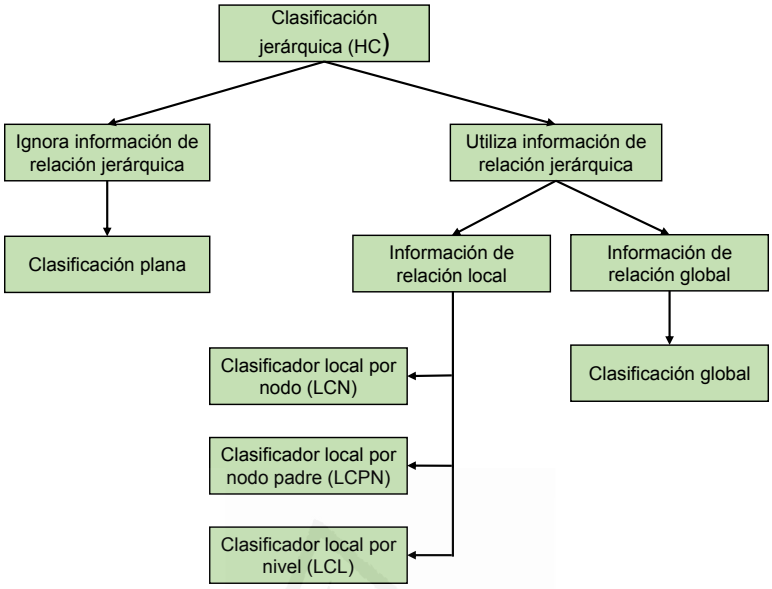


Figura 2.3: Enfoques para solucionar el problema de la clasificación jerárquica.

El enfoque de clasificación local se puede clasificar en tres categorías en función de la extracción de información local en la fase de entrenamiento del modelo de clasificación (Silla y Freitas, 2010; Freitas y Carvalho, 2007; Naik y Rangwala, 2018). Se reconocen como Clasificador Local por Nodo (LCN), Clasificador Local por Nodo Padre (LCPN) y Clasificador Local por Nivel (LCL). A continuación se muestra en detalle cada uno de los tipos de clasificadores locales:

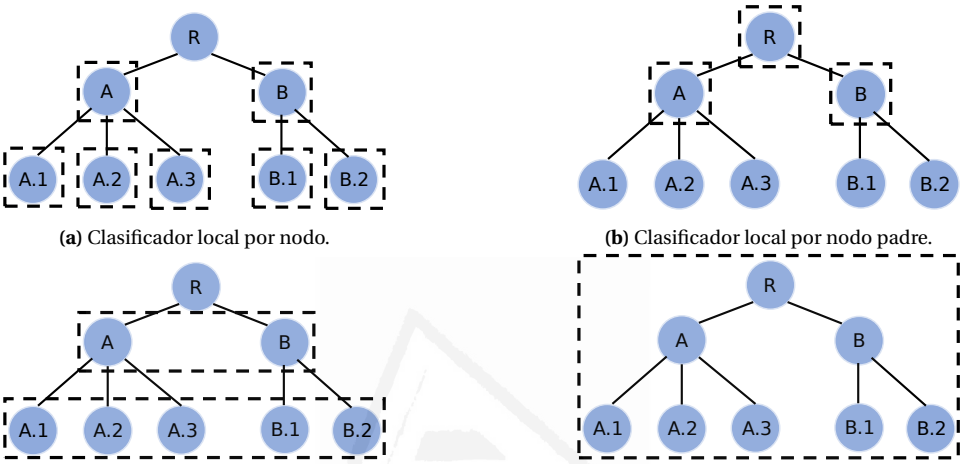
- **Clasificador local por nodo** (LCN):

  El objetivo de este enfoque es discriminar eficazmente entre los nodos hermanos en la jerarquía, para ello, crea un clasificador binario por cada nodo exceptuando al nodo raíz. Existen algunos tipos de este clasificador que se diferencian en los ejemplos que se utilizan para entrenar cada uno de los nodos. Una de sus principales desventajas es que hay que entrenar un clasificador por cada nodo, siendo el enfoque que más clasificadores necesita, sin embargo, no está exento de problemas de inconsistencia en la clasificación. La figura 2.4a muestra una representación de este clasificador.

- **Clasificador local por nodo padre** (LCPN):

  En este enfoque se crea un clasificador de múltiples clases para cada uno de los nodos padres en la jerarquía, cada clasificador está diseñado para discriminar cada uno de sus nodos hijos. En la figura 2.4b se aprecia este enfoque. En la fase de entrenamiento el clasificador en cada nodo padre solo utiliza los ejemplos de sus descendientes. Para la fase de predicción el

enfoque habitualmente se combina con el enfoque de arriba hacia abajo (Silla y Freitas, 2010). Se parte del clasificador entrenado para el nodo raíz y así se va descendiendo hasta llegar al nodo hoja o clase. Aplicando esta metodología para la predicción se evita hacer predicciones inconsistentes, respetando las limitaciones naturales de la pertenencia a una clase.



(a) Clasificador local por nodo.

(b) Clasificador local por nodo padre.

(c) Clasificador local por nivel.

(d) Clasificación global.

Figura 2.4: Clasificadores locales y globales.[8]

- **Clasificador local por nivel** (LCL):

  Entre los enfoques locales, este es el enfoque menos popular en la literatura. Este consiste en crear un clasificador de clases múltiples para cada uno de los niveles de la jerarquía, como se muestra en la figura 2.4c. En la fase de entrenamiento, el clasificador de cada nivel utiliza los ejemplos de clases de ese nivel y los ejemplos de los descendientes. La predicción por su parte se realiza eligiendo el nodo clasificado en cada nivel de la jerarquía. Al realizarse predicciones independientes, la ejecución de la predicción se puede paralelizar fácilmente, pero puede ocasionar problemas de inconsistencia entre los niveles verticales, prediciendo una clase en un nivel inferior que no tiene relación con la predicción del nivel anterior.

Clasificación global

El enfoque de clasificador global construye un modelo único de clasificación (relativamente complejo) a partir del conjunto de entrenamiento. A diferencia de los clasificadores planos se tiene en cuenta la jerarquía de clases (Silla y Freitas, 2010). Como ventaja significativa con respecto a los clasificadores locales se encuentra que no es necesario entrenar numerosos clasificadores y no existe

[8] Cada recuadro representado con líneas discontinuas expresa el alcance de la clasificación de cada enfoque. Reproducida de (Silla y Freitas, 2010).

el problema de la inconsistencia en la clasificación como ocurre en algunos enfoques de clasificación local. La principal desventaja es la alta complejidad del proceso de aprendizaje y de creación del modelo (Naik y Rangwala, 2018). La figura 2.4d muestra un ejemplo de este clasificador.

Consideraciones generales sobre clasificadores jerárquicos

En las descripciones anteriores se muestran las soluciones más comunes al problema de clasificación jerárquica. Desde las más sencillas y rápidas con el uso de clasificación plana (sin explotar la estructura jerárquica de clases); hasta soluciones mucho más complejas como la clasificación global con un único clasificador con alto nivel de complejidad, pasando por soluciones de complejidad intermedia como son los clasificadores locales (que implican crear varios clasificadores para discriminar las clases de la jerarquía). Algunas de estas soluciones presentan problemas que deben ser resueltos con posterioridad a la clasificación con un post-procesamiento entre las que se puede mencionar la aparición de inconsistencias horizontales y verticales, y otras implícitas en este tipo de clasificación como son la propagación del error (Naik y Rangwala, 2018).

Una ventaja significativa de los clasificadores jerárquicos es la posibilidad de usar características específicas en cada clasificador que ayuden a discriminar mejor las clases bajo análisis (Balyan y cols., 2020). Teniendo en cuenta el desbalance de clases en conjuntos de datos jerárquicos, en (Babbar y cols., 2013) se constata que los clasificadores que tienen en cuenta la jerarquía de clases obtienen mejores resultados que los planos, sin embargo, en conjuntos balanceados y que no tengan clases extrañas, los resultados se equiparan o incluso son mejores con la clasificación plana (Deng, Satheesh, Berg, y Li, 2011; Perronnin, Akata, Harchaoui, y Schmid, 2012).

Trabajos que utilizan clasificación jerárquica

Los problemas de clasificación donde las clases están organizadas en una jerarquía funcionan mejor con enfoques que exploten la jerarquía de clases en comparación con el uso de una clasificación plana (Balyan y cols., 2020). En este sentido, aunque los enfoques de clasificación plana siguen siendo mayoritarios, existen numerosos trabajos que abordan la clasificación con un enfoque jerárquico (R. M. Pereira y cols., 2020; Kumar y cols., 2002; Beyan y Fisher, 2015; Balyan y cols., 2020; L. H. Pereira, Silla Junior, y Nievola, 2019).

En (R. M. Pereira y cols., 2020) se propone un clasificador jerárquico aplicando el enfoque de clasificación global para clasificar imágenes de rayos-x de neumonías provocadas por el COVID-19. Los autores usan una taxonomía de neumonías causadas por microorganismos. Este enfoque mejora significativamente los resultados alcanzados por sistemas de clasificación plana.

Una comparación exhaustiva entre los enfoques de clasificación local en la tarea de clasificación de proteínas es realizado en (L. H. Pereira y cols., 2019). Es-

te trabajo concluye que el enfoque LCL es estadísticamente mejor si se compara el acierto promedio jerárquico. En orden decreciente le siguen LCPN y LCN.

El concepto de descomposición jerárquica se utiliza en (Silla y Freitas, 2010) para referirse a las situaciones donde la estructura de clases jerárquica no está predefinida. Sin embargo, en esta tesis se abordan de igual forma que aquellas que tienen una jerarquía predefinida como las taxonomías. Ambos enfoques consisten en agrupar una tarea de clasificación de múltiples clases en una jerarquía de clases (creando clases ficticias o meta-clases), lo que se ha demostrado que tiene sus ventajas en problemas de este tipo y en conjuntos de datos que estén desbalancedos o contengan clases muy extrañas (Balyan y cols., 2020; Babbar y cols., 2013).

Algunos trabajos proponen crear la jerarquía de clases automáticamente, teniendo en cuenta la similitud entre las clases (Kumar y cols., 2002; Silva-Palacios, Ferri, y Ramírez-Quintana, 2018). Una de las primeras propuestas de división de un problema de clasificación de múltiples clases en problemas de clasificación binaria agrupando los clasificadores en una jerarquía aparece en (Kumar y cols., 2002). En este caso cada clasificador binario tiene su propio espacio de características específicas.

Por otra parte, un enfoque más actual experimenta con la creación automática de la jerarquía de clases basado en la obtención de una matriz de similitud generada a partir de una clasificación plana. A partir del análisis de la similitud se crea la jerarquía de clases, esta posteriormente es comprimida para reducir el número de clasificadores locales. Con esta propuesta se experimenta en quince (15) conjuntos de datos obteniendo resultados prometedores aplicando el enfoque de LCPN (Silva-Palacios y cols., 2018).

En el área de la clasificación de textos es menos habitual encontrar trabajos que aborden la clasificación con enfoque jerárquico (Balyan y cols., 2020; Xiao, Dellandrea, Dou, y Chen, 2007; Gao y cols., 2020). El estudio realizado en (Balyan y cols., 2020) convierte una clasificación de múltiples clases en clasificaciones binarias, creando una relación jerárquica entre clases, introduciendo clases ficticias. Compara varias propuestas de clasificadores jerárquicos con clasificadores planos, obteniendo mejoras significativas usando algunas combinaciones de estos.

Este propio trabajo concluye que la clasificación jerárquica puede ayudar a tareas complejas de clasificación de textos con la inclusión de características específicas y en corpus con clases desbalanceadas.

Por su parte, en (Xiao y cols., 2007) se crea un clasificador jerárquico que clasifica seis (6) tipos de clases de discurso emocional. La agrupación de clases se lleva a cabo en función de características comunes de cada una y todos los clasificadores aplicados son binarios. Los resultados obtenidos por este clasificador mejoran a los de clasificadores planos, sin embargo, al ser una estructura jerárquica artificial, los autores no garantizan que sea la mejor configuración del clasificador jerárquico.

En (Gao y cols., 2020) se aborda el problema de la clasificación jerárquica

en textos de contenidos económicos. Se define una red neuronal jerárquica que analiza la jerarquía representada en un árbol por niveles. Los autores logran obtener un enfoque conceptualmente similar a un LCL utilizando un solo clasificador.

La mayoría de las soluciones para clasificación jerárquica utilizan el enfoque LCN, por lo que es normal encontrar clasificaciones binarias en las implementaciones de estos clasificadores. Lo que no resulta normal es que la mayor parte de las jerarquías creadas artificialmente, incluyendo meta-clases se representen en un árbol binario, lo que constituye una diferencia notable con las estructuras jerárquicas que modelan situaciones de la vida real, como la taxonomía animal o vegetal; donde un nodo padre puede tener más de dos nodos hijos.

La agrupación de clases en grupos binarios representados por una meta-clase, puede ocasionar que obvien agrupaciones mayores de dos clases que comparten características comunes.

En este trabajo, la utilización de jerarquías creadas artificialmente no solo brinda una solución consistente al problema tratado y constituye un aspecto novedoso para el problema de clasificación de múltiples clases, sino que aparenta tener un significado de valor semántico en la mayoría de los casos.

2.6 Modelos neuronales

En los últimos años la IA se ha convertido en una solución popular (por efectiva) a problemas reales de las sociedades modernas (Janiesch, Zschech, y Heinrich, 2021). Algunos ejemplos de soluciones exitosas son los asistentes virtuales (Siri, Alexa, etc.) que integran una serie de tecnologías fundamentalmente relacionadas con la IA, específicamente ML y PLN. Este auge está justificado en gran medida a la mayor disponibilidad de datos y el notable avance en las tecnologías de hardware (como Unidad de Procesamiento Gráfico (GPU), Unidad de Procesamiento Tensorial (TPU), etc.) que han permitido crear arquitecturas más complejas basadas en Redes Neuronales Artificiales (ANN) (Pouyanfar y cols., 2018) con lo que se superan barreras de diseño e implementación que pueden haber presentado los enfoques de este tipo en los primeros años de su conceptualización.

Los avances tecnológicos también han permitido una evolución considerable de las ANNs, aumentado la complejidad interna de las neuronas que contienen y sus estructuras propias, para concretar en lo que actualmente se conoce como DL (Janiesch y cols., 2021). DL ha superado a los modelos tradicionales basados en ML en tareas de alta complejidad (Pouyanfar y cols., 2018). Entre estas tareas se pueden destacar el PLN (De Cao, Aziz, y Titov, 2019; Y.-C. Chen y Bansal, 2018), procesamiento de datos visuales (Vrbancic, Zorman, y Podgorelec, 2019; Ayala, Aranda, y Galar, 2021), audio (Nagrani, Chung, Xie, y Zisserman, 2020; Tanaka, Kameoka, Kaneko, y Hojo, 2019), en áreas vitales como la salud (Lavanya y Sasikala, 2021; R. M. Pereira y cols., 2020) entre otras aplicaciones

conocidas (C. Chen, Ye, Zuo, Zheng, y Ong, 2019).

Profundizando en las soluciones basadas en DL en el ámbito del PLN se encuentran enfoques de cierta generalidad y universalidad. Los conocidos algoritmos de incrustación de palabras (word2vec, GloVe (Pennington, Socher, y Manning, 2014) y FastTest (Bojanowski, Grave, Joulin, y Mikolov, 2017)) y los potentes modelos de lenguaje (ELMo (Peters, Ammar, Bhagavatula, y Power, 2017), GPT-3 (T. B. Brown y cols., 2020), BERT) han sido desarrollados gracias a arquitecturas de DL entrenadas sobre conjuntos de datos extensos. En tareas específicas como análisis de sentimientos, traducción automática, generación de lenguaje y clasificación de textos también son la solución más habitual.

Una de las principales desventajas de los modelos basados en DL es que necesitan por lo general un gran volumen de información para descubrir patrones que sean generalizables (Qiu y cols., 2020), de aquí que la precisión de los modelos se encuentre en función de la cantidad de datos disponibles para el entrenamiento (Ghosh y cols., 2019; Vrbančič y Podgorelec, 2020). Por ejemplo, en clasificación de imágenes y textos para que una solución sea exitosa son necesarios conjuntos de datos de alta cardinalidad con elevada consistencia. La obtención de conjuntos de datos con estas características introduce una complejidad práctica especialmente en cuanto a esfuerzos de anotación, especialmente si se presentan relaciones semánticas o sintácticas (Cer y cols., 2018). A ello se suma que no todos los investigadores y sus colectivos pueden tener acceso a dispositivos de hardware con altas prestaciones para entrenar modelos de DL desde cero con conjuntos de datos suficientemente potentes (Pouyanfar y cols., 2018). Una de las soluciones actuales a este problema es el uso de aprendizaje por transferencia (Vrbančič y Podgorelec, 2020).

2.6.1 Modelos de aprendizaje por transferencia

El aprendizaje por transferencia es una de las técnicas actuales más importantes dentro del DL (Vrbancic y cols., 2019). Se define como transferir el conocimiento de un modelo previamente entrenado en una tarea general, hacia tareas específicas mediante el uso de un conjunto de datos de menor cardinalidad, con lo que se intenta evitar el esfuerzo de iniciar el aprendizaje desde cero (Pan y Yang, 2010).

Al evitar entrenar una red neuronal desde cero, que implicaría precisar de una gran cantidad de datos y requerir mucho tiempo para ello, el proceso se realiza desde una red pre-entrenada (Qiu y cols., 2020). Esta técnica permite descargar un modelo de código abierto que ha sido pre-entrenado en un conjunto de datos de gran cardinalidad y utilizar sus parámetros (miles o millones) como punto de partida para continuar entrenando el modelo con un conjunto de datos de menor cardinalidad y específico para una tarea determinada.

Estos modelos se pueden utilizar en dos modos: extracción de características y ajuste fino (fine-tuning) (Vrbancic y cols., 2019):

- **Extracción de características**: En este modo los pesos del modelo pre-

entrenado son congelados y se extraen características de nuevos datos para entrenar desde cero un nuevo modelo (Peters, Ruder, y Smith, 2019).

- **Ajuste fino (fine-tuning)**: Se utiliza un modelo con un conjunto de pesos previamente entrenados para reentrenarlo en otro conjuntos de datos, los pesos de las capas de las red son reajustados (Tajbakhsh y cols., 2016).

En el campo del procesamiento de imágenes, los modelos basados en aprendizaje por transferencia (Krizhevsky, Sutskever, y Hinton, 2012; Simonyan y Zisserman, 2014), generalmente son entrenados haciendo uso de aprendizaje supervisado en conjunto de datos extensos y etiquetados (Raffel y cols., 2019), como ImageNet (Deng y cols., 2009). Sin embargo, los modelos de aprendizaje por transferencia en PLN a menudo se pre-entrenan utilizando el aprendizaje no supervisado en conjuntos de datos sin etiquetar (Houlsby y cols., 2019; Qiu y cols., 2020). Algunos de estos modelos (Devlin y cols., 2019; Y. Liu, Ott, y cols., 2019; Canete y cols., 2020), principalmente los basados en arquitectura Transformer (Vaswani y cols., 2017), están siendo utilizados para crear soluciones que actualmente reportan los mejores resultados (Qiu y cols., 2020).

En este trabajo en su mayoría se utilizan modelos de lenguajes basados en aprendizaje por transferencia, a los que se les aplica un ajuste fino para adaptarlos a las tareas específicas abordadas. Los modelos empleados son RoBERTa y BETO.

Tanto RoBERTa como BETO están basados en la arquitectura de BERT, el cual está diseñado para entrenar representaciones bidireccionales profundas a partir de texto sin etiquetar. BERT es un modelo de lenguaje basado en la arquitectura Transformer (Vaswani y cols., 2017), que es una arquitectura con estructura codificador-decodificador. Esta arquitectura es una red neuronal basada únicamente en mecanismos de atención, prescindiendo por completo de redes tipo RNN y CNN. Los detalles de la arquitectura Transformer se pueden consultar en la descripción de la misma (Vaswani y cols., 2017).

Antes de continuar con la explicación de los modelos basados en esta arquitectura se definirá el componente principal de esta, el mecanismo de autoatención. El concepto de atención en modelo DL fue introducido por (Bahdanau, Cho, y Bengio, 2014). Este consiste en un método simple que puede ser usado para codificar datos de una secuencia en función de la puntuación de importancia que se le asigna a cada elemento (Hu, 2020). En (Bahdanau y cols., 2014) fue utilizado en una arquitectura de traducción de tipo codificador-decodificador con el fin de seleccionar la información más importante de la secuencia. Por su parte, la autoatención es un tipo de atención altamente extendida en modelos actuales (Devlin y cols., 2019; Y. Liu, Ott, y cols., 2019; He, Liu, Gao, y Chen, 2020). Esta consiste en que cada *token* de una secuencia se relaciona con los restantes, calculándose una puntuación de atención, esta relación condiciona fuertemente el significado de los *tokens* en las secuencias (Vaswani y cols., 2017).

BERT por su parte utiliza casi íntegramente la parte de los codificadores de la arquitectura *Transformer*, utilizando el mecanismo de atención para obser-

var los conjuntos de *tokens* del contexto izquierdo y derecho en todas las capas (Devlin y cols., 2019). Este pre-entrena el modelo mediante dos tareas no supervisadas: modelado de lenguaje mediante máscaras y predicción de la próxima oración. En el proceso de pre-entrenamiento se utiliza como corpus el BooksCorpus (800 M palabras) (Y. Zhu y cols., 2015) y la enciclopedia libre Wikipedia en inglés (2500 M palabras).

El modelo RoBERTa por su parte, propone una serie de modificaciones pero sin alterar la arquitectura original. Las modificaciones incluyen eliminar la predicción de la siguiente oración, realizar el entrenamiento en un mayor volumen de datos, ampliar el tamaño del lote y alargar la secuencia de entrada.

En el proceso de entrenamiento de RoBERTa se utiliza el mismo corpus utilizado en BERT además de tres nuevos conjuntos de datos, llegando a entrenar con 160 GB de texto. Este modelo logra mejorar los resultados con respecto BERT en las principales tareas de PLN (Y. Liu, Ott, y cols., 2019).

Tanto el modelo BERT como RoBERTa tienen dos versiones: la base y la extendida; estas se diferencian principalmente en la cantidad de capas, 12 y 24, con 12 y 16 cabezas de atención y 768 y 1024 de tamaño oculto respectivamente (Devlin y cols., 2019; Y. Liu, Ott, y cols., 2019).

Por su parte, BETO ha sido entrenado con textos en español provenientes de la Wikipedia y del proyecto OPUS[9]. Este modelo realiza una serie de optimizaciones (similares al modelo RoBERTa) pero usando la arquitectura de BERT. El modelo BETO tiene 12 capas de autoatención con 16 cabezas de atención cada una, usando 1024 como tamaño oculto (Canete y cols., 2020).

Los artículos que describen estos modelos reportan resultados notables en los principales *benchmarks* en PLN como: *General Language Understanding Evaluation* (GLUE), *Reading Comprehension Dataset From Examinations* (RACE) y *Stanford Question Answering Dataset* (SQuAD). Estos modelos pre-entrenados tienen la ventaja de poder ajustarse con solo una capa adicional de salida, una característica que les permite ser utilizados para crear modelos de vanguardia en varias tareas de PLN (Dodge y cols., 2020). Son numerosas las investigaciones que utilizan estos modelos como base para solucionar problemas específicos en PLN.

En conclusión, los modelos basado en BERT brindan una flexibilidad que permite crear sistemas competitivos con solo realizar un ajuste fino del modelo y sus hiperparámetros usando el conjunto de datos de la tarea específica (Devlin y cols., 2019).

La bibliografía reporta algunos problemas que pueden presentarse en la utilización del aprendizaje por transferencia. El principal problema que se presenta es el rápido sobreajuste debido principalmente al tamaño elevado que suelen tener estas redes neuronales y los bajos volúmenes de datos que suelen contener

[9]Corpus en paralelo con más de 90 idiomas, el par español-inglés es el más representativo con 36 millones de oraciones (Tiedemann, 2012). `https://opus.nlpl.eu/` (consultado el 15 de enero de 2022).

los corpus en PLN empleados para realizar el ajuste fino (Qiu y cols., 2020).

Específicamente, en modelos basados en arquitectura Transformer, se ha encontrado que el proceso de ajuste fino es a menudo frágil dado que la utilización de los mismos valores de hiperparámetros con distintas semillas aleatorias pueden conducir a resultados sustancialmente diferentes (Dodge y cols., 2020). Ello parece contradecir el principio de convergencia a un espacio de soluciones común.

Los modelos basados en aprendizaje por transferencia y los algoritmos de DL tradicionales comparten entre sí los problemas asociados a la explicabilidad de las decisiones tomadas por los modelos y la complejidad del proceso del ajuste de hiperparámetros (Kowsari y cols., 2019).

Para el problema de la explicabilidad, uno de los atributos iniciales autoexigidos para la Inteligencia Artificial (Winston y Prendergast, 1984), se están haciendo avances significativos que contribuyen a mejorar la interpretabilidad de los modelos (Kotonya y Toni, 2020; Lu y Li, 2020). Las soluciones al problema de ajustes de parámetros se tienden a abordar de la forma tradicional en que se resuelve en el resto de los algoritmos basados en DL (Koch y cols., 2018).

2.6.2 Procesamiento de textos largos versus DL

Las soluciones basadas en DL suelen presentar problemas para procesar largas secuencias de textos. Por ejemplo, las RNNs tienen un alto rendimiento procesando textos largos cuando se utiliza una arquitectura de codificador-decodificador, típico de tareas como la traducción automática, sin embargo, en tareas de clasificación de textos el rendimiento disminuye grandemente y puede aparecer el problema del *vanishing gradient*. Este fenómeno impide que se pueda hacer la propagación hacía atrás (*backpropagation*) del error en el entrenamiento, por la cantidad de variación no lineales introducidas, resultado en que la red neuronal no logra aprender (A. Hassan y Mahmood, 2017). Además, su naturaleza secuencial les impide paralelizarse, por lo que el tiempo de ejecución se eleva considerablemente al aumentar la secuencia de entrada (Cai, Li, Li, y Wang, 2018). Algunas soluciones posibles a este problema son truncar la secuencia de entrada a un tamaño de secuencia estable, resumir la secuencia o usar arquitectura de codificación-decodificación para clasificación de texto, lo que puede resultar en una red neuronal bastante compleja.

Por su parte, los modelos neuronales basados en arquitectura Transformer teóricamente no tienen una limitación de tamaño en la entrada que pueden procesar. Esta arquitectura tiene un mecanismo de atención que se traduce en que cada *token* puede atender al resto de *tokens* de entrada, creando una relación que tiene complejidad de tiempo y memoria $O(n^2)$, donde n es el tamaño máximo de secuencia de entrada (Vaswani y cols., 2017). Esta complejidad computacional del mecanismo de atención restringe en la practica su uso cuando el tamaño de secuencia de entrada es considerable (Tay, Dehghani, Bahri, y Metzler, 2020). Soluciones novedosas para procesar textos largos con arquitectura

Transformer han sido propuestas (Beltagy, Peters, y Cohan, 2020; Kitaev, Kaiser, y Levskaya, 2020; Ainslie y cols., 2020).

La mayoría de los modelos que utilizan la arquitectura Transformer limitan experimentalmente el tamaño máximo de secuencia de entrada a 512 *tokens*. En (Beltagy y cols., 2020) se propone un modelo (Longformer) que modifica el mecanismo de atención de la arquitectura Transformer extrayendo información local y global del contexto. En este caso cada *token* atiende a los w *tokens* que tiene a su derecha e izquierda, siendo w mucho menor que la secuencia de entrada n. La complejidad algorítmica se reduce de $O(n^2)$ a $O(n \cdot w)$ donde w es el tamaño de la ventana de atención en cada capa. Este modelo mejora los resultados en varios conjuntos de datos con secuencias largas.

Reformer (Kitaev y cols., 2020) no modifica el mecanismo de atención pero introduce una medida de similitud basada en *hash* para agrupar fragmentos de *tokens* de manera eficiente. Además, utiliza capas reversas de Transformer, con estas modificaciones la complejidad algorítmica baja hasta $O(n \log n)$.

Estos modelos han sido utilizado en diversos trabajos (Roush y Balaji, 2020). El modelo Longformer ha obtenido los mejores resultados entre otros modelos basados en arquitectura Transformer en una tarea de generación de resúmenes extractivos utilizando un conjunto de datos de minería de argumentos y resúmenes (Roush y Balaji, 2020).

La utilización de estos modelos permite abordar tareas que precisen procesar largas cadenas de texto sin necesidad de truncar automáticamente secuencias de entrada, como hace RoBERTa, BERT y BETO.

2.7 Conclusiones

La detección de titulares engañosos habitualmente es abordada desde la óptica de la detección de posturas. El análisis bibliográfico demuestra una marcada relación con el dominio de aplicación de la detección de posturas, en cuanto a los principales recursos y enfoques que se presentan.

En el caso de detección de posturas en tuits se suelen usar técnicas de preprocesamiento y extracción de características típicas de estos mensajes. Por su parte, en la detección de posturas para la detección de titulares engañosos en noticias es necesario procesar el contenido completo de la noticia, lo que conlleva retos adicionales.

La tipología de las soluciones relacionadas con detección de posturas y de titulares engañosos exhiben similitudes a otras tareas dentro del PLN. Se observa, que las soluciones basadas en ML son superadas por las que utilizan DL.

Una interpretación de los conceptos y soluciones utilizadas en investigaciones previas permite demostrar la variedad de tratamientos conceptuales al problema de la detección de titulares engañosos y brinda una ubicación al enfoque seguido por la investigación desarrollada.

Las investigaciones consultadas en detección de titulares engañosos y de-

tección de contradicciones permiten constatar, que a pesar de que el español es una de las lenguas más habladas en el mundo, no existen recursos para llevar a cabo estas tareas desde la perspectiva directa de esta lengua. El conjunto XNLI, que contiene ejemplos en idioma español en sus particiones de desarrollo y prueba, anota la relación semántica entre dos parejas de oraciones. Aunque conceptualmente sea necesario identificar la relación semántica, en la práctica la tarea de detectar relaciones semánticas entre titular y contenido de noticia presenta retos adicionales (diferencias de longitud entre titular y contenido, entender el discurso como una sucesión de oraciones coherentes y utilizar técnicas apropiadas para abordar estos aspectos).

Sería conveniente contar con un conjunto de datos en español que permita la creación de sistemas específicos para detectar contradicciones con el fin de determinar qué titulares pueden ser engañosos en esta lengua. Además, los conjuntos de datos existentes en la tarea de detección de contradicciones no discriminan entre los diferentes tipos de contradicciones, factor que podría ser determinante para el éxito de la tarea.

Teniendo en cuenta el análisis realizado acerca de los problemas de clasificación de múltiples clases, una posible solución podría ser el uso de clasificadores jerárquicos. Este tipo de clasificador es recomendable cuando existe una relación jerárquica predefinida entre las etiquetas a clasificar, pero también es útil cuando esta relación es creada en función de la similitud entre etiquetas. Por último, existen trabajos que experimentan mejoras sustanciales utilizando este enfoque en conjuntos de datos con un alto desbalance entre etiquetas.

Los modelos neuronales representan la principal herramienta de solución de problemas de clasificación en el ámbito del PLN. Técnicas actuales como el aprendizaje por transferencia han permitido la democratización de estas soluciones debido a que en muchos casos ya no es necesario entrenar un modelo desde cero, dado que es posible transferir el conocimiento de tareas generales a dominios específicos, reutilizando entrenamientos previos realizados por terceros.

Como se ha señalado, los modelos neuronales, específicamente los basados en DL, presentan problemas para procesar textos extensos. Sin embargo, se han propuesto soluciones que son capaces de procesar eficientemente esas entradas, reduciendo la complejidad del algoritmo y el uso de recursos computacionales. Otra opción consiste en crear arquitecturas complejas que permitan fraccionar el texto y procesarlo por lotes. En este trabajo, se evalúa la utilización de resúmenes automáticos como vía de reducir el tamaño de la entrada al modelo de DL sin pérdida de información relevante.

CAPÍTULO 3

Análisis de la aplicación de resúmenes a la detección de posturas en titulares

3.1 Introducción

La detección de posturas es una tarea ampliamente abordada en la comunidad científica. Como se mostró en las secciones 2.2 y 2.3 esta tarea además está siendo utilizada para detectar la relación semántica entre un titular y el contenido de una noticia. Retomando el problema de la sobrecarga de información a la que estamos sometidos hoy en día y la imposibilidad de evaluar manualmente la veracidad de las noticias en un tiempo razonable, el uso de resúmenes automáticos podría ser un recurso de interés para sistemas que pretenden detectar automáticamente titulares engañosos. El uso de resúmenes podría evitar tener que diseñar arquitecturas complejas como la utilizada en (Yoon y cols., 2018) para evadir la limitación de procesar textos largos de modelos computacionales del estado del arte en PLN.

En este capítulo, se presenta un estudio exploratorio mediante el cual se analiza las potencialidades y limitaciones de las técnicas actuales de resumen de textos para abordar el problema de la desinformación, específicamente, la detección de posturas entre titulares y contenidos de noticias. En particular, se seleccionaron métodos de resumen automático de última generación: extractivos, abstractivos e híbridos. La selección se realizó asegurando que los métodos propuestos emplearan diferentes estrategias, como técnicas basadas en grafos, discursivas, estadísticas y basadas en DL. Esta variedad de métodos permitirá comparar el rendimiento y la posible pertinencia de cada uno. Se seleccionaron algunos métodos y estrategias de resumen en base a la investigación realizada en (R. Ferreira y cols., 2013), donde se determinaron cuáles son las mejores téc-

nicas de resumen en términos de eficacia y eficiencia en el dominio periodístico.

El objetivo principal de este capítulo es evaluar si los resúmenes son lo suficientemente efectivos para reducir la información de entrada a los hechos más importantes y al mismo tiempo no perder precisión para verificar la idoneidad de una afirmación realizada en un titular. Este objetivo se intenta cubrir con la respuesta a las siguientes preguntas de investigación:

1. ¿Se pueden utilizar los resúmenes en lugar del contenido completo para la tarea de detección titulares engañosos?

2. ¿Cuál es el mejor método de generación de resúmenes para la tarea de detección de titulares engañosos aplicando detección de posturas?

3. ¿Cuál es la longitud más apropiada para un resumen en este contexto?

3.2 Definición y contexto

Un resumen de texto se puede definir como un texto que se produce a partir de uno o más textos, que contiene una parte significativa de la información en el texto original y que no supera la mitad del texto original (Tas y Kiyani, 2007). El ser humano para resumir un texto, por lo general lo lee por completo para comprenderlo y luego escribe un resumen con los aspectos principales (Allahyari y cols., 2017). Sin embargo, realizar esta tarea de forma automática resulta realmente desafiante y para nada trivial (Gambhir y Gupta, 2017), dado que es necesario diseñar algoritmos computacionales con capacidad de generar resúmenes de textos y de adquirir conocimientos lingüísticos (Allahyari y cols., 2017).

Investigaciones previas en resúmenes de textos han demostrado el impacto positivo que pueden tener en la sociedad, debido a que el uso de resúmenes ha sido muy beneficioso en áreas tales como la educación (debido a su uso como apoyo a las tareas de comprensión de la lectura (S. A. Brown, 2018; Engelen, Camp, van de Pol, y de Bruin, 2018; Xu, Dong, y Jhang, 2018; Barreiro, 2019)), en los negocios, (para producir, por ejemplo, un resumen automático de una bitácora de eventos (*logs*) con el objeto de apoyar a los analistas (Dijkman y Wilbik, 2017)) o en la salud (con independencia de la forma en que fueron creados los resúmenes, ya sea manual (Hartling, Gates, Pillay, Nuspl, y Newton, 2018) o automatizada (Y. Liu, Song, y Chen, 2019)). Esto en parte se debe a la capacidad de los métodos de resumen de identificar la información más relevante de un documento y condensarlo en un nuevo documento, con lo que ayudan a reducir tiempo, espacio y otros recursos cuando se trata de manejar grandes volúmenes de información.

Los resúmenes automáticos han probado su efectividad en un amplio rango de tareas de PLN, donde se suelen usar como componentes de sistemas más complejos. Esto incluye por ejemplo, clasificación de textos (Saggion, Lloret, y Palomar, 2012; Tsarev, Petrovskiy, y Mashechkin, 2013; Jeong, Ko, y Seo, 2016),

question answering (QA) (Lloret, Llorens, Moreda, Saquete, y Palomar, 2011) y IR (Perea-Ortega, Lloret, Ureña-López, y Palomar, 2013; Raposo, Ribeiro, y Martins de Matos, 2016).

Se han utilizado diferentes técnicas y enfoques para crear resúmenes automáticos que son usados como sustitutos del texto original. De esta manera, las tareas subyacentes, por ejemplo, clasificación de texto o QA, emplean directamente los resúmenes generados, lo suficientemente cortos e informativos, para cumplir con sus objetivos, mientras que al mismo tiempo optimizan los recursos requeridos para estas tareas.

El campo del periodismo, específicamente el dominio de las noticias, ha sido una de las áreas más representativas en las que el resumen se ha centrado tradicionalmente desde el principio, en parte gracias al desarrollo de corpus apropiados (por ejemplo, DUC, Gigaword, CNN/DailyMail) (Dernoncourt, Ghassemi, y Chang, 2018) y la amplia gama de técnicas y enfoques para ayudar la compresión de este tipo de información (Nenkova, 2005; Mackie, McCreadie, Macdonald, y Ounis, 2016; Duan y Jatowt, 2019; C. Zhu, Yang, Gmyr, Zeng, y Huang, 2019). Además de los diversos tipos de resúmenes que se han desarrollado para este dominio (documento único, multi-documento, extractivo, abstractivo, genérico, orientado a temas, etc.), existe una cantidad significativa de investigaciones sobre la tarea de generación de titulares utilizando técnicas de resúmenes (Banko, Mittal, y Witbrock, 2000; Dorr, Zajic, y Schwartz, 2003; Zajic, Dorr, y Schwartz, 2002) y, más recientemente, usando DL (Tan, Wan, y Xiao, 2017; Gavrilov, Kalaidin, y Malykh, 2019; Iwama y Kano, 2019). Sin embargo, aunque existen trabajos que usan resúmenes para determinar si una noticia es real o no (Esmaeilzadeh y cols., 2019; ShimJae-Seung y cols., 2019), rara vez se han considerado en el campo de la detección de información engañosa (ShimJae-Seung y cols., 2019).

3.3 Enfoques de detección de posturas

El análisis de si una afirmación hecha en un titular corresponde al contenido de la noticia, se aborda como una tarea de detección de posturas. En este capítulo se utilizan enfoques tradicionales de ML y DL[1] para medir la validez de aplicar resúmenes al problema de detección de titulares engañosos. Se utilizan estos dos enfoques (uno desarrollado para la tarea de detección de posturas y otro que ha sido adaptado a ella) con lo que se realiza una serie de experimentos sobre dos conjuntos de datos y algunos métodos empleados en la obtención de resúmenes.

[1] DL es un tipo específico de ML. Se utiliza la nomenclatura DL para indicar la diferencia entre los enfoques sin DL y los que tienen.

3.3.1 Enfoque de aprendizaje automático para detección de posturas

El enfoque de ML utilizado ha sido desarrollado por (W. Ferreira y Vlachos, 2016)[2] para abordar el problema de la detección de posturas en el conjunto de datos Emergent (ver sección 3.5.1). Estos autores crean un clasificador basado en regresión logística como modelo de ML, con regularización L1 (Pedregosa y cols., 2011) para clasificar las etiquetas del conjunto de datos. Como entrada este enfoque recibe la afirmación o titular y el resumen del contenido del artículo de noticias, los cuales son preprocesados, extrayendo características de su relación, así como de estos independientemente. A continuación, se muestran las características extraídas, las tres primeras solo se extraen de la afirmación, el resto de la relación entre la afirmación y el resumen del contenido del artículo:

- *Presencia de signos de preguntas*: se identifica la presencia de signos de preguntas en cada afirmación.
- *Distancia mínima entre la raíz y palabras de refutación y reporte*: Se anotan automáticamente palabras que indiquen refutación y reporte. A continuación se calcula la distancia mínima entre esas palabras y el elemento anotado como raíz de la oración.
- *Representación de bolsa de palabras (bag of words)*: Se extrae esta representación de palabras.
- *Similitud de coseno con vectores word embedding*: Esta característica mide cuan similares son dos textos, independientemente de su tamaño. Matemáticamente, representa el coseno del ángulo entre dos vectores proyectados en un espacio multidimensional (Passalis y Tefas, 2018). El tamaño de los textos no afecta demasiado a la precisión de esta medida como si ocurre con la distancia euclidiana. Cuanto menor sea el ángulo, mayor será la similitud de coseno (B. Li y Han, 2013). Aunque esta medida de similitud es relativamente básica (Tata y Patel, 2007), generalmente aporta mejoras significativas a los modelos de recuperación de información (Passalis y Tefas, 2018). La medida de similitud del coseno (SC) entre dos vectores X y Y se obtiene siguiendo la ecuación 3.1 (Kotu y Deshpande, 2019):

$$SC(|X,Y|) = \frac{X \cdot Y}{\|X\| \, \|Y\|} \tag{3.1}$$

donde:

$$X \cdot Y = \sum_{i=1}^{n} X_i Y_i \; ; \; \|X\| = \sqrt{X \cdot X} \; ; \; \|Y\| = \sqrt{Y \cdot Y}$$

En esta implementación el cálculo de la similitud de coseno se realiza con vectores word2vec (Mikolov y cols., 2013) de la afirmación y el resumen del contenido de la noticia.

- *Alineación de paráfrasis y negación*: Similar al trabajo desarrollado por (Rus y Lintean, 2012) se calcula la alineación entre los elementos utilizando la base de datos de alineamiento entre frases (*Paraphrase Database* (PPDB)) y el algoritmo desarrollado por (Kuhn, 1955). Para más detalles puede consultar la utilización de esta característica en (W. Ferreira y Vlachos, 2016).

- *Similitud entre tripletas sujeto-verbo-objeto*: Se extraen tripletas del tipo sujeto-verbo-objeto, posteriormente se calcula la similitud entre las tripletas extraídas.

La implementación de estas características se puede consultar en el enlace de Github[2] del proyecto.

3.3.2 Enfoque de aprendizaje profundo para detección de posturas

Para el enfoque de aprendizaje profundo, el clasificador de titulares engaños que se utiliza se inspiró en *Enhanced Sequential Inference Model* (ESIM) (Q. Chen y cols., 2016). Este modelo, comúnmente es utilizado en la tarea de reconocimiento de vinculación textual, pero ha sido adaptado en esta investigación a la tarea de detección de posturas. Para usar este modelo, se realizó una implementación basada en el sistema propuesto por (Hanselowski, Zhang, y cols., 2018). Este sistema[3], obtuvo resultados competitivos en la tarea compartida FEVER (Thorne, Vlachos, Christodoulopoulos, y Mittal, 2018).

Las entradas al modelo de clasificación son un par: i) afirmación o titular de un artículo de noticias y ii) el resumen obtenido del contenido del artículo utilizando los métodos explicados en la sección previa. La primera capa del modelo de DL es una capa de *embedding* para representar las palabras como un vector de *word embedding*. Se experimentó usando diferentes modelos de *word embedding*. Los mejores resultados se obtienen cuando se concatenan dos de ellos, en este caso FastText (Bojanowski y cols., 2017) y GloVe (Pennington y cols., 2014)(las dimensiones de los vectores *word embedding* son de 300, luego de la concatenación quedarían en 600 dimensiones). A continuación de la capa de entrada de *embedding* se coloca el modelo ESIM. Este utiliza capas bidireccionales LSTM que son las encargadas junto a un mecanismo de atención de obtener una inferencia del procesamiento del titular y del resumen. Una vez que la inferencia se realiza, este vector de salida se pasa a través de una capa de reducción de dimensiones, en este caso se usa agrupación máxima y media, y finalmente a una red neuronal de tipo MLP para realizar la clasificación. Los detalles de implementación pueden ser consultados en (Hanselowski, Zhang, y cols., 2018).

[2]Disponible en `https://github.com/willferreira/mscproject` (consultado el 10 de enero de 2022).

[3]Disponible en `https://github.com/rsepulveda911112/DL_Stance_Detection` (consultado el 10 de enero 2022).

3.4 Tipos de resúmenes automáticos

Las técnicas de generación automática de resúmenes de texto se pueden dividir en extractivas y abstractivas (Lloret y Palomar, 2012). Los enfoques extractivos se centran en detectar la información más relevante en un texto, que luego se copia y pega literalmente en el resumen final. Por el contrario, los enfoques abstractivos intentan replicar la técnica de resúmenes de los humanos, donde estos detectan la información relevante y luego se realiza un proceso más elaborado mediante el cual la información se fusiona, comprime o incluso se infiere. Además de estos dos, el desarrollo de enfoques híbridos, que combinan métodos extractivos y abstractivos, también es posible (Kirmani, Manzoor Hakak, Mohd, y Mohd, 2019). En este capítulo se experimenta con un subconjunto de técnicas de resumen que se detallan a continuación.

3.4.1 Tipos de resúmenes extractivos

Se eligieron tres técnicas de generación de resúmenes extractivos, una que comúnmente es utilizada como *baseline*, la segunda que es una técnica basada en grafos y la tercera es un enfoque basado en estadística.

Lead Summarizer: Este método trunca un texto por una cantidad determinada de oraciones. Siendo un método básico, en el ámbito de las noticias puede ser muy competitivo (Widyassari y cols., 2019) debido a que la información más relevante de estas suele encontrarse al inicio del contenido, dejando los detalles menos importantes para el final (Pöttker, 2003; Lajusticia, 2000).

TextRank Summarizer: TextRank (Mihalcea y Tarau, 2004) es un modelo de *ranking* basado en grafos para el procesamiento de texto. En esta investigación, el resumen del texto se extrae a partir de las oraciones más relevantes. Este método construye un grafo asociado a partir de un texto, siendo los nodos del grafo representativos de las oraciones a seleccionar. Se calcula un peso para cada una de las aristas del grafo que indica la fuerza de la conexión entre los pares de oraciones/nodos vinculados por ellas. Estas conexiones están determinadas por la similitud entre las oraciones de texto medidas por la superposición de su contenido. Una vez que se construye el grafo, se realiza un *ranking* ponderado para asignar una puntuación a cada una de las oraciones del texto. Luego, las oraciones se ordenan en orden inverso a su puntuación. Finalmente, se seleccionan las oraciones mejor clasificadas del ranking para incluirlas en el resumen final. Para obtener el resumen con este enfoque se utilizó la biblioteca Sumy[4].

PLM Summarizer: Este método de resumen se aprovecha del potencial de los *Positional Language Model*s (PLMs). Estos son modelos de lenguaje que captu-

[4] `https://pypi.org/project/sumy/` (consultado el 6 de octubre de 2021).

ran de manera eficiente las dependencias de términos no adyacentes, para ubicar la información relevante dentro de un documento (Vicente y Lloret, 2020). La idea básica detrás de los PLMs es que para cada posición dentro de un documento, es posible calcular una puntuación para cada palabra del vocabulario. Este valor representa la relevancia de dicho término en esa posición precisa, tomando en cuenta la distancia a otras apariciones de la misma palabra a lo largo del documento, de modo que cuanto más cerca de esa posición aparezcan los términos, mayor es la puntuación obtenida. Por lo tanto, el modelo expresa el significado de los elementos considerando el texto completo como su contexto, en lugar de limitarse al alcance de una sola oración. Estos modelos han sido usados para crear resúmenes extractivos con éxito en (Vicente Moreno, 2021). En esta investigación los incluimos para comparar con otros enfoques de referencia.

Para la realización de este resumen se puede seleccionar un conjunto de palabras (semilla de inicialización) que pueden ser representativas para el texto y ayudan al sistema a descartar partes irrelevantes del discurso (se utilizan palabras del titular). Las puntuaciones asignadas a las palabras contribuyen a una puntuación integrada de cada oración, el resumen será creado seleccionando las oraciones mejor puntuadas hasta la cantidad de oraciones requeridas por el usuario.

3.4.2 Tipos de resúmenes abstractivos

Con respecto a los métodos de resumen abstractivo, se seleccionó un método que se apoya en técnicas de generación de lenguaje natural, que ha demostrado funcionar eficazmente para la generación de este tipo de resumen (Barros, Lloret, Saquete, y Navarro-Colorado, 2019). Específicamente, se utiliza una adaptación del enfoque HanaNLG (Barros y Lloret, 2019), un enfoque de generación de lenguaje natural que se adapta fácilmente a diferentes dominios y tareas. La generación de lenguaje en este enfoque se realiza siguiendo una estrategia de sobregeneración y *ranking* que se basa en el uso de *Factored Language Model*s (FLMs) (entrenando sobre documentos que se procesaron previamente con la herramienta Freeling (Padró y Stanilovsky, 2012) para anotar factores) (Bilmes y Kirchhoff, 2003), recursos lingüísticos (WordNet (Fellbaum, 1998) y VerbNet (Schuler, 2005)) y un conjunto de semillas (fonemas, sentimientos, polaridades, etc.) que guiará el proceso de generación en términos de vocabulario.

Para adaptar HanaNLG con el fin de generar resúmenes abstractivos (de ahora en adelante, **HanaNLG Summarizer**), las palabras clave del titular se utilizan como semilla inicial. Esta semilla permite la generación de oraciones relacionadas con una palabra clave o tema específico. Estos temas deben ser relevantes, por lo que es necesario establecer un método para determinar los temas relevantes a partir de los cuales se va a generar la oración. Esto permitirá la generación de resúmenes basados en la información esencial de los documentos fuente. En este caso, las entidades nombradas se utilizan como palabras clave para la ge-

neración de resúmenes, ya que representan información relevante relacionada con los nombres de personas, ubicaciones, organizaciones, etc., de un artículo de noticias.

3.4.3 Resumen híbrido

Fast Abstractive Summarizer fue seleccionado como el método de resumen híbrido, que explota los beneficios de los paradigmas extractivos y abstractivos (Y.-C. Chen y Bansal, 2018). Este método propone una estrategia de dos pasos que no necesita preprocesamiento de documentos. En primer lugar, se seleccionan las oraciones destacadas y, en segundo lugar, se reescriben para generar el resumen final. El proceso de reescritura consiste en comprimir o parafrasear las frases elegidas inicialmente como relevantes. Para ambas etapas, se utilizan modelos neuronales. Para la detección de información destacada, se utiliza una red neuronal llamada *pointer networks*[5], entrenada haciendo uso de Aprendizaje Reforzado (RL) (Bello y cols., 2016). Luego, para el módulo de reescritura, se emplea una arquitectura de codificador-decodificador[6] simple al que se agrega un mecanismo de copia para ayudar a copiar directamente algunas palabras que se encuentren fuera del vocabulario[7].

3.5 Entorno de evaluación

En esta sección se describen dos conjuntos de datos que fueron desarrollados para la tarea de detección de posturas entre titulares y contenido de noticias. El uso de estos conjuntos de datos es ampliamente extendido en la comunidad científica, por lo que se ha decidido usarlos para validar si los resúmenes son suficientemente efectivos sobre esta tarea. Los conjuntos de datos son: Emergent y Fake News Challenge (FNC-1). En el caso del conjunto de datos Emergent se cuenta con titular, afirmación y el contenido de la noticia; de los cuales se anota la relación de postura entre la afirmación y el par (titular/contenido de la noticia). Por otro lado el conjunto de datos FNC-1 no cuenta con la afirmación por lo que la relación es anotada entre el titular y el contenido de la noticia.

En estos conjuntos de datos se crean los ejemplos de entrenamiento y prueba, relacionando o asignando una afirmación en el caso del corpus Emergent y un titular para FNC-1 con varios contenidos de noticias. Una información más detallada acerca de estos conjuntos de datos, así como el alineamiento de las etiquetas entre ambos, se ofrece más adelante.

[5] Permite que el modelo señale efectivamente una posición específica en la secuencia de entrada en lugar de predecir un valor de índice a partir de un vocabulario de tamaño fijo (Bello, Pham, Le, Norouzi, y Bengio, 2016).

[6] Arquitectura típica para traducción y generación de textos (Bello y cols., 2016).

[7] Implementación disponible en `https://github.com/ChenRocks/fast_abs_rl` (consultado el 6 de octubre de 2021).

3.5.1 Conjunto de datos Emergent

Emergent es un conjunto de datos[8] de refutación de rumores creado bajo el proyecto del mismo nombre (Silverman, 2015). Este contiene 300 afirmaciones y 2571 contenidos de noticias, además del titular real de cada noticia (W. Ferreira y Vlachos, 2016), ver tabla 3.1. Estos elementos fueron elegidos y anotados por periodistas con una de las siguientes etiquetas:

- *for*: si la afirmación apoya el texto del artículo.
- *against*: si la afirmación está en contra del texto del artículo.
- *observing*: cuando la afirmación se reporta en el artículo pero no se puede concluir una relación de las anteriores.

Tabla 3.1: Descripción de los subconjuntos de Emergent, considerando números de documentos, titulares y afirmaciones.

Particiones	**Contenidos de noticias**	**Titulares**	**Afirmaciones**	**Total de asignaciones**
Entrenamiento	2048	2023	240	2071
Prueba	523	513	60	524
Total	2571	2536	300	2595

Resulta importante explicar que en este conjunto de datos interviene el concepto de afirmación (*claim*) típico de tareas de verificación automática de hechos. Este corpus se ha venido usando para encontrar la relación semántica entre una afirmación y el contenido de la noticia, no la relación entre el titular real y el contenido de la noticia como es habitual en tareas de detección de titulares engañosos y detección de posturas sobre noticias. Sin embargo, en (Q. Zhang y cols., 2019) se utiliza para esta tarea con el fin de evaluar la generalidad del enfoque desarrollado.

En la tabla 3.2 se aprecia la distribución de elementos en cada etiqueta. Los porcentajes de las etiquetas son similares en cada conjunto y en el corpus completo.

3.5.2 Conjunto de datos Fake News Challenge (FNC-1)

Este conjunto de datos (conocido por las siglas FNC-1)[9] fue desarrollado en el contexto del Fake News Challenge (Babakar y cols., 2016). Sus instancias fueron anotadas como:

[8] Disponible en `https://github.com/willferreira/mscproject` (consultado el 6 de octubre de 2021).

[9] Disponible en `http://www.fakenewschallenge.org/` (consultado el 6 de octubre de 2021).

- *agree*: cuando el titular y el contenido de la noticia se corresponden.
- *disagree*: cuando el titular se contradice con el contenido.
- *discuss*: cuando en el contenido se discute el mismo tema que en el titular, pero no toma postura alguna.
- *unrelated*: cuando el cuerpo del texto no se relaciona con el titular, usualmente comparten entidades nombradas con el fin de tergiversar las informaciones.

Tabla 3.2: Descripción de los subconjuntos de Emergent: distribución y porcentajes de etiquetas asignadas.

Particiones	**For**	**Against**	**Observing**
Entrenamiento	992 (47,8%)	304 (14,6%)	775 (37,4%)
Prueba	246 (46,9%)	91 (17,3%)	187 (35,6%)
Total	1238 (47,7%)	395 (15,2%)	962 (37,1%)

La distribución de documentos (contenidos, titulares y asignaciones) se presenta en la Tabla 3.3. A diferencia del conjunto de datos Emergent descrito anteriormente, solo se cuenta con contenidos de noticias y sus titulares. Para la creación del corpus se mezclan contenidos de noticias con distintos titulares y se anota la relación entre ellos.

Tabla 3.3: Descripción de los subconjuntos de FNC-1, considerando números de documentos y titulares.

Particiones	**Contenidos de noticias**	**Titulares**	**Total de asignaciones**
Entrenamiento	1683	1683	49972
Prueba	904	904	25413
Total	2587	2587	75385

En la tabla 3.4 se aprecia que la distribución de etiquetas en este conjunto es bastante desbalanceada, existiendo una gran diferencia entre la clase mayoritaria (*unrelated*) y la minoritaria (*disagree*).

Sin embargo, para hacer corresponder las etiquetas de este corpus con el Emergent, se han desechado las instancias no relacionadas (etiquetadas como *unrelated*) puesto que el Emergent no cuenta con este tipo de relación, extrayendo el subconjunto formado por todos los ejemplos etiquetados como *agree*, *disagree* y *discuss*. Posteriormente se realiza una equivalencia entre las etiquetas de ambos conjuntos de datos, según su significado la alineación de etiquetas quedaría *for* $\simeq$ *agree*, *against* $\simeq$ *disagree* y *observing* $\simeq$ *discuss*.

Tabla 3.4: Descripción de los subconjuntos de FNC-1: distribución y porcentajes de etiquetas asignadas.

Particiones	**Agree**	**Disagree**	**Discuss**	**Unrelated**
Entrenamiento	3678 (7,36%)	840 (1,68%)	8909 (17,82%)	36545 (73,13%)
Prueba	1903 (7,48%)	697 (2,74%)	4464 (17,56%)	18349 (72,20%)
Total	5581 (7,4%)	1537 (2,03%)	13373 (17,73%)	54894 (72,81%)

3.5.3 Experimentos

Para examinar la efectividad de la utilización de resúmenes de noticias para determinar si el titular y la afirmación se ajustan al contenido de la noticia, se ha diseñado un grupo de experimentos. Todos los experimentos se realizan empleando los dos modelos de detección de posturas explicados en la sección 3.3 (ML y DL) y sobre los dos conjuntos de datos descritos en la sección 3.5 (Emergent y FNC-1). Los experimentos se diseñan para demostrar que tipo de técnicas de generación y longitud del resumen es más apropiado para la realización de la tarea. Estos se agrupan en dos tipos:

- *Validación del tipo de resumen*: En este experimento, se analizan las técnicas de generación extractivas (**Lead Summarizer**, **TextRank Summarizer** y **PLM Summarizer**), la abstractiva (**HanaNLG Summarizer**) y la híbrida (**Fast Abstractive Summarizer**) con el fin de determinar la más apropiada para la tarea de detección de titulares engañosos.
- *Validación de la extensión del resumen*: Se proponen empíricamente tres longitudes de resumen diferentes: una, tres y cinco oraciones, para determinar cuál de ellas arroja un mayor beneficio al ejecutar la tarea de detección de titulares engañosos y, por tanto, es la más apropiada. Estas longitudes de resumen podrían significar un ratio de compresión medio de 12%, 37% o 62%, en el caso de uno, tres o cinco oraciones, respectivamente.

Para llevar a cabo estos experimentos, se deben generar resúmenes para el total de documentos en los conjuntos de datos, así como para cada método de resumen y cada longitud. En el caso de los métodos de resumen no orientados a un tópico (**PLM Summarizer**, **TextRank Summarizer** y **Fast Abstractive Summarizer**) el resumen se obtiene solamente del contenido de la noticia, siendo necesario generar un total de 46422 resúmenes (resultante de multiplicar cantidad de documentos provenientes de ambos conjuntos de datos (5158) por las técnicas de generación de resumen (3) y por la cantidad de longitudes (3)).

Por otro lado, **PLM Summarizer** y **HanaNLG Summarizer** son enfoques orientados a un tema (por lo que toma importancia la cantidad de asignaciones entre titulares y contenidos, en lugar de la cantidad de documentos). Para hacer estos enfoques orientados a un tema se inicializan con una semilla, en esta investigación se inicializan con la información del titular o de la afirmación para obtener el resumen orientado al tema que tratan. En total se obtienen 467780 resúmenes (resultante de multiplicar cantidad de asignaciones de los dos conjuntos de datos (77980) por las técnicas de resumen (2) y por la cantidad de longitudes (3)). Los resúmenes obtenidos están disponibles en el siguiente enlace de Github[10].

Además de las técnicas de generación de resúmenes descritas en la Sección 3.4, también se incluyen dos configuraciones adicionales que fueron utilizadas con la intención de comparar la utilización de:

- *El contenido del artículo completo*: Este experimento se realiza en ambos conjuntos de datos y en ambos enfoques de detección de posturas. Para ello, se utiliza el contenido de la noticia completa en lugar de los resúmenes como entrada de los modelos. Esto permite chequear si la utilización de resúmenes conlleva beneficios a la tarea de detección de posturas y cuán efectivos pueden resultar.

- *El título escrito por humanos (Límite superior)*: Este experimento se realiza para cada enfoque de detección titulares engañosos, utilizando el titular real de cada artículo en lugar de usar el contenido del artículo como entrada, lo que se considera un resumen perfecto. Este límite superior solo es posible con el conjunto de datos Emergent debido a que FNC-1 no cuenta con afirmaciones acerca del contenido del artículo.

3.5.4 Métricas de evaluación

Con el objetivo de evaluar el rendimiento de los experimentos se han utilizado métricas típicas de problemas de clasificación como el que abordamos en este capítulo. Estos experimentos son medidos mediante la puntuación F_1 por clase y macro-promedio F_1 ($F_1 m$). Las medidas F representan una media ponderada de la precisión y la cobertura (dos métricas muy comunes en problemas de clasificación), en el caso particular del $F_1 m$, es una puntuación que otorga la misma importancia a cada etiqueta/clase (Grandini, Bagli, y Visani, 2020). Será bajo para los modelos de clasificación que solo se desempeñan bien en las clases comunes y se desempeñan mal en las clases minoritarias. Estas métricas tienen una ventaja adicional: no se ven afectadas por el tamaño de la clase mayoritaria. Estas métricas son calculadas por mediación de la biblioteca scikit-learn[11] (Buitinck y cols., 2013).

[10] https://github.com/rsepulveda911112/FNC_Emergent_summary_dataset (consultado el 20 de enero del 2022).

[11] Documentación disponible en https://scikit-learn.org/stable/ (consultada el 6 de octubre de 2021).

En el caso particular de estos experimentos, ambos conjuntos de datos utilizados están significativamente desequilibrados, especialmente en las clases *disagree* y *against*, por lo que usar estas métricas permite evaluar objetivamente los resultados sin favorecer a las clases más representativas.

3.6 Resultados y discusión

Una vez generados los resúmenes correspondientes para cada técnica y las tres longitudes mencionadas, se experimentó con los dos modelos de detección de titulares engañosos utilizando las métricas ya mencionadas. Se muestran los resultados obtenidos en las Tablas 3.5 y 3.6 para los enfoques basados en ML y DL, respectivamente. Se resaltan en negrita los resultados superiores a los obtenidos utilizando los contenidos de noticias completos. El mejor resultado de cada columna, se indica con un asterisco (*). Los resultados de las métricas se expresan en porcentaje.

Se retoman las preguntas planteadas en la sección inicial de este capítulo para discutir los resultados obtenidos en los experimentos:

1. ¿Se pueden utilizar los resúmenes en lugar del contenido completo para la tarea de detección titulares engañosos? Definitivamente, un resumen ofrece ventajas al empleo del contenido completo de la noticia. Los titulares redactados por periodistas profesionales que solo están disponibles para el conjunto de datos Emergent (W. Ferreira y Vlachos, 2016), han demostrado ser el mejor resumen posible con un impacto positivo en la tarea de detección de postura en titulares cuando se compara con el contenido completo del texto. Estos resultados son muy prometedores por lo que la hipótesis de que el empleo de resúmenes en lugar del contenido completo puede obtener buenos resultados es consistente. El titular generado por periodistas profesionales es considerado el límite superior. Los resultados experimentales relacionados con resúmenes generados automáticamente son inferiores por los errores inherentes a este tipo de generación, la que no supera la calidad de un resumen redactado por un profesional.

Con el uso de resúmenes automáticos, los resultados son similares o mejores que con el contenido completo. Este es un hallazgo positivo que significa que el resumen es apropiado para esta tarea, siempre que el método de resumen sea lo suficientemente efectivo. Si bien es cierto que la mejora de los resúmenes se nota muy levemente en los experimentos con el modelo basado en ML con respecto al conjunto de datos FNC-1 (solo el *Lead Summarizer-5* mejora ligeramente los resultados sobre el contenido completo), esto se puede entender debido a que el modelo basado en ML fue desarrollado específicamente para el conjunto de datos Emergent, siendo muy específico.

Tabla 3.5: Experimentos con el modelo basado en ML. Experimentos con Emergent (izquierda) y FNC-1 (derecha), se muestra el F_1 por clases y el $F_1\,m$.

	Emergent				FNC-1			
	F_1 Score			$F_1\,m$	F_1 Score			$F_1\,m$
Experimentos	**For**	**Against**	**Observing**		**Agree**	**Disagree**	**Discuss**	
Contenido completo	68,03	42,10	54,49	54,88	44,88	**13,71***	75,35	**44,65**
Límite superior	81,53	74,53	68,23	74,76	-	-	-	-
Lead Summarizer-1	67,24	36,36	51,17	51,59	**46,73**	0,56	**77,66**	41,65
Lead Summarizer-3	66,79	**53,06**	54,09	**57,98**	**50,34**	1,13	**78,05**	43,17
Lead Summarizer-5	**68,92**	**54,16**	**57,68**	**60,25***	**50,91***	7,39	**75,94**	**44,75***
TextRank Summarizer-1	65,54	**43,24**	**55,64**	54,81	41,18	1,13	73,52	38,59
TextRank Summarizer-3	67,39	**43,05**	53,46	54,61	**48,52**	2,71	73,17	41,47
TextRank Summarizer-5	66,42	40,90	50,13	52,49	**49,15**	6,95	74,83	43,64
PLM Summarizer-1	64,20	21,42	44,24	43,29	28,98	0,0	**77,53**	35,50
PLM Summarizer-3	**68,27**	**43,20**	**55,49**	**55,65**	39,40	0,28	**76,66**	38,78
PLM Summarizer-5	**68,32**	**43,54**	**58,29***	**56,72**	36,43	2,72	**78,23***	39,13
HanaNLG Summarizer-1	59,66	2,17	41,24	43,43	25,50	0,0	**75,39**	33,63
HanaNLG Summarizer-3	61,98	12,96	44,63	39,86	41,86	11,69	72,99	42,18
HanaNLG Summarizer-5	62,93	17,24	47,48	42,55	41,09	11,90	73,46	42,15
Fast Abstractive Summ.-1	**69,14**	35,93	52,59	52,55	35,19	0,0	**75,82**	37,00
Fast Abstractive Summ.-3	**70,73***	**54,66***	**55,13**	**60,17**	**46,05**	1,13	71,75	39,26
Fast Abstractive Summ.-5	66,66	**44,92**	47,05	52,88	42,81	0,56	68,27	37,21

Tabla 3.6: Experimentos con el modelo basado en DL. Experimentos con Emergent (izquierda) y FNC-1 (derecha), se muestra el F_1 por clases y el $F_1 m$.

	Emergent				FNC-1			
	F_1 Score			$F_1 m$	F_1 Score			$F_1 m$
Experimentos	**For**	**Against**	**Observing**		**Agree**	**Disagree**	**Discuss**	
Contenido completo	67,08	26,44	58,03	50,52	56,98	23,02	80,96	53,65
Límite superior	77,47	67,09	71,31	71,96	-	-	-	-
Lead Summarizer-1	63,71	**33,33**	51,25	49,43	47,77	14,59	77,93	46,76
Lead Summarizer-3	**69,14***	**43,66**	54,64	**55,81**	55,70	21,85	**81,21***	52,92
Lead Summarizer-5	61,20	**27,80**	45,57	44,86	53,72	**24,65**	79,80	52,73
TextRank Summarizer-1	56,50	18,29	54,13	42,97	46,40	23,49	79,76	49,96
TextRank Summarizer-3	63,49	**29,94**	48,80	47,41	55,90	**27,01**	79,67	**54,19**
TextRank Summarizer-5	61,07	16,51	57,53	45,04	**57,32***	**29,56**	77,69	**54,85**
PLM Summarizer-1	52,43	**44,73***	**59,16***	**52,10**	53,66	9,36	77,86	46,96
PLM Summarizer-3	62,84	**30,33**	54,91	49,36	45,62	21,78	80,36	49,25
PLM Summarizer-5	**67,56**	**44,72**	55,70	**55,99***	54,84	**34,44***	79,16	**56,15***
HanaNLG Summarizer-1	61,45	22,38	47,42	43,75	49,29	13,86	75,08	46,07
HanaNLG Summarizer-3	59,63	14,03	55,58	43,08	45,20	11,12	76,12	44,15
HanaNLG Summarizer-5	59,91	19,17	50,73	43,27	51,03	7,79	76,65	45,15
Fast Abstractive Summ.-1	55,79	**31,81**	46,30	44,63	51,17	22,18	76,43	49,92
Fast Abstractive Summ.-3	65,18	**40,20**	41,40	48,93	49,92	22,30	77,47	49,89
Fast Abstractive Summ.-5	58,94	22,01	55,38	45,77	49,98	18,80	78,52	49,10

Como se puede observar, los experimentos para el enfoque de ML con FNC-1 (tabla 3.5), han conducido a resultados mucho peores que los experimentos para el enfoque DL (tabla 3.6). Este último enfoque no depende de las características propias del conjunto de datos implementadas para el enfoque ML y, por lo tanto, es más generalizable, lo que permite que algunos de los métodos de resumen mejoren los resultados del experimento que usa el contenido completo. Además, el conjunto de datos de FNC-1 tiene una clase *disagree* muy desequilibrada con respecto a las otras clases, lo que impide el aprendizaje correcto en el caso del enfoque ML.

Un inconveniente de usar resúmenes en el enfoque de ML es que se fundamenta en características específicas, y al reducir la entrada puede implicar la afectación de algunas ellas. Este fenómeno puede afectar en gran medida los resultados de clases minoritarias como es el caso de la clase *disagree* en los experimentos del conjunto de datos FNC-1. Usar el contenido completo aumentaría la precisión de las características obtenidas, mejorando así el rendimiento. Sin embargo, como se informa en los resultados, los enfoques de resumen actuales pueden ser usados para generar resúmenes que reemplacen al contenido de la noticia, sin empeorar en gran medida el rendimiento de los modelos de detección de titulares engañosos. En particular, vale la pena enfatizar que algunos de los enfoques de resumen (por ejemplo, **PLM Summarizer** y **TextRank Summarizer**) exhiben mejor desempeño que con el texto completo, incluso usando una porción mucho más pequeña de texto.

2. ¿Cuál es el mejor método de generación de resúmenes para la tarea de detección de titulares engañosos aplicando detección de posturas? En general, los sistemas extractivos obtienen mejores resultados que los abstractivos o híbridos, aunque los resultados alcanzados por el sistema híbrido **Fast Abstractive Summarizer** son muy competitivos. En particular los buenos resultados del **Fast Abstractive Summarizer** se deben al hecho de que combina un módulo extractivo con uno abstractivo y, por tanto, la parte extractiva que selecciona las frases relevantes exhibe un funcionamiento correcto. Los buenos resultados obtenidos por **Lead Summarizer** que simplemente extrae las primeras oraciones, tiene su explicación en la estructura típica del contenido de noticias producidas por periodistas, las cuales (Benson y Hallin, 2007; Pöttker, 2003) se describen como una pirámide invertida. Esto implica que la información más importante de una noticia está expresada al inicio del contenido y que la importancia del contenido decrece a medida que se avanza hacia el final del texto. En el caso de los experimentos realizados con DL, los mejores resultados fueron obtenidos con los resúmenes generados el enfoque basado en PLM, en ambos conjuntos de datos. El **TextRank Summarizer** también obtiene resultados competitivos pero en este caso solo para el corpus FNC-1 con el enfoque basado en DL.

Como ya se indicó, el método DL es el enfoque más general, dado que no está desarrollado para ningún conjunto de datos y además no usa características

específicas; simplemente es un enfoque de detección de vinculación textual que hemos aplicado en la tarea de detección de posturas. Al respecto, DL podría ser más apropiado para la tarea de detección de postura en general.

3. ¿Cuál es la longitud más apropiada para un resumen en este contexto? No se obtiene una conclusión clara acerca de la longitud apropiada para un resumen, debido a que existe una fuerte dependencia entre el método de resumen y el enfoque de detección de posturas empleado. La reducción entre un 12% y un 60% aparenta ser una buena razón de compresión para los métodos de resumen experimentados, excepto para los métodos abstractivos. Esto se debe a que los métodos abstractivos son un tipo de resumen más retador, y en ocasiones, las oraciones generadas pueden ser más cortas, de menos significado en sí mismas o no perfectas en contraste con los métodos extractivos, que extraen literalmente las oraciones más relevantes del texto.

Con independencia de las longitudes de resúmenes involucradas, **Lead Summarizer** y **PLM Summarizer** exhiben un comportamiento más estable para ambos conjuntos de datos, comparados con otros métodos. Además, cuando se aumenta la longitud del resumen, los resultados se vuelven más robustos para las tres configuraciones posibles ofrecidas por ambos enfoques de resumen. Estos métodos son significativamente mejores en el ámbito de las noticias debido a que consideran la estructura del contenido de la misma.

Tomando en consideración que esta estructura se define en términos de la teoría de la pirámide invertida, como en el caso del **Lead Summarizer** o si es expresada a través del uso de la técnica de los PLMs, los resultados indican que tomar en cuenta la estructura del documento es beneficioso para preservar la información relevante en el resumen generado, lo que tiene gran impacto en los resultados de detección. Por tanto, para respaldar la tarea de detección de titulares engañosos aplicando detección de posturas, **PLM Summarizer** podría ser considerado un método de resumen simple y muy efectivo. Este puede capturar información relevante cuando esta sigue una estructura determinada por el tipo o categoría de documento, como ocurre con los textos de noticias, pero es también aplicable a otros tipos y categorías de documentos.

3.7 Conclusiones

El uso de resúmenes para la tarea de detección de titulares engañosos sobre dos conjuntos de datos (Emergent y FNC-1) considerando dos métodos diferentes basados en (ML y DL) permiten concluir lo siguiente:

La calidad del enfoque de resumen aplicado influye en la efectividad de los sistemas probados para la detección de titulares engañosos. Este hallazgo hace presuponer que en futuros sistemas sobre detección de titulares engañosos puede ser beneficioso el uso de técnicas avanzadas de resúmenes.

Si se parte de considerar que el resumen ideal es el creado por un periodista,

es difícil identificar una técnica de resumen que se destaque en la detección de todas las clases, para los conjuntos de datos y enfoques de detección empleados.

La experimentación realizada muestra que tiene validez el empleo de resúmenes como sustitutos competentes del contenido completo, lo que beneficia la concreción de la tarea de detección de titulares engañosos. De manera conjunta, los resultados sustentan el empleo de **PLM Summarizer** y **TextRank Summarizer** al producir resultados muy competitivos y estables para la tarea, brindando la ventaja adicional de utilizar una porción pequeña del artículo de noticias, lo que puede impactar positivamente en el desempeño de una implementación.

En este sentido es oportuno diseñar una arquitectura de detección de titulares engañosos aplicando la tarea de detección de posturas, que pueda explotar los beneficios que brinda usar técnicas de resúmenes, principalmente **PLM Summarizer** y **TextRank Summarizer**. Los resultados obtenidos validan la creación de esta arquitectura basada en DL para garantizar su generalidad.

CAPÍTULO 4

Propuesta de arquitectura de detección de posturas en titulares

4.1 Introducción

En la actualidad la información engañosa se manifiesta de disímiles formas. Quizás una de las más habituales se ejecuta mediante la manipulación de titulares. Estas manipulaciones ocasionan terribles problemas (en muchos casos irreversibles) para personas, marcas, empresas, etc (Normala y cols., 2021). Si ello se une a la certeza de que cada vez más personas se informan a través de redes sociales y medios digitales, siendo un número considerable los que simplemente leen y luego comparten los titulares sin ni siquiera realizar una lectura de la noticia[1], este fenómeno convierte a los titulares en un grave problema que es necesario monitorear.

Se han identificado tres tipos principales de titulares (ciberanzuelos, incongruentes y engañosos), aunque conceptualmente no son mutuamente excluyentes (podemos encontrar titulares que satisfacen múltiples clasificaciones) es importante destacar que las técnicas para detectarlos difieren considerablemente. Esta investigación solo analiza los titulares engañosos, los cuales implican un análisis semántico con respecto al contenido de la noticia que refieren.

Delineando la necesidad de detectar titulares engañosos, se retoman dos de los objetivos principales de esta investigación: diseñar una arquitectura de detección de titulares engañosos y llevar a cabo algunas implementaciones para evaluar su comportamiento y posible generalidad a otras tareas.

Este capítulo muestra el diseño de una arquitectura de detección de titulares engañosos específica, haciendo uso de la tarea de detección de posturas. Ade-

[1] Incuantificable en la vida real, pero estudios realizados en Twitter informan que cerca del 59% de las url compartidas nunca son accedidas por los usuarios que las comparten (Gabielkov y cols., 2016).

más, se proponen dos implementaciones de la arquitectura que difieren en elementos claves de esta, evaluando el rendimiento puntual de cada una de ellas. Los hallazgos obtenidos en el capítulo anterior sobre el uso de resúmenes también han influido en aspectos determinantes en el diseño de esta arquitectura.

4.2 Arquitectura de detección de posturas entre titulares y contenidos de noticias

En la sección 2.3 se planteó que una de las formas más habituales de abordar la detección de titulares engañosos es mediante el uso de la detección de posturas. Los principales trabajos en esta área se han abordado sobre el corpus de referencia FNC-1. Con el objetivo de comparar los resultados con sistemas tomados del estado del arte en la tarea de detección de posturas entre titulares y contenidos de noticias, se ha decidido usar este propio conjunto de datos.

Por otra parte, el conjunto de datos Emergent introducido en el capítulo anterior también se utiliza con motivos de validación de la arquitectura, principalmente porque es un corpus de menor tamaño y que es desarrollado para la tarea de detección de posturas. Este conjunto de datos carece de titulares *unrelated*, lo que no es común en contextos reales. La descripción de los conjuntos de datos, así como la distribución de etiquetas se pueden consultar en la sección 3.5.

El corpus FNC-1 contiene cuatro tipos de etiquetas *agree*, *disagree*, *discuss* y *unrelated*. Retomando sus definiciones, específicamente la clase *unrelated*, plantea que un titular y el contenido de la noticia abordan temas diferentes, lo que indica la distancia semántica entre este tipo de ejemplos. Por otra parte, en las clases *agree*, *disagree* y *discuss* el titular y el contenido tratan el mismo tema pero con diferencias marcadas en la postura, lo que hace presuponer que agruparlas en una clase ficticia *related* y discriminar en primer lugar *related* y *unrelated* y posteriormente el resto de posturas podría ser beneficioso para los resultados finales de clasificación.

En la sección 2.5.1 se realizó un profundo análisis sobre las clasificaciones jerárquicas. Este tipo de clasificador generalmente obtiene mejores resultados en clasificaciones que los clasificadores planos siempre que exista una relación de jerarquía entre las clases. Se mostraron trabajos que generan la jerarquía de clases de forma automática basados en la similitud entre etiquetas o para abordar el desbalance de un conjunto de datos. En este capítulo se experimenta sobre la clasificación jerárquica creando la jerarquía de clases en función de la similitud semántica que exhiben las etiquetas.

La figura 4.1 muestra la estructura jerárquica de clases creada. Esta jerarquía representa una estructura arbórea desde el nodo raíz (Titular-Contenido) hasta las hojas que representan las etiquetas a clasificar del conjunto de datos.

Para solucionar este problema de clasificación se utiliza el enfoque de Clasificador Local por Nodo Padre (LCPN), que como su nombre indica, propone crear un clasificador por cada nodo padre de la jerarquía. En el caso particu-

lar de esta estructura se necesitaría un primer clasificador en el nodo Titular-Contenido para discriminar en *related* y *unrelated* y un segundo clasificador en el nodo *related* para clasificar posteriormente en *agree, disagree* y *discuss* los ejemplos clasificados como *related* en el primer nivel de clasificación.

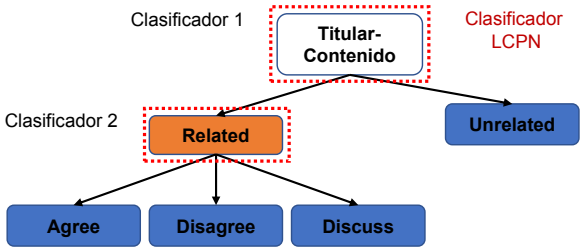


Figura 4.1: Estructura jerárquica de clasificación.

El enfoque de clasificación escogido es bastante intuitivo y fácil de implementar, de aquí que haya sido la elección en este trabajo. Su propia lógica indica que cada clasificador se especializa en un conjunto de clases similares, lo que permite hacer uso de características específicas que compartan y por tanto llegar a una mejor discriminación. Los algoritmos de clasificación a emplear pueden seleccionarse según las características de las clases. Los clasificadores son entrenados con el subconjunto de ejemplos que corresponden a la clase padre y en la fase de clasificación un ejemplo solo puede ser clasificado en una clase en concreto, evitando problemas de inconsistencia.

Explotando las ventajas del enfoque LCPN para crear clasificadores con características específicas en cada nodo padre, se define **HeadlineStanceChecker**, una arquitectura que involucra dos etapas representando a cada clasificador: una para detectar si el titular y el contenido de la noticia abordan el mismo tema (**Etapa de relación**) y otra para detectar la relación de posturas entre el titular y el contenido (**Etapa de postura**).

Cada etapa de la arquitectura está compuesta por módulos que encapsulan diferentes tipos de procesamiento. El diseño de módulos independientes permite su reutilización para futuras configuraciones de la arquitectura. Una etapa se estructura esencialmente con al menos un módulo de clasificación aunque se pueden adicionar otros de carácter específico según las características de la tarea. Los módulos que pueden estar presentes en las etapas son:

- **Módulo detección de información relevante**: se define como un módulo encargado de extraer información relevante de un texto, ya sea mediante un resumen de este o usando métodos de extracción de información.
- **Módulo de extracción de características**: es el encargado de extraer características específicas relacionadas con las etiquetas que se clasifican en el módulo de clasificación.
- **Módulo de clasificación**: este módulo en efecto es un clasificador que se

encarga de clasificar las etiquetas definidas para la etapa. Se beneficia de otros módulos antes descritos.

Se muestra la estructura interna de una configuración de la arquitectura en la figura 4.2. Esta configuración tiene como novedad el uso de resúmenes generados en la primera etapa para todo el proceso en lugar del contenido completo, es decir, en la **Etapa de relación**. La decisión de probar el comportamiento de un sistema de clasificación utilizando resumen en lugar del contenido del artículo, está justificada por la experimentación realizada en el capítulo 3 sobre la validez del uso de resúmenes en la tarea de detección de posturas.

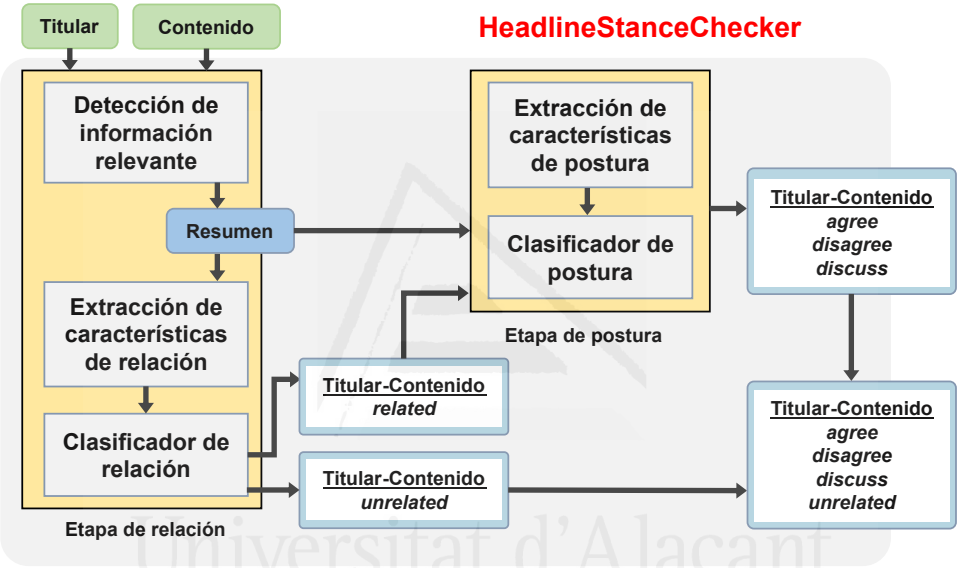


Figura 4.2: Estructura interna de una configuración de la arquitectura HeadlineStanceChecker.

Las entradas de la arquitectura son el titular y el contenido de la noticia. En la **Etapa de relación** se obtiene un resumen del contenido de la noticia y posteriormente se lleva a cabo una clasificación binaria (*related* y *unrelated*). Los resultados de esta etapa son:

- El par (titular-resumen del contenido) clasificados como *related* o *unrelated*.
- El resumen del contenido de la noticia, obtenido en el **Módulo de detección de información relevante** correspondiente a cada titular.

Posteriormente, los pares (titular-resumen de contenido) clasificados como *related* son ingresados a la **Etapa de postura**, clasificándose la relación entre ellos en: *agree*, *disagree* o *discuss*. Estos titulares clasificados junto con los titulares *unrelated* determinados anteriormente, constituirán el resultado final de todo el proceso desarrollado por la arquitectura.

Los detalles de diseño presentados en esta sección permiten crear diferentes configuraciones de la arquitectura con variaciones en sus módulos internos. Como se puede apreciar, el diseño de la arquitectura es específico para la tarea de detección de posturas entre titulares sobre el corpus FNC-1. Este incluye algunas decisiones basadas en experimentaciones realizadas sobre la tarea de detección de posturas como la de usar resúmenes en sustitución del contenido completo del artículo, así como la inclusión de características externas que puedan apoyar el proceso de clasificación.

4.3 Implementaciones de la arquitectura de detección

Se implementaron dos prototipos de la arquitectura **HeadlineStanceChecker** usando dos enfoques de resúmenes y con diferentes características en ambas etapas. Basado en el estudio de la influencia de los enfoques de resumen en la tarea de detección de posturas del capítulo 3, se decidió experimentar en estas implementaciones con los enfoques **TextRank Summarizer** y **PLM Summarizer** debido a que se obtienen los resultados más competitivos sobre sistemas basados en DL. Estas implementaciones permiten evaluar el rendimiento de la arquitectura propuesta con diferentes configuraciones de esta así como su modularidad y futura generalización. Las implementaciones se explican en las secciones 4.3.1 y 4.3.2. Se describen los aspectos específicos de implementación en cada una de las secciones, así como elementos conceptuales que sirven para entender modelos de lenguaje y características de similitud utilizados.

4.3.1 Implementación utilizando TextRank Summarizer

La implementación presentada en esta sección corresponde íntegramente con la configuración de la arquitectura mostrada en la figura 4.2. Se explica cada uno de los módulos presentes en la arquitectura y se utiliza el enfoque de resumen **TextRank Summarizer** para llevar a cabo la clasificación.

Etapa de relación

Esta etapa es la encargada de determinar si el titular y el contenido de la noticia están relacionados o no. Para esta primera etapa se han incluido tres módulos: **Módulo de detección de información relevante**, **Módulo de extracción de características de relación** y **Módulo de clasificación de relación**. Las funciones de cada uno de los módulos se describen a continuación.

Módulo de detección de información relevante:

En esta implementación de la arquitectura propuesta se usa el popular y efectivo algoritmo de resumen extractivo **TextRank Summarizer** (Mihalcea y Tarau, 2004), debido a su buen rendimiento, tiempo de ejecución y disponibilidad de implementación. Este enfoque de resumen tiene la limitación de no generar

resúmenes orientados a temas (por ejemplo, al titular), sin embargo, siendo un algoritmo sencillo, sus resúmenes suelen ser bastante competitivos. Se utiliza como longitud de resumen cinco oraciones.

Módulo de extracción de características de relación:

Este módulo extrae las características específicas, en este caso medidas de similitud entre textos. Para la obtención de estas características son usados tanto el titular como el resumen:

- *Similitud de coseno con vectores* TF-IDF: Se calcula la similitud de coseno con los vectores TF-IDF del titular y del resumen, utilizando las herramientas proporcionadas por la biblioteca scikit-learn. Más detalles sobre esta similitud se pueden encontrar en la sección 3.3.1.

- *Similitud de coseno con vectores* BERT: El titular y resumen del contenido son convertidos a vectores de embedding (Reimers y Gurevych, 2019). Se utiliza la biblioteca SentenceTransformers[2] para convertir textos en vectores y posteriormente calcular su similitud.

- *Similitud de coseno con word embedding*: Las palabras del titular y del resumen del contenido son convertidos a vectores word2vec (Mikolov y cols., 2013).

- *Coeficiente de superposición*: Esta característica mide la similitud entre dos conjuntos A y B. En el escenario actual, estos conjuntos contienen los n-gramas que pertenecen al titular y al resumen (Šarić, Glavaš, Karan, Šnajder, y Dalbelo Bašić, 2012). El coeficiente de superposición (CS) viene dado por la ecuación 4.1 (Metcalf y Casey, 2016):

$$CS(A,B) = \frac{|A \cap B|}{\min(|A|,|B|)} \tag{4.1}$$

donde:

$|X|$: cardinalidad del conjunto X

Si el conjunto A es un subconjunto de B o a la inversa, entonces el coeficiente de superposición es igual a 1, de lo contrario el coeficiente de superposición debe estar entre 0 y 1 (Vijaymeena y Kavitha, 2016).

En algunas medidas de similitud el texto del titular y del resumen son preprocesados, extrayendo *stop words* para la *Similitud de coseno* y *Coeficiente de superposición*; además de tokenizar y lematizar previo al cálculo del *Coeficiente de superposición*. Los valores de estas características externas se encuentran en el rango de 0 a 1.

[2] Documentación disponible en `https://www.sbert.net/` (consultada 6 de octubre de 2021).

Módulo de clasificación de relación:

Este módulo es el encargado de clasificar la relación entre el titular y resumen en *related* y *unrelated*. Para llevar a cabo esta clasificación se crea una arquitectura de clasificación que está formada por dos componentes principales: un modelo de lenguaje y una red neuronal de clasificación. En la figura 4.3 se muestra la arquitectura interna del módulo de clasificación propuesto. Las entradas de esta arquitectura de clasificación son: el titular, el resumen (los cuales son concatenados y pasados al modelo de lenguaje) y la otra entrada en caso de existir son las características externas (son concatenadas con la salida del modelo de lenguaje para alimentar la entrada a la red neuronal de clasificación).

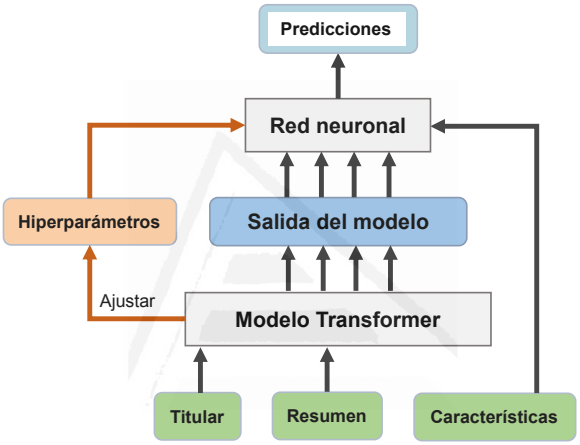


Figura 4.3: Arquitectura interna del módulo de clasificación.

El modelo de lenguaje elegido es un modelo que utiliza la arquitectura Transformer (Vaswani y cols., 2017). Actualmente este tipo de modelo está obteniendo resultados de vanguardia en las principales tareas dentro de PLN. En este caso se ha utilizado el modelo de lenguaje pre-entrenado RoBERTa (Y. Liu, Ott, y cols., 2019). A este modelo se le realiza un proceso de ajuste fino (*fine-tuning*), que consiste en reentrenar el modelo haciendo uso de un corpus de entrenamiento para ajustar los pesos de cada una de las capas que lo contienen. Posteriormente se utiliza una red neuronal encargada de llevar a cabo la clasificación, a continuación se muestra la estructura interna:

1. Se utiliza una capa densa[3] con función de activación Tanh para extender las características externas a un vector que contenga las mismas dimensiones que la salida del modelo de lenguaje.

2. Una capa para multiplicar las características extendidas en la capa anterior y las salidas del modelo de lenguaje. Las dos primeras capas no se utilizan si no se cuenta con características externas.

[3]Capa de red neuronal que está profundamente conectada, recibe información directa de todas las neuronas de la capa anterior.

3. Una capa *dropout*[4] con tasa de 0,2 para evitar que el modelo de clasificación se sobreajuste.

4. Una capa densa con una neurona de salida que permite clasificar en *related* y *unrelated* con función de activación Sigmoid.

5. Se utiliza como función de pérdida *binary cross-entropy*.

Para crear el clasificador, se utilizó la biblioteca Simple Transformers[5]. Esta es una biblioteca de PLN que permite la modificación de hiperparámetros para entrenar, evaluar y hacer predicciones utilizando modelos de última generación. Específicamente se utiliza el modelo de lenguaje RoBERTa *large* con 24 capas y 1024 unidades ocultas. Por último, para ajustar el modelo de lenguaje y la red neuronal posterior se utilizan los siguientes hiperparámetros: longitud máxima de secuencia de 512, tamaño de lote de 4, tasa de entrenamiento de 1e-5 y entrenamiento realizado durante 3 iteraciones. Estos valores se establecieron tras evaluaciones sucesivas, partiendo de experimentos previos sobre este modelo (Dulhanty y cols., 2019; Slovikovskaya, 2019; Y. Liu, Ott, y cols., 2019). En este caso se utilizan cuatro características externas (*similitud de coseno con vectores* TF-IDF, *similitud de coseno con word embedding*, *similitud de coseno con* BERT y *coeficiente de superposición*).

Etapa de postura

Una vez que se ha podido identificar los titulares que están relacionados con el resumen de su contenido de noticia, el objetivo principal de esta etapa es determinar las posturas: *agree*, *disagree* o *discuss*. Esta etapa está compuesta por dos módulos: **Módulo de extracción de características de postura** y **Módulo de clasificación de postura**.

Módulo de extracción de características de postura:

En este módulo se obtienen características de polaridad del titular y del resumen independientemente. Estas características muestran la intensidad de sentimientos positivos y negativos y sus valores se encuentran entre 0 y 1. Se calculan utilizando la herramienta NLTK[6] (Bird, Klein, y Loper, 2009). Estas características son: *Polaridad positiva y negativa del titular (Pol_head_pos, Pol_head_neg)* y *Polaridad positiva y negativa del resumen (Pol_sum_pos, Pol_sum_neg)*.

Módulo de clasificación de postura:

[4]Capa para reducir el sobreajuste en los modelos de redes neuronales, elimina el ruido que puede estar presente en la entrada de las neuronas.

[5]Documentación disponible en `https://simpletransformers.ai/` (consultada el 20 de enero de 2022).

[6]Documentación disponible en `https://www.nltk.org/` (consultada el 10 de enero de 2022).

El módulo de clasificación creado en esta implementación utiliza la arquitectura de clasificación descrita en la figura 4.3. Esta incluye las cuatro características extraídas en el **Módulo de extracción de características de postura**. Similar al **Módulo de clasificación de relación** se utilizan características externas (para mejorar los resultados finales de clasificación) y el modelo de lenguaje RoBERTa.

La estructura de la red neuronal utilizada se compone de las dos primeras especificaciones de la red neuronal descrita en el **Módulo de clasificación de relación**, seguido de las siguientes:

1. Una capa *dropout* con tasa de 0,2 para evitar que el modelo de clasificación se sobreajuste.
2. Una capa densa con tres neuronas de salida que permite clasificar en *agree*, *disagree* y *discuss* con función de activación Softmax.
3. Se utiliza como función de pérdida *cross-entropy*.

Información complementaria relacionada con esta implementación se puede consultar en el apéndice A.

4.3.2 Implementación utilizando PLM Summarizer

Esta implementación corresponde con una configuración de la arquitectura **HeadlineStanceChecker** diferente a la mostrada en la figura 4.2. Posterior a una serie de experimentos se decidió no incluir características externas en la **Etapa de postura** por lo que no se necesita el **Módulo de extracción de características de postura**. Esta decisión conlleva una diferencia marcada con la implementación anterior, además del uso del enfoque de resumen automático **PLM Summarizer** el cual es semánticamente es más complejo y se espera que pueda generar un resumen más competitivo. Se omiten algunos detalles que han sido presentados en la implementación anterior, entre ellos, la definición formal de algunas medidas de similitud.

Etapa de relación

Esta etapa se corresponde con la configuración mostrada de la arquitectura en la figura 4.2. Es la encargada de obtener el resumen del contenido utilizando el enfoque de resumen **PLM Summarizer**.

Módulo de detección de información relevante:

En el caso particular de esta implementación este módulo contiene un enfoque de resumen automático para resumir la información más relevante del contenido de la noticia. El análisis llevado a cabo en el capítulo 3 muestra que el enfoque basado en PLM produjo los resultados más estables de todos los métodos automáticos de resumen analizados. Teniendo en cuenta esto, se eligió el **PLM Summarizer** con cinco oraciones como piedra angular de este módulo.

Para la realización del resumen se utilizan como semillas del enfoque de PLM las palabras claves del titular de la noticia, lo que permite obtener un resumen orientado al tema tratado en el titular. Además del resumen, el enfoque también calcula una *puntuación de parentesco del* PLM que representa la similitud entre el titular y el contenido de la noticia. Esta puntuación será usada como característica de relación en el módulo de clasificación.

Módulo de extracción de características de relación:

Además de la información relevante (es decir, el resumen) y la *puntuación de parentesco del* PLM obtenida en el módulo anterior, se extraen dos características de similitud (*Similitud de coseno con vectores* TF-IDF y *Coeficiente de superposición*) que se utilizan de entrada para el clasificador de relación que se aplica a continuación. Estas dos medidas han sido introducidas en la implementación anterior y el preprocesamiento y el cálculo son idénticas, solo que en este caso se utiliza el resumen extraído por **PLM Summarizer**.

Módulo de clasificación de relación:

En este módulo se utiliza íntegramente la arquitectura de clasificación que se muestra en la figura 4.3, además el modelo de lenguaje, la red neuronal utilizada y los hiperparámetros son los mismos. En este caso se utilizan solo tres características externas (*similitud de coseno con vectores* TF-IDF, *coeficiente de superposición* y *puntuación de parentesco del* PLM).

Etapa de postura

Como se explicó anteriormente, esta etapa no contiene **Módulo de extracción de características de postura** lo que permite probar experimentalmente el comportamiento de la arquitectura ante cambios en su configuración.

Módulo de clasificación de postura:

Similar al **Módulo de clasificación de relación**, esta etapa se ha construido utilizando el modelo de lenguaje RoBERTa. En este caso, no se utilizan características externas por lo que solo se lleva a cabo un ajuste del modelo con la partición de entrenamiento del corpus FNC-1 excluyendo la etiqueta *unrelated*. La estructura interna de la red neuronal de clasificación es:

1. Una capa *dropout* con tasa de 0,2 para evitar que el modelo de clasificación se sobreajuste.

2. Una capa densa con tres neuronas de salida que permite clasificar en *agree*, *disagree* y *discuss* con función de activación Softmax.

3. Se utiliza como función de pérdida *cross-entropy*.

4.3.3 Resumen de las implementaciones

En este apartado se discuten las diferencias conceptuales de cada implementación. Como se puede apreciar en la tabla 4.1 la configuración de la arquitectura en cada implementación difiere principalmente en la cantidad de módulos de la **Etapa de postura**, el algoritmo de resumen, así como los tipos y cantidades de características externas utilizadas. Estos cambios son principalmente decisiones de diseño tomadas después de sucesivas experimentaciones.

Tabla 4.1: Detalles de las implementaciones.

	Implementación sección 4.3.1	**Implementación sección 4.3.2**
Etapa de relación		
Cantidad de módulos	3	3
Enfoque de resumen	**TextRank Summarizer**	**PLM Summarizer**
Longitud de resumen	5 oraciones	5 oraciones
Cantidad de características	4	3
Modelo de clasificación	RoBERTa	RoBERTa
Etapa de postura		
Cantidad de módulos	2	1
Cantidad de características	4	0
Modelo de clasificación	RoBERTa	RoBERTa

Un semejanza importante entre las implementaciones es que ambas utilizan como modelo de lenguaje para crear los módulos de clasificación a RoBERTa. Esta característica permite centrarse en el análisis de otros componentes de las implementaciones como las características externas y los enfoques de resúmenes para evaluar su rendimiento, sin que se introduzcan variaciones provocadas por el cambio del modelo de clasificación. Por otra parte es importante notar que los hiperparámetros de los modelos de clasificación y las herramientas de implementación son exactamente los mismos para todos los módulos de clasificación en ambas implementaciones, usándose la configuración descrita en la **Etapa de relación** de la implementación que usa **TextRank Summarizer**.

4.4 Experimentos

Para medir el rendimiento de la clasificación de las implementaciones de la arquitectura de detección de posturas en titulares, se propone un conjunto de experimentos, cuyos resultados se muestran y discuten en la sección 4.5. Los dos primeros experimentos se pueden considerar pruebas unitarias de cada eta-

pa y el tercero es una prueba de integración de las etapas. A continuación se describen los experimentos propuestos:

1. **Validación de la etapa de relación**: El primer experimento fue diseñado para evaluar la primera etapa de la arquitectura como un elemento aislado. En primer lugar se analizan y comparan las implementaciones de la arquitectura en términos de rendimiento. En segundo lugar se analiza la influencia de las características externas en el comportamiento del módulo de clasificación de la primera etapa mediante un estudio de ablación de características. En esta etapa se detectan los ejemplos *unrelated* y se agrupan el resto de elementos (*agree*, *disagree* y *discuss*) en la clase ficticia *related*.

2. **Validación de la etapa de postura**: El objetivo de este experimento es determinar la precisión de la **Etapa de postura** cuando se evitan posibles errores producidos por la **Etapa de relación**. En este experimento se utiliza una porción del corpus FNC-1 como entrada ideal para esta etapa, es decir, los titulares agrupados como *related*. Además, para validar hasta qué punto la arquitectura de detección de posturas propuesta puede generalizarse, se aplica a otro conjunto de datos de detección de posturas entre titulares y contenidos de noticias, el conjunto de datos Emergent.

3. **Validación de la arquitectura HeadlineStanceChecker**: Se prueba todo el sistema integrando la **Etapa de relación** y la **Etapa de postura**, usando resúmenes como entrada para todo el proceso en lugar del contenido completo. De esa manera su rendimiento se compara con otras configuraciones del modelo, así como con sistemas competitivos de última generación.

4. **Comparación de los resúmenes contra el contenido completo de los artículos**: Se diseña una serie de experimentos que tienen como objetivo probar la validez del uso de resúmenes. Se prueba la arquitectura con la utilización de contenidos de noticias. Además se muestra un análisis de los principales corpus de noticias.

5. **Modelo de lenguaje para procesar textos extensos**: Se experimenta sobre un modelo de lenguaje capaz de procesar textos extensos (Longformer). Se compara el rendimiento de este utilizando un clasificador plano y la arquitectura de detección creada.

4.5 Resultados y discusión

Esta sección presenta los resultados obtenidos en cada uno de los experimentos descritos en la Sección 4.4. Las tablas presentadas con los resultados que incluyen el rendimiento por clases F_1 (F_1 *score*) y el macro-promedio F_1

($F_1\,m$). Los valores se expresan en modo de porcentaje. Los experimentos presentados en esta sección pueden ser replicados haciendo uso del siguiente enlace de GitHub[7].

4.5.1 Validación de la etapa de relación

La **Etapa de relación** se validó comparando los resultados finales de clasificación de la etapa de dos configuraciones distintas de esta: *Etapa de relación con TextRank Summarizer* (implementación de la sección 4.3.1) y *Etapa de relación con PLM Summarizer* (implementación de la sección 4.3.2). El objetivo principal de este experimento es analizar si el uso de resúmenes y las características extraídas tienen un impacto positivo en el resultado de clasificación. Los resultados se muestran en la Tabla 4.2.

Tabla 4.2: Resultados de clasificación de la Etapa de relación: puntuación F_1 por clase y $F_1\,m$ usando resúmenes automáticos.

	F_1 Score		$F_1\,m$
Experimentos	**Related**	**Unrelated**	
Etapa de relación con TextRank Summarizer	98,22	99,31	98,77
Etapa de relación con PLM Summarizer	**98,38**	**99,40**	**98,89**

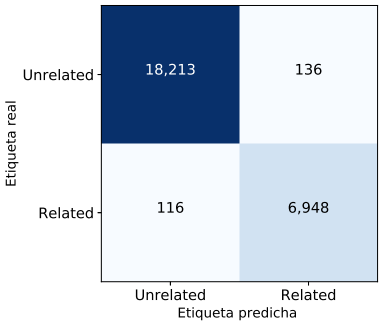
La *Etapa de relación con TextRank Summarizer* y *Etapa de relación con PLM Summarizer* utilizan resúmenes para calcular características y como entrada al modelo de clasificación. Los resultados de los dos experimentos son relativamente cercanos, lo que hace concluir que ambos enfoques tienen un impacto similar en la clasificación. Las figuras 4.4a y 4.4b muestran cada matriz de confusión de los experimentos de esta etapa.

Analizando las matrices de confusión de ambas implementaciones no se nota una diferencia marcada, estas clasificaciones alcanzan resultados cercanos al 99% de $F_1\,m$ lo que es un resultado muy bueno para una clasificación binaria.

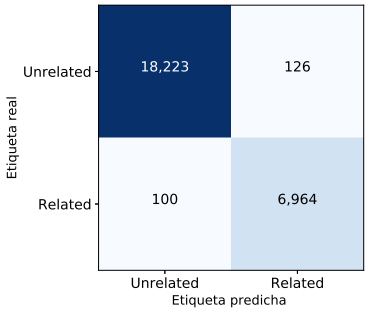
Con el objetivo de evaluar la influencia de las características añadidas en la **Etapa de relación**, se ha realizado un estudio de ablación de las características extraídas del resumen. Se ha eliminado cada característica y se ha diseñado un experimento que devolvió los resultados de la clasificación sin la incidencia de la característica eliminada. El estudio de ablación de características se realiza sobre cada una de las implementaciones. La tabla 4.3 muestra el estudio de ablación.

En la medida en que el resultado de la clasificación sea peor, se infiere que la característica eliminada tiene una mayor influencia en los resultados. Para que

[7] https://github.com/rsepulveda911112/HeadlineStanceChecker (consultado 20 de enero de 2022).



(a) Matriz de confusión: *Etapa de relación con TextRank Summarizer.*



(b) Matriz de confusión: *Etapa de relación con PLM Summarizer.*

Figura 4.4: Matrices de la Etapa de relación.

Tabla 4.3: Resultados del estudio de ablación para las características utilizadas en la Etapa de relación: TextRank Summarizer y PLM Summarizer.

	F_1 Score		$F_1\,m$
Características eliminadas	**Related**	**Unrelated**	
TextRank Summarizer			
Similitud de coseno con vectores TF-IDF	**97,52**	**99,04**	**98,28**
Coeficiente de superposición	98,04	99,24	98,64
Similitud de coseno con vectores BERT	97,66	99,11	98,38
Similitud de coseno con word embedding	98,05	99,26	98,66
Etapa de relación con TextRank Summarizer	98,22	99,31	98,77
PLM Summarizer			
Puntuación de parentesco del PLM	**98,00**	**99,23**	**98,61**
Similitud de coseno con vectores TF-IDF	98,24	99,32	98,78
Coeficiente de superposición	98,10	99,27	98,68
Etapa de relación con PLM Summarizer	98,38	99,40	98,89

el análisis sea más sencillo se marca con negritas los peores resultados de puntuación F_1 y de $F_1\,m$ y se colocan los resultados de la etapa con todas las características para cada implementación. En el caso de la implementación que utiliza **PLM Summarizer** la característica más influyente es la *Puntuación de parentesco del* PLM ya que el experimento que no la usa obtiene los peores resultados en todas las métricas evaluadas. Sin embargo, en el caso de la implementación que utiliza **TextRank Summarizer** la característica más influyente es *Similitud de coseno con vectores* TF-IDF. Como se puede apreciar la eliminación de cada característica provocó un impacto negativo en los resultados, evidenciando su influencia positiva en la clasificación.

4.5.2 Validación de la etapa de postura

El segundo experimento como se describió en la Sección 4.4 es el encargado de probar primeramente la validez y posteriormente la generalidad de la etapa de postura. La etapa es evaluada con la fracción del corpus FNC-1 que contiene los ejemplos *related*. Similar al capítulo 3 se utiliza el corpus Emergent, introducido en la Sección 3.5, para analizar si es posible generalizar la solución a otros corpus similares. Para alinear las clases de los dos corpus se sigue la mismas especificación antes presentada en la Sección 3.5. Por lo tanto, para abordar estas validaciones, la tabla 4.4 incluye las siguientes configuraciones de esta etapa:

- *TextRank Summarizer* FNC-1: Esta configuración se corresponde con la descrita en la **Etapa de postura** de la sección 4.3.1 para validar el uso de resúmenes generados por **TextRank Summarizer**.
- *PLM Summarizer* FNC-1: Esta configuración se corresponde con la descrita en la **Etapa de postura** de la sección 4.3.2 para validar el uso de resúmenes generados por **PLM Summarizer**.
- *Límite superior Emergent*: Esta configuración se considera un límite superior dado que utiliza el titular (creado por un periodista y considerado un resumen perfecto) y la afirmación para entrenar y predecir el modelo. Este titular tiene completa correspondencia con el contenido de la noticia. Sin embargo, este límite superior solo es aplicable al conjunto de datos Emergent ya que en el caso de FNC-1 lo que se anota es la relación entre el titular y el contenido, no entre el titular y una afirmación.
- *Etapa de postura con Emergent*: Se utiliza el conjunto de datos Emergent en todo el proceso de la etapa, incluido el entrenamiento y la validación del clasificador. La entrada de la etapa es la afirmación y el resumen del contenido de la noticia generado por **PLM Summarizer**.
- *Etapa de postura con entrenamiento* FNC-1 *y validación Emergent*: Esta configuración utiliza el conjunto de datos Emergent para validar la etapa de postura, pero en este caso el **Módulo de clasificación de postura** es entrenado en el conjunto de datos FNC-1 para demostrar hasta qué punto esta propuesta se puede generalizar.

En la tabla 4.4 se muestran los resultados de la **Etapa de postura** de postura de forma aislada, cada fila corresponde a las dos primeras configuraciones mencionadas previamente.

El desempeño de la **Etapa de postura** de forma aislada, es decir, sin considerar la **Etapa de relación**, es notablemente inferior en términos de $F_1 m$ que la **Etapa de relación**. Esto era de esperar ya que aunque se evitan los errores derivados de la **Etapa de relación** esta clasificación semánticamente es más compleja y el conjunto de datos está desbalanceado, observándose una gran diferencia en

Tabla 4.4: Resultados de Etapa de postura: puntuación F_1 por clase y $F_1 m$ en el conjuntos de datos FNC-1.

	F_1 Score			$F_1 m$
Experimentos	**Agree**	**Disagree**	**Discuss**	
TextRank Summarizer FNC-1	**74,54**	**64,54**	87,69	**75,59**
PLM Summarizer FNC-1	72,87	63,50	**88,74**	75,04

cantidad de ejemplos de la clase mayoritaria (*discuss*) y las minoritarias (*agree* y *disagree*). Los resultados alcanzados por las configuraciones que usan distintos enfoques de resúmenes difieren de los obtenidos en la **Etapa de relación**. En este caso, los mejores resultados son alcanzados por la implementación que utiliza **TextRank Summarizer** en todas las clases con excepción de la clase *discuss*.

El resto de configuraciones sobre el conjunto de datos Emergent son realizadas utilizando la **Etapa de postura** de la implementación descrita en 4.3.2. La tabla 4.5 muestra los resultados de cada configuración.

Tabla 4.5: Generalización de Etapa de postura: puntuación F_1 por clase y $F_1 m$ en el conjuntos de datos Emergent.

	F_1 Score			$F_1 m$
Experimentos	**Agree**	**Disagree**	**Discuss**	
Límite superior Emergent	**85,82**	**84,44**	**78,06**	**82,77**
Etapa de postura Emergent	75,15	77,77	65,49	72,80
Etapa de postura con entrenamiento FNC-1 y prueba Emergent	73,15	73,68	70,61	72,48

El análisis de los resultados obtenidos en esta etapa con respecto a la comparación del desempeño usando el conjunto de datos Emergent es muy prometedor, considerando las configuraciones que están usando resúmenes automáticos. Sin embargo, los resultados difieren notablemente a los alcanzados con el experimento *Límite superior Emergent*, obtenido al encontrar la postura entre las afirmaciones y titulares confeccionados por humanos en lugar de resúmenes generados automáticamente. Al analizar por clase, se puede apreciar mejor la diferencia entre el experimento *Límite superior Emergent* y el resto.

Por otra parte, con la configuración de la *Etapa de postura con entrenamiento* FNC-1 *y validación Emergent*, se obtienen resultados similares al experimento *Etapa de postura con Emergent* que realiza todo el proceso (entrenamiento y prueba) con el conjunto de datos Emergent. Este resultado muestra que tanto la arquitectura como el entrenamiento con el conjunto de datos FNC-1 pueden ser generalizables para la tarea, al menos con los experimentos realizados en el

conjunto de datos Emergent.

Tal y como se realizó en la **Etapa de relación**, se llevó a cabo un estudio de ablación (Tabla 4.6), donde se probó el clasificador de la **Etapa de postura** eliminando cada una de las características propuestas (*Pol_head_pos*, *Pol_head_neg*, *Pol_sum_pos*, *Pol_sum_neg*). En este caso el estudio de ablación solo es posible para la configuración que utiliza **TextRank Summarizer** porque es la que incluye características externas al clasificador. Las características incluidas muestran claramente su influencia positiva en el desempeño del clasificador, siendo la más influyente *Pol_head_pos*.

Tabla 4.6: Resultados del estudio de ablación para las características utilizadas en la Etapa de postura: TextRank Summarizer.

	F_1 Score			$F_1\,m$
Características eliminadas	**Agree**	**Disagree**	**Discuss**	
Pol_head_pos	**71,64**	**56,99**	**87,10**	**71,91**
Pol_head_neg	72,19	58,84	88,12	73,05
Pol_sum_neg	71,68	61,31	88,11	73,70
Pol_sum_pos	73,08	59,94	88,26	73,76

4.5.3 Validación de la arquitectura HeadlineStanceChecker

Se realizaron cuatro experimentos con el objetivo de evaluar el rendimiento de las implementaciones de la arquitectura con respecto a sistemas del estado del arte en la tarea de detección de posturas sobre titulares, usando el corpus FNC-1. Estos experimentos constituyen la integración de las dos etapas en la arquitectura. Los resultados del **HeadlineStanceChecker** se muestran en la Tabla 4.7 filas 8 y 10. Además de las métricas mencionadas en el inicio de la sección, se incluyen la exactitud (Acc.) y la puntuación relativa (Rel. Score). Esta última métrica fue propuesta originalmente por los organizadores del desafío FNC-1, la cual asigna una puntuación de 0,25 a los ejemplos *unrelated* y *related* clasificados correctamente, esta puntuación se incrementa en 0,75 si además los ejemplos *related* son bien clasificados en sus etiquetas (*agree*, *disagree* y *discuss*).

Las tres primeras filas son los tres mejores sistemas que participaron en el desafío FNC-1 (Talos (Baird y cols., 2017), Athene (Andreas Hanselowski y Caspelherr, 2017) y UCLMR (Riedel y cols., 2017)). Los resultados para cada una de las métricas de evaluación se calcularon utilizando las matrices de confusión y los resultados publicados (Riedel y cols., 2017) o puestos a disposición por los autores[8] [9].

[8] https://github.com/hanselowski/athene_system/ (consultado 6 de octubre de 2021).
[9] https://github.com/CiscoTalos/fnc1 (consultado 6 de octubre de 2021).

La cuarta fila corresponde al *Límite superior humano*, es el resultado de realizar la tarea de detección de posturas FNC-1 manualmente. Este límite superior fue definido por (Andreas Hanselowski y Caspelherr, 2017). Se utilizaron cinco anotadores humanos que etiquetaron manualmente 200 instancias aleatorias, obteniendo un acuerdo global entre anotadores de k de Fleiss de 0,686. Debido al hecho de que no hay un límite superior informado en los datos de FNC-1, también se consideran estos valores como referencia para fines de comparación.

A continuación, las filas quinta y sexta incluyen los resultados de enfoques recientes (Q. Zhang y cols., 2019; Dulhanty y cols., 2019) que también abordaron la tarea de detección de posturas de los titulares utilizando el conjunto de datos FNC-1, pero que no participaron en el desafío. Dado que no había códigos públicos disponibles, estos resultados se calcularon a partir de las matrices de confusión proporcionadas en los artículos científicos. Es importante aclarar que el trabajo desarrollado por (Dulhanty y cols., 2019) utiliza el modelo de lenguaje RoBERTa para crear un clasificador plano, lo que permitió una comparación directa entre la opción de utilizar clasificación plana o la arquitectura creada en el contexto de esta tarea específica.

La séptima y la novena fila muestran los resultados de dos experimentos diseñados para comparar el rendimiento de cada implementación de **HeadlineStanceChecker** con clasificadores planos, que también utilizan como entrada los resúmenes generados por **PLM Summarizer** y **TextRank Summarizer** así como las características de la primera y segunda implementación, respectivamente.

Las filas octava y décima muestran resultados utilizando íntegramente la propuesta de arquitectura en dos etapas (**HeadlineStanceChecker**), para ambas implementaciones. Tanto los clasificadores planos como las implementaciones de la arquitectura han utilizado únicamente los resúmenes automáticos creados a partir del contenido completo durante todo el proceso.

Como se puede apreciar en la Tabla 4.7, los sistemas basados en la arquitectura **HeadlineStanceChecker** utilizando las dos etapas y resúmenes automáticos en todo el proceso son competitivos comparados con otros que utilizan el contenido completo de la noticia. La reducción del contenido de la noticia implica una pérdida de rendimiento en los resultados obtenidos por los experimentos que utilizan resúmenes. Sin embargo, se obtienen resultados notablemente mejores que los participantes del FNC-1, el *Límite superior humano* y (Dulhanty y cols., 2019).

Tabla 4.7: Resultados del sistema HeadlineStanceChecker, comparación de rendimiento con otros sistemas en el corpus FNC-1.

	F_1 Score				$F_1\,m$	Acc.	Rel. Score
Sistemas	**Agree**	**Disagree**	**Discuss**	**Unrelated**			
1-*Talos*	53,90	3,54	76,00	99,40	58,21	89,08	82,02
2-*Athene*	48,70	15,12	78,00	99,60	60,40	89,48	82,00
3-*UCLMR*	47,94	11,44	74,70	98,90	58,30	88,46	81,72
4-*Límite superior humano*	58,80	66,70	76,50	99,70	75,40	–	85,90
5-*(Dulhanty y cols., 2019)*	73,76	55,26	85,53	99,12	78,42	93,71	90,00
6-*(Q. Zhang y cols., 2019)*	67,47	**81,30**	83,90	**99,73**	**83,10**	93,77	89,30
7-*Clasificador-plana-PLM*	70,34	53,42	85,30	99,41	77,12	93,64	89,80
8-*HeadlineStanceChecker-PLM*	72,34	62,53	**87,32**	99,38	80,39	**94,31**	**91,02**
9-*Clasificador-plana-TextRank*	71,64	53,31	85,25	99,29	77,37	93,58	89,92
10-*HeadlineStanceChecker-TextRank*	**74,22**	64,29	86,00	99,31	80,95	94,13	90,73

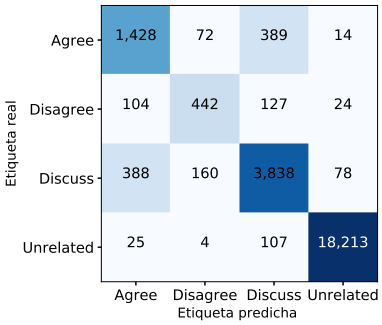
Teniendo en cuenta, la estrategia de divide y vencerás concretada en la arquitectura de detección en varias etapas (este caso particular en dos etapas) muestra beneficios dado que produce un mejor rendimiento cuando se utiliza en lugar de una sola etapa de clasificación (filas 5, 7 y 9).

La mejora más notable para *HeadlineStanceChecker-PLM* se logra en la categoría *discuss* sobre el resto de sistemas. El F_1 de esta clase mejora alrededor de 2% en comparación con el segundo mejor enfoque, *HeadlineStanceChecker-TextRank*, y 13% puntos sobre el sistema de rendimiento más bajo (Riedel y cols., 2017) en esta categoría. Por su parte *HeadlineStanceChecker-TextRank* obtiene los mejores resultados en la clase *agree* y el segundo mejor resultado en la clase *disagree* y *discuss*. En general, los sistemas basados en la arquitectura propuesta logran valores competitivos en todas las clases, *HeadlineStanceChecker-PLM* obtiene un valor $F_1 m$ final de 80,39% y *HeadlineStanceChecker-TextRank* de 80,95%, solo superado por el sistema propuesto en (Q. Zhang y cols., 2019), que aprovecha un número considerable de características externas más allá de la similitud para enriquecer el modelo neuronal. Cabe destacar que en términos de exactitud y puntuación relativa, una de las implementaciones propuestas, es decir, *HeadlineStanceChecker- PLM* obtiene el mejor resultado entre estos sistemas en ambos casos, logrando 94,31% y 91,02%, respectivamente.

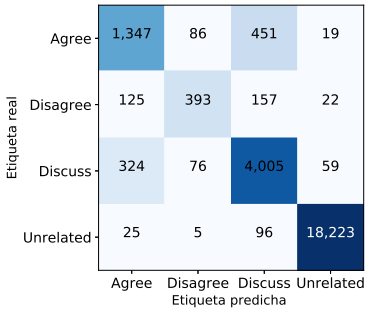
Centrándose en los resultados obtenidos por los participantes en la competición FNC-1 se observa que, cuando estos resultados se analizan de forma independiente para cada una de las clases, salvo la clasificación de titulares *unrelated* (cuyos resultados son cercanos al 100% en la métrica $F_1 m$), en el resto, los resultados son muy limitados. Los sistemas que participaron en la competencia tienen un rendimiento muy reducido especialmente en la detección de la postura *disagree*, mientras que la detección de *agree* ronda el 50% en la métrica $F_1 m$ y para *discuss* alrededor del 75% para el mejor sistema. Estos resultados fueron alcanzados usando técnicas más limitadas que los actuales modelos de lenguaje basados en DL. Fuera de la competencia, el desempeño aumenta en todas las categorías, siendo la categoría de *disagree* una de las más desafiantes de clasificar, en la que solo el enfoque propuesto en (Q. Zhang y cols., 2019) obtiene resultados sorprendentemente altos para esta categoría en comparación con los sistemas restantes.

Después de haber demostrado que **HeadlineStanceChecker** funciona adecuadamente detectando correctamente el 94,31% de las clases del conjunto de prueba, se presentan las matrices de confusión en la Figura 4.5 para proporcionar más detalles sobre el rendimiento real del sistema para cada clase.

A partir de las matrices se puede observar que por clases, los principales problemas de clasificación ocurren con la clase *disagree* que es la que menos ejemplos clasificados correctamente tiene. Los datos reflejan que el 18,2% y 22,5% del número total de posturas *disagree* se clasifican como *discuss*, mientras que el 14,9% y 17,9% son clasificadas como *agree*. Por otra parte, en el caso de la clase *agree* el 20,4% y 23,6% del número total de posturas son clasificadas como *discuss* y tan solo 3,7% y el 4,5% son clasificadas como *disagree*. Por último, el



(a) Matriz de confusión: *Arquitectura completa con TextRank Summarizer.*



(b) Matriz de confusión: *Arquitectura completa con PLM Summarizer.*

Figura 4.5: Matrices de la arquitectura completa.

8,6% y 7,2% del número total de posturas *discuss* se clasifican como *agree* y el 3,5% y 1,7% son clasificadas como *disagree*. La clasificación incorrecta de estas clases afecta el rendimiento de la tarea de clasificación.

4.5.4 Comparación de los resúmenes contra el contenido completo de los artículos

Se realizó un análisis adicional con el fin de probar la conveniencia de utilizar el resumen o el contenido completo como entrada para cada componente de la arquitectura. Se diseñó un experimento que permite comparar los resultados para ambas entradas.

Como se explicó en la sección 2.6.2 los modelos basados en DL presentan algunas limitaciones para procesar largas secuencias de entradas. El modelo de lenguaje RoBERTa, utilizado para crear los clasificadores, tiene la limitación de procesar como máximo 512 *tokens* como secuencia de entrada. En los experimentos presentados en las secciones anteriores se ha utilizado la secuencia máxima. Las entradas que sobrepasan este tamaño máximo, son truncadas automáticamente, utilizándose solamente la primera parte del contenido de la noticia. Analizando el truncamiento llevado a cabo por este modelo, se ha concluido que conceptualmente es un resumen extractivo de tipo *Lead* (consultar sección 3.4.1).

Teniendo en cuenta la teoría de la pirámide invertida que plantea que la información más importante de una noticia debe encontrarse en sus primeras secciones, se puede concluir que el truncamiento que realiza el modelo por la limitación de su arquitectura interna debería ser válido para analizar noticias.

Sin embargo, según investigaciones recientes que evidencian la baja calidad en la estructura y el relato noticioso (Mompart, Lozano, y Sampío, 2015; Urban y Schweiger, 2014), unido a que el reconocimiento de la teoría de la pirámide invertida no excluye que el resto de secciones de la noticia disponga de infor-

mación importante, hace oportuno al menos probar algún enfoque de resumen automático. Estos enfoques pueden reducir la información a procesar, pero de una manera más precisa.

El modelo de lenguaje escogido y la mayoría de los modelos basados en arquitectura Transformer tienen la misma limitación al procesar textos extensos. En esencia, el titular y el contenido de la noticia o el resumen son concatenados y pasados al modelo RoBERTa. Si el texto supera los 512 *tokens* se trunca y desecha el resto que excede ese tamaño. Es importe explicar que los tokenizadores que se utilizan en los modelos basados en arquitectura Transformer tokenizan a nivel de subpalabras, por lo que 512 *tokens* en este caso representan menos de 512 palabras.

En (Dulhanty y cols., 2019) se realiza un análisis de los resultados de clasificación en función de la longitud del contenido de la noticia, mostrando que para los ejemplos en los que la secuencia de entrada es mayor a 512 *tokens*, la exactitud de la clasificación disminuye considerablemente con respecto a secuencias más pequeñas. En este contexto, se trazó una hipótesis parcial sustentada en que la aplicación del resumen al texto antes de la clasificación implica una mejora en los resultados para textos largos (en este trabajo mayores de 512 *tokens* que es la limitación del modelo RoBERTa). Para demostrarlo, primero creamos dos subconjuntos del corpus FNC-1 de acuerdo con la longitud de la noticia: *Subconjunto A* (con los contenidos de noticias menores de 512 *tokens*) y *Subconjunto B* (con los contenidos de noticias mayores o iguales de 512 *tokens*). Las tablas 4.8 y 4.9 muestran la distribución de clases para ambos subconjuntos. Como se puede apreciar en las tablas, en este conjunto de datos existen muchos más ejemplos con contenidos de noticias menores de 512 *tokens* (*Subconjunto A*) que de mayor longitud (*Subconjunto B*), siendo la relación de 2 a 1, aproximadamente.

Tabla 4.8: Distribución de etiquetas para el *Subconjunto A*, menores de 512 *tokens*.

	Agree	**Disagree**	**Discuss**	**Unrelated**	**Total**
Entrenamiento	2566	526	5373	23659	32124
Prueba	1258	376	3205	12848	17687
Total	3824	902	8578	36507	49811

Se utilizan los subconjuntos creados para entrenar y probar sobre la arquitectura **HeadlineStanceChecker**. Los resultados en la tabla 4.10 muestran que para noticias largas (*Subconjunto B*), el sistema funciona mejor con resúmenes como entrada que truncando el texto del artículo completo. Esto podría suceder porque reducir la entrada simplemente quedándose con los primeros párrafos del documento (*Lead Summarizer)*) da como resultado la pérdida de información relevante, mientras que al utilizar un resumen más complejo como

TextRank Summarizer y **PLM Summarizer** prevalece información de carácter relevante.

Tabla 4.9: Distribución de etiquetas para el *Subconjunto B*, mayores o iguales de 512 *tokens*.

	Agree	**Disagree**	**Discuss**	**Unrelated**	**Total**
Entrenamiento	1112	314	3536	12886	17848
Prueba	645	321	1259	5501	15452
Total	1757	635	4795	18387	25574

Tabla 4.10: Resultados *HeadlineStanceChecker* para *Subconjunto B* con diferentes entradas: contenido de noticias y resumen de noticias.

		F_1 **Score**			$F_1\,m$
Entrada	**Agree**	**Disagree**	**Discuss**	**Unrelated**	
Contenido de noticias	54,45	12,69	78,97	**99,52**	61,40
Resumen de noticias	**59,61**	**28,06**	**80,85**	99,32	**66,96**

De manera similar, los resultados del *Subconjunto A* (noticias más breves) se muestran en la Tabla 4.11. En este caso, los resultados son mejores cuando se utiliza el contenido completo, lo que podría indicar que toda la información necesaria para una clasificación adecuada está presente al considerar el texto completo (un escenario inviable con textos más largos).

Tabla 4.11: Resultados *HeadlineStanceChecker* para *Subconjunto A* con diferentes entradas: contenido de noticias y resumen de noticias.

		F_1 **Score**			$F_1\,m$
Entrada	**Agree**	**Disagree**	**Discuss**	**Unrelated**	
Contenido de noticias	**78,64**	**69,38**	**89,81**	**99,59**	**84,35**
Resumen de noticias	74,17	58,91	87,69	99,36	80,03

No existen reglas explícitas que determinen cuál debe ser la longitud de un artículo de noticias, pero en cambio hay cierta evidencia que respalda que las noticias tienden a ser mayores a 512 *tokens*. En la Tabla 4.12 se han recopilado estadísticas de los conjuntos de datos de noticias más populares que se utilizan en tareas de PLN. En conjunto, los cuatro corpus contienen más de 2 millones de artículos de diferentes fuentes, con una extensión media muy superior a 512 *tokens* en todos los casos. Con el objetivo de comparar con la extensión de los

contenidos del conjunto de datos FNC-1 se utiliza el tokenizador del modelo RoBERTa utilizado.

Tabla 4.12: Estadísticas de extensión media en palabras para corpus de noticias.

Corpus	**Ejemplos**	**Extensión media**
CNN (Hermann y cols., 2015)	92 K	760,50
DailyMail (Hermann y cols., 2015)	310 K	653,33
NY times (Sandhaus, 2008)	650 K	800,04
Newsroom (Grusky, Naaman, y Artzi, 2018)	1,210 M	770,09
Total	2,260 M	745,83

En cualquier caso la extensión de las noticias dependerá de la fuente que las aporta. En la mayoría de los casos, el resumen de noticias podría ser una estrategia correcta.

Por último, se realiza una comparación del rendimiento de la arquitectura de detección de titulares utilizando los contenidos de noticias completas.

Experimentación de la arquitectura con el contenido completo de la noticia

Se diseña un experimento para evaluar el comportamiento de la arquitectura de detección cuando se utiliza como entrada el contenido de la noticia, en lugar de resúmenes. Se utiliza la misma configuración de la implementación de la sección 4.3.2, utilizándose las mismas características externas (**similitud de coseno con vectores** TF-IDF, **coeficiente de superposición** y **puntuación de parentesco del** PLM), calculadas sobre la relación titular-contenido. La tabla 4.13 se refleja el comportamiento de la arquitectura procesando el contenido completo y se compara con los resultados obtenidos por la mejor configuración de la arquitectura que utiliza resúmenes.

Tabla 4.13: Resultados de la arquitectura HeadlineStanceChecker con el contenido de la noticia.

			F_1 **Score**		$F_1\,m$
Experimentos	**Agree**	**Disagree**	**Discuss**	**Unrelated**	
HeadlineStanceChecker-TextRank	74,22	64,29	86,00	99,31	80,95
HeadlineStanceChecker-Contenido	**78,64**	**69,67**	**88,69**	**99,45**	**84,11**

Los resultados alcanzados por el experimento utilizando el contenido de la noticia (*HeadlineStanceChecker-Contenido*) superan los obtenidos por las im-

plementaciones de la arquitectura utilizando resúmenes. La estructura del conjunto de datos FNC-1, con mayoría de ejemplos de noticias de longitud menor que 512 *tokens*, permite la utilización del contenido completo de la noticia que obviamente contiene más información que un resumen y por tanto en estos casos la aplicación de resúmenes resultaría irrelevante. El experimento *HeadlineStanceChecker-Contenido* mejora los resultados en todas las clases, siendo la clase *disagree* la que más ventaja muestra. El rendimiento de la métrica $F_1 m$ aumenta en poco más de 3%. La utilización del contenido completo de la noticia obviamente contiene más información que un resumen y por tanto en estos casos la aplicación de resúmenes resultaría irrelevante.

Acorde con la experimentación realizada en el capítulo 3 y la de esta sección, se presume que la utilización de un corpus con noticias en su mayoría compuestas por más de 512 *tokens* sería más representativo de un contexto real de aplicación, donde probablemente se obtendrían indicadores de desempeño superiores utilizando resúmenes automáticos.

Se considera que la utilización de resúmenes en el contexto de procesamiento de textos extensos representa soluciones más genéricas, evitando la dependencia de modelos basados en DL que puedan cortar o no en un número determinado de *token*. Por tanto, esto permitiría trabajar siempre con información relevante.

4.5.5 Modelo de lenguaje para procesar textos extensos

En esta sección se analiza el comportamiento de un modelo de lenguaje que es capaz de procesar textos extensos como entrada. En la sección 2.6.2 se analizó el problema de los modelos basados en DL para procesar este tipo de texto, el modelo RoBERTa, utilizado en este capítulo, no queda exento de este problema. En los experimentos realizados se han utilizado técnicas de resumen automático con el fin de mitigar este inconveniente. Sin embargo, como se mencionó en la sección 2.6.2 existen modelos de lenguajes como Longformer que son capaces de procesar estas entradas.

Se evalúa el rendimiento de Longformer en dos experimentos sobre el conjunto de datos FNC-1. Primeramente se entrena un clasificador plano y a continuación se entrena una instancia de la arquitectura **HeadlineStanceChecker** con dos etapas. Estos experimentos son similares a los de secciones anteriores, basados en el contenido completo de la noticia, pero sin la utilización de características adicionales. En la tabla 4.14 se muestran los resultados de los experimentos realizados en esta sección.

El *Clasificador-plano-Longformer* alcanza resultados notables, mejorando los obtenidos por un clasificador plano que utiliza el modelo RoBERTa (Dulhanty y cols., 2019) y la arquitectura **HeadlineStanceChecker** utilizando las técnicas de resumen automático **TextRank Summarizer** y **PLM Summarizer**. Por otra parte, el experimento *HeadlineStanceChecker-Longformer* que utiliza la arquitectura de detección creada y el modelo de lenguaje Longformer obtiene 83,46%

en la métrica $F_1\,m$, mejorando en más de un punto lo obtenido por el experimento que no la utiliza.

Tabla 4.14: Resultados del modelo Longformer: puntuación F_1 por clase y $F_1\,m$.

	F_1 Score				$F_1\,m$
Experimentos	**Agree**	**Disagree**	**Discuss**	**Unrelated**	
Clasificador-plano-Longformer	77,76	64,13	87,69	**99,60**	82,30
HeadlineStanceChecker-Longformer	**79,54**	**65,86**	**88,99**	99,44	**83,46**

Con estos experimentos nuevamente queda demostrado la validez de la arquitectura de detección, la que es capaz de explotar la relación jerárquica entre clases. El modelo Longformer, como mostraban los trabajos consultados logra mejorar en algunos experimentos los resultados de RoBERTa. Sin embargo, es importante aclarar que el tiempo de ejecución del entrenamiento y prueba es mucho más elevado que el modelo RoBERTa, llegando en algunos experimentos a consumir nueve veces el tiempo requerido por RoBERTa.

Longformer logra procesar textos muy extensos, hasta 4096 *tokens*, lo que puede permitir analizar completamente la mayoría de las noticias con una alta precisión. Es evidente que Longformer y los modelos similares a este constituye una solución novedosa y que es necesario evaluar su evolución futura.

4.6 Conclusiones

En este capítulo se ha presentado la arquitectura **HeadlineStanceChecker** con el objetivo de proponer una solución novedosa al problema de la detección de titulares engañosos.

Se ha demostrado que **HeadlineStanceChecker** es una arquitectura eficaz para detectar información errónea en las noticias, específicamente cuando un titular debe compararse con el contenido de la noticia. La novedad de este enfoque se basa en dos premisas clave: i) la adopción de una estrategia de divide y vencerás, abordando así el problema de clasificación de posturas mediante una arquitectura neuronal de dos etapas; y ii) el uso de un resumen semántico extractivo como sustituto del contenido completo de la noticia, además de características externas de similitud que ayudan a determinar la relación del titular con el artículo noticioso.

Para demostrar la idoneidad de **HeadlineStanceChecker**, se llevaron a cabo dos implementaciones variando módulos internos de esta. Se realizaron diferentes experimentos en el contexto de una tarea existente FNC-1, donde la postura de un titular con respecto a un contenido de noticia tenía que clasificarse en

una de las siguientes clases: *unrelated*, *agree*, *disagree* y *discuss*. Los experimentos consistieron en validar cada una de las etapas de clasificación propuestas de forma aislada junto con el enfoque completo, así como una comparación con respecto al estado del arte en esta tarea.

Adicionalmente, se realizaron experimentos en la etapa de detección de posturas con otro corpus (conjunto de datos Emergent), para verificar la generalidad de este enfoque. Los resultados obtenidos por la arquitectura fueron muy competitivos en comparación con otros sistemas de referencia del estado del arte, obteniendo 94,31% de exactitud, así como el resultado más alto en puntuación relativa FNC-1 en comparación con el estado del arte (91,02%).

La arquitectura se define para que sea flexible y adaptable a tareas similares donde la división de clases pueda tener un efecto positivo en los resultados finales de clasificación. Una característica propia es que la división en módulos más concretos permite tener pequeñas funcionalidades que pueden ser reutilizadas por varias etapas y a su vez sustituidas con relativa facilidad sin afectar el funcionamiento de las etapas.

El conjunto de datos utilizado (FNC-1) se encuentra extremadamente desbalanceado, lo que provoca que los sistemas de clasificación sesguen el aprendizaje hacía las clases mayoritarias (en este caso la clase *unrelated*), pero sean menos precisos cuando se trata de las clases restantes. Aun así, los resultados obtenidos por **HeadlineStanceChecker** para las diferentes categorías con menos ejemplos (*agree, disagree* y *discuss*) son prometedores, lo que indica que el enfoque elegido es apropiado para la tarea. Además, el modelo de lenguaje RoBERTa utilizado en ambas implementaciones exhibe un alto rendimiento para esta tarea.

Se analizó profundamente el comportamiento de la arquitectura de detección en función de la longitud del contenido de la noticia. Se encontró una clara disminución de la exactitud cuando las noticias sobrepasaban los 512 *tokens* permitidos en la entrada del modelo RoBERTa, mejorándose estos resultados con la utilización de técnicas de resúmenes aplicadas con el fin de reducir el tamaño de la entrada. En el conjunto de datos utilizado se encontró, que cerca de dos tercios (2/3) de los contenidos de noticias son menores que 512 *tokens*, aspecto que difiere notablemente de la mayoría de noticias que circulan en medios digitales.

Por tanto, aunque para este formato de corpus con 2/3 de las noticias menores de 512 *tokens* no se aprecia la mejora de utilizar resúmenes, puesto que las noticias completas son pequeñas, se refuerza la hipótesis, basado en la experimentación de la sección 4.5.4, que en el contexto real, donde la longitud de las noticias es más elevada, es más plausible utilizar la información relevante que puede proporcionar un resumen automático.

Finalmente, como un objetivo futuro que contribuye a investigar el problema de la detección de información engañosa, se espera aplicar **HeadlineStanceChecker** a la detección de titulares engañosos en idioma español e integrar esta solución a un escenario del mundo real para detectar la introducción de errores y desinformación en los titulares en perjuicio de los lectores.

Esta contribución aporta a la investigación actual en el campo, nuevas estrategias de aprendizaje y análisis tradicionales de similitud, dado que podría ayudar a combatir las noticias falsas en línea, un problema social que requiere una acción conjunta.

CAPÍTULO 5

Titulares engañosos aplicando detección de contradicciones

Los resultados obtenidos con el uso de una arquitectura de detección de titulares engañosos son realmente prometedores. La propuesta inicial se orientaba a crear un sistema de detección de titulares engañosos para aplicar al idioma español, uno de los idiomas más hablados en el mundo pero que carece de este tipo de soluciones. Sin embargo, el conjunto de datos utilizado (FNC-1) para la creación del sistema de detección solo está disponible para el idioma inglés. Además la distribución de etiquetas de este corpus se encuentra extremadamente desbalanceanda lo que ocasionó problemas para detectar las clases minoritarias.

Una solución general a la creación de sistemas de detección para varios idiomas podría ser utilizar modelos multilingües que permitan entrenar en idioma inglés (el idioma en el que se encuentra el corpus FNC-1) y predecir para otros idiomas. Sin embargo, aún faltaría generar ejemplos en otros idiomas para validar el rendimiento del sistema, sumado a los problemas que presenta el corpus con el que se cuenta para la tarea. En este contexto tiene sentido desarrollar un conjunto de datos en idioma español para la detección de titulares engañosos.

La detección de titulares engañosos implica realizar un análisis semántico de la relación entre el titular y el contenido de la noticia, a diferencia de los titulares ciberanzuelos (*clickbaits*) que exhiben estructuras bien definidas por lo que soluciones a nivel sintáctico pueden ser efectivas. Analizando la perspectiva de soluciones, estas se podrían reducir a detectar las relaciones semánticas entre textos, como son: la detección de implicación textual y la detección de contradicciones.

En la sección 2.3 se analizó en profundidad la tarea de detección de titulares engañosos, mostrando la estrecha relación que mantiene con los titulares incongruentes. La principal diferencia parece radicar en que los titulares incon-

gruentes implican una exageración sutil o una tergiversación de los hechos y los engañosos se contradicen abiertamente con el contenido de la noticia. Lo cierto es que en ambos casos se necesita detectar contradicciones entre el titular y el contenido de la noticia. En este capítulo se propone un conjunto de datos para la detección de titulares engañosos con base en la detección de las contradicciones que aparecen en el texto.

Es frecuente encontrar información diversa sobre un mismo hecho en distintos medios, a veces sesgada por un determinado espectro político, religioso, económico o social. Por ejemplo, se presenta a continuación un caso real de contradicción entre tres medios de comunicación españoles sobre la misma información[1]:

1. ABC[2]: "Sánchez retiró el helicóptero que buscaba al desaparecido de Mallorca para usarlo él"

2. 20 Minutos[3]: "El Gobierno desmiente que Pedro Sánchez utilizara un helicóptero que buscaba a un desaparecido en Mallorca"

3. El Mundo[4]: "De cinco helicópteros posibles Sánchez utilizó el único destinado al rescate"

Los titulares de noticias de los medios ABC y El Mundo son contradictorios con respecto al publicado por 20 Minutos. Por una parte, se afirma la utilización de un helicóptero que estaba destinado a una búsqueda y el otro medio publica que la información fue desmentida por el gobierno. La difusión de titulares como este puede causar polarización de los lectores y problemas para determinadas personas y entidades, con posibles impactos económicos, de reputación y de salud, entre otros. Por tanto, se considera de vital importancia alertar a los usuarios de estas contradicciones antes que se saquen conclusiones precipitadas, en muchos casos sin haber leído la noticia completa u otros medios que publiquen noticias similares.

En el contexto actual de desinformación, la detección automática de contradicciones contribuiría a identificar información poco confiable, ya que encontrar contradicciones entre dos piezas de información relacionadas con el mismo hecho sería un indicio de que al menos uno de los dos contiene elementos demostrables de falsedad o incluso es errónea su información; y por tanto pondría en entredicho la confiabilidad del contenido en cuestión. Teniendo en cuenta el escenario de la relación entre titulares y contenidos de noticias, encontrar contradicciones también es una tarea crucial en la lucha contra la propagación perjudicial de la desinformación.

[1] La fecha de publicación de los titulares de noticias extraídas de ABC, 20 Minutos y El Mundo son el 25, 26 y 28 de enero de 2020.

[2] `https://bit.ly/3sDOu0P` (consultado el 20 de enero de 2022).

[3] `https://bit.ly/3ur3phe` (consultado el 20 de enero de 2022).

[4] `https://bit.ly/3gt5e5a` (consultado el 20 de enero de 2022).

Considerando el contexto que se ha delineado y con el principal objetivo de cubrir la carencia de conjuntos de datos para las tareas de detección de contradicciones y detección de titulares engañosos en idioma español, se ha decidido crear un conjunto de datos que anote la relación semántica entre el titular y el contenido de una noticia.

5.1 Conjunto de datos de titulares engañosos aplicando detección de contradicciones

Similar al conjunto de datos FNC-1, el corpus propuesto anota la relación semántica entre titulares y contenidos de noticias. A diferencia del conjunto de datos FNC-1 donde se definía la relación semántica en términos de la postura entre las dos piezas de textos, en este conjunto de datos la relación se define en términos de la existencia de contradicciones.

Retomando la conexión realizada en la sección 5.1 entre las tareas de detección de titulares engañosos y la detección de contradicciones, donde se partía de la definición de contradicción como una tarea que podía ser transversal a cualquier otra que implique detectar una contradicción entre textos, se propone una definición formal de la relación semántica entre textos en términos de las contradicciones.

Siguiendo este enfoque, se puede definir una relación semántica en términos de contradicciones si se define una declaración como $s = (i, f, t)$, donde i se refiere a la información proporcionada sobre el hecho f que ocurre en el momento t. Se pueden clasificar dos pares de texto como:

- **Información compatible** (*compatible*): dos piezas de textos, s_1 y s_2, son consideradas compatibles si, dado $s_1 = (i_1, f_1, t_1)$ y $s_2 = (i_2, f_2, t_2)$, se cumple la siguiente expresión:

$$(i_1 \cong i_2) \wedge (f_1 \cong f_2) \wedge (t_1 \cong t_2) \tag{5.1}$$

- **Información contradictoria** (*contradiction*): dos piezas de textos, s_1 y s_2, son consideradas contradictorias si, dado $s_1 = (i_1, f_1, t_1)$ y $s_2 = (i_2, f_2, t_2)$, se cumple la siguiente expresión:

$$(i_1 \ncong i_2) \wedge (f_1 \cong f_2) \wedge (t_1 \cong t_2) \tag{5.2}$$

- **Información no relacionada** (*unrelated*): dos piezas de textos, s_1 y s_2, son consideradas no relacionadas si, dado $s_1 = (i_1, f_1, t_1)$ y $s_2 = (i_2, f_2, t_2)$, se cumple la siguiente expresión:

$$f_1 \neq f_2 \tag{5.3}$$

Así, una noticia se clasifica como *contradiction* cuando se da el mismo hecho (se considera que el mismo hecho en dos noticias diferentes podría expresarse con diferentes menciones de eventos, con cierta equivalencia semántica) dentro del mismo marco temporal[5], sin embargo, la información relacionada con el hecho es incongruente en las dos noticias que se están considerando, expresión 5.2.

En la práctica no se suele encontrar referencias a la variable tiempo en la verificación de relaciones semánticas entre noticias. Para ello, se hace una abstracción previendo que si se está comparando dos textos en búsqueda de contradicciones es porque ya se ha verificado que pertenecen al mismo marco temporal.

En el corpus creado se anota la relación semántica entre el titular y el contenido de una noticia en *compatible*, *contradiction* o *unrelated*. Este conjunto de datos podrá ser utilizado para crear sistemas de detección de titulares engañosos, centrados en las contradicciones existentes entre el titular y el contenido de la noticia.

Además, en el caso de que la relación sea de tipo *contradiction*, se sigue la definición propuesta por De Marneffe y cols. y se incluye una anotación más precisa para el tipo de contradicción en *negación*, *antónimo*, *numérica/fecha*, *fáctica* o *estructura*. Se parte de dos frases relacionadas que exhiben contradicción para definirlas:

1. *negación*: el evento principal en alguna de las frases analizadas se encuentra negado, logrando que la frase cambie totalmente su significado. Se utilizan marcas de negación (no, ninguna, nunca, etc).

2. *antónimo*: los dos eventos principales de cada frase son antónimos, convirtiendo a dos frases semánticamente compatibles en contradictorias.

3. *numérica/fecha*: existen diferencias entre partes de las frases que expresen datos numéricos o fechas, convirtiendo las frases en contradictorias.

4. *estructura*: la estructura de una de las frases no es compatible con la otra. La entidad nombrada que realiza una acción es diferente a la encontrada en la otra frase o se encuentran intercambiadas entidades nombradas en una frase.

5. *fáctica*: una de las frases no presenta hechos concretos, utiliza palabras modales, hace suposiciones sobre hechos ocurridos.

Esta anotación podría ser beneficiosa para generar la explicación de las decisiones tomadas por futuros sistemas desarrollados sobre este corpus.

Una vez explicada la necesidad de desarrollo de este conjunto de datos y las relaciones existentes entre el titular y el contenido de noticias, a continuación se explica detalladamente el proceso de construcción del conjunto de datos.

[5]Se refiere a un rango de tiempo determinado.

5.1.1 Proceso de construcción del corpus

Con el objetivo de explicar el proceso de construcción del conjunto de datos se definieron tres fases principales: planificación, primera fase de anotación y segunda fase de anotación. A continuación se explica cada una de las fases propuestas.

Planificación

El primer paso de esta fase es escoger una fuente de datos confiable para extraer las noticias. En este caso se utilizó la agencia de noticias EFE[6] debido a que es una agencia de noticias reconocida por exhibir neutralidad en sus publicaciones. Se extraen el título, el contenido y la fecha de la noticia. Las noticias extraídas pertenecen a los dominios político y económico, asumiendo que el titular y el contenido de las noticias son *compatible*, aunque se verifica posteriormente su relación en la fase de anotación 1.

Se implementa un *web crawler* utilizando la biblioteca BeautifulSoup[7] de python, descargándose un total de 25945 noticias entre enero del 2019 y marzo del 2021.

Además, se desarrolló una guía de anotación que explica en detalle el procedimiento que se debe seguir para modificar los titulares y hacerlos contradictorios en correspondencia con la definición de contradicciones presentada por (De Marneffe y cols., 2008). Esta guía se puede consultar en el siguiente enlace de Zenodo[8] y en el apéndice B.

Primera fase de anotación

La primera fase de anotación tiene como objetivo desarrollar una versión preliminar del conjunto de datos. Esta primera versión tendría un costo de anotación no muy elevado debido a que se anotaría una fracción de las noticias descargadas. Además podría permitir realizar experimentos para identificar como se comportaría un sistema de detección de titulares engañosos aplicando detección de contradicciones en idioma español.

En esta primera fase de anotación se escoge un subconjunto de las noticias descargadas (7403) para desarrollar una primera versión del conjunto de datos. De las 7403 noticias se reservan 2499 para la generación de ejemplos de tipo *unrelated*. Esta fase se divide en tres tareas principales: **Modificar manualmente el titular de la noticia**, **Clasificar la relación entre el titular y el contenido de la noticia** y **Mezclar aleatoriamente titulares y contenidos de noticias**. A continuación se explica cada una de las tareas.

[6] https://www.efe.com/efe/espana/1 (consultado el 15 de enero de 2022).

[7] Documentación disponible en https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (consultada 15 de enero de 2022).

[8] https://zenodo.org/badge/latestdoi/344923645 (consultado el 20 de enero de 2022).

1. **Modificar manualmente el titular de la noticia**: el objetivo de esta tarea es hacer que el titular de la noticia sea contradictorio con el contenido mediante la inclusión de modificaciones simples en su semántica. Los cambios en el titular junto con algunos ejemplos se muestran a continuación:

    - *negación (neg)*: Esta alteración consiste en negar el titular de la noticia, incluyendo un indicador de tiempo en una posición específica de la oración.
        (a) Titular original: "El comité de empresa **debatirá** mañana la propuesta final de Alcoa"
        (b) Titular modificado: "El comité de empresa **no debatirá** mañana la propuesta final de Alcoa"
    - *antónimo (ant)*: Esta transformación consiste en reemplazar el verbo del evento principal del titular por un antónimo.
        (a) Titular original: "El Gobierno se compromete a **subir** los salarios a los empleados públicos tras los comicios"
        (b) Titular modificado: "El Gobierno se compromete a **bajar** los salarios a los empleados públicos tras los comicios"
    - *numérica/fecha (num)*: Esta modificación consiste en cambiar los números, fechas que aparecen en el título.
        (a) Titular original: "La economía británica ha crecido un **3%** menos por el brexit, según S&P"
        (b) Titular modificado: "La economía británica ha crecido un **5%** menos por el brexit, según S&P"
    - *estructura (str)*: Esta modificación consiste en cambiar la posición de una entidad nombrada por otra o sustituir entidades nombradas en la oración.
        (a) Titular original: "**Arvind Krishna** sustituirá a **Ginni Rometty** como consejero delegado de IBM"
        (b) Titular modificado: "**Ginni Rometty** sustituirá a **Arvind Krishna** como consejero delegado de IBM"
    - *fáctica (fac)*: Esta transformación consiste en reemplazar el verbo del evento principal por una construcción verbal no factual o viceversa.
        (a) Titular original: "Isuzu y Volvo **pactan crear** una alianza estratégica en camiones pesados"
        (b) Titular modificado: "Isuzu y Volvo **crean** una alianza estratégica en camiones pesados"

    Estas alteraciones cambian el significado semántico de la oración, haciéndola contradictoria con el titular original y el contenido. El proceso de anotación fue realizado por dos anotadores independientes que fueron entrenados por un anotador experto.

2. **Clasificar la relación entre el titular y el contenido de la noticia**: La relación semántica entre el titular y el contenido se anotó en dos fases. La primera fase consistió en anotar la información en *compatible* o *contradiction*. En la segunda fase, en el caso de haberse anotado la relación titular-contenido como *contradiction*, también se anotó el tipo de contradicción (*negación*, *antónimo*, *numérica/fecha* o *estructura*). Esta tarea involucró a cuatro anotadores entrenados para detectar relaciones semánticas entre pares de textos.

3. **Mezclar aleatoriamente el titular y el contenido de las noticias**: Las noticias reservadas al inicio de esta fase se utilizaron para generar ejemplos *unrelated*. El titular se separó del contenido correspondiente y todos los titulares se mezclaron al azar con los contenidos. En el proceso se verificó que el titular no se mezclara con el contenido que le corresponde. Este paso se realizó automáticamente sin la intervención de los anotadores.

Como se puede apreciar en la explicación de las tareas de esta fase, en las dos primeras fue necesaria la intervención de anotadores humanos. Por el contrario, la tercera tarea se realizó automáticamente. En esta primera fase de anotación no se anota el tipo de contradicción *fáctica* debido a que es una contradicción que requiere más esfuerzo para modificar los titulares de las noticias.

Segunda fase de anotación

La segunda fase de anotación tiene como objetivo aumentar la cantidad de ejemplos anotándolos de forma automática. Se planifica la cantidad de ejemplos a anotar por tipos de contradicción con el fin de maximizar ejemplos de contradicciones más complejas como la de *estructura* y la *fáctica*.

Las noticias que no fueron utilizadas en la primera fase de anotación (18542) se utilizaron en esta fase para aumentar el tamaño del conjunto de datos. La segunda fase de anotación se lleva a cabo con posterioridad a la evaluación y experimentación sobre el conjunto de datos obtenido en la primera fase, de aquí que se tomen decisiones al respecto, como la eliminación de la tarea **Clasificar la relación entre el titular y el contenido de la noticia**. Estos aspectos serán explicados en la validación de la primera fase. Esta fase consta de tres tareas: **Modificar automáticamente el titular de la noticia**, **Modificar manualmente el titular de la noticia** y la de **Mezclar aleatoriamente el titular y el contenido de las noticias**.

1. **Modificar automáticamente el titular de la noticia**: En esta tarea se crea un mecanismo automático de modificación de los titulares para inducir los tipos de contradicción sin intervención de anotadores humanos.

   Basado en las experiencias adquiridas de la anotación manual llevada a cabo en la fase anterior, se crea un *pipeline* que induce tres tipos de con-

tradicciones (*antónimo, numérica* o *estructura*). El *pipeline* fue creado haciendo uso de la biblioteca Spacy[9], especializada en PLN.

Cada titular es preprocesado por el *pipeline*. La figura 5.1 muestra los componentes internos del *pipeline*.

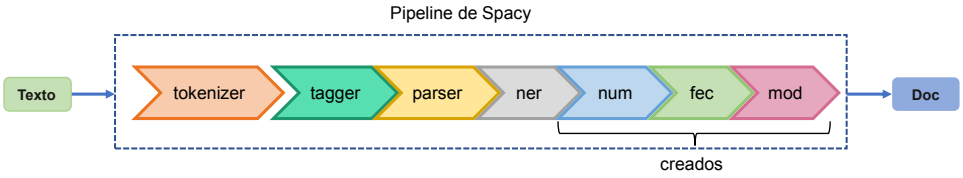


Figura 5.1: Pipeline creado utilizando la biblioteca Spacy[10].

Spacy permite crear *pipeline* de una forma sencilla, reutilizando componentes disponibles y desarrollando propios para ser integrados. A continuación se explican cada uno de los componentes utilizados y creados:

(a) *tokenizer*: segmenta el texto en *tokens*.

(b) *tagger*: asigna las etiquetas del discurso.

(c) *parser*: realiza un análisis de dependencias entre *tokens*.

(d) *ner*: detecta las entidades nombradas.

(e) *num*: detecta las formas de expresar un número (ordinal, cardinal y dígitos).

(f) *fec*: detecta las fechas incluidas en un texto (días, meses, años, etc.).

(g) *mod*: modifica los titulares haciendo uso de la anotación de los componentes previos, para inducir tres tipos de contradicción: *numérica/fecha, antónimo* y de *estructura*. Este componente analiza la anotación y prioriza realizar una anotación u otra en función de la cantidad de ejemplos anotados en la fase previa. El orden de prioridad es *estructura, numérica* y *antónimo*.

Los tres últimos componentes del *pipeline* han sido creados en esta investigación para llevar a cabo la modificación automática de los titulares.

2. **Modificar manualmente el titular de la noticia**: Esta tarea tiene el mismo objetivo que el de la fase anterior, para anotar el tipo de contradicción *fáctica* debido a la complejidad que supuso incluirla en el *pipeline* automático de inducción de contradicciones. Adicionalmente se modificaron algunos titulares introduciendo la contradicción de *negación* que en pruebas experimentales generaba ejemplos inconsistentes en la versión automática de la modificación de titulares.

[9] Documentación disponible en `https://spacy.io/` (consultada 15 de enero de 2022).

[10] Basado en la figura tomada de `https://spacy.io/usage/processing-pipelines/` (consultado el 15 de enero de 2022).

3. **Mezclar aleatoriamente el titular y el contenido de las noticias**: Esta tarea coincide íntegramente con la descrita en la primera fase de anotación. En esta ocasión se utilizan 3610 pares titular-contenido para generar los ejemplos *unrelated*.

5.2 Conjunto de datos para la detección de titulares engañosos

El conjunto de datos se desarrolló en dos fases de anotación por lo que se cuenta con dos versiones del mismo. En la primera fase se anotan cuatro tipos de contradicciones (*negación*, *antónimo*, *numérica* y *estructura*) y en la segunda fase se adiciona la contradicción *fáctica*.

5.2.1 Primera versión del conjunto de datos

La primera versión del corpus desarrollado consta de 7403 noticias, de los cuales 2431 han sido anotados como *compatible*, 2473 como *contradiction* y 2499 como *unrelated*. Esto representa un conjunto de datos equilibrado con tres elementos de clasificación principales. Se divide el conjunto de datos en particiones de entrenamiento y prueba. En la tabla 5.1 se muestra la distribución para cada partición.

Tabla 5.1: Distribución de clases en cada partición de la primera versión del conjunto de datos.

Particiones	Compatible	contradiction	unrelated	Total
Entrenamiento	1703	1733	1755	5191
Prueba	728	740	744	2212
Total	2431	2473	2499	7403

Las particiones de entrenamiento y prueba se crearon fraccionando el conjunto de datos en un 70% para entrenamiento y 30% para prueba.

Como se puede ver en la tabla 5.2, el conjunto de datos contiene ejemplos de cada tipo de contradicción. Sin embargo, teniendo en cuenta los tipos de contradicciones existe una diferencia marcada entre la cantidad de ejemplos de la clase *str* y el resto. Los anotadores tenían total libertad para decidir qué tipo de contradicción inducían en los titulares, lo que claramente ha ocasionado que se anoten pocos ejemplos del tipo de contradicción *str*.

Validación de la primera versión de la anotación

Debido a las particularidades del proceso de anotación del conjunto de datos, fue necesario validar las tareas **Modificar manualmente el titular de la no-**

Tabla 5.2: Distribución por tipos de contradicciones en la primera versión del conjunto de datos.

Particiones	**Neg**	**Ant**	**Num**	**Str**	**Total**
Entrenamiento	674	552	430	77	1733
Prueba	287	236	184	33	740
Total	961	788	614	110	2473

ticia y **Clasificar la relación entre el titular y el contenido de la noticia**. Para la primera tarea se realizó una validación por un experto que participó en la creación de la guía de anotación, mientras que para la tercera tarea se realizó un acuerdo entre anotadores. Se seleccionó aleatoriamente el 4% de los pares *compatible* y *contradiction* (200 ejemplos) para realizar las validaciones de la primera versión del conjunto de datos. En la práctica, alrededor de la mitad son ejemplos de tipo *contradiction*, verificándose que hubiera un balance entre cada tipo de contradicción.

- **Validación de un experto**: Para la primera tarea, no es posible realizar un acuerdo entre anotadores dado que esta consiste en la modificación de los titulares y las posibles variantes pueden ser inmanejables o tender a infinito. En este caso, un experto en la anotación realiza la revisión manual de los titulares modificados a fin de detectar inconsistencias con las indicaciones presentes en la guía de anotación. Se encuentra que solamente el 2% de los ejemplos analizados presentan inconsistencias con la guía de anotación, lo que corrobora la validez del proceso desarrollado en esta tarea.

- **Acuerdo entre anotadores**: Con el objetivo de medir la calidad de la tercera tarea de anotación se realizó un acuerdo entre dos anotadores. Estos anotaron independientemente 200 ejemplos entre *compatible* y *contradiction*, calculando un índice de acuerdo en la anotación. Se utilizó el índice kappa de Cohen (Cohen, 1960) para calcular el acuerdo en las anotaciones (índice común en procesos de validación de anotaciones entre dos anotadores). Se obtuvo un kappa de Cohen de 0,83, lo que representa un valor alto de acuerdo entre dos anotadores, validando el proceso de anotación.

  En los casos en que no hubo acuerdo (o coincidencia), se desarrolló un proceso de consenso entre los anotadores, resultando en interpretaciones erróneas de la guía de anotación. En este proceso se encontró que la mayoría de los problemas en la anotación estaban relacionados con anotar los tipos de contradicciones. Además se evidenció que los titulares originales

(sin modificación) son clasificados en su mayoría como *compatible* lo que indica la confiabilidad de este medio de noticias. Este proceso de validación hace presuponer qué si se automatiza el proceso de modificación de los titulares, estos serán de tipo *contradiction* debido a la contradicción introducida y los que no se modifiquen serán de tipo *compatible*. Con ello se evita tener que anotar la relación semántica realizada en la tercera tarea.

5.2.2 Segunda versión del conjunto de datos

La segunda versión del corpus desarrollado consta de 18542 noticias, de los cuales 7346 han sido anotados como *compatible*, 7586 como *contradiction* y 3610 como *unrelated*. En esta versión se anotan tres contradicciones de forma automática (*str*, *ant* y *num*) y dos de forma manual *neg* y *fac*. Al igual que en la versión anterior se divide en particiones de entrenamiento y prueba, se muestra la distribución de clases en la tabla 5.3.

Tabla 5.3: Distribución de clases en cada partición de la segunda versión del conjunto de datos.

Particiones	**Compatible**	**Contradiction**	**Unrelated**	**Total**
Entrenamiento	5142	5308	2527	12977
Prueba	2204	2278	1083	5565
Total	7346	7586	3610	18542

Por último, la tabla 5.4 muestra la distribución de clases por tipos de contradicción. Esta versión muestra una distribución de clases menos uniforme debido a que se planificó maximizar algunos tipos de contradicciones con menos ejemplos.

Tabla 5.4: Distribución por tipos de contradicciones en la segunda versión del conjunto de datos.

Particiones	**Neg**	**Ant**	**Num**	**Str**	**Fac**	**Total**
Entrenamiento	168	1366	1505	1873	396	5308
Prueba	73	586	645	803	171	2278
Total	241	1952	2150	2676	567	7586

Validación de la segunda versión de la anotación

Similar a la versión anterior del conjunto de datos, se validan las dos primeras tareas. En este caso ambas tareas corresponden con la modificación de

los titulares. Se validan las tareas de **Modificar automáticamente el titular de la noticia** y **Modificar manualmente el titular de la noticia** de forma conjunta, mediante la validación por parte de un experto.

La segunda fase de anotación no utiliza una tarea extra para anotar la relación semántica. Se parte de la experiencia adquirida en la fase anterior, donde se anotaron muy pocos pares titular-contenido de noticia original como *contradiction*, evidenciando la calidad de las noticias de la fuente escogida. Sin embargo, adicionalmente se lleva a cabo una validación de acuerdo entre anotadores para asegurarse de la validez del conjunto. Se seleccionó aleatoriamente el 4% de los pares *compatible* y *contradiction* (590 ejemplos) para la primera validación y 200 ejemplos para la segunda, de los cuales la mitad eran ejemplos de titulares modificados (automáticamente o manualmente) para ambas validaciones.

- **Validación de un experto**: Se realiza la verificación manual por parte de un experto. Los ejemplos modificados manualmente exhiben una alta correspondencia con las indicaciones de la guía. Por su parte en una primera validación los ejemplos modificados automáticamente inducían un número elevado de problemas de concordancia a los titulares, cerca del 15%, lo que obligó a modificar el *pipeline* de modificación. En una segunda validación se redujo significativamente estos errores a un 4%. La mayoría se encuentran en las contradicciones de *ant* y *num*. Esta validación corrobora la validez del proceso de modificación manual y automática de titulares.

- **Acuerdo entre anotadores**: Con las dos tareas de modificación de titulares validadas se realiza una anotación adicional entre dos anotadores. Se utiliza igualmente el índice kappa de Cohen con un resultado de 0,79, lo que se considera un valor aceptable para una modificación automática de titulares.

5.2.3 Consolidación del conjunto de datos

Con las dos fases de anotación se cuenta con suficientes ejemplos para tener un conjunto de datos pertinente para la tarea de detección de titulares engañosos. En este sentido la tabla 5.5 muestra la versión consolidada de conjunto de datos que ha sido nombrada **ES_Headline_Contradiction**.

Adicionalmente se muestra la tabla 5.6 que indica la distribución por tipos de contradicción. Se aprecia claramente que las contradicciones *ant*, *num* y *str* contienen una cantidad similar de ejemplos anotados. Sin embargo, el tipo de contradicción *fac* presenta muy pocos ejemplos anotados lo que puede afectar el rendimiento de un futuro modelo de detección de contradicciones para este tipo de contradicción. Tanto la versión final como las versiones parciales se encuentran disponibles en este enlace de Zenodo[11].

[11] `https://zenodo.org/badge/latestdoi/344923645` (consultado el 20 de enero de 2022).

Tabla 5.5: Distribución de clases en cada partición en el corpus ES_Headline_-Contradiction.

Particiones	**Compatible**	**Contradiction**	**Unrelated**	**Total**
Entrenamiento	6845	7041	4282	18168
Prueba	2932	3018	1827	7777
Total	9777	10059	6109	25945

Tabla 5.6: Distribución por tipos de contradicciones en el corpus ES_Headline_-Contradiction.

Particiones	**Neg**	**Ant**	**Num**	**Str**	**Fac**	**Total**
Entrenamiento	842	1918	1935	1950	396	7041
Prueba	360	822	829	836	171	3018
Total	1202	2740	2764	2786	567	10059

En la tabla 5.7 se muestran las estadísticas del conjunto de datos creado. Se calcula la extensión media (ext. media) de *tokens* en el titular y el contenido de la noticia. Para ello se utiliza el tokenizador de la biblioteca Spacy que opera a nivel de palabras. Adicionalmente, se incluye la extensión media de *tokens* utilizando el tokenizador de un modelo de lenguaje basado en arquitectura Transformer. Este último tokenizador se diferencia en que tokeniza a nivel de subpalabras por lo que la media será mayor que el tokequizador de Spacy que es a nivel de palabras.

Tabla 5.7: Descripción general de estadísticas del conjunto de datos.

Particiones	**Ext. media titular**	**Ext. media contenido**	**Ext. media contenido subpalabras**
Entrenamiento	13,6	545,2	976,9
Prueba	13,6	553,8	991,7
Total	13,6	547,8	980,7

Teniendo en cuenta el análisis de la tabla 4.12 de la sección 4.5.4 sobre la extensión de los contenidos en corpus de noticias, se encuentra que el corpus creado tiene una longitud media de 980 *tokens*. Para la comparación se utiliza el tokenizador a nivel de subpalabras.

5.3 Experimentos

Se realizó un conjunto de experimentos para demostrar que un sistema de clasificación es capaz de aprender a detectar titulares engañosos y contradicciones procedentes del conjunto de datos propuesto. Se utiliza el modelo de lenguaje basado en arquitectura *Transformer* (BETO[12]). En la sección 2.6.1 se puede consultar la descripción de este modelo. Los experimentos presentados en esta sección pueden ser replicados haciendo uso del siguiente enlace de GitHub[13].

Los experimentos conducidos consisten en hacer un ajuste fino del modelo BETO haciendo uso de la partición de entrenamiento del conjunto de datos desarrollado.

Similar al capítulo 4 los experimentos son llevados a cabo utilizando la biblioteca Simple Transformers. La configuración de hiperparámetros para todos los experimentos es: longitud máxima de secuencia de 512, tamaño de lote de 4, tasa de entrenamiento de 2e-5 y entrenamiento realizado durante 3 iteraciones.

Para evaluar los experimentos, se utilizan tanto la métrica de F_1 por clase como un el macro-promedio F_1 ($F_1 m$).

Cada experimento es desarrollado sobre las dos versiones preliminares del conjunto de datos, así como el conjunto en su totalidad. Se propusieron los siguientes experimentos:

1. **Predicción de todas las clases**: Este experimento tiene como objetivo predecir las clases principales del conjunto de datos *compatible*, *contradiction* y *unrelated*.

2. **Detección de titulares contradictorios vs compatibles**: Este experimento se enfoca en detectar solo los ejemplos de tipo *contradiction* y *compatible* de nuestro conjunto de datos (Tabla 5.1, 5.3 y 5.5 sin los ejemplos *unrelated*).

3. **Detección de tipos específicos de contradicciones**: este experimento utiliza solo los ejemplos de tipo *contradiction* del conjunto de datos descritos en las tablas 5.2, 5.4 y 5.6, para detectar cada tipo de contradicción.

4. **Predicción de todas las clases con la arquitectura**: este experimento utiliza una implementación de la arquitectura propuesta en el capítulo 4 con el fin de evaluar el comportamiento de esta utilizándola para predecir las etiquetas principales del conjunto de datos.

[12] `https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased` (consultada 20 de enero de 2022).

[13] `https://github.com/rsepulveda911112/ES-Contradiction-baseline` (consultada 20 de enero de 2022).

5.4 Resultados y discusión

Esta sección presenta los resultados obtenidos en cada uno de los experimentos descritos en la Sección 5.3. Los valores se expresan en modo porcentual.

5.4.1 Predicción de todas las clases

Este experimento se realiza en todo el conjunto de datos para predecir las tres (3) clases definidas anteriormente. El sistema creado es capaz de detectar la clase *unrelated* con un alto nivel de precisión y consigue resultados significativamente notables en las clases *compatible* y *contradiction*. La tabla 5.8 presenta los resultados.

Tabla 5.8: Resultados obtenidos en el experimento 1: Predicción *compatible*, *contradiction* y *unrelated*.

	F_1 Score			$F_1 m$
Experimentos	**Compatible**	**Contradiction**	**Unrelated**	
Corpus_v1	88,70	89,12	99,59	92,47
Corpus_v2	90,84	90,79	99,40	93,67
ES_Headline_Contradiction	91,60	91,66	99,58	94,28

Los resultados obtenidos en la clase *unrelated* para las dos particiones y el conjunto completo indican que el sistema es capaz de detectar con alto rendimiento este tipo de ejemplos, corroborando los resultados obtenidos en la literatura sobre este tipo de relación semántica entre textos (Q. Zhang y cols., 2019). Con respecto a las otras dos clases se aprecia un margen de mejora. Sin embargo, con la unión de las dos versiones, es decir ES_Headline_Contradiction, los resultados mejoran con respecto a las versiones independientes.

Una posible opción a mejorar estos resultados podría ser incluir conocimiento externo. Una futura línea de trabajo consistiría en incluir recursos que detecten antónimos y sinónimos en línea con (Kang y cols., 2020) con el fin de mejorar los resultados de la clase *contradiction*. Además, incluir información sintáctica y semántica podría mejorar la detección de otras contradicciones más complejas, como la *str* y la *fac*, sin necesidad de conjuntos de datos de tan alta cardinalidad.

5.4.2 Detección de titulares contradictorios vs compatibles

En este experimento, la clase *unrelated* se elimina del conjunto de datos para medir la precisión del enfoque en términos de distinguir entre titulares *compatible* o *contradiction*, asumiendo que la información está relacionada. Los resultados se muestran en la Tabla 5.9.

Tabla 5.9: Resultados obtenidos en el experimento 2: Detectando entre *compatible* y *contradiction* cuando el texto esta relacionado.

	F_1 Score		$F_1\,m$
Experimentos	**Compatible**	**Contradiction**	
Corpus_v1	88.63	88.75	88.69
Corpus_v2	90.93	90.90	90.91
ES_Headline_Contradiction	92.01	92.18	92.09

El enfoque obtiene resultados similares en ambas clases predichas. Esto se debe a la calidad de los ejemplos de entrenamiento y al número equilibrado de ejemplos de cada clase en este conjunto de datos. Similar al experimento 1 el corpus completo obtiene los mejores resultados. Como se indicó en la discusión del primer experimento, los resultados para la predicción de clases podrían mejorarse mediante la introducción de información semántica externa, similar a la introducción de SRL (Z. Zhang y cols., 2020) y el uso de las relaciones de Wordnet (Q. Chen y cols., 2017), las cuales mejoren los resultados de los modelos de aprendizaje profundo.

5.4.3 Detección de tipos específicos de contradicciones

Este experimento tiene como objetivo analizar la capacidad de detección del enfoque por tipos de contradicción. La tabla 5.10 muestra los resultados obtenidos exclusivamente para la detección de tipos de contradicción.

Tabla 5.10: Resultados obtenidos en el experimento 3: Detectando los tipos de contradicción.

	F_1 Score					$F_1\,m$
Experimentos	**Neg**	**Ant**	**Num**	**Str**	**Fac**	
Corpus_v1	97,90	93,20	92,39	68,75	-	88,06
Corpus_v2	83,33	93,28	97,35	95,24	66,88	87,21
ES_Headline_Contradiction	94,21	93,49	97,53	94,85	69,04	89,82

En la primera versión del corpus se contaba con pocos ejemplos anotados del tipo de contradicción *str*, de aquí que se obtengan resultados relativamente bajos con 66,88 en la métrica F_1. Con el aumento de los ejemplos anotados en la segunda versión se alcanzan resultados muy competitivos (95,35 de F_1). También en la segunda versión del corpus se anotaron ejemplos del tipo de contradicción *fac*, alcanzando resultados discretos (66,88 de F_1). Sin embargo, cuando se aumenta la cantidad de ejemplos de otros tipos de clasificación se logra

discriminar un poco mejor este tipo de contradicción (69,04 de F_1). Estos tipos de contradicciones son consideradas más complicadas de detectar en comparación con las otras contradicciones (De Marneffe y cols., 2008). Resulta interesante que el tipo de contradicción *neg* con muchos menos ejemplos que las clases *ant*, *num* y *str* alcanza resultados notablemente buenos usando el corpus completo (94,21 de F_1).

5.4.4 Predicción de todas las clases con la arquitectura

Por último, se realiza un experimento que prueba una instancia de la arquitectura de detección del capítulo 4 con dos etapas de clasificación. En este caso no se utilizan resúmenes ni características externas. La primera etapa de clasificación detecta ejemplos *unrelated* y *related* (incluye las clases *compatible* y *contradiction*) y la segunda detecta *compatible* y *contradiction*.

Tabla 5.11: Resultados obtenidos en el experimento 4: Predicción *compatible*, *contradiction* y *unrelated* mediante la utilización de la arquitectura.

	F_1 Score			
Experimentos	**Compatible**	**Contradiction**	**Unrelated**	**$F_1 m$**
ES_Headline_Contradiction_arq	**91,96**	**92,00**	**99,61**	**94,52**
ES_Headline_Contradiction	91,60	91,66	99,58	94,28

Se añaden los resultados alcanzados por el sistema entrenado en el primer experimento (*ES_Headline_Contradiction*) para predecir todas las clases pero sin utilizar la arquitectura. El experimento que utiliza la arquitectura (*ES_Headline_Contradiction_arq*) obtiene mejor resultado que el realizado mediante una clasificación plana. La diferencia entre ambos experimentos no es tan clara como la del capítulo 4, debido principalmente a que el conjunto de datos desarrollado parece tener menos complejidad para la detección exitosa de sus clases, debido principalmente a que los tipos de contradicción introducidos no son de los más complejos que se pueden encontrar entre textos, además de que sus clases están balanceadas y los modelos basados en DL como BETO aprenden correctamente este tipo de patrones.

5.5 Conclusiones

ES_Headline_Contradiction es un nuevo conjunto de datos en español para la detección de titulares engañosos. Este corpus anota la relación semántica entre titulares y contenidos de noticias teniendo en cuenta la definición de contradicciones entre textos en *compatible*, *contradiction* y *unrelated*. Contiene

además una anotación detallada que distingue el tipo de contradicción según sus características, lo que lo distingue del resto de los conjuntos de datos de la tarea.

Se cubren cinco tipos de contradicciones, lo que representa un amplio espectro de las contradicciones definidas por (De Marneffe y cols., 2008). Se creó un proceso de anotación dividido en dos fases, la primera manual y la segunda de forma automática para las contradicciones de tipo antónimo, numérica y estructura, y el resto de forma manual. El conjunto de datos tiene una cardinalidad de 25945 noticias anotadas.

La validación realizada demuestra que tanto el proceso de construcción como el corpus resultante exhiben la calidad exigida por la guía de anotación y sugiere la reutilización del proceso para otros escenarios de anotación automática.

Los resultados obtenidos por los experimentos muestran que el conjunto de datos creado es una buena opción para entrenar un sistema de detección de titulares engañosos en idioma español mediante la detección de contradicciones. Además, permite detectar el tipo específico de contradicción con una alta precisión, lo que puede contribuir a explicar las decisiones tomadas en auxilio de periodistas, verificadores de hechos y otros usuarios.

Adicionalmente, se corrobora que los modelos de lenguaje basados en arquitectura *Transformer*, como BETO, constituyen una opción viable para la construcción de *baselines* con un ajuste fino sobre el corpus de la tarea, lo que se hace evidente en los resultados experimentales obtenidos.

La utilización de una instancia de la arquitectura del capítulo 4 obtuvo los mejores resultados entre todos los experimentos. Con ello se puede concluir que un enfoque que divide la clasificación de múltiples clases en clasificaciones más pequeñas puede ser apropiado en diversas tareas en el área del PLN.

En futuras investigaciones se propone extender el conjunto de datos siguiendo el enfoque automático de anotación, descargando noticias de otras fuentes de datos confiables para que exista diversidad de emisores y temas abordados. Se deben incluir más ejemplos de la contradicción de tipo *fáctica*, así como el resto de contradicciones definidas por (De Marneffe y cols., 2008). Además, sería de interés crear un sistema de detección automática de titulares engañosos en idioma español, que podría enriquecerse introduciendo información externa de recursos que permitan mejorar los resultados de contradicciones con menor cantidad de ejemplos para entrenar.

CAPÍTULO 6

Verificación automática de hechos

6.1 Introducción

En la introducción de este trabajo se destacaba el problema de la creciente desinformación a la que se está expuesto hoy en día y la incapacidad que tienen los periodistas de verificar manualmente, en un tiempo razonable, las noticias que circulan por redes sociales y medios digitales. Las redes sociales se han convertido en la principal fuente de consumo y diseminación de noticias (Alam y cols., 2020), lo que propicia la proliferación de rumores y noticias falsas. La cantidad de organizaciones dedicadas a la verificación de hechos ha aumentado de 44 en 2014 a 114 a principios de 2017 (N. Hassan y cols., 2017), lo que también es una evidencia contundente del reconocimiento y la atención que se le presta al problema.

En este entorno han surgido esfuerzos para intentar automatizar la verificación de hechos, practicada por periodistas y verificadores de hechos, como tarea principal dentro de la detección de información engañosa (Thorne y Vlachos, 2018).

En la comunidad de investigación es muy habitual encontrar competiciones, laboratorios o talleres que retan a los investigadores a desarrollar soluciones novedosas para avanzar en la creación de recursos, modelos y sistemas en un área determinada. Dos muy importantes en la verificación automática de hechos son: la tarea compartida *Fact Extraction and VERification* (FEVER)[1] y Laboratorio CheckThat! (CheckThat! Lab)[2]. Por ejemplo, FEVER pone a libre disposición un conjunto de datos que contiene afirmaciones y sus respectivas clasifica-

[1] `https://fever.ai/index.html` (consultado el 10 de septiembre de 2022).

[2] `https://sites.google.com/view/clef2021-checkthat` (consultado el 10 de septiembre de 2022).

ciones. Su objetivo es dada una afirmación recuperar evidencias necesarias para llevar a cabo la clasificación usando como base de conocimiento la enciclopedia libre Wikipedia[3]. Este taller ha contribuido a la creación de modelos y sistemas de recuperación de información (IR) y reconocimiento de vinculación textual (RTE). Investigar en estas tareas podría ser un paso de avance significativo hacia la creación de sistemas complejos de verificación automática de hechos. En este sentido, una de las contribuciones adicionales de este trabajo de tesis es abordar problemas abiertos enunciados en una de estas competiciones. Este puede aportar conocimiento al área de la verificación automática de hechos.

El objetivo de este capítulo es el desarrollo de soluciones a problemas particulares del estado del arte que puedan contribuir a mejorar sistemas de verificación automática futuros y por tanto contribuir en un sistema automático de detección de desinformación que es el objetivo principal de este trabajo.

6.2 Verificación de hechos en contexto

La verificación automática de hechos es una tarea que ha surgido para atacar el problema actual de la desinformación y sobrecarga de información. Usualmente se encuentran afirmaciones que están circulando en redes sociales sin su debida verificación, por lo que contar con sistemas que permitan alertar automáticamente sobre su veracidad y apoyar a las tareas de verificación de los periodistas es de mucho interés en la comunidad internacional (Zeng, Abumansour, y Zubiaga, 2021). En este sentido se han propuesto numerosos recursos, modelos y sistemas para abordar esta tarea.

Es muy común encontrar trabajos que propongan el desarrollo de conjuntos de datos para esta tarea (Augenstein y cols., 2019; Gupta y Srikumar, 2021; Thorne y cols., 2018).

En (Augenstein y cols., 2019) se propone un conjunto de datos en idioma inglés. Este es construido extrayendo afirmaciones reales desde 26 sitios web de organizaciones reconocidas de verificación automática de hechos. Por otra parte, (Gupta y Srikumar, 2021) desarrolla un conjunto de datos multilingüe en 25 idiomas con afirmaciones reales. Ambos trabajos crean sistemas *baseline* que alcanzan resultados realmente bajos, lo que indica la complejidad de los conjuntos de datos.

Por último en (Thorne y cols., 2018) se desarrolló un conjunto de datos a gran escala que contiene afirmaciones hechas por anotadores humanos sobre artículos de la Wikipedia. Además se anotan una serie de oraciones que pueden servir de evidencias para verificar una afirmación y la relación semántica con esta. Este conjunto de datos ha sido utilizando en las primeras dos ediciones de la tarea compartida FEVER, desarrollándose numerosas investigaciones en torno a este conjunto (Yoneda, Mitchell, Welbl, Stenetorp, y Riedel, 2018; Miranda y cols., 2019).

[3] `https://www.wikipedia.org/` (consultado el 10 de enero del 2022).

Teniendo en cuenta los *pipelines* de verificación automática de hechos creados, es necesario mencionar a ClaimBuster[4], uno de los pioneros. Actualmente es un proyecto que automatiza varias tareas dentro de la verificación de hechos. Este hace un monitoreo constante de redes sociales, debates políticos y otras fuentes; con el fin de detectar afirmaciones que deban ser chequeadas, lo que constituye un primer gran problema a resolver dento de esta tarea. Posteriormente, verifica si las afirmaciones chequeables detectadas ya han sido verificadas por organizaciones de verificación de hechos. Por último, en el caso que la afirmación no haya sido verificada se realiza un proceso de verificación automática (N. Hassan y cols., 2017).

En (Yoneda y cols., 2018) se propone un *pipeline* de detección compuesto por cuatro etapas (recuperación de documentos, recuperación de oraciones, NLI y agregación) que utiliza el conjunto de datos FEVER. Las dos primeras etapas consisten en recuperar las evidencias necesarias de la Wikipedia. Posteriormente, en la etapa NLI, se desarrolla una red neuronal encargada de encontrar la relación semántica entre la afirmación y las evidencias, calculando una puntuación en esta relación. En la última etapa se realiza una votación con las quince primeras evidencias relacionadas con la afirmación para escoger cual es la clasificación correcta.

En (Miranda y cols., 2019) se crea otro *pipeline* de verificación automática de hechos, destinado a apoyar a periodistas en procesos de verificación. Es un sistema que recibe de entrada una afirmación, recupera evidencias relacionadas y clasifica la relación con la afirmación. Utiliza el conjunto de datos FEVER para entrenar el modelo de clasificación.

Es común encontrar en las verificaciones de hechos realizadas por periodistas, los juicios y la explicación de las verificaciones. Esto es un proceso necesario para poder explicar la veracidad de los hechos, sin embargo, actualmente es un reto que tienen que superar los *pipelines* de verificación automática de hechos (Kotonya y Toni, 2020).

Una red neuronal basada en BERT es entrenada en (Atanasova, Simonsen, Lioma, y Augenstein, 2020) conjuntamente para dos tareas, predecir la veracidad de una afirmación y generar la explicación de su veracidad. Se utiliza el conjunto de datos LIAR-PLUS que fue desarrollado extrayendo afirmaciones verificadas y sus explicaciones de PolitiFact[5]. El entrenamiento conjunto de estas tareas obtiene mejores resultados que entrenar las tareas por separado en modelos independientes.

Otro trabajo tiene como salida del modelo de clasificación además de la veracidad de una afirmación, las frases de evidencia que llevaron a tomar esa decisión (Portelli, Zhao, Schuster, Serra, y Santus, 2020). Esta solución, aunque es muy básica, puede ser utilizada por los usuarios para entender las causas de su

[4] `https://idir.uta.edu/claimbuster/` (consultado el 10 de enero de 2022).

[5] Organización sin ánimos de lucro que se dedica a la verificación de hechos. `https://www.politifact.com/` (consultado el 10 de enero 2022).

decisión. Estos son solo algunos ejemplos de propuestas que intentan abordar el problema de la explicabilidad de los modelos, sin embargo, estas soluciones son bastante lejanas a las explicaciones que brindan los verificadores. En un corto plazo lo más probable es encontrar soluciones híbridas que tengan que ser complementadas con tareas desarrolladas por humanos.

La mayoría de los trabajos en el área se centran en desarrollar conjuntos de datos, sistemas y *pipelines* de verificación automática de hechos. Sin embargo, existen espacios de desarrollo de soluciones que dividen la verificación automática de hechos en tareas específicas con posibilidad de configurarse o integrarse entre sí. En la próxima sección se profundiza en uno de ellos, el laboratorio CheckThat!.

6.3 Laboratorio CheckThat!

Una de las primeras fases dentro de un sistema de verificación automática de hechos es la identificación y determinación del valor de verificación y factualidad de afirmaciones (Alam y cols., 2020). Este tipo de tarea podría automatizarse para tener un control efectivo de las afirmaciones que circulan por medios digitales o incluso son planteadas por figuras públicas como políticos y empresarios, entre otros. Esta tarea es una de las que aborda el laboratorio CheckThat!, este pretende fomentar el desarrollo de soluciones colaborativas y novedosas a problemas dentro de la verificación automática de afirmaciones (Nakov y cols., 2021).

El laboratorio presenta tareas concretas para desarrollar las soluciones en un tiempo determinado, por lo que funciona como un concurso, donde ganan las soluciones mejor punteadas. Se brindan conjuntos de datos de las tareas específicas, los que son utilizados en una fase de construcción de sistemas. Finalmente se pone a disposición un conjunto de datos de prueba (sin anotación de clases) para que los sistemas desarrollados hagan sus predicciones con el objeto de evaluar posteriormente los resultados de cada solución.

En esta sección se abordarán algunos enfoques desarrollados bajo los principios de este laboratorio, los que pueden considerarse contribuciones de importancia para la creación de sistemas de verificación automática de hechos.

6.3.1 Definición del laboratorio CheckThat! 2021

El laboratorio CheckThat! 2021 en su cuarta edición contó con tres tareas independientes: Tarea 1 (*Check-Worthiness Estimation*), Tarea 2: Recuperación de afirmaciones (*Claim Retrieval*) y Tarea 3: Detección de noticias falsas (*Fake News Detection*). Estas tareas se definen como (Nakov y cols., 2021):

- Tarea 1 - (*Check-Worthiness Estimation*): consiste en dado un fragmento de texto detectar si es factible chequearlo. La tarea 1 se divide en dos sub-

tareas la 1A aplicada sobre tuits y la 1B sobre debates políticos, ambas problemas de clasificación.

- Tarea 2 - Recuperación de afirmaciones (*Claim Retrieval*): dada una afirmación verificable y un conjunto de afirmaciones previamente verificadas, determinar si la afirmación ya ha sido verificada. Esta tarea se divide en dos problemas de ranking, la subtarea 2A aplicada sobre tuits y la 2B sobre debates políticos.

- Tarea 3 - Detección de noticias falsas (*Fake News Detection*): consiste en dado un titular y el contenido de una noticia determinar la veracidad de la afirmación principal del artículo y detectar el dominio temático del artículo. La tarea se divide en dos problemas de clasificación de múltiples clases, la subtarea 3A para determinar la veracidad de una noticia y la 3B para determinar el tema que aborda la noticia. (Shahi, StruSS, y Mandl, 2021).

De las tareas presentadas en el laboratorio se escogieron la 1 y la 3 debido a que implicaban problemas de clasificación relacionados con problemas abiertos dentro de la desinformación. La tarea 2 es más cercana a problemas de recuperación de información, los cuales no son abordados directamente en esta investigación.

Teniendo en cuenta la tarea 1 se aborda únicamente la subtarea 1A sobre tuits. Los tuits son mensajes característicos de la red social Twitter, esta solo es superada por Facebook cuando los usuarios de redes sociales discuten sobre noticias (Gabielkov y cols., 2016). Esta red social presenta un alto grado de polarización y es muy común encontrar información engañosa (Grinberg, Joseph, Friedland, Swire-Thompson, y Lazer, 2019). Este tipo de mensaje es considerado distinto al de una noticia, por lo que que puede demandar análisis diferentes, en este sentido se ha decidido profundizar en una solución específica para esta subtarea.

Con respecto a la tarea 3 solo se participó en la subtarea 3A debido a que implica analizar una noticia para evaluar su veracidad, a diferencia de la 3B que es una subtarea de clasificación de temas.

A continuación se explican en detalles las subtareas 1A y 3A de este laboratorio, las que fueron abordadas como parte de esta investigación:

1. La subtarea 1A tiene como objetivo clasificar tuits según su valor de comprobación para un tema, basado en el análisis de la información que expone, partiendo de un tema y un flujo de tuits potencialmente relacionados. Esta tarea podría estar presente en una fase de monitorización de afirmaciones previa a un proceso de verificación automática de hechos para ayudar a verificadores de hechos a priorizar sus esfuerzos en la identificación de afirmaciones chequeables (Nakov y cols., 2018). Las etiquetas a clasificar se definen de la siguiente forma (Shaar y cols., 2021):

- *check-worthy*: son afirmaciones que establecen una definición, mencionan una cantidad en el presente o en el pasado, hacen una predicción verificable sobre el futuro, hacen referencia a leyes, procedimientos y reglas de operación, discuten imágenes o vídeos y afirman correlación o causalidad, entre otros.
- *not check-worthy*: no presenta elementos de la definición anterior.

Esta subtarea es ofrecida en esta edición en idioma árabe, búlgaro, inglés, turco y español. En esta sección se describen dos soluciones sobre la versión en inglés y en español de la propia subtarea.

2. La subtarea 3A, por su parte, persigue el objetivo de detectar noticias falsas como un problema de clasificación de cuatro clases. Desde el titular y el contenido de una noticia se debe determinar si la afirmación principal hecha en un artículo es *false*, *partially false*, *true* o *other* (Shahi y cols., 2021). La subtarea 3A solo se encuentra disponible para idioma inglés. Las etiquetas a clasificar se definen de la siguiente forma:

   - *false*: La afirmación principal realizada en el artículo es falsa.
   - *partially false*: La afirmación principal del artículo es una mezcla de información verdadera y falsa. El artículo contiene información parcialmente verdadera y parcialmente falsa, pero no puede considerarse 100% verdadero.
   - *true*: Esta clasificación indica que los elementos principales de la afirmación son verdaderos.
   - *other*: Un artículo que no se puede clasificar como *False*, *Partially False* o *True* debido a la falta de evidencias sobre sus afirmaciones.

6.3.2 Conjunto de datos CheckThat! 2021

Como se explicó anteriormente, el laboratorio cuenta con tres tareas y con algunas subtareas dentro de estas. Esto implica que se pongan a disposición varios conjuntos de datos dependiendo de la tarea, subtarea y el idioma disponible para las mismas. Este capítulo explica enfoques que han sido desarrollados sobre la subtarea 1A en idioma inglés y español, así como la subtarea 3A para idioma inglés por lo que se explican en detalles los corpus de estas subtareas.

Los conjuntos de datos disponibles para la subtarea 1A cuentan con tres particiones: entrenamiento, desarrollo y prueba. La tabla 6.1 muestra la distribución de clases de cada uno.

Cada ejemplo de entrenamiento y desarrollo contiene el texto, la url, el tema y el identificador único de un tuit, y se anota si el tuit es chequeable (*check-worthy* o *not check-worthy*). En el caso de los ejemplos de prueba en la fase de desarrollo no contienen la anotación de chequeable. A continuación se muestra un ejemplo de tuit chequeable extraído de la partición de entrenamiento.

"En España hay un 40% de energía renovable con el impulso de los gobiernos del PP. Creamos el primer Ministerio de Medio Ambiente, ratificamos los acuerdos Kioto y París, legislamos para acabar con los residuos plásticos... seguiremos luchando contra el cambio climático."

Como se puede apreciar en el ejemplo, el texto del tuit menciona hechos concretos realizados por una persona, entidad, etc. Además se mencionan datos numéricos que pueden ser verificados.

Tabla 6.1: Distribución de etiquetas por partición de los conjuntos de datos de la subtarea 1A.

Particiones	**Check-worthy**	**Not check-worthy**	**Total**
Subtarea 1A para inglés			
Entrenamiento	290	532	822
Desarrollo	60	80	140
Prueba	19	331	350
Total	369	943	1312
Subtarea 1A para español			
Entrenamiento	200	2295	2495
Desarrollo	109	1138	1247
Prueba	120	1128	1248
Total	429	4561	4990

Por otra parte, el conjunto de datos disponible para la tarea 3A solo presenta particiones de entrenamiento y prueba. Los ejemplos de la partición de entrenamiento contienen el titular, el contenido de la noticia y la anotación de la veracidad de la noticia (*false*, *partially False*, *true* y *other*). En el caso de la partición de prueba para la fase de desarrollo de sistemas solo tenían el titular y el contenido de la noticia. Con el fin de evaluar distintos experimentos sobre esta subtarea se decidió fraccionar la partición disponible para entrenar en entrenamiento y desarrollo, con una distribución de 70% y 30% respectivamente. La distribución de clases de los conjuntos creados, así como la partición de prueba se muestran en la tabla 6.2.

Como se puede apreciar en la distribución de clases de los corpus existe un gran desbalance entre la clase mayoritaria y minoritaria. En la subtarea 1A la clase *not check-worthy* tiene casi el doble de ejemplos de la clase minoritaria *check-worthy* para el corpus en inglés y en el corpus en español tiene más de 10 veces el tamaño de la clase minoritaria.

En la subtarea 3A el problema del desbalance del corpus puede ocasionar peores resultados, debido a que es una tarea de clasificación de múltiples clases, implica más complejidad (Lorena y cols., 2008) y la distribución de clases no es

uniforme. La cantidad de elementos que representan la clase mayoritaria *false* es notablemente superior al resto de clases y en peor medida a las clases *true* y *other*. Este desbalance puede ocasionar que los sistemas creados utilizando estos corpus, sesguen su aprendizaje hacia la clase mayoritaria en detrimento del resto de las clases.

Tabla 6.2: Distribución de etiquetas por partición del conjunto de datos de la subtarea 3A.

Particiones	**True**	**False**	**Partially False**	**Other**	**Total**
Entrenamiento	98	323	155	54	630
Desarrollo	44	142	62	22	270
Prueba	65	111	138	40	354
Total	207	576	255	116	1254

6.3.3 Trabajos relacionados con la tarea 1 y 3 del CheckThat!

La primera edición del laboratorio CheckThat! se desarrolló en el año 2018 y hasta el momento se han realizado cuatro ediciones en las que siempre ha estado presente la tarea *check-worthiness estimation*, tarea 1 en la edición del año 2021. Sin embargo, en las primeras dos ediciones solo se aborda la tarea sobre debates políticos. No fue hasta la tercera edición donde se introduce esta tarea sobre tuits. A continuación se profundiza en algunas investigaciones sobre esta tarea, ya sea en debates políticos o en tuits.

En las dos primeras ediciones se encuentran trabajos utilizando técnicas como SVM (Agez, Bosc, Lespagnol, Mothe, y Petitcol, 2018; Su, Macdonald, y Ounis, 2019), K-NN (Ghanem, y Gómez, Pardo, y Rosso, 2018), bosques aleatorios (Agez y cols., 2018), *naïve bayes* (Coca, Cusmuliuc, y Iftene, 2019) y ANNs (Dhar, Dutta, y Das, 2019; Favano, Carman, y Lanzi, 2019). En estos trabajos se suelen utilizar desde representaciones de tipo *word embeddings* (Dhar y cols., 2019; Favano y cols., 2019), bolsa de palabras (Coca y cols., 2019; Su y cols., 2019) y características relacionadas con la subjetividad (Dhar y cols., 2019) o sentimientos(Dhar y cols., 2019) entre otras más específicas. En las dos primeras ediciones la mayoría de las soluciones se abordan utilizando algoritmos de ML tradicionales.

A partir de la tercera edición hay un cambio marcado en la tipología de las soluciones, siendo la mayoría basada en algoritmos de DL. Se utilizan modelos basados en arquitectura Transformer como BERT (Zengin, Kartal, y Kutlu, 2021), RoBERTa (E. Williams, Rodrigues, y Novak, 2020), BERTweet[6] (J. Martinez-Rico, Martinez-Romo, y Araujo, 2021), SBERT[7] (Schlicht, de Paula, y Rosso, 2021) o

[6]Modelo BERT entrenados sobre millones de tuits (Nguyen, Vu, y Tuan Nguyen, 2020).

[7]Versión multilingüe de un modelo *Transformer* de representación de oraciones (Reimers y Gurevych, 2019).

redes neuronales de tipo CNN (Alkhalifa, Yoong, Kochkina, Zubiaga, y Liakata, 2020) y LSTM (J. R. Martinez-Rico, Araujo, y Martinez-Romo, 2020).

Además, también se encuentran algunas estrategias de preprocesamiento de los tuits, como extraer url (Nikolov, Martino, Koychev, y Nakov, 2020), emoji[8], *stop words*, convertir las palabras en minúsculas y lematizar (McDonald y cols., 2020).

Algunos trabajos muestran soluciones de notable interés (Dhar y cols., 2019; Schlicht y cols., 2021; Nikolov y cols., 2020). En (Dhar y cols., 2019) se combina una red neuronal de tipo LSTM que recibe como entrada una representación de tipo *word embeddings* de los textos de los debates, con un clasificador de tipo LR que procesa características externas relacionadas con la subjetividad, análisis de sentimientos, etc. Posteriormente, se concatenan ambas representaciones y son clasificadas en un red neuronal de tipo MLP. Este sistema obtiene alta precisión.

(Nikolov y cols., 2020) entrena un clasificador basado en LR utilizando vectores extraídos de RoBERTa y los concatena con características externas provenientes de metadatos de los tuits.

Por último, el trabajo desarrollado en (Schlicht y cols., 2021) propone un sistema multilingüe utilizando el modelo SBERT. Este sistema se entrena para dos tareas, la subtarea 1A y para que detecte el idioma del tuit. La estrategia de entrenar estas dos tareas al mismo tiempo hace que se complementen y mejoren los resultados de clasificación.

La tarea de detección de noticias falsas ha sido introducida por primera vez en la cuarta edición del laboratorio CheckThat!. Analizando los sistemas desarrollados por los equipos se confirma la tendencia actual a desarrollar sistemas utilizando modelos basados en arquitectura Transformer (Kumari, 2021; Lekshmiammal y Madasamy, 2021).

En (Kumari, 2021) se utiliza el modelo de lenguaje RoBERTa entrenado sobre un conjunto de datos extenso, creado extrayendo noticias de 92 sitios web de verificación de hechos. Con el corpus desarrollado en (Kumari, 2021) se logra abordar el desbalance del conjunto de datos con el que se cuenta en la subtarea 3A.

Un clasificador *ensemble* que integra dos clasificadores es desarrollado en (Lekshmiammal y Madasamy, 2021). El primero se entrenó utilizando el modelo de lenguaje RoBERTa haciendo un truncamiento de los contenidos de noticias en fracciones de 450 *tokens* y entrenando con esas fracciones. El segundo modelo se entrenó utilizando el modelo Longformer con una cantidad máxima de 3000 *tokens*.

Sin embargo, una diferencia marcada con la subtarea 1A es que de los cinco sistemas mejor ranqueados de la subtarea 3A, dos de ellos (equipos SAUD[9] y DLRG[9]) utilizan algoritmos tradicionales de ML. El equipo SAUD utiliza SVM

[8]Pequeña imagen o icono digital que se usa en las comunicaciones electrónicas para representar una emoción, un objeto, una idea, etc. `https://dle.rae.es/emoji` (consultado el 10 de septiembre de 2022).

con vectores TF-IDF y el DLRG crea un clasificador *ensemble* con *naïve bayes*, LR y *passive aggressive*.

En la tabla 6.3 se muestra un resumen de los trabajos consultados relacionados con los enfoques de ML y DL utilizados. Como se aprecia existe bastante diversidad en las soluciones aplicadas, pero es claro el aumento de soluciones basadas en arquitectura Transformer de los últimos años. Esta tendencia no siempre se traduce en los mejores resultados, como es el ejemplo del segundo y tercer equipos mejores ranqueados de la subtarea 3A que no la utilizan.

Tabla 6.3: Resumen de soluciones para las tareas 1 y 3.

Investigaciones	**Tarea**	SVM	K-NN	LR	**Naïve bayes**	CNN	LSTM	**Transformer**
(Agez y cols., 2018)	1	✓						
(Su y cols., 2019)	1	✓						
(Ghanem y cols., 2018)	1		✓					
(Nikolov y cols., 2020)	1			✓				✓
(Coca y cols., 2019)	1				✓			
(Alkhalifa y cols., 2020)	1					✓		
(J. R. Martinez-Rico y cols., 2020)	1						✓	
(Dhar y cols., 2019)	1			✓			✓	
(Zengin y cols., 2021)	1							✓
(E. Williams y cols., 2020)	1							✓
(J. Martinez-Rico y cols., 2021)	1 y 3							✓
(Schlicht y cols., 2021)	1							✓
(Kumari, 2021)	3							✓
(Lekshmiammal y Madasamy, 2021)	3							✓
Equipo SAUD	3	✓						
Equipo DLRG	3			✓	✓			

Poniendo en contexto las principales soluciones a estas subtareas dentro del laboratorio CheckThat!, en la próxima sección se explican las soluciones propuestas en este trabajo.

6.3.4 Propuesta de solución subtareas 1A y 3A en CheckThat! 2021

Para dar solución a las subtareas seleccionadas de este laboratorio seguimos una estrategia de escalar una solución básica incluyendo características específicas y realizar diferentes experimentos sobre estas. En concreto, se crea un

[9] No envió artículo de descripción del sistema. Descripción extraída de (Shahi y cols., 2021).

sistema *baseline* para cada subtarea haciendo uso de modelos de lenguajes pre-entrenados del estado del arte (basados en arquitectura Transformer).

En el caso de las subtareas 1A y 3A para idioma inglés se utiliza el modelo RoBERTa y para la subtarea 1A en idioma español se utiliza el modelo BETO. En ambos casos los modelos de lenguaje utilizados han sido seleccionados basados en la calidad de los resultados obtenidos sobre otras tareas similares en los idiomas utilizados. Se pueden consultar más detalles sobre los modelos escogidos en la sección 2.6.

Arquitectura de clasificación utilizando modelos Transformer

Para la creación de los sistemas *baselines* así como para proponer sistemas más complejos que utilicen características externas se ha utilizado la arquitectura de clasificación propuesta en la figura 4.3. Las entradas de esta arquitectura descrita en la sección 4.3.1 eran el titular, el resumen de la noticia, así como las característica externas. En el caso de las soluciones para la subtarea 1A las entradas serán el tuit a verificar y las características externas. Al igual que las implementaciones explicadas en las secciones 4.3.2 y 4.3.1, posterior a la salida de los modelos de lenguaje (BETO y RoBERTa) se utiliza una red neuronal de clasificación. Esta red neuronal es la encargada de llevar a cabo la clasificación del tuit o la noticia. A continuación se muestra la estructura utilizando estas características:

1. Capa densa con función de activación Tanh para extender las características externas a un vector que contenga las mismas dimensiones que la salida del modelo de lenguaje.
2. Capa para multiplicar las características extendidas en la capa anterior y las salidas del modelo de lenguaje.
3. Capa *dropout* para evitar que el modelo de clasificación se sobreajuste.
4. Capa densa con función de activación Tanh.
5. Subtarea 1A: capa densa con una neurona de salida y función de activación Sigmoid debido a que consiste en una clasificación binaria. Subtarea 3A: capa densa con cuatro neuronas de salida y función de activación Softmax porque consiste en una clasificación de múltiples clases.
6. Subtarea 1A: función de pérdida *binary cross-entropy*. Subtarea 3A: función de pérdida *cross-entropy*.

La estructura interna de la red neuronal viene determinada por el empleo de características externas. Si no se utilizan características externas, se excluyen las dos primeras especificaciones, comenzando la estructura con la número 3.

En la Subtarea 1A, tanto para los idiomas inglés y español, el tuit de entrada a los modelos se preprocesa con el fin de extraer emojis y las url que contiene.

Este preprocesameinto permite disminuir símbolos fuera del vocabulario de los modelos.

Posterior al preprocesamiento se extraen características relacionadas con la presencia y cantidad de números y fechas en los tuits. Se utiliza la biblioteca Stanza[10] de python para anotar las dependencias del texto y reconocer entidades nombradas. Se decide utilizar estas características debido a que es común encontrar en tuits potencialmente verificables datos numéricos, fechas e incluso entidades nombradas (Konstantinovskiy, Price, Babakar, y Zubiaga, 2021).

Para la Subtarea 1A se ha probado una modificación consistente en la inclusión de características lingüísticas de la herramienta *Linguistic Inquiry and Word Count* (LIWC). Este es un recurso para detectar características en una amplia variedad de entornos experimentales, que incluyen foco de atención, emoción, relaciones sociales, estilos de pensamiento y las diferencias individuales (Pennebaker, Boyd, Jordan, y Blackburn, 2015). Los diccionarios de LIWC se han traducido a varios idiomas, incluidos español, alemán, italiano y portugués. Se usa LIWC traducido al español para la subtarea de la versión en español y la versión original de LIWC para la subtarea de la versión en inglés.

6.3.5 Experimentos y métricas de evaluación

En esta sección se describe una serie de experimentos que tienen como objetivo encontrar el modelo, los hiperparámetros y las características externas, así como las estrategias que mejor se ajusten a las subtareas señaladas. Los experimentos realizados se basan en los modelos de lenguajes descritos anteriormente.

Experimentos subtarea 1A

Esta subtarea tiene como objetivo determinar si un tuit debe ser verificado o no, por lo tanto, la entrada a cada modelo es un tuit y en el caso de algunos experimentos, la entrada también agrega características externas. Los seis experimentos propuestos para esta subtarea utilizan técnicas que son comúnmente utilizadas por la comunidad científica. La entrada al modelo de lenguaje es limitada a una longitud máxima de secuencia de 125 *tokens*, debido a que se procesan tuits que tienen como máximo 280 caracteres. Estos experimentos se describen a continuación:

1. *Los sistemas baseline con* RoBERTa *o BETO*: Este experimento hace un *fine-tuning* de los modelos de lenguaje sobre el corpus de la subtarea. Similar a otras utilizaciones de estos modelos de lenguaje en capítulos anteriores, estos *baselines* tienen los siguientes hiperparámetros: tamaño de lote de 4, tasa de entrenamiento de 1,5e-5 y número de iteraciones para entrenar de 3.

[10] Documentación disponible en `https://stanfordnlp.github.io/stanza/` (consultada el 20 de enero de 2022).

2. *Optimización de hiperparámetros*: Este experimento realiza una búsqueda bayesiana para optimizar los hiperparámetros. Se utiliza la herramienta Weights & Biases[11] para automatizar el ajuste de hiperparámetros y explorar el espacio de posibles configuraciones de los modelos. Esta herramienta permite la visualización y comparación de los resultados de cada modelo (Biewald, 2020). La configuración de búsqueda se muestra en la tabla 6.4.

Tabla 6.4: Configuración de búsqueda de hiperparámetros

Hiperparámetros	**Valores**
Tamaño de lote	2, 4 y 8
Tasa de entrenamiento	1e-5, 1,5e-5, 2e-5, 2,5e-5 y 3e-5
Tasa de *dropout*	0,1, 0,2, 0,3, 0,4 y 0,5
Número de iteraciones para entrenar	2, 3 y 4

3. *Modelos* RoBERTa *o BETO con indicadores de números y fecha*: Se concatena la salida de la última capa de RoBERTa o BETO con los indicadores de número y fecha.

4. *Modelos con* RoBERTa *o BETO con características de* LIWC: Se concatena la salida de la última capa de RoBERTa o BETO con características de LIWC.

5. *Modelos con* RoBERTa *o BETO con oversampling*: El conjunto de entrenamiento se amplía con ejemplos de la clase menos representativa para equilibrar el conjunto de datos.

6. *Modelos con* RoBERTa *o BETO con undersampling*: Se eliminan ejemplos de la clase más representativa para equilibrar el conjunto de datos.

Los cuatro últimos experimentos se desarrollan sobre los modelos de lenguajes con los hiperparámetros optimizados.

Con el objetivo de evaluar los resultados, en esta subtarea los organizadores proponen utilizar *Mean Average Precision* (MAP) como métrica oficial. MAP es una métrica popular utilizada para medir el rendimiento de los modelos que realizan tareas de recuperación de documentos/información y detección de objetos (Nakov y cols., 2018). Esta métrica se define como:

$$mAP = \frac{\sum_{d=1}^{D} AveP(d)}{D} \tag{6.1}$$

donde $d \in D$ y es uno de los tuits a evaluar, y *AveP* es:

[11] Documentación disponible en `https://wandb.ai/site` (consultada el 20 de enero de 2022).

$$AveP = \frac{\sum_{k=1}^{K}(P(k) \times \delta(k))}{\mathrm{N}} \tag{6.2}$$

donde *P(k)* se refiere al valor de precisión en el rango *k*, $\delta(k) = 1$ si el reclamo en esa posición es verificable y N es cantidad de tuits clasificados como *check-worthy*.

Además, son incluidas las métricas R-precision(RP) y precision@k, con *P@k* para $k \in \{1, 3, 5, 10, 20, 30\}$.

R-precision(RP) es la Precisión en R, donde R es el número de tuit relevantes para un tema Q y r es la cantidad de documentos recuperados que son relevante. Esta métrica mide la proporción entre los documentos recuperados relevantes y el total de documentos recuperados, se define en (Craswell, 2009b) como:

$$RP = \frac{r}{R} \tag{6.3}$$

Precision@k es una métrica utilizada en recuperación de información que mide la presión de la recuperación de r elementos relacionados en k elementos. Se define en (Craswell, 2009a) como:

$$Precision@k = \frac{r}{k} \tag{6.4}$$

Por último, el macro-promedio F_1 ($F_1 m$) es incluido como métrica común de problemas de clasificación (ver descripción en la sección 3.5.4).

Experimentos subtarea 3A

El objetivo de esta subtarea es clasificar noticias según su veracidad. La subtarea 3A se realiza en inglés, por lo tanto, de los modelos explicados en la sección anterior solo se puede utilizar el modelo de lenguaje RoBERTa. Este procesa tanto el título como el contenido del artículo, representándolos como una secuencia de palabras concatenadas. Se realizan tres experimentos con el objetivo de desarrollar el sistema enviado a la tarea 3A del Checkthat!. En este caso la entrada del modelo es configurada en la longitud máxima de secuencia que permite el modelo RoBERTa, 512 *tokens*, debido a que las noticias suelen tener mayor longitud. A continuación se explican los experimentos:

1. *Sistema baseline*: Este experimento hace un *fine-tuning* del modelo de lenguaje RoBERTa sobre el corpus de la subtarea. Se utiliza la misma configuración de hiperparámetros del experimento 1 de la subtarea 1A.

2. *Optimización de hiperparámetros*: Similar a la subtarea anterior se realiza una búsqueda bayesiana para optimizar los hiperparámetros. Se utiliza la misma configuración de hiperparámetros descrita en la tabla 6.4.

3. *Modelo con tres clasificadores*: Se agrega un tercer experimento que pretende abordar el problema del desbalance del conjunto de datos de la subtarea 3A, haciendo uso de clasificadores jerárquicos. Este experimento divide la clasificación de cuatro clases en dos clasificaciones binarias y una clasificación de tres clases.

Se utiliza como métrica oficial para evaluar los resultados el macro-promedio F_1 ($F_1 m$), consultar sección 3.5.4 para más detalles sobre esta métrica.

6.3.6 Resultados y discusión

Cada uno de los experimentos presentados en la sección anterior, que forman parte del desarrollo de los sistemas presentados en el laboratorio Checkthat! han sido entrenados y evaluados haciendo uso de las particiones de entrenamiento y desarrollo descritas en la sección 6.3.2. Estos experimentos son referidos teniendo en cuenta los números asignados en su descripción. Los sistemas presentados en esta sección fueron enviados al laboratorio CheckThat! como parte del equipo GPLSI.

Con el objetivo de evaluar críticamente los sistemas finales enviados al laboratorio se incluyen experimentos posteriores que intentan descubrir y solucionar algunos problemas en las soluciones planteadas en el corto margen de tiempo de desarrollo de propuestas para este tipo de competición.

Subtarea 1A

Los resultados a los experimentos explicados en esta sección también incluyen la métrica $F_1 m$ porque aunque no es posible comparar con el sistema *baseline* propuesto por los organizadores permite tener una idea de la pertinencia y calidad de una clasificación binaria. En la etapa de construcción de los sistemas, solo estaban disponibles las particiones de entrenamiento y desarrollo, por lo que son las usadas en estos experimentos.

En el experimento 1 se desarrolla el *baseline* para ambos idiomas. Las versiones para ambos idiomas obtienen buenos resultados en la métrica $F_1 m$ siendo un sistema de clasificación binaria, ambos por encima de 70%. Sin embargo en la métrica oficial MAP para el sistema que utiliza el modelo BETO el resultado es de 0,485, resultado solo 0,073 puntos mejor que el sistema *baseline* de los organizadores; por el contrario el sistema que utiliza el modelo RoBERTa alcanza resultados muy superiores al sistema de referencia.

En la búsqueda de la mejor configuración de hiperparámetros del experimento 2, se encontró una configuración para cada sistema de clasificación que obtiene resultados notables. Sin embargo, no es posible garantizar que la configuración encontrada sea la mejor, debido a que la búsqueda está acotada a un subconjunto de hiperparámetros y sus posibles valores se encuentran en el rango definido para la búsqueda (esto significa que la búsqueda no es exhaustiva). La tabla 6.5 muestra la configuración de hiperparámetros para ambos idiomas.

Tabla 6.5: Configuración de hiperparámetros obtenida para los modelos de lenguaje RoBERTa y BETO.

	Valor	
Hiperparámetros	**Español**	**Inglés**
Tamaño de lote	8	4
Tasa de entrenamiento	1e-5	1e-5
Tasa de *dropout*	0,2	0,2
Número de iteraciones para entrenar	2	3

Para la realización de los experimentos restantes se utiliza la configuración de hiperparámetros descubierta en el experimento 2. Los experimentos 3 y 4 modifican la estructura de la red neuronal de clasificación, similar a la mostrada en la figura 4.3, la estructura exacta se puede consultar en la sección 6.3.4. Los resultados alcanzados por ambos experimentos son bajos incluso si los comparamos con el experimento *baseline*.

Este comportamiento es similar en ambos idiomas, lo que evidentemente significa que las características utilizadas no son determinantes para ordenar la importancia de un tuit para verificar un tema en específico. Por otra parte, los experimentos 5 y 6, que aplican técnicas reconocidas para abordar el problema de la diferencia en la distribución de clases, tampoco logran mejorar los resultados obtenidos por el experimento 2. En ambos casos, las dos métricas descendieron en relación con el experimento 2. En esta subtarea no se logró encontrar un conjunto de características que mejoraran los resultados de clasificación. La Tabla 6.6 muestra los resultados de los experimentos.

Tabla 6.6: Resultados de los experimentos de la subtarea 1A con los modelos de lenguaje RoBERTa y BETO sobre la partición de desarrollo.

	Español		Inglés	
Experimentos	**MAP**	$F_1\,m$	**MAP**	$F_1\,m$
1 - Sistemas baseline con RoBERTa y BETO	0,485	70,9	0,762	70,2
2 - Búsqueda de hiperparámetros	**0,549**	**71,2**	**0,825**	**75,0**
3 - Modelo con indicadores de números y fecha	0,500	63,3	0,652	70,6
4 - Modelo con características de LIWC	0,497	68,5	0,624	62,4
5 - Modelo con oversampling	0,387	69,3	0,772	73,5
6 - Modelo con undersampling	0,455	56,4	0,795	70,9
Sistema baseline propuesto por la organización	0,412	-	0,591	-

Teniendo en cuenta los resultados obtenidos evaluando los experimentos con el conjunto de desarrollo, se seleccionó el experimento 2 para hacer las pre-

dicciones sobre el conjunto de prueba para ambos idiomas.

El equipo GPLSI alcanzó el segundo lugar en la versión en español de la subtarea 1A. La diferencia en la métrica MAP con el equipo TOBB ETU (Zengin y cols., 2021) que lideró la clasificación de la subtarea es de tan solo 0,008 puntos. Sin embargo, el siguiente equipo (bigIR[12]) se situó a 0,033 de GPLSI. Los tres mejores resultados son mostrados en la tabla 6.7.

Tabla 6.7: Resultados de la subtarea 1A del CheckThat! en idioma español sobre la partición de prueba.

Equipo	MAP	RP	P@1	P@3	P@5	P@10	P@20	P@30
TOBB ETU	**0,537**	0,525	**1,000**	**1,000**	**0,800**	**0,900**	0,700	**0,680**
GPLSI	0,529	**0,533**	0,000	0,667	0,600	0,800	**0,750**	0,620
bigIR	0,496	0,483	**1,000**	**1,000**	**0,800**	0,800	0,600	0,620

Para la versión en idioma inglés de la subtarea 1A el equipo GPLSI alcanzó el quinto puesto. En este caso, la diferencia en la métrica MAP entre el primer puesto, equipo NLP&IR@UNED (J. Martinez-Rico y cols., 2021), y el equipo GPLSI es de 0,092, siendo bastante mayor que en la versión en español. En esta versión de la subtarea todos los equipos que alcanzaron mejores resultados que el equipo GPLSI utilizan modelos Transformer como: BERTweet o en el caso del equipo UPV (Schlicht y cols., 2021) usa SBERT.

En el equipo GPLSI por su parte se utilizó el modelo RoBERTa, los resultados alcanzados en este laboratorio hacen pensar que quizás RoBERTa no sea el modelo de lenguaje más adecuado para la tarea. Sin embargo, el modelo que alcanzó mejores resultados (BERTweet) ha sido creado usando los procedimientos de entrenamientos propuestos por RoBERTa pero entrenado sobre un corpus con 850 millones de tuits, indicando la mejor adaptación de este modelo de lenguaje a la tarea. La tabla 6.8 muestra los resultados alcanzados por el equipo GPLSI y los equipos que obtuvieron el primer y tercer lugar.

Tabla 6.8: Resultados de la subtarea 1A del CheckThat! en idioma inglés sobre la partición de prueba.

Equipo	MAP	RP	P@1	P@3	P@5	P@10	P@20	P@30
NLP&IRUNED	**0,22**	**0,21**	**1,00**	**0,66**	**0,40**	**0,30**	**0,20**	**0,16**
UPV	0,14	0,10	**1,00**	0,33	0,20	0,20	0,10	0,12
GPLSI	0,13	0,15	0,00	0,00	0,00	0,20	0,15	0,14

En resumen, los numerosos experimentos que se llevaron a cabo para cada versión de la subtarea no lograron mejorar los resultados alcanzados por el experimento 2. Estos experimentos permiten concluir que los modelos basados

[12]No envió artículo de descripción del sistema.

en aprendizaje por transferencia y específicamente los basados en arquitectura Transformer (BETO, RoBERTa, etc.) permiten crear sistemas competitivos para tareas específicas realizando un ajuste sobre los datos que se necesitan predecir. Evidentemente, no se han encontrado características externas adecuadas para mejorar los resultados obtenidos en el experimento 2. Los equipos mejor posicionados en ambos idiomas utilizan estrategias que atacan el desbalance de las etiquetas de los conjuntos de datos incluyendo ejemplos ficticios, en ocasiones modificando ejemplos que ya se encontraban en los conjuntos de datos.

Subtarea 3A

Similar a la subtarea anterior, el *baseline* desarrollado para esta subtarea realiza un ajuste del modelo de lenguaje RoBERTa. Este *baseline* utiliza los hiperparámetros descritos en el experimento 1 de la sección anterior, solo cambiando la longitud máxima de secuencia a 512. La subtarea 3A tiene como entrada a la arquitectura de clasificación el titular y el contenido de la noticia. Esta subtarea se considera más complicada que la 1A debido a que es necesario encontrar patrones para clasificar la veracidad de las noticias en cuatro clases, sin utilizar información externa. El *baseline* propuesto obtiene 51,6% en la métrica oficial ($F_1 m$), siendo un resultado notablemente bajo, lo que corrobora la complejidad de esta subtarea.

El experimento 2 realiza una búsqueda de la mejor configuración de hiperparámetros y como muestran los resultados (52,0% de $F_1 m$), la mejora es insignificante con respecto al sistema *baseline*. En otra tarea, un ajuste de hiperparámetros como el realizado debería haber mejorado significativamente los resultados obtenidos por el *baseline*. Teniendo en cuenta los valores alcanzados de $F_1 m$ por clases, solo para la clase *other* el valor es superior al del sistema *baseline*, corroborando que la mejora de este experimento es ínfima. Retomando la distribución de etiquetas del corpus utilizado en esta subtarea (ver tabla 6.2), es evidente la poca cantidad de ejemplos que contienen la partición de entrenamiento y desarrollo. Esta característica puede ocasionar que el sistema de clasificación no sea capaz de generalizar patrones de detección con tan pocos ejemplos de entrenamiento de algunas clases.

La tabla 6.9 muestra los resultados de los experimentos realizados con el objetivo de encontrar el mejor modelo de clasificación durante el proceso de desarrollo de sistemas para la subtarea 3A dentro del laboratorio.

En la sección 6.3.2 se planteó el problema del desbalance del corpus de esta subtarea, ya que la diferencia es marcada entre la clase mayoritaria (*false*) y las minoritarias (*true* y *other*). En este sentido, en el análisis realizado sobre clasificaciones jerárquicas en la sección 2.5.1 se encontraban algunos trabajos que aplican este tipo de clasificadores para intentar solucionar el desbalance entre etiquetas y así mejorar los resultados de clasificación.

En el tercer experimento realizado para el desarrollo de este sistema de clasificación, se aplican algunas de las pautas planteadas en el capítulo 4 sobre la

creación de clasificadores divididos en etapas de clasificación. De forma similar a ese capítulo se crea una estructura de clases relacionadas jerárquicamente, la que se muestra en la figura 6.1.

Tabla 6.9: Resultados de los experimentos de la subtarea 3A, ajustando el modelo de lenguaje RoBERTa para predecir la partición de desarrollo creada.

Experimento	F_1 Score				$F_1 m$
	False	Partially False	True	Other	
1 - Sistema baseline	**81,3**	**57,8**	59,0	8,30	51,6
2 - Optimización de hiperparámetros	79,3	56,6	54,9	17,6	52,0
3 - Modelo con tres clasificadores	80,9	57,4	**60,0**	**21,0**	**54,8**

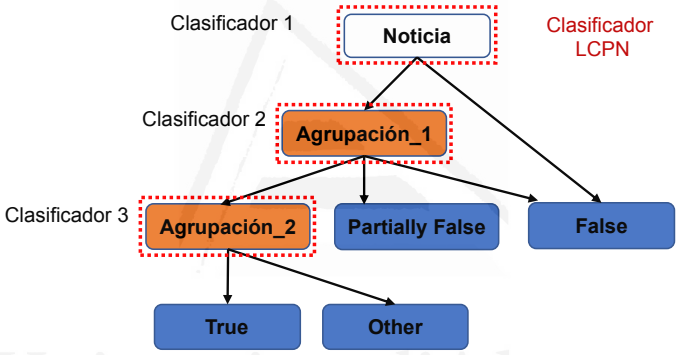


Figura 6.1: Estructura jerárquica de clasificación (caso particular).

La estructura jerárquica de este experimento se crea basada en el desbalance de las etiquetas y está representada por un Grafo Acíclico Dirigido (DAG), a diferencia del capítulo anterior en que las clases son agrupadas según su relación semántica en un árbol. Se sigue una estrategia de agrupar las clases minoritarias en clases ficticias para que sea clasificada la agrupación junto a clases mayoritarias y a medida que se desciende en la jerarquía ir clasificando las minoritarias.

Se crean dos clases ficticias, *agrupación_1* que contiene a *agrupación_2*, *partially false* y *false*. Por otra parte, la *agrupación_2* contiene a *true* y *other*. En este caso también se aplica el enfoque de Clasificador Local por Nodo Padre (LCPN) para solucionar el problema de clasificación jerárquica. En el nodo Noticia de la figura 6.1 se aplica el primer clasificador para discriminar entre *agrupación_1* y *false*, en el nodo *agrupación_1* se aplica el segundo clasificador para obtener las clases *agrupación_2*, *partially false* y *false*. Como se puede apreciar, tanto el primer clasificador como el segundo clasifican la clase *false*, aspecto que permite la representación en DAG con el objetivo de mejorar la precisión de la clase *false*. Finalmente el tercer clasificador es colocado en el nodo *agrupación_2* para clasificar en *true* y *other*.

Con este análisis se decide, a partir de la experiencia en el capítulo 4, crear una arquitectura de clasificación dividida en etapas. Para la creación de esta arquitectura solo se tiene en cuenta la distribución de clases en la partición de entrenamiento y no la semántica de cada clase, como se hace en la propuesta de arquitectura de detección de titulares engañosos del capítulo 4. Esta arquitectura de clasificación se muestra en la figura 6.2.

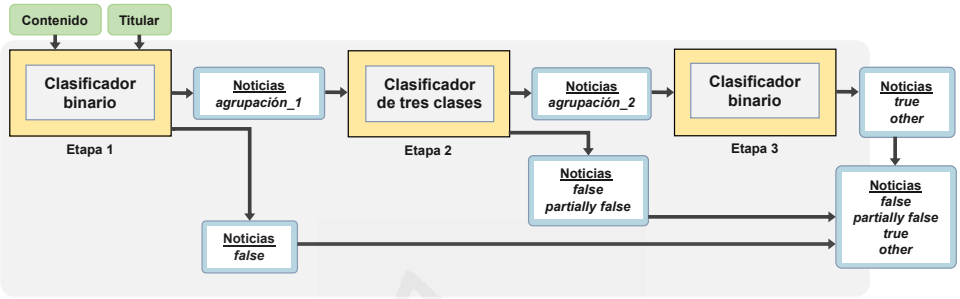


Figura 6.2: Arquitectura de clasificación dividida en tres etapas

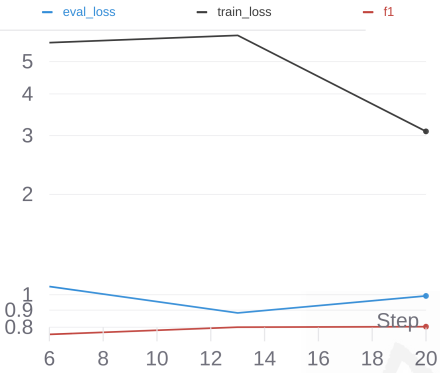
Esta arquitectura difiere de la creada en el capítulo 4 principalmente en que no cuenta con un módulo de extracción de características y no se utiliza el módulo de extracción de información relevante, se diferencia en el método seguido para dividir las etapas de clasificación. Sin embargo, los notables resultados alcanzados por la arquitectura dividida en etapas para la detección de titulares engañosos ha motivado la experimentación en otras tareas que impliquen clasificación en múltiples clases.

Similares a los experimentos 1 y 2 de esta subtarea, la creación de cada uno de los clasificadores del experimento 3 implicó crear un clasificador inicial y buscar una configuración de hiperparámetros adecuada. Una vez encontrada, para cada configuración de hiperparámetros se realizan treinta (30) entrenamientos para cada clasificador, variándose solamente la semilla que inicializa los pesos de los modelos para poder replicar los entrenamientos. Con estos entrenamientos se evalúa la evolución de la función de pérdida durante el proceso de entrenamiento y evaluación del modelo, prestando especial énfasis en el comportamiento de cada clasificador.

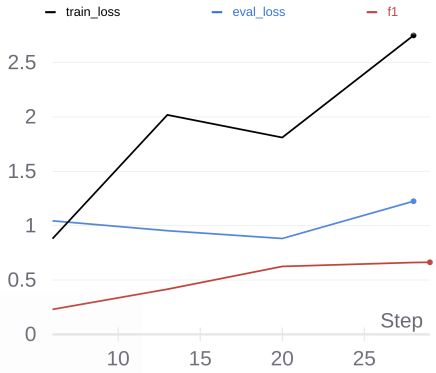
Los clasificadores se entrenan con la partición de entrenamiento y se evalúan durante cada iteración con la partición de desarrollo. Las gráficas de la figura 6.3 muestran las curvas de pérdida para el entrenamiento, la evaluación y además la evolución de la métrica $F_1\,m$ durante la evaluación con la partición de desarrollo en cada iteración. En el eje de las abscisas se encuentra la cantidad de iteraciones de entrenamiento y el eje de las ordenadas representa el rango de valores de la pérdida y la métrica de evaluación.

Las gráficas de la figura 6.3 representan a cada clasificador de la arquitectura. En la gráfica de la figura 6.3a se observa como la pérdida usando el conjunto de entrenamiento bajó desde 5,99 hasta 3,08 en la última etapa de entrenamiento,

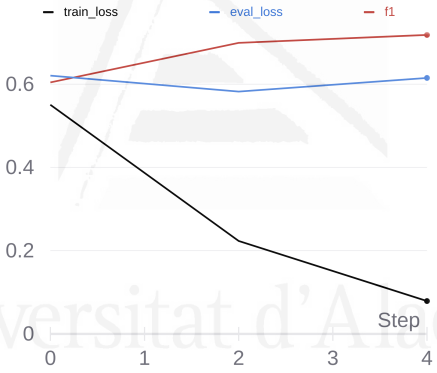
la pérdida en el conjunto de desarrollo subió ligeramente de 0,88 a 0,99. Con respecto a la métrica $F_1\,m$ el aumento es ínfimo de 79,3% a 80,3%, pero teniendo en cuenta la pérdida tan elevada en el conjunto de entrenamiento para la segunda iteración, es pertinente escoger el modelo entrenado con tres (3) iteraciones.



(a) Clasificador 1.



(b) Clasificador 2.



(c) Clasificador 3.

Figura 6.3: Clasificadores enviados al laboratorio CheckThat!. Pérdida usando la partición de entrenamiento y desarrollo, métrica $F_1\,m$ durante el entrenamiento.

Siguiendo con el análisis independiente de cada clasificador, la gráfica 6.3b, corresponde al clasificador de tres clases. Esta gráfica no indica que haya existido un sobreajuste y precisamente se escogió este entrenamiento debido a que las curvas de las pérdidas no mostraban un sobreajuste como ocurría con el resto de entrenamientos probados. Este modelo logra alcanzar valores bajos de pérdida tanto en el entrenamiento como en la evaluación, siendo tan solo de 2,75 en el conjunto de entrenamiento. La pérdida en el conjunto de desarrollo aumenta de 0,88 a 1,22 de la iteración 3 a la 4 pero por el contrario la métrica $F_1\,m$ aumenta considerablemente de 62,0% a 66,6%, de aquí que se haya escogido cuatro (4) iteraciones de entrenamiento como la más adecuada.

Por último, en la gráfica 6.3c se aprecia claramente que la curva de la pérdida de entrenamiento es mucho menor que la pérdida de evaluación, lo que hace indicar que este clasificador se sobreajustó desde la primera iteración. Para la

creación de estos clasificadores se siguieron pautas para evitar el sobreajuste de los modelos de clasificación. Evidentemente esto no se logró en el caso del tercer clasificador que claramente se sobreajustó. Sin embargo, este es el utilizado en la arquitectura debido a que todos los clasificadores entrenados se sobreajustaban rápidamente y los resultados en la métrica $F_1 m$ es mayor que en el resto de entrenamientos.

Las configuraciones presentadas en el laboratorio fueron escogidas después de analizar las variaciones de las pérdidas y la métrica $F_1 m$. En la tabla 6.10 se muestra la configuración de hiperparámetros para los tres clasificadores. Teniendo en cuenta la métrica $F_1 m$, el primer clasificador obtiene 80,3%, el segundo 66,6% y el tercero 72,7%.

Tabla 6.10: Configuración de hiperparámetros obtenida para cada uno de los clasificadores de la arquitectura dividida en etapas.

	Valores		
Hiperparámetros	**Primer**	**Segundo**	**Tercero**
Tamaño de lote	2	2	4
Tasa de entrenamiento	2e-5	2e-5	1e-5
Tasa de *dropout*	0,2	0,2	0,2
Número de iteraciones para entrenar	3	4	3

Las particiones de entrenamiento y desarrollo son subconjuntos de las mostradas en la tabla 6.2 debido a que solo se utilizan los ejemplos correspondientes a las etiquetas que deben determinar los clasificadores. La primera y la tercera clasificación son binarias y obtienen buenos resultados en la métrica $F_1 m$, siendo un poco menor para el tercer clasificador para el cual se cuentan con muy pocos ejemplos de entrenamiento, fundamentalmente para la clase *other*. Con respecto al segundo clasificador los resultados de la misma métrica son algo menores, pero son aceptables para una clasificación de tres clases.

El experimento 3 obtuvo los mejores resultados, así que fue usado para predecir los ejemplos en la partición de prueba. El equipo GPLSI ocupó el puesto 16 en esta subtarea, teniendo un amplio margen de mejora. Los experimentos realizados durante la realización del laboratorio no lograron obtener un sistema competitivo para la subtarea.

Con el objetivo de entender las causas del discreto resultado y a su vez proponer posibles mejoras del sistema de clasificación para esta subtarea, se realizan otros experimentos al finalizar el laboratorio CheckThat!. Primeramente se incluyen las predicciones sobre el corpus de prueba de los tres experimentos realizados para conformar el sistema definitivo, ver tabla 6.11.

Los resultados usando la partición de prueba contradicen los alcanzados con la partición de desarrollo creada. El experimento 2, el cual realiza la optimización de hiperparámetros alcanza los mejores resultados, con una ventaja

de 0,5 con respecto al experimento 3 que fue el enviado como sistema a evaluar. La diferencia del experimento 3 con el 2 no era demasiado grande evaluado sobre la partición de desarrollo, pero no dejan de llamar la atención los resultados obtenidos sobre la partición de prueba. Una posible y habitual respuesta a este comportamiento es que el experimento con tres clasificadores se haya sobreajustado a la partición de entrenamiento sin lograr generalizar el aprendizaje de cada etiqueta.

Tabla 6.11: Resultados de los experimentos de la subtarea 3A con el modelo RoBERTa sobre la partición de prueba.

Experimento	**F_1 Score**				**$F_1 m$**
	False	**Partially False**	**True**	**Other**	
1 - Sistema baseline	48,0	**40,5**	12,0	20,0	30,3
2 - Optimización de hiperparámetros	**53,0**	40,3	14,6	**35,8**	**35,9**
3 - Modelo con tres clasificadores	50,5	28,5	**24,2**	19,8	30,8

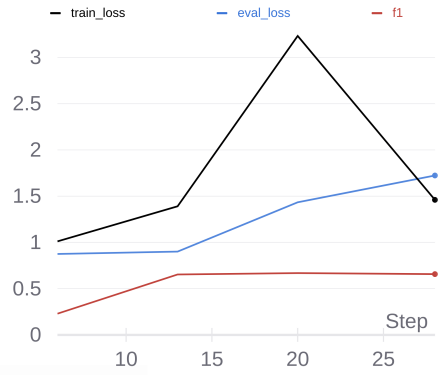
Siguiendo los principios del laboratorio Checkthat! para desarrollar los sistemas, se volvió a entrenar cada uno de los clasificadores independientemente con 60 inicializaciones aleatorias de los modelos de clasificación. Solo en el caso del tercer clasificador se modificaron algunos hiperparámetros para intentar ralentizar el sobreajuste de este clasificador. Se disminuyó el tamaño de lote de 4 a 2, la tasa de aprendizaje de 1e-5 a 2e-5 y solo se entrenó el clasificador durante 3 iteraciones. Similar a la figura anterior, en la figura 6.4 se muestran la evolución de las curvas para poder decidir qué etapa del entrenamiento se ajusta mejor a la tarea.

Las gráficas de la figura 6.4 representan los clasificadores entrenados posteriormente a la competición. En la gráfica de la figura 6.4a se observa como la pérdida usando el conjunto de entrenamiento bajó desde 5,64 hasta 2,77 en la última etapa de entrenamiento, siendo un poco menor que el modelo enviado al laboratorio. Al mismo tiempo la curva de pérdida en el conjunto de desarrollo bajó ligeramente de 1,06 a 1,02 lo que indica que no existe sobreajuste en la última iteración. La métrica $F_1 m$ por su parte tuvo un aumento significativo de 73,0% a 81,0% y junto con la disminución en ambas curvas de pérdida, hacen perfecto escoger el modelo entrenado con tres (3) iteraciones.

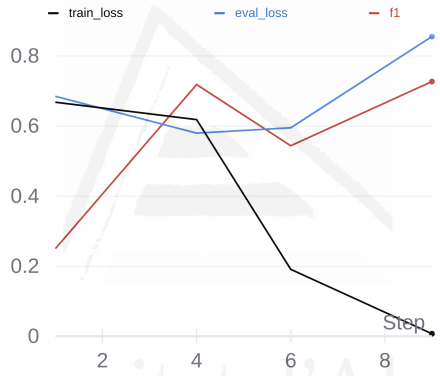
La gráfica 6.4b, corresponde al clasificador de tres clases. Al contrario del entrenamiento enviado al laboratorio, en este caso la gráfica muestra que el modelo empezó a sobreajustarse de la iteración 3 a la 4, con una disminución repentina de la pérdida en la curva de entrenamiento de 3,22 a 1,46. Al mismo tiempo la pérdida en la curva de desarrollo aumenta de 1,43 a 1,72 y la métrica $F_1 m$ disminuye de 66,8% a 65,7%. Este modelo tiene una peculiaridad con respecto al anterior y es que la métrica $F_1 m$ se logra mantener en valores altos desde la segunda iteración con un valor de 65,3% y alcanzando el mayor valor

en la tercera iteración. De aquí que se haya elegido este modelo en la iteración 3, para sustituir el anterior.

(a) Clasificador 1.



(b) Clasificador 2.



(c) Clasificador 3.

Figura 6.4: Clasificadores entrenados posterior al laboratorio. Pérdida usando la partición de entrenamiento y desarrollo, métrica $F_1\,m$ durante el entrenamiento.

Por último, en la gráfica 6.4c se encuentra un modelo que en la dos primeras iteraciones no se sobreajustó. La curva de pérdida en el conjunto de entrenamiento es ligeramente superior a la del conjunto de desarrollo. Coincide que en esa segunda iteración la métrica $F_1\,m$ tiene un valor elevado de 71,8%, ligeramente inferior que el modelo escogido para enviar al laboratorio con 72,7%, pero este último se encontraba sobreajustado desde la primera iteración.

La tabla 6.12 muestra los resultados de la arquitectura de clasificación, primero sobre la partición de desarrollo y posteriormente sobre la de prueba. Como se puede apreciar los resultados en la partición de desarrollo son bastante similares a los obtenidos por el sistema que fue enviado al laboratorio. Sin embargo, en la partición de prueba los resultados son sustancialmente mayores en este experimento. La métrica $F_1\,m$ alcanza un valor de 41,8%. Teniendo en cuenta los resultados obtenidos por clases, todas las clases mejoran sus resultados siendo mucho más significativos en las clases minoritarias (*other* y *true*), las que estaban siendo clasificadas por un clasificador totalmente sobreajustado.

Tabla 6.12: Resultados de los experimentos de la subtarea 3A con el modelo RoBERTa sobre la partición de desarrollo y prueba.

Experimento	**F_1 Score**				**$F_1 m$**
	False	**Partially False**	**True**	**Other**	
Desarrollo					
Modelo con tres clasificadores	83,4	59,8	56,1	23,5	55,7
Prueba					
Modelo con tres clasificadores	57,7	36,6	47,3	25,6	41,8

En la subtarea 3A, a diferencia de la 1A, el conjunto de datos fue liberado en tres partes con un tiempo reducido antes de la fecha límite. Claramente esto comprometió el tiempo de experimentación para la tarea. Con el análisis posterior se pudo validar la pertinencia de la arquitectura de detección dividida en etapas.

6.4 Conclusiones

Se abordaron dos tareas abiertas sobre la verificación automática de hechos en el contexto del laboratorio CheckThat! en su cuarta edición. Estas solo representan una parte dentro del complejo problema de la verificación automática de hechos.

Con respecto a la tarea 1A los modelos de clasificación propuestos para idioma español e inglés obtuvieron resultados notables, posicionándose en los primeros puestos de la subtarea. Se realiza una profunda experimentación utilizando técnicas de reconocido prestigio dentro la comunidad científica.

Aún así, los resultados alcanzados tienen un amplio rango de mejoras incluyendo modelos de lenguajes más específicos, por ejemplo entrenados sobre textos en Twitter o incluso modelos multilingües para poder crear sistemas más generales que permitan detectar tuits que deban ser chequeados para varios idiomas. Otra posible mejora puede ser encontrar características externas que apoyen a modelos más complejos como el propuesto.

En la subtarea 3A se utilizó un clasificador dividido en tres etapas para abordar el desbalance de clases en el conjunto de entrenamiento. Para crear este clasificador se obtiene primeramente una jerarquía de clases, se agrupan la clases según la cantidad de ejemplos de entrenamiento en clases ficticias, diferenciándose del utilizado en el capítulo 4 que agrupa las clases según su similitud semántica.

Sin embargo, los resultados alcanzados en la subtarea 3A no son demasiado significativos. En esta subtarea se aprecia una diferencia marcada entre la experimentación realizada para desarrollar el sistema y los resultados prediciendo la

partición de prueba.

Como se evidencia en los experimentos posteriores al laboratorio Checkthat!, el clasificador enviado obtenía buenos resultados en la partición de desarrollo porque estaba sobreajustando su entrenamiento. Se aplicaron técnicas para reducir este problema, además de analizar los entrenamientos realizados para evitar que los modelos se sobreajusten.

Siguiendo los mismos principios planteados por el laboratorio, pero con una experimentación más extensa se lograron alcanzar mejores resultados que situarían al equipo GPLSI en un hipotético séptimo puesto del decimosexto obtenido. En investigaciones futuras se experimentará con la utilización de clasificadores *ensemble* con el fin de obtener sistemas más potentes para este tipo de clasificación.

Se encontró que los clasificadores creados utilizando los modelos de lenguaje RoBERTa y BETO tiene una alta varianza en sus resultados en dependencia de la semilla que se utilice para inicializar los pesos del modelo. Es recomendable realizar entrenamientos sucesivos con la misma configuración de hiperparámetros, variando únicamente el valor de esta semilla y escoger el modelo que mejor se ajuste a la partición de desarrollo pero que exhiba sobreajuste.

El laboratorio CheckThat! resulta un entorno adecuado para consultar investigaciones notables en el campo de la verificación automática de hechos. La participación en las subtareas 1A y 3A ha permitido adentrarnos en esta área con soluciones concretas a problemas específicos del estado del arte, en los que habrá que seguir profundizando en trabajos futuros ya que puede considerarse un factor complementario e importante en la detección de desinformación.

CAPÍTULO 7

Conclusiones y trabajos futuros

7.1 Conclusiones generales

La proliferación de las infraestructuras móviles y sus herramientas de comunicación y de intercambio de información han cambiado drásticamente el comportamiento de las sociedades modernas en torno al análisis y al consumo de noticias, convirtiéndose en un escenario propicio para manejar la opinión pública de manera irresponsable por medio de información engañosa, lo que se refleja en formas tales como noticias falsas, titulares engañosos y ciberanzuelos.

Las investigaciones en torno a la búsqueda de soluciones a esta problemática convocan a varias disciplinas científicas que intentan captar la esencia misma de un trabajo de verificación desarrollado tradicionalmente por periodistas y su traducción a una algoritmia efectiva y convincente para la detección de información engañosa.

La verificación manual, aunque termina considerándose un patrón de referencia, no se puede concretar en la práctica para un caudal informativo que crece exponencialmente y se propaga de manera descontrolada causando efectos nocivos e inmediatos a los consumidores. Por ello, las aproximaciones tienden a solucionar el problema y actuar sobre sus causas a través de herramientas automatizadas que van resolviendo problemas parciales y escalando hacia soluciones más complejas.

El foco de esta investigación se ha acercado a las tareas de detección de titulares engañosos, detección de posturas, verificación automática de hechos, detección de noticias falsas y detección de contradicciones. Comúnmente estas tareas se reducen a la solución de problemas complejos de clasificación de textos.

Aunque la tipología de soluciones se encuentra situada en el entorno del PLN, ML y DL , la diversidad de técnicas, algoritmos y conceptos, no siempre esencialmente excluyentes, trae consigo cierta complejidad adicional para ubi-

carse en el estado del arte en el área. El método tradicional con que se conduce una investigación en estas tareas parte de analizar en profundidad el problema particular, la tipología de soluciones habituales, conjuntos de datos (corpus), así como sus anotaciones, con el fin de entrenar enfoques de ML o DL con el objeto de aprender patrones generalizables para alcanzar una clasificación en específico.

Se observa un interés marcado en el par titular-noticia, en la búsqueda de posibles contradicciones entre ellos o en la detección de posibles posturas en un titular. Para ello, es común mezclar enfoques lexicográficos, sintácticos y semánticos. A este contexto se suman ciertas restricciones prácticas que impiden manejar de forma exhaustiva toda la información disponible e incapacidades de los algoritmos de clasificación para manejar el contenido completo de la noticia.

Teniendo en cuenta el símil de la pirámide invertida que se aplica en la redacción de noticias, se demuestra experimentalmente que es posible disminuir la longitud de entrada de los datos a procesar mediante técnicas de resúmenes automáticos. Sin embargo, no siempre es posible dirimir con certeza las circunstancias de aplicación de estas técnicas y el tipo de algoritmo para la obtención de resúmenes.

Teniendo en cuenta los modelos de lenguaje utilizados se valida que son dos alternativas competitivas del estado del arte. Sin embargo, sigue siendo importante llevar a cabo experimentaciones profundas para el desarrollo de soluciones que utilicen estos modelos. Se corroboran los hallazgos obtenidos por otras investigaciones que plantean la alta varianza que pueden presentar los entrenamientos utilizando modelos como RoBERTa y BETO.

7.2 Principales aportaciones

Las aportaciones de la investigación doctoral desarrollada se enmarcan en los siguientes aspectos:

- **Reformulación de problemas de clasificación de múltiples clases en problemas de clasificación jerárquica:**

  A partir de la similitud semántica entre las etiquetas de clasificación o para abordar el desbalance de conjuntos de datos se representa un problema de clasificación de múltiples clases como un problema de clasificación jerárquica, por medio de la introducción de clases ficticias. Esto posibilita la utilización de características específicas en cada uno de los niveles de la jerarquía, mejorando las clasificaciones planas dentro de la tarea. Con las experimentaciones realizadas en esta tesis se aprecia que muchos problemas de clasificación se ven beneficiados por esta técnica, siendo oportuno valorar su experimentación en otras tareas.

- **Introducción de técnicas de resúmenes de textos en sustitución de contenidos de noticias:**

Se utilizaron resúmenes automáticos como sustitutos del contenido completo de la noticia, obteniéndose resultados competitivos en problemas de clasificación. Con ello se evaden algunas limitaciones de los algoritmos de DL para estas tareas. Se demostró que la ventaja de esta técnica está estrechamente relacionada con los tamaños de los textos y la tarea específica en la que sean aplicados.

- **Diseño de una arquitectura de detección de titulares engañosos:**

  Se diseñó una arquitectura específica para la detección de titulares engañosos, que expone ciertos elementos de flexibilidad y escalabilidad. Esta arquitectura se construyó para resolver un problema de clasificación jerárquica. Bajo los principios de esta arquitectura se han instanciado dos prototipos novedosos variando algunos de sus componentes internos, lo que valida la flexibilidad que presenta ante modificaciones al menos en el contexto de la detección de titulares engañosos.

  Ambos prototipos utilizan enfoques de resúmenes para reducir la noticia a la información esencial. Se utiliza el modelo de lenguaje basado en arquitectura Transformer (RoBERTa). Se distinguen entre sí en el empleo de características específicas incorporadas a las etapas de clasificación y en el tipo de algoritmo de obtención de resúmenes empleado. Con ello, se demuestra, la flexibilidad de la arquitectura y su posible ajuste a tareas específicas.

- **Obtención de un conjunto de datos para la detección de titulares engañosos:**

  Se desarrolló un conjunto de datos (ES_Headline_Contradiction) en idioma español para la tarea de detección de titulares engañosos desde la perspectiva de las contradicciones entre textos. El conjunto anota la relación semántica entre el titular y el contenido de la noticia (*compatible*, *contradiction* y *unrelated*). Además, se añade el tipo de contradicción en caso de existir. La anotación de tipos de contradicciones puede ser utilizada para mejorar la explicabilidad de los modelos de clasificación utilizados. Se logra automatizar el proceso de introducción de determinadas contradicciones en los titulares de noticias. Según el conocimiento del autor no se cuenta con un recurso similar a este en idioma español.

- **Resultados experimentales sobre la arquitectura y el conjunto de datos desarrollado:**

  Los experimentos realizados aportan evidencias de las posibilidades de explotación de la arquitectura y el conjunto de datos ES_Headline_Contradiction para la detección de titulares engañosos y detección de contradicciones en español. La arquitectura de detección de titulares engañosos desarrollada obtiene los mejores resultados sobre este corpus evidenciando nuevamente su validez para la tarea.

- **Avances en tareas aisladas relacionadas con la verificación automática de hechos**:

  Se desarrollan soluciones en la detección de afirmaciones que deban ser chequedas, obteniéndose resultados notables para los idiomas español e inglés. En la detección de noticias falsas se experimenta con la utilización de una arquitectura dividida en etapas, desarrollada siguiendo principios similares que la diseñada para detectar titulares engañosos. Sin embargo, aunque los resultados no son comparables con los encontrados en el estado del arte, se mejoran los obtenidos por un clasificador plano.

7.3 Trabajos futuros

Un área a profundizar en futuras investigaciones es la verificación automática de hechos. Con las investigaciones desarrolladas en este trabajo de tesis que solucionan determinados problemas dentro de esta tarea, se ha entendido la importancia que presenta para la verificación de información engañosa como concepto de mayor generalidad.

Una propuesta interesante y novedosa para atacar el problema de la información engañosa en medios digitales podría ser contar con un sistema automático de alerta ante este tipo de fenómenos. Las redes sociales y medios digitales por lo general no dedican suficientes recursos a detectar este tipo de información. Este servicio podría indicar a usuarios comunes de estas tecnologías cuando exista la presencia de información que merece ser contrastada o al menos dudar de sus afirmaciones. Este sistema de alerta no debe objetar o censurar la información a la que acceden los usuarios pero sí podría indicar en el momento que esa información no sea confiable, presente indicios de contradicciones en su contenido o la relación entre el titular y el contenido de la noticia no sea coherente.

La ampliación del conjunto de datos ES_Headline_Contradiction con las contradicciones menos representativas y otras que no han sido anotadas es una tarea que debe ser abordada para lograr cubrir el amplio espectro de las contradicciones entre textos.

Otro campo de investigación que se pretende abordar en un futuro próximo es la de la explicabilidad de los modelos computacionales. En los últimos años, influenciado por garantizar la integridad y equidad en decisiones que se confían a sistemas computacionales, urge entender el fundamento de estas decisiones. La explicabilidad tiene un interés especial si se utilizan herramientas conexionistas como es el caso de las redes neuronales cuyo funcionamiento es de caja negra. Por ejemplo, los sistemas automatizados que deciden si a una fuente se le debe otorgar un crédito bancario, habitualmente no son capaces de explicar el motivo de una decisión. Este tipo de explicación podría ser útil especialmente en caso de auditorías, dotando al enfoque de un atributo de seguridad que puede serle imprescindible en determinadas situaciones de cuestionamientos.

Analizando las soluciones planteadas en este trabajo, se detecta claramente la ventaja que podría aportar un sistema de detección de contradicciones entre textos, que adicionalmente a la detección indique el tipo de contradicción y una explicación al respecto. El atributo de explicabilidad podría añadir credibilidad a la arquitectura de detección de titulares engañosos diseñada.

7.4 Publicaciones

En el transcurso de esta investigación se realizaron las siguientes publicaciones en revistas y congresos:

- Sepúlveda-Torres, R., Vicente, M., Saquete, E., Lloret, E., & Palomar, M. (2021). HeadlineStanceChecker: Exploiting summarization to detect headline disinformation. Journal of Web Semantics, 71, 100660.
- Sepúlveda-Torres, R., Bonet-Jover, A., & Saquete, E. (2021). Here Are the Rules: Ignore All Rules: Automatic Contradiction Detection in Spanish. Applied Sciences, 11(7), 3060.
- Sepúlveda-Torres, R., Vicente, M., Saquete, E., Lloret, E., & Palomar, M. (2021). Exploring Summarization to Enhance Headline Stance Detection. In International Conference on Applications of Natural Language to Information Systems (pp. 243-254). Springer, Cham.
- Sepúlveda-Torres, R., & Saquete, E. (2021). GPLSI team at CLEF CheckThat! 2021: fine-tuning BETO and RoBERTa. CLEF (Working Notes) 2021: (pp. 628-638). http://ceur-ws.org/Vol-2936/paper-52.pdf.
- Vicente, M., Sepúlveda-Torrres, R., Barros, C., Saquete, E., & Lloret, E. (2021). Can Text Summarization Enhance the Headline Stance Detection Task? Benefits and Drawbacks. In International Conference on Document Analysis and Recognition (pp. 53-67). Springer, Cham.
- Alonso-Reina, A., Sepúlveda-Torres, R., Saquete, E., & Palomar, M. (2019). Team GPLSI. approach for automated fact checking. In Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER) (pp. 110-114).

APÉNDICE A

Demo de detección de titulares engañosos

Se puede consultar un prototipo de la arquitectura **HeadlineStanceChecker** que realiza detección de titulares engañosos en idioma inglés, en `https://headlinechecker.demos.gplsi.es/`. El prototipo exhibe funcionalidades básicas que se escalarán a modelos entrenados sobre idioma español. Este prototipo utiliza el algoritmo **TextRank Summarizer** para resumir el contenido de la noticia en caso necesario.

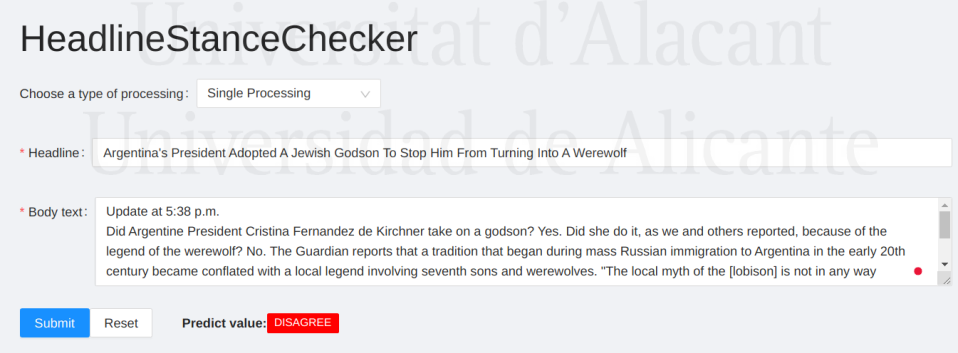


Figura A.1: Interfaz de usuario del prototipo HeadlineStanceChecker (en desarrollo).

APÉNDICE B

Guía de anotación del conjunto de datos

B.1 Introducción

Esta guía ha sido creada para la anotación de contradicciones entre titulares y contenidos de noticias. Este corpus puede ser una herramienta importante para enfrentar la desinformación en noticias. La detección automática de información contradictoria puede permitir detectar titulares no relacionados con sus contenidos. Se explicará la estructura del conjunto de datos y se pondrán algunos ejemplos de cada tipo de contradicción.

El conjunto de datos se encuentra en formato json y presenta la siguiente estructura:

- id: número de identificación de la noticia.
- headline: titular de la noticia.
- body: parte que incluye todo el contenido de la noticia.
- date: fecha de creación de la noticia.
- label: etiqueta que indica la relación entre un titular y el contenido de la noticia. (*compatible*, *contradiction* y *unrelated*).
- label_contradiction: etiqueta que indica el tipo de contradicción entre el titular y el contenido de la noticia. Esta etiqueta contiene varios valores que serán definidos en detalle en esta guía. (*num*, *ant*, *str*, *neg* y *fac*).

B.2 Anotación manual del conjunto de datos

B.2.1 Modificar manualmente el titular de la noticia

En esta etapa de anotación se persigue el objetivo de incluir modificaciones específicas en el titular para contradecir el contenido de la noticia. Estas modificaciones cambian la semántica de la oración y hacen el titular totalmente contradictorio con lo que se plantea en el contenido. Las técnicas utilizadas para modificar los titulares son las que siguen:

- negación: Se deben adicionar adverbios de negación (no, nunca, etc.) en una posición adecuada de cada oración, esta modificación se puede encontrar negando el verbo principal pero también en otras posiciones. Es importante analizar la oración completa para determinar si podrían ser oraciones incompatibles con este tipo de anotación.

  Ejemplo 1:

  - Titular original: Garzón defiende ante la Asamblea de IU **no limitar** la unidad a las elecciones.
  - Titular modificado: Garzón defiende ante la Asamblea de IU **limitar** la unidad a las elecciones.

  Ejemplo 2:

  - Titular original: Próximo objetivo, presupuesto.
  - Titular modificado: no se puede modificar (falta el verbo).

- antónimo: Se debe encontrar una acción, sustantivo de la oración y proceder a cambiarla por un antónimo. El cambio debe inducir una contradicción. Para encontrar antónimos de verbos se pueden consultar sitios web especializados como `https://www.wordreference.com/sinonimos/`.

  Ejemplo 1:

  - Titular original: El petróleo de Texas abre con una **bajada** del 5,9%, hasta 39,22 dólares.
  - Titular modificado: El petróleo de Texas abre con una **subida** del 5,9%, hasta 39,22 dólares.

  Ejemplo 2:

  - Titular original: El gobierno **ha aprobado** ayudas por 45 millones para proyectos piloto de 5G.
  - Titular modificado: El gobierno **rechaza** ayudas por 45 millones para proyectos piloto de 5G.

- numérico: Se deben modificar elementos numéricos o de fecha en la oración (un año, un mes, un día de la semana, un por ciento, etc). Esta modificación sola es posible en titulares que incluyen fechas o números.

  Ejemplo 1:

  - Titular original: Accionistas de Nissan aprueban la incorporación de **cuatro** nuevos consejeros.
  - Titular modificado: Accionistas de Nissan aprueban la incorporación de **diez** nuevos consejeros.

  Ejemplo 2:

  - Titular original: Los presidentes nacionalistas y del PP critican la descoordinación y la vuelta al trabajo.
  - Titular modificado: no se puede modificar (falta cifra o fecha).

- Estructura: Se deben invertir los elementos, entidades nombras de una oración, cambiando el orden o reemplazándolos (por ejemplo, al cambiar el sujeto y el objeto).

  Ejemplo 1:

  - Titular original: **Japón y EEUU** han hecho un gran progreso hacia un acuerdo comercial.
  - Titular modificado: **Cuba y EEUU** han hecho un gran progreso hacia un acuerdo comercial.

  Ejemplo 2:

  - Titular original: **EEUU** multa a **Citibank** con 400 millones de dólares por deficiencias significativas.
  - Titular modificado: **Citibank** multa a **EEUU** con 400 millones de dólares por deficiencias significativas.

- Fáctica: Se deben introducir palabras modales o verbos no fácticos que hagan que la frase no este asegurando hechos, sino mostrando la posibilidad de que pasen o viceversa.

  Ejemplo 1:

  - Titular original: Renfe **empezará a vender** en enero billetes de bajo coste para el AVE Madrid-Barcelona.
  - Titular modificado: Renfe **vende** billetes de bajo coste en enero para el AVE Madrid-Barcelona.

  Ejemplo 2:

- Titular original: EL FMI **aprueba** el desembolso de 498,4 millones de dólares para Ecuador.
- Titular modificado: EL FMI **considera aprobar** un desembolso de 498,4 millones de dólares para Ecuador.

B.2.2 Anotar la relación semántica entre el titular y el contenido de la noticia

Una vez que los titulares han sido modificados, el próximo paso consiste en la clasificación de la relación entre el titular y el contenido de la noticia. Los anotadores deben detectar las relaciones semánticas y las contradicciones que han sido anotadas para clasificarlas en *compatible* (cuando la información del titular y el cuerpo coinciden) o *contradiction* (cuando la información entre ambos es contradictoria).

Cuando los ejemplos son clasificados con la etiqueta *contradiction*, los anotadores tienen que indicar el tipo de la relación de contradicción con los valores antes descritos: neg (negación), ant (antónimo), num (numérico), str (estructura) o fac (fáctica).

Bibliografía

Agez, R., Bosc, C., Lespagnol, C., Mothe, J., y Petitcol, N. (2018). Irit at checkthat! 2018. En *9th conference and labs of the evaluation forum, living labs (clef 2018)* (Vol. 2125).

Agrawal, R., Rajagopalan, S., Srikant, R., y Xu, Y. (2003). Mining newsgroups using networks arising from social behavior. En *Proceedings of the 12th international conference on world wide web* (pp. 529–535).

Ainslie, J., Ontañón, S., Alberti, C., Cvicek, V., Fisher, Z., Pham, P., Ravula, A., Sanghai, S., Wang, Q., y Yang, L. (2020). ETC: Encoding Long and Structured Inputs in Transformers. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 268–284.

Alam, F., Shaar, S., Dalvi, F., Sajjad, H., Nikolov, A., Mubarak, H., Martino, G. D. S., Abdelali, A., Durrani, N., y Darwish, K. (2020). Fighting the covid-19 infodemic: modeling the perspective of journalists, fact-checkers, social media platforms, policy makers, and the society. *arXiv preprint arXiv:2005.00033*.

ALDayel, A., y Magdy, W. (2021). Stance detection on social media: State of the art and trends. *Information Processing Management*, *58*(4), 102597.

Al-Ghadir, A. I., Azmi, A. M., y Hussain, A. (2021). A novel approach to stance detection in social media tweets by fusing ranked lists and sentiments. *Information Fusion*, *67*, 29–40.

Alkhalifa, R., Yoong, T., Kochkina, E., Zubiaga, A., y Liakata, M. (2020). Qmulsds at checkthat! 2020: determining covid-19 tweet check-worthiness using an enhanced ct-bert with numeric expressions. *arXiv preprint arXiv:2008.13160*.

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., y Kochut, K. (2017). Text summarization techniques: A brief survey. *International Journal of Advanced Computer Science and Applications*, *8*(10).

Allcott, H., y Gentzkow, M. (2017). Social media and fake news in the 2016 election. *Journal of economic perspectives*, *31*(2), 211–36.

Andreas Hanselowski, B. S., Avinesh PVS, y Caspelherr, F. (2017). *Description of*

the system developed by team athene in the FNC-1. https://github.com/hanselowski/athene_system,lastaccessedon29/05/20.

Apuke, O. D., y Omar, B. (2021). Fake news and COVID-19: modelling the predictors of fake news sharing among social media users. *Telematics and Informatics, 56*, 101475.

Atanasova, P., Simonsen, J. G., Lioma, C., y Augenstein, I. (2020). Generating fact checking explanations. En *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 7352–7364). Association for Computational Linguistics.

Augenstein, I., Lioma, C., Wang, D., Lima, L. C., Hansen, C., Hansen, C., y Simonsen, J. G. (2019). Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims. *arXiv preprint arXiv:1909.03242*.

Augenstein, I., Rocktäschel, T., Vlachos, A., y Bontcheva, K. (2016). Stance detection with bidirectional conditional encoding. *arXiv preprint arXiv:1606.05464*.

Ayala, C., Aranda, C., y Galar, M. (2021). Multi-class strategies for joint building footprint and road detection in remote sensing. *Applied Sciences (Switzerland), 11*(18).

Azman, S. N., Ishak, I., Sharef, N. M., y Sidi, F. (2017). Towards an enhanced aspect-based contradiction detection approach for online review content. *Journal of Physics: Conference Series, 892*, 012006.

Babakar, M., Bakos, N., Daumé, H., Mantzarlis, A., Seddah, D., Vlachos, A., y Wardle, C. (2016). *Fake news challenge - i.* http://www.fakenewschallenge.org/ , last accessed on 21/01/21.

Babbar, R., Partalas, I., Gaussier, E., y Amini, M.-R. (2013). On Flat versus Hierarchical Classification in Large-Scale Taxonomies. *Advances in Neural Information Processing Systems, 26*.

Bahdanau, D., Cho, K., y Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Baird, S., Sibley, D., y Pan, Y. (2017). *Talos targets disinformation with fake news challenge victory*. https://blog.talosintelligence.com/2017/06/talos-fake-news-challenge.html,lastaccessedon20/02/21.

Baly, R., Mohtarami, M., Glass, J., Màrquez, L., Moschitti, A., y Nakov, P. (2018). Integrating stance detection and fact checking in a unified corpus. En *Proceedings of the 2018 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 2 (short papers)* (pp. 21–27). Association for Computational Linguistics.

Balyan, R., McCarthy, K. S., y McNamara, D. S. (2020). Applying Natural Lan-

guage Processing and Hierarchical Machine Learning Approaches to Text Difficulty Classification. *International Journal of Artificial Intelligence in Education*, *30*(3), 337–370.

Banko, M., Mittal, V. O., y Witbrock, M. J. (2000). Headline generation based on statistical translation. En *Proceedings of the 38th annual meeting on association for computational linguistics* (pp. 318–325). Association for Computational Linguistics.

Barreiro, J. P. (2019). Improving reading comprehension of narrative texts through summaries. *PhD Thesis. Universidad Casa Grande*.

Barros, C., y Lloret, E. (2019). HanaNLG: A Flexible Hybrid Approach for Natural Language Generation. En *Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing*.

Barros, C., Lloret, E., Saquete, E., y Navarro-Colorado, B. (2019). NATSUM: Narrative abstractive summarization through cross-document timeline generation. *Information Processing & Management*, *56*(5), 1775–1793.

Bastos, M. T., y Mercea, D. (2019). The Brexit botnet and user-generated hyperpartisan news. *Social Science Computer Review*, *37*(1), 38-54.

Bello, I., Pham, H., Le, Q. V., Norouzi, M., y Bengio, S. (2016). Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*.

Beltagy, I., Peters, M. E., y Cohan, A. (2020). Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Benson, R., y Hallin, D. (2007). How states, markets and globalization shape the newsthe french and us national press, 1965-97. *European Journal of Communication*, *22*, 27-48.

Beyan, C., y Fisher, R. (2015). Classifying imbalanced data sets using similarity based hierarchical decomposition. *Pattern Recognition*, *48*(5), 1653–1672.

Biewald, L. (2020). *Experiment tracking with weights and biases.* Descargado de `https://www.wandb.com/` (Software available from wandb.com)

Bilmes, J. A., y Kirchhoff, K. (2003). Factored Language Models and Generalized Parallel Backoff. En *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 4–6). Association for Computational Linguistics.

Bird, S., Klein, E., y Loper, E. (2009). *Natural language processing with python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Blom, J. N., y Hansen, K. R. (2015). Click bait: Forward-reference as lure in online news headlines. *Journal of Pragmatics*, *76*, 87–100.

Bojanowski, P., Grave, E., Joulin, A., y Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146.

Bondielli, A., y Marcelloni, F. (2019). A survey on fake news and rumour detection techniques. *Information Sciences*, *497*, 38–55.

Bonet-Jover, A., Piad-Morffis, A., Saquete, E., Martínez-Barco, P., y Ángel García-Cumbreras, M. (2020). Exploiting discourse structure of traditional digital media to enhance automatic fake news detection. *Expert Systems with Applications*, 114340.

Boró, E. S., Tomás, D., Moreda, P., Martínez-Barco, P., y Palomar, M. (2020). Fighting post-truth using natural language processing: A review and open challenges. *Expert Systems with Applications*, *141*.

Bovet, A., y Makse, H. A. (2019). Influence of fake news in Twitter during the 2016 US presidential election. *Nature Communications*, *10(1):7*.

Bowman, S. R., Angeli, G., Potts, C., y Manning, C. D. (2015). A large annotated corpus for learning natural language inference. En *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 632–642). Association for Computational Linguistics.

Brown, S. A. (2018). The effects of explicit main idea and summarization instruction on reading comprehension of expository text for alternative high school students. *PhD Thesis. Utah State University*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., y Askell, A. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., y Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. En *Ecml pkdd workshop: Languages for data mining and machine learning* (pp. 108–122).

Cai, J., Li, J., Li, W., y Wang, J. (2018). Deeplearning model used in text classification. En *2018 15th international computer conference on wavelet active media technology and information processing (iccwamtip)* (pp. 123–126).

Canete, J., Chaperon, G., Fuentes, R., y Pérez, J. (2020). Spanish pre-trained bert model and evaluation data. *PML4DC at ICLR, 2020*.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Céspedes, M., Yuan, S., y Tar, C. (2018). Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Chen, C., Ye, W., Zuo, Y., Zheng, C., y Ong, S. P. (2019). Graph Networks as a Uni-

versal Machine Learning Framework for Molecules and Crystals. *Chemistry of Materials*, *31*(9), 3564–3572.

Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., y Inkpen, D. (2016). Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Chen, Q., Zhu, X., Ling, Z.-H., Inkpen, D., y Wei, S. (2017). Natural language inference with external knowledge. *arXiv preprint arXiv:1711.04289*, *42*, 43.

Chen, Y., Conroy, N. J., y Rubin, V. L. (2015). News in an online world: The need for an automatic crap detector. En *Proceedings of the 78th asis&t annual meeting: Information science with impact: Research in and for the community.* American Society for Information Science.

Chen, Y.-C., y Bansal, M. (2018). Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, *1*, 675–686.

Chesney, S., Liakata, M., Poesio, M., y Purver, M. (2017). Incongruent headlines: Yet another way to mislead your readers. En *Proceedings of natural language processing meets journalism* (pp. 56–61).

Ciampaglia, G. L., Shiralkar, P., Rocha, L. M., Bollen, J., Menczer, F., y Flammini, A. (2015). Computational Fact Checking from Knowledge Networks. *PLOS ONE*, *10*(6), e0128193.

Coca, L. G., Cusmuliuc, C.-G., y Iftene, A. (2019). Checkthat! 2019 uaics. En *Working notes of clef 2019-conference and labs of the evaluation forum* (Vol. 2380).

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, *20*(1), 37–46.

Colomina, C. (s.f.). *CIDOB - Coronavirus: Infodemia y desinformación.*

Conneau, A., Lample, G., Rinott, R., Williams, A., Bowman, S. R., Schwenk, H., y Stoyanov, V. (2018). XNLI: evaluating cross-lingual sentence representations. *CoRR*, *abs/1809.05053*.

Conroy, N. J., Rubin, V. L., y Chen, Y. (2015). Automatic deception detection: Methods for finding fake news. En *Proceedings of the 78th asis&t annual meeting: Information science with impact: Research in and for the community* (p. 82). American Society for Information Science.

Craswell, N. (2009a). Precision at n. En L. LIU y M. T. ÖZSU (Eds.), *Encyclopedia of database systems* (pp. 2127–2128). Springer US.

Craswell, N. (2009b). R-precision. En L. LIU y M. T. ÖZSU (Eds.), *Encyclopedia of database systems* (pp. 2453–2453). Springer US.

Dale, R. (2017). NLP in a post-truth world. *Natural Language Engineering*, *23*(2), 319–324.

De Cao, N., Aziz, W., y Titov, I. (2019). Question Answering by Reasoning Across Documents with Graph Convolutional Networks. En *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 2306–2317). Association for Computational Linguistics.

De Marneffe, M.-C., Rafferty, A. N., y Manning, C. D. (2008). Finding contradictions in text. *Proceedings of Association for Computational Linguistics*, 1039–1047.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., y Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. En *2009 ieee conference on computer vision and pattern recognition* (pp. 248–255).

Deng, J., Satheesh, S., Berg, A. C., y Li, F.-F. (2011). Fast and balanced: Efficient label tree learning for large scale object recognition. En *Nips* (Vol. 24, pp. 567–575).

Derczynski, L., Bontcheva, K., Liakata, M., Procter, R., Hoi, G. W. S., y Zubiaga, A. (2017). Semeval-2017 task 8: Rumoureval: Determining rumour veracity and support for rumours. *arXiv preprint arXiv:1704.05972*.

Dernoncourt, F., Ghassemi, M., y Chang, W. (2018). A repository of corpora for summarization. En *Proceedings of the eleventh international conference on language resources and evaluation*. European Language Resources Association (ELRA).

Devlin, J., Chang, M.-W., Lee, K., y Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. En *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Association for Computational Linguistics.

Dhar, R., Dutta, S., y Das, D. (2019). A hybrid model to rank sentences for checkworthiness. En *Working notes of clef 2019-conference and labs of the evaluation forum* (Vol. 2380).

Dias, P. (2014). From infoxicationto infosaturation: a theoretical overview of the cognitive and social effects of digital immersion. *Ámbitos. Revista Internacional de Comunicación, 24*.

Di Domenico, G., Sit, J., Ishizaka, A., y Nunan, D. (2021). Fake news, social media and marketing: A systematic review. *Journal of Business Research, 124*, 329–341.

Dijkman, R., y Wilbik, A. (2017). Linguistic summarization of event logs a practical approach. *Information Systems*, *67*, 114 - 125.

Dodge, J., Ilharco, G., Schwartz, R., Farhadi, A., Hajishirzi, H., y Smith, N. (2020). Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Dor, D. (2003). On newspaper headlines as relevance optimizers. *Journal of Pragmatics*, *35*, 695-721.

Dorr, B., Zajic, D. M., y Schwartz, R. M. (2003). Hedge trimmer: A parse-and-trim approach to headline generation. En *Hlt-naacl 2003*.

Doval Avendaño, M., y Domínguez Quintas, S. (2016). Los jóvenes españoles, habitantes de los medios: una propuesta de ayuno digital. En *Actas del i congreso internacional comunicación y pensamiento. comunicracia y desarrollo social (2016), p 1302-1317*.

Dragos, V. (2017). Detection of contradictions by relation matching and uncertainty assessment. En *Procedia computer science* (Vol. 112, pp. 71–80). Elsevier B.V.

Duan, Y., y Jatowt, A. (2019). Across-time comparative summarization of news articles. En *Proceedings of the twelfth acm international conference on web search and data mining* (p. 735743). Association for Computing Machinery.

Dulhanty, C., Deglint, J. L., Daya, I. B., y Wong, A. (2019). Taking a stance on fake news: Towards automatic disinformation assessment via deep bidirectional transformer language models for stance detection. *arXiv preprint arXiv:1911.11951*.

Ecker, U. K., Lewandowsky, S., Chang, E. P., y Pillai, R. (2014). The effects of subtle misinformation in news headlines. *Journal of experimental psychology: applied*, *20*(4), 323.

Engelen, J., Camp, G., van de Pol, J., y de Bruin, A. (2018). Teachers' monitoring of students' text comprehension: can students' keywords and summaries improve teachers' judgment accuracy? *Metacognition and learning*, *13*(3), 287–307.

Esmaeilzadeh, S., Peh, G. X., y Xu, A. (2019). Neural abstractive text summarization and fake news detection. *arXiv preprint arXiv:1904.00788*.

Estrada-Cuzcano, A., Alfaro-Mendives, K., y Saavedra-Vásquez, V. (2020). Disinformation and misinformation, post-truth and fake news: Conceptual precisions, differences, similarities and juxtapositions. *Informacion, Cultura y Sociedad*(42), 93–106.

Faulkner, A. (2014). Automated classification of stance in student essays: An ap-

proach using stance target information and the wikipedia link-based measure. En *The twenty-seventh international flairs conference*.

Favano, L., Carman, M. J., y Lanzi, P. L. (2019). Theearthisflat's submission to clef'19checkthat! challenge. En *Working notes of clef 2019-conference and labs of the evaluation forum* (Vol. 2380, pp. 1–11).

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Ferreira, R., de Souza Cabral, L., Lins, R. D., e Silva, G. P., Freitas, F., Cavalcanti, G. D., Lima, R., Simske, S. J., y Favaro, L. (2013). Assessing sentence scoring techniques for extractive text summarization. *Expert systems with applications*, *40*(14), 5755–5764.

Ferreira, W., y Vlachos, A. (2016). Emergent: a novel data-set for stance classification. En *Proceedings of the conference of the north american chapter of the association for computational linguistics* (pp. 1163–1168). Association for Computational Linguistics.

Freitas, A., y Carvalho, A. (2007). A tutorial on hierarchical classification with applications in bioinformatics. *Research and trends in data mining technologies and applications*, 175–208.

FullFact.org. (2016). *The state of automated factchecking (2016)*. `https://fullfact.org/blog/2016/aug/automated-factchecking/`, accedido el 15/01/2022.

Gabielkov, M., Ramachandran, A., Chaintreau, A., y Legout, A. (2016). Social clicks: What and who gets read on twitter? *ACM SIGMETRICS Performance Evaluation Review*, *44*, 179-192.

Gambhir, M., y Gupta, V. (2017). Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, *47*(1), 1–66.

Gao, D., Yang, W., Zhou, H., Wei, Y., Hu, Y., y Wang, H. (2020). Deep hierarchical classification for category prediction in e-commerce system. *arXiv preprint arXiv:2005.06692*.

Gavrilov, D., Kalaidin, P., y Malykh, V. (2019). Self-attentive model for headline generation. En *Advances in information retrieval* (pp. 87–93). Springer International Publishing.

Ghanem, B., y Gómez, M. M., Pardo, F. M. R., y Rosso, P. (2018). Upv-inaoe - check that: Preliminary approach for checking worthiness of claims. En *9th conference and labs of the evaluation forum, living labs (clef 2018)* (Vol. 2125).

Ghosh, S., Singhania, P., Singh, S., Rudra, K., y Ghosh, S. (2019). Stance detection in web and social media: a comparative study. En *International conference of the cross-language evaluation forum for european languages* (pp. 75–87).

Gorrell, G., Kochkina, E., Liakata, M., Aker, A., Zubiaga, A., Bontcheva, K., y Derczynski, L. (2019). Semeval-2019 task 7: Rumoureval, determining rumour veracity and support for rumours. En *Proceedings of the 13th international workshop on semantic evaluation* (pp. 845–854).

Grandini, M., Bagli, E., y Visani, G. (2020). Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*.

Grinberg, N., Joseph, K., Friedland, L., Swire-Thompson, B., y Lazer, D. (2019). Fake news on twitter during the 2016 us presidential election. *Science*, *363*(6425), 374–378.

Grusky, M., Naaman, M., y Artzi, Y. (2018). Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv preprint arXiv:1804.11283*.

Gupta, A., y Srikumar, V. (2021). X-fact: A new benchmark dataset for multilingual fact checking. *arXiv preprint arXiv:2106.09248*.

Hanselowski, A., PVS, A., Schiller, B., Caspelherr, F., Chaudhuri, D., Meyer, C. M., y Gurevych, I. (2018). A retrospective analysis of the fake news challenge stance-detection task. En *Proceedings of the 27th international conference on computational linguistics* (pp. 1859–1874). Association for Computational Linguistics.

Hanselowski, A., Zhang, H., Li, Z., Sorokin, D., Schiller, B., Schulz, C., y Gurevych, I. (2018). Ukp-athene: Multi-sentence textual entailment for claim verification. *arXiv preprint arXiv:1809.01479*.

Harabagiu, S., Hickl, A., y Lacatusu, F. (2006). Negation, contrast and contradiction in text processing. En *Proceedings of the 21st national conference on artificial intelligence - volume 1* (p. 755762). AAAI Press.

Hartling, L., Gates, A., Pillay, J., Nuspl, M., y Newton, A. (2018). Development and usability testing of epc evidence review dissemination summaries for health systems decisionmakers. *Methods Research Report. Technical Report.*.

Hassan, A., y Mahmood, A. (2017). Efficient deep learning model for text classification based on recurrent and convolutional layers. En *2017 16th ieee international conference on machine learning and applications (icmla)* (pp. 1108–1113).

Hassan, N., Zhang, G., Arslan, F., Caraballo, J., Jimenez, D., Gawsane, S., Hasan, S., Joseph, M., Kulkarni, A., y Nayak, A. K. (2017). Claimbuster: The first-ever end-to-end fact-checking system. *Proceedings of the VLDB Endowment*, *10*(12), 1945–1948.

Hayashi, Y., y Yanagimoto, H. (2018). Headline generation with recurrent neu-

ral network. En *New trends in e-service and smart computing* (pp. 81–96). Springer.

He, P., Liu, X., Gao, J., y Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., y Blunsom, P. (2015). Teaching machines to read and comprehend. En *Advances in neural information processing systems* (pp. 1693–1701).

Hooper, V. (2018). Fake news and social media: The role of the receiver. En *5th european conference on social media 2018* (p. 62).

Houlsby, N., Giurgiu, A., Jastrzçbski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., y Gelly, S. (2019). Parameter-Efficient Transfer Learning for NLP. *36th International Conference on Machine Learning, ICML 2019, 2019-June*, 4944–4953.

Hu, D. (2020). An introductory survey on attention mechanisms in nlp problems. En Y. Bi, R. Bhatia, y S. Kapoor (Eds.), *Intelligent systems and applications* (pp. 432–448). Cham: Springer International Publishing.

Huang, B., y Carley, K. M. (2020). Disinformation and misinformation on twitter during the novel coronavirus outbreak. *arXiv preprint arXiv:2006.04278*.

Huang, Z., Ye, Z., Li, S., y Pan, R. (2017). Length adaptive recurrent model for text classification. En *Proceedings of the acm on conference on information and knowledge management* (p. 1019-1027). Association for Computing Machinery.

Iwama, K., y Kano, Y. (2019). Multiple news headlines generation using page metadata. En *Proceedings of the 12th international conference on natural language generation* (pp. 101–105). Association for Computational Linguistics.

Janiesch, C., Zschech, P., y Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685–695.

Jeong, H., Ko, Y., y Seo, J. (2016). How to improve text summarization and classification by mutual cooperation on an integrated framework. *Expert Systems with Applications*, *60*, 222-233.

Kang, X., Li, B., Yao, H., Liang, Q., Li, S., Gong, J., y Li, X. (2020). Incorporating synonym for lexical sememe prediction: An attention-based model. *Applied Sciences*, *10*(17), 5996.

Kim, G., y Ko, Y. (2021). Graph-based fake news detection using a summarization technique. En *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: Main volume* (pp. 3276–3280).

Kirmani, M., Manzoor Hakak, N., Mohd, M., y Mohd, M. (2019). Hybrid text summarization: A survey. En K. Ray, T. K. Sharma, S. Rawat, R. K. Saini, y A. Bandyopadhyay (Eds.), *Soft computing: Theories and applications* (pp. 63–73). Springer Singapore.

Kitaev, N., Kaiser, Ł., y Levskaya, A. (2020). Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Koch, P., Golovidov, O., Gardner, S., Wujek, B., Griffin, J., y Xu, Y. (2018). Autotune: A derivative-free optimization framework for hyperparameter tuning. En *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* (pp. 443–452).

Konjengbam, A., Ghosh, S., Kumar, N., y Singh, M. (2018). Debate stance classification using word embeddings. En *International conference on big data analytics and knowledge discovery* (pp. 382–395).

Konstantinovskiy, L., Price, O., Babakar, M., y Zubiaga, A. (2021). Toward automated factchecking: Developing an annotation schema and benchmark for consistent automated claim detection. *Digital Threats: Research and Practice*, *2*(2), 1–16.

Kotonya, N., y Toni, F. (2020). Explainable automated fact-checking: A survey. *arXiv preprint arXiv:2011.03870*.

Kotu, V., y Deshpande, B. (2019). Classification. En *Data science* (pp. 65–163). Elsevier.

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., y Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150.

Krejzl, P. (2018). Stance detection and summarization in social networks. *Report*.

Krejzl, P., Hourová, B., y Steinberger, J. (2017). Stance detection in online discussions. *Computing Research Repository, CoRR, abs/1701.00504*.

Krizhevsky, A., Sutskever, I., y Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, 1097–1105.

Küçük, D., y Fazli, C. A. (2020). Stance detection: A survey. *ACM Computing Surveys*, *53*(1).

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, *2*(1-2), 83–97.

Kuiken, J., Schuth, A., Spitters, M., y Marx, M. (2017). Effective headlines of newspaper articles in a digital environment. *Digital Journalism*, *5*(10), 1300-1314.

Kumar, S., Ghosh, J., y Crawford, M. M. (2002). Hierarchical Fusion of Multiple Classifiers for Hyperspectral Data Analysis. *Pattern Analysis and Applications*, *2*(5), 210–220.

Kumari, S. (2021). Nofake at checkthat! 2021: fake news detection using bert. *arXiv preprint arXiv:2108.05419*.

Lai, M., Cignarella, A. T., Farías, D. I. H., Bosco, C., Patti, V., y Rosso, P. (2020). Multilingual stance detection in social media political debates. *Computer Speech & Language*, *63*, 101075.

Lajusticia, M. R. B. (2000). Estructura textual, macroestructura semántica y superestructura formal de la noticia. *Estudios sobre el mensaje periodístico*(6), 239.

Lavanya, P. M., y Sasikala, E. (2021). Deep learning techniques on text classification using Natural language processing (NLP) in social healthcare network: A comprehensive survey. *2021 3rd International Conference on Signal Processing and Communication, ICPSC 2021*, 603–609.

Lazarski, E., Al-Khassaweneh, M., y Howard, C. (2021). Using nlp for fact checking: A survey. *Designs 2021, Vol. 5, Page 42*, *5*, 42.

Lekshmiammal, H., y Madasamy, A. K. (2021). Nitk _ nlp at checkthat! 2021: Ensemble transformer model for fake news classification. En *Conference and labs ofthe evaluation forum (clef 2021)*.

Lendvai, P., y Reichel, U. D. (2016). Contradiction detection for rumorous claims. *arXiv preprint arXiv:1611.02588*.

Levy, O., Zesch, T., Dagan, I., y Gurevych, I. (2013). Recognizing partial textual entailment. En *Proceedings of the 51st annual meeting of the association for computational linguistics* (Vol. 2, pp. 451–455).

Li, B., y Han, L. (2013). Distance weighted cosine similarity measure for text classification. En *Proceedings of the 14th international conference on intelligent data engineering and automated learning* (p. 611-618). Springer-Verlag.

Li, C., Porco, A., y Goldwasser, D. (2018). Structured representation learning for online debate stance prediction. En *Proceedings of the 27th international conference on computational linguistics* (pp. 3728–3739).

Li, L., Qin, B., y Liu, T. (2017). Contradiction detection with contradiction-specific word embedding. *Algorithms*, *10*(2), 1–14.

Lingam, V., Bhuria, S., Nair, M., Gurpreetsingh, D., Goyal, A., y Sureka, A. (2018). Deep learning for conflicting statements detection in text. *PeerJ Prepr.*.

Liu, L., Feng, S., Wang, D., y Zhang, Y. (2016). An Empirical Study on Chinese Microblog Stance Detection Using Supervised and Semi-supervised Machine

Learning Methods. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10102*, 753–765.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., y Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Liu, Y., Song, X., y Chen, S.-F. (2019). Long story short: finding health advice with informative summaries on health social media. *Aslib Journal of Information Management*.

Lloret, E., Llorens, H., Moreda, P., Saquete, E., y Palomar, M. (2011). Text summarization contribution to semantic question answering: New approaches for finding answers on the web. *International Journal of Intelligent Systems*, *26*(12), 1125-1152.

Lloret, E., y Palomar, M. (2012). Text summarisation in progress: a literature review. *Artificial Intelligence Review*, *37*(1), 1–41.

Lorena, A. C., De Carvalho, A. C., y Gama, J. M. (2008). A review on the combination of binary classifiers in multiclass problems. *Artificial Intelligence Review*, *30*(1-4), 19–37.

Lu, Y.-J., y Li, C.-T. (2020). GCAN: Graph-aware co-attention networks for explainable fake news detection on social media. En *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 505–514). Association for Computational Linguistics.

Lutz, B., Adam, M. T. P., Feuerriegel, S., Pröllochs, N., y Neumann, D. (2020). Affective information processing of fake news: Evidence from neurois. En *Information systems and neuroscience* (pp. 121–128). Springer International Publishing.

Ma, Y., Liu, X., Zhao, L., Liang, Y., Zhang, P., y Jin, B. (2022). Hybrid embedding-based text representation for hierarchical multi-label text classification. *Expert Systems with Applications*, *187*, 115905.

Mackie, S., McCreadie, R., Macdonald, C., y Ounis, I. (2016). Experiments in newswire summarisation. En N. Ferro y cols. (Eds.), *Advances in information retrieval* (pp. 421–435). Cham: Springer International Publishing.

Marelli, M., Bentivogli, L., Baroni, M., Bernardi, R., Menini, S., y Zamparelli, R. (2014). SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. En *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)* (pp. 1–8). Association for Computational Linguistics.

Martinez-Rico, J., Martinez-Romo, J., y Araujo, L. (2021). NLPIR@UNED at CheckThat! 2021: Check-worthiness estimation and fake news detection using transformer models. En *Ceur workshop proceedings* (Vol. 2936, pp. 545–557).

Martinez-Rico, J. R., Araujo, L., y Martinez-Romo, J. (2020). Nlp&ir@ uned at checkthat! 2020: A preliminary approach for check-worthiness and claim retrieval tasks using neural networks and graphs. En *Clef (working notes)*.

McDonald, T., Dong, Z., Zhang, Y., Hampson, R., Young, J., Cao, Q., Leidner, J. L., y Stevenson, M. (2020). The university of sheffield at checkthat! 2020: Claim identification and verification on twitter. En *Clef (working notes)*.

Metcalf, L., y Casey, W. (2016). Metrics, similarity, and sets. En *Cybersecurity and applied mathematics* (pp. 3–22). Elsevier.

Mihalcea, R., y Tarau, P. (2004). TextRank: Bringing order into text. En *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 404–411). Association for Computational Linguistics.

Mikolov, T., Chen, K., Corrado, G., y Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miranda, S., Nogueira, D., Mendes, A., Vlachos, A., Secker, A., Garrett, R., Mitchel, J., y Marinho, Z. (2019). Automated fact checking in the news room. En *The world wide web conference* (pp. 3579–3583).

Mishra, R., Yadav, P., Calizzano, R., y Leippold, M. (2020). Musem: Detecting incongruent news headlines using mutual attentive semantic matching. *Proceedings - 19th IEEE International Conference on Machine Learning and Applications, ICMLA 2020*, 709-716.

Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., y Cherry, C. (2016). SemEval-2016 task 6: Detecting stance in tweets. En *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)* (pp. 31–41). San Diego, California: Association for Computational Linguistics.

Molina, M. D., Sundar, S. S., Le, T., y Lee, D. (2021). Fake news is not simply false information: A concept explication and taxonomy of online content. *American behavioral scientist*, *65*(2), 180–212.

Mompart, J. L. G., Lozano, J. F. G., y Sampío, D. P. (2015). Los periodistas españoles y la pérdida de la calidad de la información: el juicio profesional. *Comunicar*, *23*, 143-150.

Moscadelli, A., Albora, G., Biamonte, M. A., Giorgetti, D., Innocenzio, M., Paoli, S., Lorini, C., Bonanni, P., y Bonaccorsi, G. (2020). Fake news and covid-19 in Italy: Results of a quantitative observational study. *International Journal of Environmental Research and Public Health*, *17*(16), 1–13.

Murakami, A., y Raymond, R. (2010). Support or oppose? classifying positions in online debates from reply activities and opinion expressions. En *Proceedings of the 23rd international conference on computational linguistics: Posters* (pp. 869–875).

Nagrani, A., Chung, J. S., Xie, W., y Zisserman, A. (2020). Voxceleb: Large-scale speaker verification in the wild. *Computer Speech and Language, 60*.

Naik, A., y Rangwala, H. (2018). *Large Scale Hierarchical Classification: State of the Art*. Springer International Publishing.

Nakov, P., Barrón-Cedeno, A., Elsayed, T., Suwaileh, R., Màrquez, L., Zaghouani, W., Atanasova, P., Kyuchukov, S., y Da San Martino, G. (2018). Overview of the clef-2018 checkthat! lab on automatic identification and verification of political claims. En *International conference of the cross-language evaluation forum for european languages* (pp. 372–387).

Nakov, P., Martino, G. D. S., Elsayed, T., Barrón-Cedeño, A., Míguez, R., Shaar, S., Alam, F., Haouari, F., Hasanain, M., Babulkov, N., Nikolov, A., Shahi, G. K., Struß, J. M., y Mandl, T. (2021). The CLEF-2021 checkthat! lab on detecting check-worthy claims, previously fact-checked claims, and fake news. En D. Hiemstra, M. Moens, J. Mothe, R. Perego, M. Potthast, y F. Sebastiani (Eds.), *Advances in information retrieval - 43rd european conference on IR research, ECIR 2021, virtual event, march 28 - april 1, 2021, proceedings, part II* (Vol. 12657, pp. 639–649). Springer.

Narwal, B. (2018). Fake News in Digital Media. *Proceedings - IEEE 2018 International Conference on Advances in Computing, Communication Control and Networking, ICACCCN 2018*, 977–981.

Nenkova, A. (2005). Automatic text summarization of newswire: Lessons learned from the document understanding conference. En *Proceedings of the 20th national conference on artificial intelligence - volume 3* (p. 14361441). AAAI Press.

Nguyen, D. Q., Vu, T., y Tuan Nguyen, A. (2020). BERTweet: A pre-trained language model for English tweets. En *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations* (pp. 9–14). Association for Computational Linguistics.

Nikolov, A., Martino, G. D. S., Koychev, I., y Nakov, P. (2020). Team alex at clef checkthat! 2020: Identifying check-worthy tweets with transformer models. *arXiv preprint arXiv:2009.02931*.

Normala, C. E., Jamil, I., Ishak, F. S., y Lilly Suriani, A. (2021). Fakeheader: A tool to detect deceptive online news based on misleading news headlines and contents. *Turkish Journal of Computer and Mathematics Education Vol, 12*(3), 2217–2223.

Omidvar, A., Poormodheji, H., An, A., y Edall, G. (2019). Learning to determine the quality of news headlines. *ICAART 2020 - Proceedings of the 12th International Conference on Agents and Artificial Intelligence, 1*, 401-409.

Padró, L., y Stanilovsky, E. (2012). Freeling 3.0: Towards wider multilinguality. En *Proceedings of the language resources and evaluation conference.*

Pan, S. J., y Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Park, C. S. (2019). Does too much news on social media discourage news seeking? mediating role of news efficacy between perceived news overload and news avoidance on social media. *Social Media + Society*, *5*(3), 1–12.

Park, K., Kim, T., Yoon, S., Cha, M., y Jung, K. (2020). Baitwatcher: A lightweight web interface for the detection of incongruent news headlines. En *Disinformation, misinformation, and fake news in social media* (pp. 229–252). Springer.

Passalis, N., y Tefas, A. (2018). Learning bag-of-embedded-words representations for textual information retrieval. *Pattern Recognition, 81*, 254–267.

Patra, B. G., Das, D., y Bandyopadhyay, S. (2016). Ju_nlp at semeval-2016 task 6: detecting stance in tweets using support vector machines. En *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)* (pp. 440–444).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., y Dubourg, V. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12*(Oct), 2825–2830.

Pennebaker, J. W., Boyd, R. L., Jordan, K., y Blackburn, K. (2015). *The development and psychometric properties of liwc2015* (Inf. Téc.).

Pennington, J., Socher, R., y Manning, C. D. (2014). Glove: Global vectors for word representation. En *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1532–1543).

Perea-Ortega, J. M., Lloret, E., Ureña-López, L. A., y Palomar, M. (2013). Application of text summarization techniques to the geographical information retrieval task. *Expert Systems with Applications*, *40*(8), 2966 - 2974.

Pereira, L. H., Silla Junior, C. N., y Nievola, J. C. (2019). Local hierarchical classification techniques analysis using attribute selection for protein function prediction. *Proceedings - 6th Annual Conference on Computational Science and Computational Intelligence, CSCI 2019*, 1476–1481.

Pereira, R. M., Bertolini, D., Teixeira, L. O., Silla, C. N., y Costa, Y. M. (2020). COVID-19 identification in chest X-ray images on flat and hierarchical clas-

sification scenarios. *Computer Methods and Programs in Biomedicine*, *194*, 105532.

Perronnin, F., Akata, Z., Harchaoui, Z., y Schmid, C. (2012). Towards good practice in large-scale learning for image classification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3482–3489.

Peters, M. E., Ammar, W., Bhagavatula, C., y Power, R. (2017). Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*.

Peters, M. E., Ruder, S., y Smith, N. A. (2019). To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*.

Pham, M. Q. N., Nguyen, M. L., y Shimazu, A. (2013). Using shallow semantic parsing and relation extraction for finding contradiction in text. En *Proceedings of the sixth international joint conference on natural language processing* (pp. 1017–1021). Asian Federation of Natural Language Processing.

Portelli, B., Zhao, J., Schuster, T., Serra, G., y Santus, E. (2020). Distilling the evidence to augment fact verification models. En *Proceedings of the third workshop on fact extraction and verification (fever)* (pp. 47–51). Association for Computational Linguistics.

Pöttker, H. (2003). News and its communicative quality: the inverted pyramidwhen and why did it appear? *Journalism Studies*, *4*(4), 501-511.

Pouliquen, B., Steinberger, R., y Best, C. (2007). Automatic detection of quotations in multilingual news. En *Proceedings of recent advances in natural language processing* (pp. 487–492).

Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., Shyu, M. L., Chen, S. C., y Iyengar, S. S. (2018). A Survey on Deep Learning: Algorithms, Techniques, and Applications. *ACM Computing Surveys (CSUR)*, *51*(5).

Pujahari, A., y Sisodia, D. S. (2021). Clickbait detection using multiple categorisation techniques. *Journal of Information Science*, *47*, 118-128.

Qiu, X. P., Sun, T. X., Xu, Y. G., Shao, Y. F., Dai, N., y Huang, X. J. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, *63*(10), 1872–1897.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., y Liu, P. J. (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, *21*, 1–67.

Rahimi, Z., y ShamsFard, M. (2021). Contradiction detection in persian text. *arXiv preprint arXiv:2107.01987*.

Raposo, F., Ribeiro, R., y Martins de Matos, D. (2016). Using generic summarization to improve music information retrieval tasks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *24*(6), 1119-1128.

Reimers, N., y Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Reis, J., Benevenuto, F., de Melo, P. O. S. V., Prates, R., Kwak, H., y An, J. (2015). Breaking the News: First Impressions Matter on Online News. *Proceedings of the 9th International Conference on Web and Social Media, ICWSM 2015*, 357–366.

Riedel, B., Augenstein, I., Spithourakis, G. P., y Riedel, S. (2017). A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. *Computing Research Repository, CoRR, abs/1707.03264*.

Ritter, A., Soderland, S., Downey, D., y Etzioni, O. (2008). It's a contradiction – no, it's not: A case study using functional relations. En *Proceedings of the 2008 conference on empirical methods in natural language processing* (pp. 11–20). Association for Computational Linguistics.

Rodríguez, R. F., y Barrio, M. G. (2015). Infoxication: Implications of the phenomenon in journalism. *Revista de Comunicación de la SEECI*, *38*, 141–181.

Rosemblat, G., Fiszman, M., Shin, D., y Kilicoglu, H. (2019). Towards a characterization of apparent contradictions in the biomedical literature using context analysis. *Journal of Biomedical Informatics*, *98*.

Roush, A., y Balaji, A. (2020). Debatesum: A large-scale argument mining and summarization dataset. *arXiv preprint arXiv:2011.07251*.

Rubin, V. L. (2019). Disinformation and misinformation triangle. *Journal of Documentation*, *75*(5), 1013–1034.

Rus, V., y Lintean, M. (2012). A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. En *Proceedings of the seventh workshop on building educational applications using NLP* (pp. 157–162). Association for Computational Linguistics.

Saggion, H., Lloret, E., y Palomar, M. (2012). Can text summaries help predict ratings? a case study of movie reviews. En G. Bouma, A. Ittoo, E. Métais, y H. Wortmann (Eds.), *Natural language processing and information systems* (pp. 271–276). Springer Berlin Heidelberg.

Sandhaus, E. (2008). The New York Times Annotated Corpus ldc2008t19. Linguistic Data Consortium.

Šarić, F., Glavaš, G., Karan, M., Šnajder, J., y Dalbelo Bašić, B. (2012). TakeLab: Systems for measuring semantic text similarity. En *Proceedings of the first joint conference on lexical and computational semantics* (pp. 441–448). As-

sociation for Computational Linguistics.

Schlicht, I. B., de Paula, A. F. M., y Rosso, P. (2021). Upv at checkthat! 2021: Mitigating cultural differences for identifying multilingual check-worthy claims. *arXiv preprint arXiv:2109.09232*.

Schuler, K. K. (2005). *Verbnet: A broad-coverage, comprehensive verb lexicon*. University of Pennsylvania.

Sen, A., Sinha, M., Mannarswamy, S., y Roy, S. (2018). Stance classification of multi-perspective consumer health information. En *Proceedings of the acm india joint international conference on data science and management of data* (pp. 273–281).

Shaar, S., Hasanain, M., Hamdan, B., Ali, Z. S., Haouari, F., Alex Nikolov, M. K., Yavuz Selim Kartal, F. A., Da San Martino, G., Barrón-Cedeño, A., Míguez, R., Elsayed, T., y Nakov, P. (2021). Overview of the CLEF-2021 CheckThat! lab task 1 on check-worthiness estimation in tweets and political debates. En *Working notes of clef 2021—conference and labs of the evaluation forum*.

Shahi, G. K., StruSS, J. M., y Mandl, T. (2021). "Overview of the CLEF-2021 CheckThat! Lab Task 3 on Fake News Detection". En *"working notes of clef 2021—conference and labs of the evaluation forum"*.

ShimJae-Seung, WonHa-Ram, y AhnHyunchul. (2019). A study on the effect of the document summarization technique on the fake news detection model. *A Study on the Effect of the Document Summarization Technique on the Fake News Detection Model*, *25*(3), 201-220.

Shu, K., Sliva, A., Wang, S., Tang, J., y Liu, H. (2017). Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, *19*(1), 22–36.

Sifa, R., Pielka, M., Ramamurthy, R., Ladi, A., Hillebrand, L., y Bauckhage, C. (2019). Towards contradiction detection in german: a translation-driven approach. En *2019 ieee symposium series on computational intelligence (ssci)* (p. 2497-2505).

Silla, C. N., y Freitas, A. A. (2010). A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery 2010 22:1*, *22*(1), 31–72.

Silva-Palacios, D., Ferri, C., y Ramírez-Quintana, M. J. (2018). Probabilistic class hierarchies for multiclass classification. *Journal of Computational Science*, *26*, 254–263.

Silverman, C. L. (2015). Lies, damn lies and viral content..

Simonyan, K., y Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sisodia, D. S. (2019). Ensemble learning approach for clickbait detection using article headline features. *Informing Science*, *22*, 31-44.

Slovikovskaya, V. (2019). Transfer learning from transformers to fake news challenge stance detection (FNC-1) task. *arXiv preprint arXiv:1910.14353*.

Smith, T. F., y Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of molecular biology*, *147*(1), 195–197.

Somasundaran, S., y Wiebe, J. (2010). Recognizing stances in ideological on-line debates. En *Proceedings of the naacl hlt 2010 workshop on computational approaches to analysis and generation of emotion in text* (pp. 116–124).

Su, T., Macdonald, C., y Ounis, I. (2019). Entity detection for check-worthiness prediction: Glasgow terrier at clef checkthat! 2019. En *Working notes of clef 2019-conference and labs of the evaluation forum* (Vol. 2380).

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., y Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, *35*(5), 1299–1312.

Takabatake, Y., Morita, H., Kawahara, D., Kurohashi, S., Higashinaka, R., y Matsuo, Y. (2015). Classification and acquisition of contradictory event pairs using crowdsourcing. En *Proceedings of the the 3rd workshop on EVENTS: Definition, detection, coreference, and representation* (pp. 99–107). Association for Computational Linguistics.

Talwar, S., Dhir, A., Kaur, P., Zafar, N., y Alrasheedy, M. (2019). Why do people share fake news? Associations between the dark side of social media use and fake news sharing behavior. *Journal of Retailing and Consumer Services*, *51*(June), 72–82.

Tan, J., Wan, X., y Xiao, J. (2017). From neural sentence summarization to headline generation: A coarse-to-fine approach. En *Proceedings of the 26th international joint conference on artificial intelligence* (p. 4109-4115). AAAI Press.

Tanaka, K., Kameoka, H., Kaneko, T., y Hojo, N. (2019). Atts2s-vc: Sequence-to-sequence voice conversion with attention and context preservation mechanisms. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2019-May*, 6805-6809.

Tas, O., y Kiyani, F. (2007). A survey automatic text summarization. *PressAcademia Procedia*, *5*(1), 205–213.

Tata, S., y Patel, J. M. (2007). Estimating the selectivity of tf-idf based cosine similarity predicates. *SIGMOD Record*, *36*(4), 75–80.

Taulé, M., Martí, M. A., Rangel, F. M., Rosso, P., Bosco, C., y Patti, V. (2017). Over-

view of the task on stance and gender detection in tweets on Catalan independence at Ibereval 2017. En *2nd workshop on evaluation of human language technologies for iberian languages* (Vol. 1881, pp. 157–177).

Tavernisen, S. (2019). As fake news spreads lies, more readers shrug at the truth. *The New York Times*, *6*.

Tay, Y., Dehghani, M., Bahri, D., y Metzler, D. (2020). Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*.

Thorne, J., y Vlachos, A. (2018). Automated fact checking: Task formulations, methods and future directions. *arXiv preprint arXiv:1806.07687*.

Thorne, J., Vlachos, A., Christodoulopoulos, C., y Mittal, A. (2018). FEVER: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 809–819.

Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. En *Proceedings of the eighth international conference on language resources and evaluation (LREC'12)* (pp. 2214–2218). European Language Resources Association (ELRA).

Tsarev, D., Petrovskiy, M., y Mashechkin, I. (2013). Supervised and unsupervised text classification via generic summarization. *International Journal of Computer Information Systems and Industrial Management Applications. MIR Labs*, *5*, 509–515.

Tsipursky, G., Votta, F., y Roose, K. M. (2018). Fighting Fake News and Post-Truth Politics with Behavioral Science: The Pro-Truth Pledge. *Behavior and Social Issues*, *27*(1), 47–70.

Tudjmanand, M., y Mikelic Preradovic, N. (2003). Information science: Science about information. En *Proceedings of informing science & it education* (p. 1513-1527).

Tutek, M., Sekulić, I., Gombar, P., Paljak, I., Čulinović, F., Boltužić, F., Karan, M., Alagić, D., y Šnajder, J. (2016). TakeLab at SemEval-2016 Task 6: Stance Classification in Tweets Using a Genetic Algorithm Based Ensemble. *SemEval 2016 - 10th International Workshop on Semantic Evaluation, Proceedings*, 464–468.

Urban, J., y Schweiger, W. (2014). News quality from the recipients' perspective: Investigating recipients' ability to judge the normative quality of news. *Journalism Studies*, *15*(6), 821–840.

Vamvas, J., y Sennrich, R. (2020). X-stance: A multilingual multi-target dataset for stance detection. *CEUR Workshop Proceedings*, *2624*.

van Dijk, T. A. (1988). *News as discourse*. L. Erlbaum Associates.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., y Polosukhin, I. (2017). Attention is all you need. En *Advances in neural information processing systems* (pp. 5998–6008).

Vicente, M., y Lloret, E. (2020). A discourse-informed approach for cost-effective extractive summarization. En *International conference on statistical language and speech processing* (pp. 109–121).

Vicente Moreno, M. (2021). *A discourse-aware macroplanning approach for text generation and beyond* (PhD Thesis). University of Alicante.

Vijaymeena, M., y Kavitha, K. (2016). A survey on similarity measures in text mining. *Machine Learning and Applications: An International Journal*, *3*(2), 19–28.

Vlachos, A., y Riedel, S. (2015). Identification and verification of simple claims about statistical properties. En *Proceedings of the conference on empirical methods in natural language processing* (pp. 2596–2601).

Vosoughi, S., Roy, D., y Aral, S. (2018). The spread of true and false news online. *Science*, *359*(6380), 1146–1151.

Vrbančič, G., y Podgorelec, V. (2020). Transfer learning with adaptive fine-tuning. *IEEE Access*, *8*, 196197–196211.

Vrbancic, G., Zorman, M., y Podgorelec, V. (2019). Transfer learning tuning utilizing grey wolf optimizer for identification of brain hemorrhage from head ct images. *StuCoSReC. Proceedings of the 2019 6th Student Computer Science Research Conference*.

Vychegzhanin, S. V., y Kotelnikov, E. V. (2019). Stance detection based on ensembles of classifiers. *Programming and Computer Software*, *45*(5), 228–240.

Wardle, C., y Derakhshan, H. (2017). Information disorder: Toward an interdisciplinary framework for research and policy making. *Council of Europe*, *27*.

Wei, W., y Wan, X. (2017). Learning to identify ambiguous and misleading news headlines. En *Proceedings of the 26th international joint conference on artificial intelligence* (pp. 4172–4178). AAAI Press.

Widyassari, A. P., Affandy, A., Noersasongko, E., Fanani, A. Z., Syukur, A., y Basuki, R. S. (2019). Literature review of automatic text summarization: Research trend, dataset and method. En *International conference on information and communications technology* (p. 491-496).

Williams, A., Nangia, N., y Bowman, S. (2018). A broad-coverage challenge corpus for sentence understanding through inference. En *Proceedings of the 2018 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers)* (pp. 1112–1122). Association for Computational Linguistics.

Williams, E., Rodrigues, P., y Novak, V. (2020). Accenture at checkthat! 2020: If you say so: Post-hoc fact-checking of claims using transformer-based models. *arXiv preprint arXiv:2009.02431*.

Winston, P. H., y Prendergast, K. A. (1984). *The ai business: Commercial uses of artificial intelligence*. Massachusetts Institute of Technology.

Xiao, Z., Dellandrea, E., Dou, W., y Chen, L. (2007). Automatic hierarchical classification of emotional speech. En *Ninth ieee international symposium on multimedia workshops (ismw 2007)* (pp. 291–296).

Xu, L., Dong, D., y Jhang, S.-E. (2018). Investigating the effects of text summarization on linguistic quality of argumentative writing. , *60*(4), 245–268.

Yoneda, T., Mitchell, J., Welbl, J., Stenetorp, P., y Riedel, S. (2018). Ucl machine reading group: Four factor framework for fact finding (hexaf). En *Proceedings of the first workshop on fact extraction and verification (fever)* (pp. 97–102).

Yoon, S., Park, K., Lee, M., Kim, T., Cha, M., y Jung, K. (2021). Learning to detect incongruence in news headline and body text via a graph neural network. *IEEE Access*, *9*, 36195–36206.

Yoon, S., Park, K., Shin, J., Lim, H., Won, S., Cha, M., y Jung, K. (2018). Detecting Incongruity Between News Headline and Body Text via a Deep Hierarchical Encoder. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*, 791–800.

Zajic, D., Dorr, B., y Schwartz, R. (2002). Automatic headline generation for newspaper stories. En *Proceedings of the workshop on automatic summarization* (pp. 78–85).

Zarrella, G., y Marsh, A. (2016). Mitre at semeval-2016 task 6: Transfer learning for stance detection. En *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)* (pp. 458–463). Association for Computational Linguistics.

Zeng, X., Abumansour, A. S., y Zubiaga, A. (2021). Automated fact-checking: A survey. *Language and Linguistics Compass*, *15*(10), e12438.

Zengin, M., Kartal, Y., y Kutlu, M. (2021). TOBB ETU at CheckThat! 2021: Data engineering for detecting check-worthy claims. En *Ceur workshop proceedings* (Vol. 2936, pp. 670–680).

Zhang, Q., Liang, S., Lipani, A., Ren, Z., y Yilmaz, E. (2019). From stances' imbalance to their hierarchical representation and detection. En *The world wide web conference* (pp. 2323–2332). ACM.

Zhang, X., y Ghorbani, A. A. (2020). An overview of online fake news: Characterization, detection, and discussion. *Information Processing and Manage-*

ment, *57*(2).

Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X., y Zhou, X. (2020). Semantics-aware bert for language understanding. En *Proceedings of the aaai conference on artificial intelligence* (Vol. 34, pp. 9628–9635).

Zhou, S., Lin, J., Tan, L., y Liu, X. (2019). Condensed convolution neural network by attention over self-attention for stance detection in twitter. En *International joint conference on neural networks (ijcnn)* (pp. 1–8).

Zhu, C., Yang, Z., Gmyr, R., Zeng, M., y Huang, X. (2019). Make lead bias in your favor: A simple and effective method for news summarization. *arXiv preprint arXiv:1912.11602*.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., y Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. En *Proceedings of the ieee international conference on computer vision* (pp. 19–27).

Zotova, E., Agerri, R., y Rigau, G. (2021). Semi-automatic generation of multilingual datasets for stance detection in twitter. *Expert Systems with Applications*, *170*, 114547.