UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dpto. de Sistemas Informáticos y Computación

Detección y agrupamiento de noticias en fuentes periodísticas digitales

Trabajo Fin de Máster

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

AUTOR/A: Cardona Lorenzo, Víctor

Tutor/a: Segarra Soriano, Encarnación

Cotutor/a: Hurtado Oliver, Lluis Felip

Cotutor/a: Ahuir Esteve, Vicent

CURSO ACADÉMICO: 2022/2023

Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València

Detección y agrupamiento de noticias en fuentes periodísticas digitales

Trabajo Fin de Máster

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Autor: Victor Cardona Lorenzo

Tutores: Encarnación Segarra Soriano

Lluis Felip Hurtado Oliver

Curso 2022/2023

Resumen

La importancia de la información y de las noticias que circulan por la red resulta en algunos casos tan abrumadora que dificulta el acceso a los sucesos relevantes, de actualidad o en tendencia. A su vez, gracias a la automatización de todos los procesos que utilizamos hoy día y, a la introducción de las técnicas de aprendizaje automático y del procesamiento del lenguaje natural, se ha podido evolucionar en el sentido de cómo se abordan las tareas cotidianas.

La finalidad de relacionar las noticias y medios periodísticos con el NLP es con el fin de mejorar la experiencia tanto de usuarios como de profesionales. De este modo se pretende otorgar de un sistema capaz de obtener y seleccionar aquellas noticias candentes en varios medios. De esta manera se facilita el acceso a temas significantes además de poder realizar un análisis más exhaustivo semántico y poder obtener las contraposiciones entre medios, opinión, resumen objetivo u otros intereses.

El sistema desarrollado se diferencia entre una primera parte de obtención de las noticias y, una segunda, como agrupamiento de estas. Para la primera de ellas se ha implementado un raspado de datos web (*scraper*) que permite acceder a las webs de los medios y obtener las noticias dado un rango de fechas. Con ello se ha sido capaz de crear un corpus y anotarlo, que es crucial para el siguiente paso.

La segunda parte, se extraen las entidades nombradas y las palabras clave de las noticias, así como realizar un análisis de varios modelos para efectuar estas tareas. Este paso intermedio resulta necesario para poder obtener y comparar más resultados a la hora de agrupar. Finalmente, con todos los datos obtenidos y extraídos se presentan diferentes modelos para la agrupación de las noticias que sean similares en cuanto a su contenido semántico y se evalúan.

El trabajo se enmarca en un proyecto de investigación del Ministerio de Ciencia e Innovación (BEWORD-UPV: DESCUBRIENDO EL SIGNIFICADO Y LA INTENCIÓN MAS ALLÁ DE LA PALABRA HABLADA: HACIA UN ENTORNO INTELIGENTE, PID2021-126061OB-C41) consistente en mejorar la capacidad de los sistemas autónomos para procesar información recopilada de fuentes de naturaleza muy diversa.

Palabras clave: Procesamiento de Lenguaje Natural; técnicas de agrupamiento; similitud semántica; noticias periodísticas; medios digitales;

Abstract

The amount of information and news circulating on the web is in some cases so overwhelming that it makes it difficult to access relevant, current or trending events. At the same time, thanks to the automation of all the processes we use today, the introduction of automatic learning techniques and natural language processing in everyday life, we have been able to evolve in the way we deal with everyday tasks.

The purpose of linking news and media to NLP is to improve the experience of both users and professionals. In this way, it is intended to provide a system capable of obtaining and selecting hot news in various media. In this way, it facilitates the access to significant and hot topics, in addition to being able to carry out a more thorough semantic analysis and to be able to obtain the oppositions between media, opinion, objective summary or other interests.

The developed system is differentiated between a first part of obtaining the news and a second part, as a clustering of these. For the first part, a web data scraper has been implemented to access the media websites and obtain the news given a range of dates. With this we have been able to create a corpus and annotate it, which is crucial for the next step.

In the second part, named entities and keywords are extracted from the news items as well as an analysis of various models to perform these tasks. This intermediate step is crucial in order to obtain and compare more results for clustering. Finally, with all the data obtained and extracted, different models for the clustering of news items that are similar in terms of their semantic content are presented and evaluated.

The work is part of a Ministerio de Ciencia e Innovación (BEWORD-UPV: DESCUBRIENDO EL SIGNIFICADO Y LA INTENCIÓN MAS ALLÁ DE LA PALABRA HABLADA: HACIA UN ENTORNO INTELIGENTE, PID2021-126061OB-C41) aimed at improving the capacity of autonomous systems to process information gathered from a wide variety of sources.

Keywords: Natural Language Processing; clustering techniques; semantic similarity; journalistic news; digital media;

Tabla de contenidos

Índice de esquemas

Índice de gráficas

Índice de ilustraciones

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

La información es un recurso fundamental en la sociedad actual y, los medios de comunicación digitales son una de las principales fuentes de información para muchas personas. En las últimas décadas ha experimentado un crecimiento exponencial con la llegada de internet, los *smartphones* y las redes sociales. Sin embargo, la gran cantidad y diversidad de noticias que se publican cada día dificultan el acceso a la información interesante, relevante y veraz.

Gracias a la automatización de la informática se consiguió poder seleccionar y realizar búsquedas filtradas sobre noticias y palabras clave de manera automática. Sin embargo, su funcionamiento estaba basado en sistemas de búsqueda y recuperación de información (*information retrieval*). Ahora ese paradigma ha cambiado con la llegada y el auge de la inteligencia artificial (IA) y el aprendizaje profundo.

Estas nuevas técnicas ya conocidas por todo el mundo en la actualidad, debido a la cantidad de productos y herramientas en aumento resultan de interés para abordar los siguientes temas. Se está de acuerdo que la IA ha irrumpido en la vida social y cotidiana hasta el punto que ha transformado la forma en que nos comunicamos, cómo tomamos decisiones y accedemos a información. Parte de ello se encuentra relacionado con el procesamiento del lenguaje natural (NLP) ya sea en el uso de *chatbots*, traducciones automáticas, análisis de sentimientos, mejoras en búsquedas, en automatización y análisis, etc.

El interés por relacionar las noticias con el aprendizaje automático y, en concreto, con el procesamiento del lenguaje natural radica en la posibilidad de mejorar la experiencia de los usuarios al acceder a noticias en línea o temas relevantes o

tendencias. Para ello se quiere realizar un sistema capaz de obtener las noticias de los medios digitales y conseguir agrupar las noticias que se encuentren relacionadas.

1.2. Solución propuesta

El sistema que se plantea desarrollar es uno que sea capaz de obtener las noticias de las páginas webs de los periódicos españoles mediate un raspado de la web (*scraping*) y que posteriormente aplicando técnicas de NLP con similitud semántica y una posible codificación de las noticias se pueda agrupar aquellas noticias que sean similares porque traten del mismo acontecimiento o suceso. Con este sistema se aplican tanto técnicas de IA como de NLP, haciendo uso de similitud semántica, uso de *words embeddings*, medidas de distancia entre estos y agrupamiento de documentos de acuerdo a estos factores, así como una posterior evaluación y comparación de sistemas.

El interés de esta propuesta radica en su potencial de aplicabilidad en diversos ámbitos, tales como la educación, la investigación, el periodismo o la ciudadanía. Estas aplicaciones son la creación de resúmenes automáticos de las noticias más relevantes del día, realizar análisis comparativos de las fuentes informativas, detectar tendencias o eventos emergentes, analizar la cobertura de noticias sobre un tema específico, o fomentar el pensamiento crítico.

Asimismo, esta propuesta contribuye al avance del conocimiento científico en el campo del NLP y la IA aplicados a la extracción y el análisis de noticias.

1.3. Objetivos

El objetivo principal de este trabajo de fin de máster es confeccionar un sistema capaz de extraer noticias de los medios digitales y después poder aplicar técnicas de aprendizaje automático para poder agrupar las noticias que traten sobre el mismo acontecimiento o suceso. Para ello se definen los siguientes objetivos:

- Extraer noticias de al menos cinco medios periodísticos digitales
- Crear un corpus etiquetado con la agrupación de noticias similares

- Extraer información adicional que sea de utilidad para la agrupación del corpus
- Utilizar modelos preentrenados para agrupar las noticias según su similitud de contenido
- Comparar los modelos de agrupamiento y seleccionar aquel que obtenga mejor rendimiento

1.4. Estructura de la memoria

Esta memoria se encuentra dividida en diferentes capítulos, donde cada uno de ellos hace hincapié sobre un determinado ámbito.

El segundo capítulo se centra sobre la contextualización de la inteligencia artificial y de los métodos del procesamiento del lenguaje natural. Ofrece una visión para conocer el estado del arte de esta línea de investigación.

El tercer capítulo explica las diferentes herramientas, tecnologías y librerías que se han utilizado a lo largo de la implementación, tanto para la parte de extracción de noticias como de aprendizaje automático para el agrupamiento.

El cuarto capítulo es la explicación del corpus y su creación. En este punto y dado que se hace uso de ello para crear el corpus también se explica la aplicación de raspado de datos de la web para obtener las noticias. También la extracción de noticias para crear el corpus, su anotación de manera manual y preproceso para disponer de los datos y de los conjuntos para poder evaluar.

El quinto capítulo consta de la parte de inteligencia artificial donde a partir del corpus se presentan los diferentes modelos preentrenados. Este capítulo se divide en el análisis para la extracción de más información del corpus de la tarea NER, de palabras clave y, de la tarea que interesa en este caso, el agrupamiento de noticias. Aunque en las dos primeras tareas se muestran los modelos y resultados en la última únicamente se presentan los modelos.

El sexto capítulo es el encargado de mostrar los resultados obtenidos con los modelos presentados en el capítulo anterior para el agrupamiento de noticias similares. También se realiza un pequeño análisis y comentario sobre el rendimiento de estos modelos en los conjuntos de validación y *test*.

El séptimo capítulo realiza un pequeño resumen sobre cómo se ha desarrollado las partes del sistema y sus resultados brevemente. Asimismo, se justifica el cumplimiento de los objetivos y se finaliza con la conclusión a título personal.

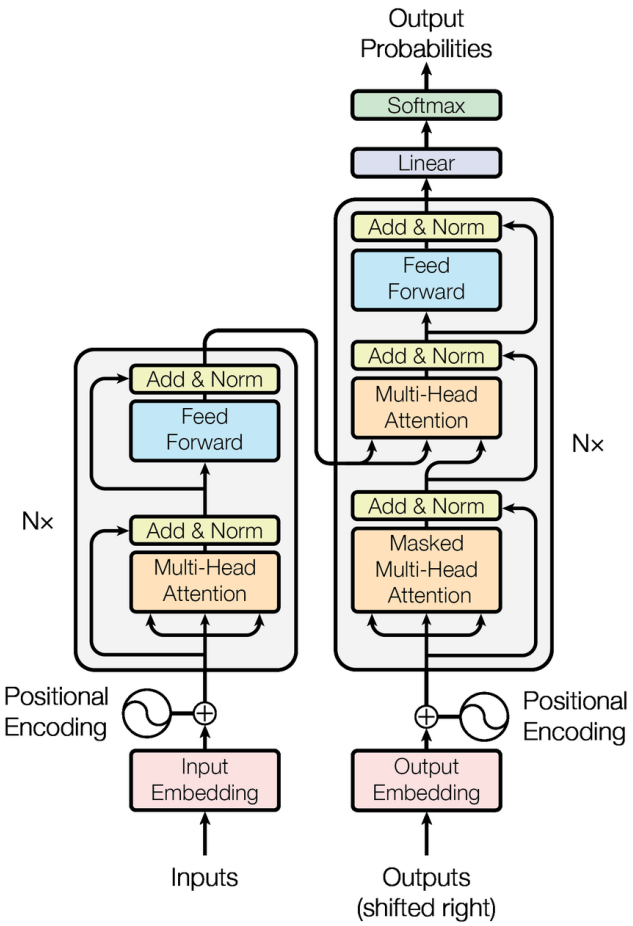
El octavo y último capítulo finaliza la memoria con las funciones de mejora, progreso y perfeccionamiento para un futuro poder disponer de una aplicación completa, útil, sencilla y de uso real para el usuario final.

Capítulo 2

Estado del arte

El procesamiento del lenguaje natural (NLP) es una rama de la inteligencia artificial (IA) que se ocupa de la comprensión y generación de lenguaje humano. El NLP abarca diversas tareas, como el análisis sintáctico, la desambiguación semántica, la extracción de información, el resumen de textos, la traducción automática, el reconocimiento de voz, la generación de lenguaje natural, entre otras. El NLP se aplica a diversos dominios y fuentes de datos, como las noticias, las redes sociales, los libros, los correos electrónicos, etc.

En cuanto a su evolución, ha sido de los campos que más ha mejorado y donde también se han conseguido varios descubrimientos que posteriormente se aplicarían al resto de áreas debido a su efectividad. La línea de investigación que aportó bastante al desarrollo del área fue la traducción automática donde los primeros modelos estadísticos fueron dando paso a las redes neuronales. En este segundo campo destacan las redes recurrentes (RNNs) [1] para solucionar el desvanecimiento del gradiente con puertas LSTM [2] y GRU [3]. Más adelante, el modelo de atención fue el punto de inflexión en el aprendizaje automático ya que dio paso a los Transformers [4], con su arquitectura basada en encoder-decoder y su representación mediante *embeddings* junto a los modelos de atención, tal y como se muestra en el Esquema 1. Aquí también fue decisivo la manera de representar las palabras de una manera vectorial. Esto se llevó a cabo mediante los *word-embeddings* o *embeddings* los cuales dotaban de un punto del espacio vectorial a una palabra que además fuera cercana matemáticamente de aquellas que semánticamente así lo fueran, añadiendo características como la posibilidad de aplicar operaciones matemáticas sobre ellas.



Esquema 1. Arquitectura del Transformer encoder-decoder de Vaswani

Esta nueva arquitectura y método dio la posibilidad de mejorar los modelos ya conocidos y poder entrenarlos con cantidades ingentes de datos. Con ello y pequeñas variaciones se presentaron modelos que hacen uso del encoder o del decoder y alguna modificación, entre los cuales se encuentran BERT [5], T5 [6] o GPT-3 [7]. Estos enormes modelos dieron paso a los modelos preentrenados, los cuales a partir de la realización de un *fine-tuning*, un pequeño entrenamiento en específico para la tarea en cuestión y, una posible modificación de la última parte de la arquitectura, era suficiente para disponer de la potencia de estos modelos de lenguaje orientados a resolver dicha tarea. De este modo se consiguieron resolver tareas como el reconocimiento de entidades nombradas (NER), la extracción de palabras clave, el resumen de texto o la obtención de la respuesta sobre una pregunta y un texto.

Recientemente también se han creado grandes modelos orientas en específico al idioma español, entre ellos se encuentra MarIA [8], un modelo preentrenado con un corpus masivo de textos limpiados y deduplicados del Archivo Web en español rastreado por la Biblioteca Nacional de España entre 2009 y 2019. Este conjunto de datos de entrenamiento constituye un conjunto mucho más diverso para el español en

comparación con la porción española del corpus en modelos multilingües como Multilingual BERT (mBERT) [9] así como en otros modelos monolingües en español como BETO [10].

Debido a estos avances cada vez se buscan tareas más complejas que resolver y conseguir modelos que puedan realizar más tareas y más profundas. Por ello otras de las tareas a mencionar son la similitud entre textos. No únicamente por su parte sintáctica sino, más importante aún, por la parte semántica. Como herramientas para probar distintas tareas sobre la similitud entre textos se encuentra SentEval [11]. Incluye más de 15 tareas de similitud semántica con la medición basada en la similitud coseno de la representación. Otra de las tareas más conocidas relacionadas es la detección de preguntas similares en Quora, Quora Question Pairs Dataset. Esto se presentó como una competición y actualmente el mejor modelo es ALICE [12] con un resultado de 0.9 de F1. Sin embargo, estas tareas son específicas para el inglés. Entre los modelos más destacados se encuentra SentenceTransformers [13] el cual incluye multitud de modelos orientados específicamente a la vectorización de las palabras para obtener la similitud adecuada. Se basa en modelos preentrenados y ofrece también la posibilidad de multilenguaje.

Otro de los problemas es la agrupación de varios elementos en función a la cercanía, parecido o relación que se tengan en un mismo grupo, esta tarea es conocida como agrupamiento o *clustering*. El objetivo es atribuir cada elemento a un grupo, *cluster*, de manera que entre los elementos del mismo grupo la distancia sea mínima y que las distancias entre grupos sea la más grande posible. Respecto a los algoritmos relacionados se encuentra el K-Means [14] donde se agrupan los datos en K *clusters*. El objetivo del algoritmo es minimizar la suma de las distancias al cuadrado entre cada punto de datos y el centroide de su *cluster* asignado.

Otro algoritmo utilizado para el agrupamiento de datos es el EM [15]. Se basa en la idea de que los datos pueden provenir de una mezcla de varias distribuciones de probabilidad (componentes latentes). El algoritmo consta de dos pasos iterativos: el paso de Expectación (E-step) y el paso de Maximización (M-step). En el E-step, se calcula las probabilidades de pertenencia de cada punto de datos a cada componente latente, basadas en las estimaciones actuales de los parámetros. En el M-step, se ajusta los parámetros del modelo para maximizar la verosimilitud de los datos dados los puntajes de pertenencia calculados en el E-step.

El siguiente algoritmo es Affinity Propagation (AP) [16] es una técnica de *clustering* que se destaca por su capacidad para identificar automáticamente el número

de *clusters* en un conjunto de datos y asignar puntos de datos a esos *clusters*. A diferencia de otros métodos de *clustering*, AP no requiere que se especifique previamente el número de *clusters*, lo que lo hace especialmente útil en situaciones donde esta información es desconocida.

El modelo que se plantea desarrollar se basa en las técnicas explicas de procesamiento del lenguaje natural para poder vectorizar de alguna manera las noticias, mediante *embbedings* por ejemplo, y de esta manera tener un valor de similitud semántica que se pueda comparar entre noticias. Con ello, se obtiene su medida con similitud coseno y finalmente, mediante el umbral se consigue establecer y diferenciar las agrupaciones, *clusters*.

Capítulo 3

Tecnologías y herramientas utilizadas

3.1. Visual Studio Code

Visual Studio Code[1], o VSCode, es un entorno de desarrollo integrado (IDE). Ofrece una interfaz amigable, soporte para múltiples lenguajes de programación, y una amplia variedad de extensiones que facilitan la escritura de código y la gestión de proyectos. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código que hace que se convierta en una herramienta necesaria para cualquier tipo de proyecto.

3.2. Git

Git[2], es un software de control de versiones diseñado por Linus Torvalds. Permite la colaboración múltiple sobre un mismo directorio de trabajo con el fin de realizar un buen mantenimiento y un trabajo en equipo de manera rápida, eficaz y segura. Sus principales características son la rapidez en la gestión por ramas, la gestión distribuida y eficiente y el almacenamiento en paquetes. Además, tanto VSCode como Google

[1] https://code.visualstudio.com/
[2] https://git-scm.com/

Colab están integrados con esta herramienta que dota de funcionalidad a la plataforma GitHub[3] para realizar el alojamiento de los servicios a través de la red.

3.3. Google Colaboratory

Google Colaboratory[4] o Colab, es una plataforma de desarrollo en la nube, permitiendo el acceso a recursos de hardware, como GPUs o TPUs, para la ejecución de modelos de aprendizaje profundo y el procesamiento intensivo de datos. Su colaboración en tiempo real y la facilidad de uso son una de sus ventajas para experimentar y compartir código y resultados.

3.4. Python

Python[5] es un lenguaje de alto nivel de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código que se utiliza para desarrollar aplicaciones de todo tipo. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma. Por estas y otras razones es ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el aprendizaje automático.

3.4.1. Pandas

Pandas[6] es una librería de Python especializada en la manipulación y el análisis de datos. Ofrece estructuras de datos y operaciones para manipular tablas numéricas y series temporales. Gracias a su integración con Python se obtiene un manejo de las bases de datos rápida, sencilla y con una gran posibilidad de poder realizar modificaciones sobre los documentos.

[3] https://github.com/
[4] https://colab.research.google.com/?hl=es
[5] https://www.python.org/
[6] https://pandas.pydata.org/

3.4.2. NumPy

NumPy[7] es una biblioteca para el lenguaje de programación Python que da soporte para crear vectores y matrices grandes multidimensionales, junto con una gran colección de funciones matemáticas de alto nivel para operar con ellas. Gracias al procesamiento que realiza sobre vectores y matrices resulta de gran ayuda para el tratamiento de los datos.

3.4.3. Matplotlib

Matplotlib[8] permite la generación de gráficos y visualizaciones de datos de alta calidad, ayudando en la presentación de resultados y detección de hallazgos de manera efectiva. Esta herramienta se complementa con las anteriores para poder alcanzar la correcta representación y análisis de datos.

3.5. Selenium

Selenium se define como un software para la automatización de pruebas y la extracción de datos de sitios web. Esto resulta especialmente útil para la recopilación de datos de fuentes externas y la automatización de tareas repetitivas. Además, se encuentra disponible en varios lenguajes de programación y, en concreto, con Python que es de interés. Debido a estas características se permite hacer uso de diferentes navegadores webs de manera automatizada como si fuese un humano quien lo manipula. Gracias a ello, resulta de gran facilidad para realizar el raspado de datos web (*scraping*).

3.6. Jupyter Notebook

Jupyter Notebook [9] es una herramienta de código abierto utilizada para desarrollar y compartir código en diferentes lenguajes de programación, incluyendo Python, R y Julia. Permite la integración de texto, código, gráficos y otros elementos

[7] https://numpy.org/
[8] https://matplotlib.org/
[9] https://jupyter.org/

multimedia en un documento interactivo, lo que lo convierte en una herramienta muy útil para la investigación reproducible. En concreto, Jupyter Notebook es una aplicación cliente/servidor que puede correr localmente en un navegador de Internet sin necesidad de tener una conexión a Internet. Sin embargo, al tratarse de una aplicación de red también puede ejecutarse remotamente a través de Internet. La facilidad de uso, flexibilidad, colaboración en tiempo real con varios usuarios y su visualización hacen que estos cuadernos o notebooks sean una opción muy utilizada en el ámbito de ciencia de datos y aprendizaje automático, entre otros. Google Colab hace uso de estos cuadernos para poder utilizar su herramienta y poder obtener el máximo beneficio de ambas dos.

3.7. Hugging Face

Hugging Face[10] se define como una de las mayores comunidades de inteligencia artificial. Gracias a la colaboración de los usuarios se disponen de miles de modelos ya preentrenados para abordar cualquier tipo de tarea, corpus de datos, entornos para probar los modelos y la potencia de la comunidad con sus blogs, artículos y foros específicos.

Debido a su importancia disponen de una biblioteca Python que proporciona acceso a modelos de NLP preentrenados y herramientas para el desarrollo de modelos personalizados. Esto resulta imprescindible para implementar modelos avanzados de procesamiento de lenguaje natural y acelerar el desarrollo de soluciones basadas en NLP.

3.8. Scikit-Learn

Scikit-learn [11] , también conocido como sk-learn, es una biblioteca para aprendizaje automático de software libre para Python. Proporciona una amplia gama de herramientas para la clasificación, regresión y evaluación de modelos lo que permite la implementación de algoritmos de aprendizaje automático y estadísticas.

[10] https://huggingface.co/
[11] https://scikit-learn.org/stable/

3.9. spaCy

La herramienta spaCy[12] es una librería de software libre para el procesamiento de lenguajes naturales en el lenguaje de programación de Python. Es capaz de lematizar y tokenizar las palabras además de realizar el análisis morfosintáctico. También incluye una gran cantidad de modelos preentrenados en diferentes idiomas, reconocimiento de entidades nombradas (NER), clasificación de texto y aprendizaje mediante *embeddings* y Transformers. Esta aplicación no solo presenta un gran uso en el ámbito educativo y de investigación sino también en el industrial. A todo este potencial se le suma la facilidad de integración con otras herramientas como Google Colab o Jupyter Notebooks.

[12] https://spacy.io/

Capítulo 4

Corpus

Una de las partes más relevantes y a la cual se debe prestar mucha atención cuando se quiere desarrollar una aplicación de aprendizaje automático es la elección de un buen corpus. El disponer de un conjunto de datos representativo de la tarea, variado, con un tamaño adecuado y una correcta anotación es una de las claves para alcanzar un buen modelo y producto final.

Debido a que parte del sistema que se realiza es la obtención de noticias de diferentes medios para posteriormente poder agruparlas, se utiliza esa extracción de noticias para así poder disponer de un conjunto que sirva para obtener el mejor modelo posible. Para ello, y el primer paso, es la realización de un raspado de datos web, *scraper*.

4.1. Fuentes periodísticas

La elección de periódicos se ha llevado a cabo con la intención de incluir los máximos posibles, obteniendo así un mayor número de datos y, así también, poder tener más probabilidad de disponer de noticias que traten sobre el mismo tema y poder llevar a cabo el objetivo de realizar el agrupamiento de noticias, pero también intentando seleccionar aquellos que no compartan la misma ideología. Esta última anotación se ha realizado para poder obtener un corpus con unas noticias que a pesar de que deberían ser lo más objetivas posibles es conocido que esto puede variar entre varias fuentes, por ello, para un posible análisis automático se ha decido realizarlo de esta manera.

En concreto se han seleccionado un total de ocho periódicos intentando que exista un equilibrio entre las ideologías de derecha, centro e izquierda, en términos muy

generales y sin querer hacer más hincapié en el asunto. Los medios digitales seleccionados son: El Diario[13], El Español[14], El Mundo[15], El País[16], EPE[17], Okdiario[18], Publico[19] y 20 Minutos[20].

4.2. Scraper web

El proceso de extracción automática de conjuntos de datos específicos que se encuentran en la web es conocido como raspado de datos web, o *scraping*. Por lo tanto, el primer paso del proyecto será desarrollar una aplicación que consiga recopilar el mayor número de noticias y volcarlas en un fichero local para su posterior procesamiento. El sistema desarrollado hace uso de las tecnologías de Python como lenguaje de programación, Pandas, para la gestión y manipulación de los datos y Selenium para acceder a las páginas web y obtener los documentos deseados.

Su funcionamiento está basado en la introducción de dos fechas las cuales actúan para establecer el rango de fechas en las cuales se desea obtener todas las noticias de las fuentes periodísticas. Únicamente con esta indicación el sistema es capaz de acceder a las páginas de los periódicos, seleccionar aquellas noticias que se encuentren dentro del rango de fechas establecido, obtener los datos de la noticia que resulten de interés y almacenarlos en un fichero csv. Sin embargo, este filtro en las fechas puede verse ligeramente limitado debido al acceso a las noticias de algunos periódicos. En este punto se ha llegado a diferenciar entre diferentes tipos de acceso a las noticias de acuerdo a cada periódico.

El primer tipo es aquel que dispone de una hemeroteca y se accede rápidamente indicando ya sea por url o con algún otro tipo, el día en cuestión, y se consigue listar todas las noticias de dicho día. El segundo tipo es aquel que no dispone de la hemeroteca anterior y sus noticias se encuentran ordenadas mostrando en primer lugar aquella que es de última hora y al llegar al final de la página, dispone de una paginación donde se puede ir accediendo a las noticias anteriores. Este proceso permite acceder a

[13] https://www.eldiario.es/
[14] https://www.elespanol.com/
[15] https://www.elmundo.es/
[16] https://elpais.com/
[17] https://www.epe.es/es/
[18] https://okdiario.com/
[19] https://www.publico.es
[20] https://www.20minutos.es/

todas las noticias independientemente del rango de fecha, pero es más lento pues siempre empieza en el día actual y debe llegar a la página donde se encuentre las noticias del día de interés. Como contrapunto, si es un rango muy alejado puede llevar más tiempo, entre pasar las páginas y realizar la comprobación de si ya ha llegado y/o sobrepasado las fechas deseadas. El tercer y, último, tipo es similar al anterior, donde no dispone de hemeroteca y se tienen las ultimas noticias primero. Sin embargo, este último tipo o no dispone de paginación o, si dispone tiene un límite, por lo que si el rango de fechas deseados es muy alejado del día actual resulta imposible obtener las noticias de esa fuente.

Una vez explicado el funcionamiento del rango de fechas, el siguiente paso es extraer las noticias de una fuente. Para las fuentes que disponen de hemeroteca se obtienen todas las noticias de ese día y se almacenan en un DataFrame que incluye categoría, titular, fecha y url. Después se recorre ese pandas y se accede a la noticia donde acaba de extraerse todo el cuerpo de la noticia, autores y si es de pago o no. Para las noticias que no disponen de hemeroteca dista un poco de este flujo. Partiendo de la página inicial se acceden a las diferentes secciones del periódico, como puedan ser nacional, internacional, deportes, … Luego por cada página de sección se aplica lo comentado anteriormente. Primero se obtienen todas las noticias con sus links y después se va entrando uno a uno para conseguir todos los datos faltantes.

Todo este proceso de *scraping* es posible gracias al uso de Selenium que permite la utilización de navegadores navegando entre todas las páginas haciendo click en los diferentes botones, como si de un humano se tratase, la extracción del contenido html para recuperar los textos y, además, ejecuciones javascript como enviar y recibir peticiones web.

Específicamente se ha realizado un raspado de las noticias de los ocho medios digitales en un rango de fechas entre el 19 de julio de 2023 y 26 de julio de 2023 junto con el rango del 2 de agosto de 2023 al 3 de agosto de 2023, ambos inclusive en los dos casos. La justificación de obtener fechas de otro rango separado en el tiempo es para así disponer de más variabilidad en los datos y dar opción a nuevos grupos de noticias.

4.3. Preprocesamiento

Una vez se dispone de un fichero con las noticias extraídas de las diferentes páginas web, es necesario realizar una limpieza y análisis de los datos. En este punto se tienen un total de 10.069 noticias con las siguientes columnas:

- Newsletter: Nombre del periódico
- Section: Sección del periódico donde se incluye la noticia
- Category: Categoría asociada a la noticia
- Headline: Titular de la noticia
- Is_premium: Boolean que indica si la noticia es para suscriptores o no
- Authors: Lista con los autores de la noticia
- Text: Cuerpo de la noticia
- Url: Link de la noticia
- Published_date: Día que se ha publicado la noticia

Partiendo de este conjunto de datos se realizan los diferentes procesos para limpiar y disponer de datos de mayor calidad.

En primer lugar, se eliminan las noticias que se encuentren repetidas. Esto puede llegar a ser posible porque se haya accedido a la misma noticia desde secciones distintas, por lo que la manera de realizar esta criba es mediante la url. Después de eliminar las noticias con misma url, se reduce el total del corpus a 9.942 noticias.

En segundo lugar, se elimina las noticias que no dispongan de texto o que sean premium. Debido a la automatización en el proceso de extracción puede darse el caso que a pesar de etiquetar una noticia como premium esta disponga de un párrafo o dos en el texto, pues no es hasta que avanza un poco que las páginas suelen bloquear la web y tampoco se encuentra en el html para poder acceder a él anteriormente. De igual manera, en este proceso puede fallar el raspado y por ello guardar noticias que no disponen de texto. Con estos borrados el corpus pasa a tener 8.835 noticias

En tercer lugar, al realizar un análisis de las noticias hay muchas de ellas que pueden no resultar de interés en un primer momento ya sea porque incluyen promociones de producto o son campañas publicitarias, recetas de cocina o artículos de opinión. Por ello, y teniendo como objetivo querer obtener noticias que se puedan agrupar y por lo tanto se encuentren en varios medios se eliminan todas las noticias que no dispongan de secciones o categorías relacionadas con acontecimientos nacionales,

provinciales, internacionales, deporte, salud, cultura y alguna más en este ámbito. Con ello se consigue reducir a una cantidad de 5.190 noticias.

En cuarto lugar, y siguiendo con el estudio de las categorías y secciones, se realiza un renombre sobre estos valores para que se tengan el mismo. Un ejemplo claro, son valores que tienen ‘más deporte’ el cual se ha renombrado a ‘deporte’, ‘más barcelona’ cambiado a ‘barcelona’ o ‘f1’ a ‘formula 1’. Con ello se pretende que sea más significativo el nombre y que se comparta entre los diferentes periódicos y noticias.

En quinto, se realiza una conversión (*casting*) de la columna de fecha de publicación de la noticia de un tipo string a un tipo date, pues al obtener los datos, estos se almacenan tal cual se extraen del html de la página web.

En sexto y, último lugar, consiste en eliminar aquellas columnas que no resulten de interés para el resto del desarrollo del proyecto. En concreto, se eliminan las columnas de si es de pago o no la noticia y los autores de esta.

4.4. Anotación manual

Con el conjunto de noticias procesado se debe obtener una anotación en base a cada noticia para posteriormente poder obtener unas métricas de los diferentes sistemas y así poder comprobar la efectividad de estos y decidir cual funciona mejor. Para ello, y dado que el corpus se está creando de cero es necesario realizar un etiquetado manual.

Llegados a este punto se disponen de un total de 5.190 noticias de ocho fuentes periodísticas digitales distintas repartidas en nueve días en diferentes rangos de fecha. La finalidad a obtener es disponer de una cantidad de noticias agrupadas por el mismo contenido que ofrecen, es decir, que estén hablando del mismo suceso. Para ello, y de manera manual, se miran los titulares del corpus para identificar aquellas noticias que traten de lo mismo. La similitud entre los titulares hace relativamente sencillo poder identificar aquellas noticias similares, sin embargo, la búsqueda a lo largo de todas las noticias es el proceso tedioso y que conlleva más tiempo.

Una vez realizado este proceso, añade una nueva columna al DataFrame con nombre *group_cluster* que se trata de un número entero que se compartirá entre aquellas noticias que correspondan con el mismo grupo y, por tanto, que relacionen las noticias que tratan sobre el mismo acontecimiento. Debido al largo proceso que lleva

etiquetar los datos se decide obtener un total de 14 grupos de noticias derivando en un total de 52 noticias las cuales fueron seleccionadas entre un total de cinco días distintos.

Con la idea de poder extraer el máximo de información posible de las noticias para posteriormente poder obtener un mejor modelo y evaluarlo se anotan otros datos distintos no relacionados con el grupo genérico de noticias. En concreto, y a nivel individual, se obtiene por cada una de las noticias sus entidades nombradas y las palabras clave.

Para las palabras clave, y debido a que no se dispone de un anotador profesional, se ha leído la noticia y se han extraído aquellas palabras más relevantes y que pudieran tener más peso para identificar a la noticia de manera esquemática, sencilla y rápida.

De manera igual que el caso anterior, se ha extraído las entidades de la noticia clasificándolas en diferentes categorías por separado. Las categorías de las entidades que se han seleccionado han sido: organizaciones, personas, lugares, productos, eventos, títulos de arte o medios, redes sociales y direcciones.

Cabe mencionar la gran cantidad de tiempo que ha llevado realizar el proceso de anotación manual, tanto agrupar entre tantos tipos distintos de noticias como su posterior extracción de información.

4.5. Particiones

Una vez con las noticias extraídas, seleccionadas y anotadas se dispone del corpus al completo para poder continuar con las tareas de aprendizaje y poder desarrollar el sistema deseado. Finalmente, las columnas que componen el corpus son las siguientes:

- Newsletter: Nombre del periódico
- Section: Sección del periódico donde se incluye la noticia
- Category: Categoría asociada a la noticia
- Headline: Titular de la noticia
- Text: Cuerpo de la noticia
- Url: Link de la noticia
- Published_date: día que se ha publicado la noticia

- Group_cluster: Entero que representa a que grupo corresponde la noticia
- Keywords: Listado de palabras claves de la noticia
- Organizations, people, places, products, events, art_media, social_network, addresses: Cada una de los grupos de entidades nombradas de la noticia

Con el corpus actual, en el cual se encuentran etiquetadas un total de 52 noticias y debido a que los modelos que se van a utilizar son preentrenados en composición entre ellos, por lo que no se va a realizar un entrenamiento, se decide dividir el corpus en dos partes. Estas partes corresponderán a la de validación (*dev*) y la de prueba (*test*). Las partes se han dividido al 50% según los grupos, que a nivel de noticia se traduce en 27 noticias para *dev* y 25 para *test*, tal y como muestra la Tabla 1. Por último, mencionar que las noticias tanto para *dev* y *test*, se encuentran mezcladas según los dos rangos de fecha, aunque siempre cumpliendo que todas las noticias de un grupo se encuentren en la misma parte de separación del corpus. Debido a la poca cantidad de muestras de las cuales se dispone, esta tarea que se quiere realizar entraría dentro del campo del *few-shot learning*.

Tabla 1. Particiones del corpus

	Dev	***Test***	**Corpus**
Nº grupos	7	7	14
Nº noticias	27	25	52

Capítulo 5

Desarrollo del sistema

El sistema completo que se desarrolla consta tanto de la extracción de noticias de medios de internet como de su agrupación por aquellas que traten sobre lo mismo. Como para la segunda parte de obtener un modelo capaz de realizar la agrupación se necesita disponer de un corpus y, este a su vez, se ha generado a partir del extractor de noticias ya explicado en el capítulo Scraper web, este capítulo va a centrarse en el desarrollo de un modelo de aprendizaje automática para agrupar las noticias.

Para la obtención de este modelo van a realizarse diferentes pasos. Estos incluyen un preprocesado previo, una posterior extracción de palabras claves y entidades nombradas para disponer de más datos y, finalmente, los métodos basados en similitud semántica para agrupar textos, en este caso, las noticias.

5.1. Preproceso

Con las particiones de *dev* y *test* ya separados se procede a realizar un preproceso previo sobre los datos para poder agrupar, modificar o limpiar aquellos que resulten de interés.

Primero, partiendo de las columnas del titular y del cuerpo de la noticia, *headline* y *text*, respectivamente se crea una nueva que sea la concatenación de estas dos. Con ello se añade al texto la semántica del titular que por norma general puede verse como un resumen con los elementos más importantes.

Segundo, dado que los textos extraídos de la web no se han manipulado, algunos de ellos se encuentran en mayúscula y otros en minúscula, se ha utilizado la

unión del paso anterior, *headline+text*, para normalizarlo. Para esta tarea se ha hecho uso de spaCy y su modelo mediano de noticias en español, *es_core_news_md*[21]. Después de convertir todo el texto a minúsculas, se ha creado un *pipeline* de spaCy para eliminar las palabras vacías (*stopwords*), los signos de puntuación, direcciones web y correos electrónicos. En última instancia se transforman las palabras restantes a su lema correspondiente.

Tercero, por una razón similar al caso segundo, las palabras claves se encuentran almacenadas como texto, por ello, para poder trabajar más cómodamente en el *DataFrame*, se transforma a una lista de palabras, en vez de un único string como se encontraba.

Cuarto, y último, la unificación de clases de las entidades nombradas. Actualmente, se disponen de las entidades separadas por varias categorías y la modificación es unir todas las entidades en una única columna que sea la lista de cada entidad nombrada.

Con este preproceso aplicado al conjunto de datos se dispone finalmente de las siguientes columnas:

- Newsletter: Nombre del periódico
- Section: Sección del periódico donde se incluye la noticia
- Category: Categoría asociada a la noticia
- Headline: Titular de la noticia
- Text: Cuerpo de la noticia
- Url: Link de la noticia
- Published_date: Día que se ha publicado la noticia
- Group_cluster: Entero que representa a que grupo corresponde la noticia
- Headline_text: Concatenación de las columnas headline y text
- Headline_text_clean: Texto normalizado y lematizado de la columna headline_text
- Keywords: Listado de palabras claves de la noticia
- Ner: Lista de las entidades nombradas de la noticia

[21] https://spacy.io/models/es#es_core_news_md

5.2. Reconocimiento de entidades nombradas (NER)

A pesar que el objetivo final del proyecto es realizar la agrupación de noticias, se ha decidido hacer una extracción de entidades nombradas con el objetivo de poder obtener toda la información posible de cada noticia. Por ello se ha optado por poner en comparativa diferentes modelos preentrenados con la intención de utilizar el mejor de ellos para la tarea de agrupación.

Como se ha explicado en el apartado Particiones, el uso de los modelos preentrenados sin ningún tipo de entrenamiento hace que estos se hayan utilizado directamente sobre los conjuntos de *dev* y *test*.

5.2.1. spaCy

El primero de los modelos que se ha evaluado ha sido los desarrollados mediante la librería spaCy. Esta librería dispone de varios modelos específicos tanto para diferentes idiomas como tamaños de modelos, que a priori se traduce en un mejor rendimiento. Aunque se define que los modelos se basan en el entrenamiento de multitarea como BERT, se ha querido separar de este último grupo a la hora de realizar las pruebas y análisis.

Los modelos que se han evaluado han sido los preentrenados para el idioma español de tamaño pequeño y mediano, *es_core_news_sm* y *es_core_news_md* respectivamente. A pesar que estos modelos ofrecen un gran rendimiento según sus métricas, en el caso actual no resultan tan altos como cabría esperar. Podría llegar a ser entendible dado que las etiquetas con las cuales se está realizando la comparación han sido anotadas por alguien no profesional ni dedicado a ello.

La evaluación de los modelos obtiene el resultado esperado donde el de mayor tamaño ha sido el que obtiene mejor resultado en el conjunto de validación. La Tabla 2 muestra los valores que se han obtenido a la hora de evaluar las métricas de precisión, *recall* y F1.

Tabla 2. Resultados en la tarea NER de modelos spaCy en validación

Model	Precision	Recall	F1
es_core_news_sm	0.30	0.85	0.44
es_core_news_md	***0.39***	***0.87***	***0.54***

5.2.2. XLM-RoBERTa

El segundo modelo es la realización de un *fine-tuning* sobre el modelo de XLM-RoBERTa, es decir un reentrenamiento sobre el modelo preentrenado para una tarea especifica.

El modelo utilizado ha sido MMG/xlm-roberta-large-ner-spanish [17], el cual ha realizado un *fine-tuning* sobre el modelo grande XLM-RoBERTa-large para la tarea de NER orientada al español sobre el corpus CoNLL-2002 [18]. Este modelo ha sido accesible gracias a Hugging Face pues es ahí donde ha podido obtenerse.

El resultado que se ha obtenido es el que se muestra en la Tabla 3, que al disponer únicamente de un modelo basado en XLM-RoBERTa es el elegido para evaluar en *test*. Sin embargo, a nivel de validación ofrece el mismo resultado F1 que los resultados anteriores de spaCy, aunque difiere en la precisión y *recall*, donde este obtiene mejor precisión mientras que spaCy mejor *recall*.

Tabla 3. Resultados en la tarea NER de modelos XLM-RoBERTa en validación

Model	Precision	Recall	F1
MMG/xlm-roberta-large-ner-spanish	**0.50**	**0.59**	**0.54**

5.2.3. BERT

Los últimos modelos se basan en el modelo ya preentrenado BERT. El primer modelo es NazaGara/NER-fine-tuned-BETO [19] el cual ha realizado el *fine-tuning* anterior de la tarea NER sobre el corpus CoNLL-2002 y Babelscape/wikineural [20]. Pero a diferencia que el modelo anterior basado en XLM-RoBERTa este lo ha realizado sobre BETO [21], el cual es un *fine-tuning* de BERT sobre el idioma español.

El segundo modelo es realiza un *fine-tuning* sobre BERT, pero en este caso para incluir más de 10 idiomas además de la tarea de NER, Davlan/bert-base-multilingual-cased-ner-hrl [22].

El tercer y último modelo es Babelscape/wikineural-multilingual-ner, el cual ha realizado un *fine-tuning* para la tarea de NER es español del corpus que también comparte el primer de los modelos vistos para BERT, Babelscape/wikineural.

Los resultados obtenidos de estos modelos son los que se muestran en la Tabla 4. En cuanto a la precisión se obtiene un empate entre el modelo de BERT multilenguaje y el de wikineural y, en último lugar el de BETO. Sin embargo, son el de BETO y wikineural los que alcanzan el mayor valor de F1. En este punto, y a pesar que la diferencia radica en medio punto que sube o baja tanto de precisión como de *recall* en ambos modelos, se decide seleccionar el último modelo para *test*. La justificación es para premiar la mayor precisión, ya que se decide que esa métrica puede ofrecer mejores resultados para que no se seleccionen entidades que sean erróneas y que luego a la hora de realizar el agrupamiento pueda suponer un conflicto con otras noticias.

Tabla 4. Resultados en la tarea NER de modelos BERT en validación

Model	Precision	Recall	F1
NazaGara/NER-fine-tuned-BETO	0.63	**0.65**	**0.64**
Davlan/bert-base-multilingual-cased-ner-hrl	**0.68**	0.57	0.62
Babelscape/wikineural-multilingual-ner	***0.68***	*0.60*	***0.64***

5.2.4. Resultados

Una vez seleccionados aquellos modelos que pueden ofrecer un mejor resultado a la hora de extraer las entidades nombradas se prueban contra el corpus de *test*. Específicamente, los modelos escogidos son el modelo mediano de spaCy, el *fine-tuning* sobre XLM-RoBERTa-large y el de wikineural que se realiza sobre BERT.

Los resultados sobre el conjunto de test que se han obtenido son los de la Tabla 5. El peor modelo es el basado en XLM-RoBERTa, y los otros dos cabe hacer algunas aclaraciones. Aunque el modelo de spaCy solo difiera el algo más de medio punto con el de wikineural, que alcanza el mejor resultado de F1, resulta necesario analizar el resto de métricas. Como se había comentado anteriormente en el caso de los modelos BERT con la precisión, ocurre de la misma manera en este caso. La precisión de spaCy es mucho menor que la de wikineural aunque su *recall* sea mayor, lo que hacen un F1 que pudiera ser equiparable. Pero se busca tener una buena precisión con un *recall* adecuado por lo que eso implica que el mejor modelo que se ha podido obtener para la extracción de entidades nombradas sea el basado en BERT, Babelscape/wikineural-multilingual-ner con un 0.56 de F1 y 0.61 de precisión.

Tabla 5. Resultados en la tarea NER en test

Method	**Model**	**Precision**	**Recall**	**F1**
spaCy	es_core_news_md	0.35	**0.83**	0.49
XLM-RoBERTa	MMG/xlm-roberta-large-ner-spanish	0.40	0.46	0.43
BERT	Babelscape/wikineural-multilingual-ner	***0.61***	***0.**51*	***0.56***

El Esquema 2 muestra la extracción de entidades que consigue realizar el modelo ganador, Babelscape/wikineural-multilingual-ner, sobre una noticia sobre la muerte de la cantante irlandesa Sinéad O'Connor. Se aprecia como consigue extraer bastantes nombres propios, así como eventos y también canciones.

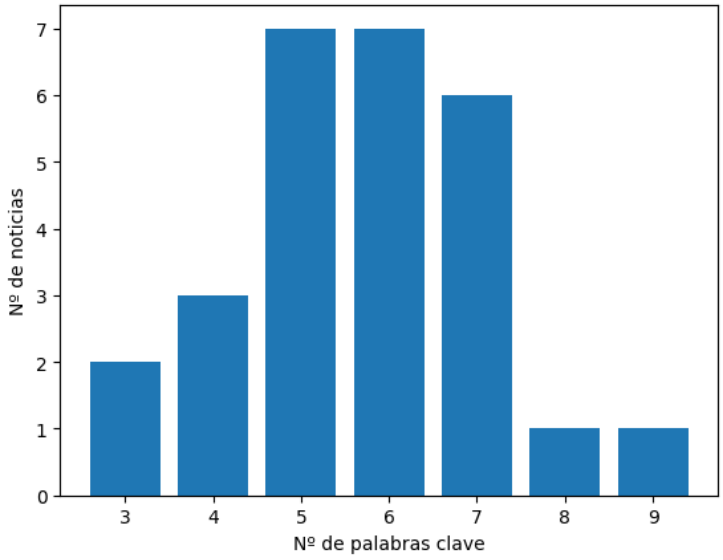
$$['Sinéad\ O\ '\ Connor','Dublín','Nothing\ Compares\ 2U','Prince','Nothing\ Compares\ 2\ U',$$
$$'Magdalenas',\ 'O'\ Connor',\ 'Disco\ Clásico\ Irlandés',\ 'Premios\ de\ la\ Música',\ 'RTÉ',$$
$$'The\ Lion\ and\ the\ Cobra',\ 'premio\ Grammy',\ 'IDo\ Not\ Want\ What\ I\ Haven'\ t\ Got',$$
$$'Grammy','Saturday\ Night\ Live','Iglesia','Juan\ Pablo\ II']$$

Esquema 2. Ejemplo NER del modelo Babelscape/wikineural-multilingual-ner para la noticia de la muerte de Sinéad O'Connor

5.3. Extracción de palabras claves (keywords)

Al igual que con el reconocimiento de entidades nombradas, se va a realizar un breve estudio para obtener aquel modelo que logre extraer las mejores palabras claves y, tener más información a la hora de agrupar las noticias más adelante. Al igual que con la tarea de NER, los modelos que se han utilizado son librerías o modelos ya preentrenados lo que evita el reentrenamiento y se evalúan directamente.

Antes de comenzar, se ha realizado un análisis de la cantidad de palabras clave de las cuales disponen las noticias, aquellas que han sido etiquetadas de manera manual. La Gráfica 1 muestra la distribución de acuerdo al número de palabras clave que dispone cada noticia. La media de palabras clave por noticia que se obtiene es de 5.7 y su mediana de 6, por lo que la cantidad de palabras claves a la cual se va a fijar para obtener en los modelos será de seis palabras.



Gráfica 1. Némero de noticias que tienen la misma cantidad de palabras claves en validación

5.3.1. Yake!

YAKE! [23] se define como un método ligero de extracción automática no supervisada de palabras clave que se basa en características estadísticas del texto extraídas de documentos individuales para seleccionar las palabras clave más importantes. Entre sus características destaca que no necesita ser entrenado en un conjunto concreto de documentos, ni depende de diccionarios, corpus externos, tamaño

del texto, idioma o dominio. Además, cuenta con interacción directa con Python para poder usarlo fácilmente mediante la instalación del paquete.

La extracción de las palabras clave se realiza a partir de un método el cual puede ser configurado levemente indicando idioma, numero máximo de n-gramas, cantidad máxima de palabras o el tamaño de la ventana, entre otros. En este caso, el idioma se ha fijado a español y se ha realizado un barrido entre varios valores de n-gramas para poder analizar con cual se obtiene mejores resultados.

La Tabla 6 muestra los diferentes resultados obtenidos iterando sobre los diferentes valores de n-gramas fijándolos a un máximo de entre uno y tres. Todas las métricas, disminuyen su valoración a medida que se aumenta el máximo valor de n-grama. Este comportamiento puede resultar previsible dado que normalmente las palabras claves no suelen ser compuestas o tener más de una para el mismo concepto o término, por ello, se obtiene el mejor resultado con el menor número de n-gramas, uno es este caso.

Tabla 6. Resultados en la tarea de extracción de palabras clave con Yake! en validación

Max n-gram	Precision	Recall	F1
1	***0.27***	***0.29***	***0.28***
2	0.16	0.17	0.16
3	0.12	0.12	0.12

5.3.2. textacy

Textacy [24] es una biblioteca de Python para realizar diversas tareas de procesamiento del lenguaje natural (NLP), como una capa de abstracción superior de spaCy.

Como esta biblioteca está por encima de spaCy debe utilizarse los modelos preentrenados de esta y cargarlos en textacy. El que se ha utilizado ha sido el modelo pequeño y seguidamente se ha vuelto a fijar el tamaño del listado y, en este caso, no se ha podido realizar ninguna configuración adicional. El resultado obtenido de manera por defecto ha sido el que se muestra en la Tabla 7, alcanzando un 0.18 de F1.

Tabla 7. Resultados en la tarea de extraccion de palabras clave con textacy en validación

Precision	Recall	F1
0.18	***0.19***	***0.18***

5.3.3. spaCy

Para la extracción de palabras clave utilizando spaCy se han seleccionado los sustantivos, adjetivos y nombres propios, en primera instancia. Esto es posible gracias a que al tokenizar las palabras incluyen también su etiqueta gramatical asociada (*pos-tag*) y se filtra fácilmente. También se han eliminado aquellas palabras que formen parte de las *stopwords* o de signos de puntuación. Una vez con las palabras seleccionadas se realiza un conteo y se ordena seleccionando aquellas que tengan más ocurrencias en el texto.

Los modelos que se han evaluado han sido el pequeño y el mediano y como sucede en el caso de NER, cuan mayor es el modelo, mejor resultado ofrece. Las puntuaciones son las mostradas en la Tabla 8, donde aumenta con el modelo, y esto se traduce en una mejor asignación de las etiquetas gramaticales que han sido la clave para poder extraer las palabras, logrando un 0.39 de F1 en validación.

Tabla 8. Resultados en la tarea de extracción de palabras clave con spaCy en validación

Modelo	Precision	Recall	F1
es_core_news_sm	0.37	0.39	0.38
es_core_news_md	***0.38***	***0.40***	***0.39***

5.3.4. KeyBERT

KeyBERT es una técnica de extracción de palabras clave minimalista y fácil de usar que aprovecha los *embeddings* de BERT para crear las palabras y frases clave más similares a un documento. En primer lugar, se extraen los *embeddings* de los documentos con BERT para obtener una representación a nivel de documento. A continuación, se extraen los *embeddings* de palabras para las palabras/frases de n-gramas. Por último, se utiliza la similitud coseno para encontrar las palabras/frases más

parecidas al documento. Estas palabras pueden identificarse como las que mejor describen todo el documento y, por lo tanto, las palabras clave.

Debido al uso de *embeddings*, resulta necesario poder incluir un modelo para realizar esa conversión. En este análisis se han comparado dos modelos de sentence-transformers [13] los cuales son multilenguaje para poder tratar correctamente con el español, distiluse-base-multilingual-cased-v1 [25] y distiluse-base-multilingual-cased-v2 [25]. A su vez, la biblioteca permite cierta configuración relacionada con los valores de n-gramas a utilizar, por lo que la comparativa se realizara entre los modelos y n-gramas.

Los resultados que se han obtenido son los mostrados en la Tabla 9. Al igual que ocurría con Yake! a medida que aumenta el tamaño de n-grama disminuyen las prestaciones, por lo que se obtienen los mejores resultados con unigramas. En la comparación entre los modelos se obtiene una leve mejora en el v2 pero muy sutil en la precisión, pero será el elegido para analizar en *test*.

Tabla 9. Resultados en la tarea de extraccion de palabras clave con KeyBERT en validación

Model	N-gram	Precision	Recall	F1
distiluse-base-multilingual-cased-v1	(1,1)	0.11	**0.12**	0.11
	(1,2)	0.03	0.03	0.03
	(1,3)	0.01	0.01	0.01
	(2,3)	0.01	0.01	0.01
distiluse-base-multilingual-cased-v2	*(1,1)*	***0.12***	***0.12***	***0.12***
	(1,2)	0.02	0.02	0.02
	(1,3)	0.00	0.00	0.00
	(2,3)	0.00	0.00	0.00

5.3.5. KeyBERT + KeyPhraseVectorizers

En esta aproximación va a hacerse una combinación entre dos librerías, KeyBERT, ya conocía de antes, y KeyPhraseVectorizers. KeyPhraseVectorizers se basa en spaCy para etiquetar los textos, luego extrae las palabras cuyas etiquetas concuerden con una expresión regular predefinida y, por último, se calcula la matriz documento-frase clave. Sus características son que extrae las frases clave de acuerdo al *pos-tag*, no es necesario especificar la cantidad de n-gramas y soporta multilenguaje.

La implementación de este hibrido, consta de una primera parte donde se inicializa KeyBERT con el mejor modelo anterior, el distiluse-base-multilingual-cased-v2. Una segunda parte que inicializa KeyPhrase que para ello debe incluir el modelo que se quiera de spaCy, ya que está basada en él, en este caso el modelo mediano. La carga de este modelo de spaCy es necesario para poder indicar como debe realizarse la extracción de características. En este caso se han utilizado dos extractores, CountVectorizer y TfidfVectorizer. Una vez con ello definido, al modelo KeyBERT, se la vincula su *vectorizer*, el texto a analizar y el número máximo de términos a recuperar y se obtiene el resultado.

Los resultados de este experimento son los que se muestran en la Tabla 10. En este caso se han obtenido los mismos resultados sin importar el vectorizador a utilizar, por lo que se va a seleccionar aquel que gasta tf-idf, decidido arbitrariamente.

Tabla 10. Resultados en la tarea de extracción de palabras clave con KeyBERT + KeyPhraseVectorizers en validación

Model	Vectorizer	Precision	Recall	F1
distiluse-base-multilingual-cased-v2	Count	0.09	0.10	0.09
	Tf-idf	0.09	0.10	0.09

5.3.6. Resultados

Después de haber analizado y puesto a prueba con el conjunto de validación diferentes aproximaciones y modelos para la extracción de palabras clave, se han seleccionado aquellos que ofrecían mejores resultados para compararlos entre ellos con el conjunto de *test*. La Tabla 11 muestra las prestaciones de los mejores métodos.

Tabla 11. Resultados en la tarea de extracción de palabras clave en test

Method	Model	n-gram	Vectorizer	Precision	Recall	F1
Yake!	-	1	-	0.27	0.36	0.31
Textacy	-	-	-	0.16	0.21	0.18
spaCy	es_core_news_md	-	-	**0.33**	**0.43**	**0.37**
KeyBERT	distiluse-base-multilingual-cased-v2	1	-	0.07	0.10	0.08
KeyBERT + KeyPhrase Vectorizers	distiluse-base-multilingual-cased-v2	-	Tf-idf	0.11	0.14	0.12

Los que ofrecen peores resultados son los derivados de KeyBERT, tanto su versión solitaria como su combinación con KeyPhrase. Parece llamativo el comprobar que ahora la combinación de ambos elementos funciona mejor que solo KeyBERT, ya que en el análisis en el conjunto de validación sucedía lo contrario. Esto hace desconfiar bastante de la ejecución de ambos métodos por su oscilación.

El siguiente mejor modelo es el basado en textacy, el cual no se ha podido modificar sus parámetros y, por lo tanto, probar varias configuraciones, lo que lo ha estancado en un valor bastante uniforme de prestación.

Finalmente, se tienen los dos mejores modelos que se han obtenido, Yake! con un 0.31 de F1 y spaCy con un 0.37, alzándose como el mejor modelo de entre todos los analizados. Aunque en el conjunto de validación Yake! se encontraba diez puntos por detrás que spaCy, en test, consigue reducir esa distancia hasta poco más de cinco puntos. Esto hace que Yake! pueda convertirse en otra opción más que aceptable para la tarea de extracción de palabras clave. Sin embargo, ha sido spaCy el que ha logrado la mejor puntuación y es por ello el seleccionado para realizar la tarea y así poder obtener más información que sirva de ayuda para la agrupación de noticias.

El Esquema 3 muestra la extracción de palabras clave que ofrece spaCy sobre la noticia de la muerte de Sinéad O'Connor. En comparación con las entidades extraídas anteriores de ejemplo, estas se encuentran fijadas a seis palabras clave y pueden incluir tanto nombres propios como sustantivos o conceptos.

$$['O'Connor', 'años', 'álbum', 'cantante', 'Sinéad', 'Nothing']$$

Esquema 3. Ejemplo extractor de keywords del modelo spaCy para la noticia de la muerte de Sinéad O'Connor

5.4. Agrupación de noticias (*clustering*)

Una vez finalizado el estudio de NER y extracción de palabras clave para poder obtener más información de las noticias se puede proceder al desarrollo final y objetivo del proyecto, la agrupación de noticias similares.

Antes de empezar a postular modelos, aproximaciones o formas para conseguir la agrupación, debe extraerse la nueva información que hemos conseguido. Para ello, en todo nuestro corpus se aplica la tarea de NER con el modelo Babelscape/wikineural-multilingual-ner para el reconocimiento de entidades nombradas y conseguir una lista de entidades. Y, de igual manera, pero para conseguir una lista de palabras clave haciendo uso del modelo mediano de spaCy.

La tarea de agrupación (*clustering*) es una tarea compleja que su objetivo consiste en agrupar los elementos en grupos o *clústers* basados en la similitud que estos presentan. En NLP, se puede calcular la similitud semántica entre dos oraciones o palabras para determinar cuán similares son en términos de su significado. Esto es esencial para tareas como la búsqueda de documentos similares, la recomendación de contenido relacionado o la traducción automática, donde es importante comprender la similitud en el significado que subyace, en este caso, la similitud en cuanto al suceso o acontecimiento de la noticia.

En cuanto a las métricas más usuales para el cálculo de similitud entre textos o documentos se incluyen las siguientes. La distancia euclidiana que mide la distancia geométrica entre dos puntos en un espacio vectorial, como vectores de características que representan palabras, documentos u otros objetos. La distancia de Jaccard la cual se utiliza para calcular la similitud entre conjuntos, midiendo la intersección de dos conjuntos dividida por su unión. La distancia de Levenshtein mide la similitud entre dos

cadenas de caracteres midiendo el número mínimo de operaciones (inserciones, eliminaciones o sustituciones) requeridas para convertir una cadena en la otra. La última a mencionar es la similitud coseno calcula el coseno del ángulo entre dos vectores en un espacio vectorial. Cuanto más cercano sea el coseno a uno, más similares son los vectores.

La métrica para la similitud semántica que se ha seleccionado es la similitud coseno. Esta será la encargada de atribuir un valor que corresponda con la distancia entre los vectores que representan el texto a comparar. Para ello, previamente se debe cambiar el espacio de representación de manera textual original a uno vectorial o derivado capaz de poder establecer ese valor. Una vez con esta similitud obtenida, se debe indicar un valor umbral a partir el cuales acepten o rechacen documentos que sean o no similares.

En cuanto a los métodos que se aplican para realizar las agrupaciones, se proponen tres aproximaciones, la primera basada en la librería sk-learn, la segunda en spaCy y la tercera en SentenceTransformer.

5.4.1. Term Frequency times Inverse Document Frequency (Tf-idf)

Esta primera aproximación se basa en el uso de la librería sk-learn con tf-idf para el agrupamiento de las noticias. *Term Frequency times Inverse Document Frequency* (Tf-idf) es una métrica para encontrar el documento más relevante para cierto término dentro de una colección de documentos, o en este caso, cuan de relevante es una palabra en un documento. Su funcionamiento se basa en el conteo de cuantas veces aparece esa palabra en el documento y en el corpus, penalizando a aquellas que aparecen mucho, como las *stopwords*, y premiando a las que aparece en contadas ocasiones.

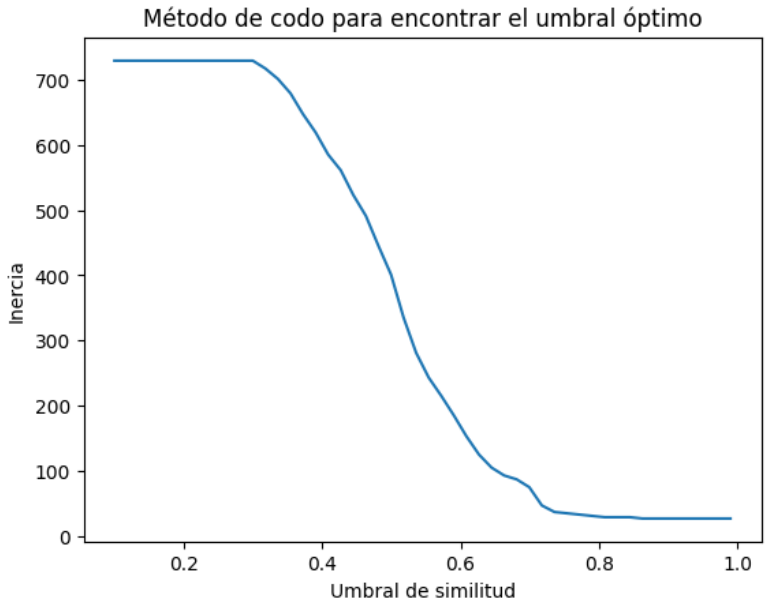
Para su implementación se ha calculado la matriz que genera la función de tfi-df de sk-learn con las noticias del corpus. Luego se ha aplicado la similitud coseno sobre la matriz, obtenido así, una matriz con los valores de similitud entre noticias. Llegados a este punto, el siguiente paso es establecer el umbral a partir del cual se consideran noticias semánticamente similares o no.

Para este cometido existen varias técnicas, pero la que se ha optado es la del método del codo (*elbow*). Esta técnica es utilizada en algoritmos de *clustering*, como K-

Means, donde se calcula la distorsión promedia de los *clusters*, que es la distancia promedia del centroide a todos los puntos del clúster. Así, cuando se va de una situación en la que el número de *clusters* es inferior al correcto a una situación en la que el número es el adecuado, el valor de la dispersión disminuye bruscamente, mientras que, si aumenta el número de *clusters* al adecuado, el valor de la dispersión se reducirá más lentamente, formando un codo en la gráfica.

A pesar de que no se ha podido identificar estos centroides, pues no se ha aplicado dicho algoritmo, al disponer de las similitudes cosenos puede obtenerse este grafico con otros datos pero que se comportan de la misma manera. En este caso se realiza el cálculo de la inercia para poder dibujar la gráfica. Primero, se seleccionan noticias similares fijando un umbral. Segundo, se cuenta el número de *clusters* que serán las noticias anteriores representada como una tupla de índices de noticias similares. Y, por último, el cálculo la inercia, que es la suma de las similitudes coseno que están por encima del umbral proporcionando una medida de cuán bien se agrupan los datos y cuán compactos son los clústeres. La Gráfica 2 es un ejemplo de esta representación sobre la cantidad de inercia en función del umbral de similitud. En este caso se han realizado los cálculos para el cuerpo de la noticia, la columna de texto. Los mejores valores para definir el umbral serán alrededor de 0.7 ya que es donde el valor de inercia disminuye más lentamente. Si se selecciona un umbral muy alto encontrará pocas noticias que etiquete como similares y, si es muy bajo, las estará agrupando todas como similares en el mismo cluster.

Una vez se tiene seleccionado el mejor umbral posible, se fija y se vuelven a buscar las noticias similares. En ese punto se tienen la relación de una noticia con sus noticias similares.



Gráfica 2. Evolución de la inercia para el texto en función del umbral de similitud

A continuación, se extraen las noticias de forma que quede una lista de listas, donde cada lista interior significa un *cluster* y, sus elementos, las noticias que se encuentran relacionadas. Esta forma de codificación se comparte en todas las aproximaciones y también para la extracción de los *clusters* de referencia y así poder compararlos y establecer las métricas.

Una vez explicado el funcionamiento de esta técnica se realizan las pruebas con las diferentes noticias e información de estas que se disponen. En concreto se plantean pruebas con el titular (*headline*), el texto (*text*), la unión de titular y texto, esta última unión, pero tokenizado y lematizado, el listado de entidades nombradas (*ner*), las palabras clave (*keywords*) y la unión de estas dos últimas listas. Esta combinación de pruebas se repite a lo largo de las otras aproximaciones. El único cambio que se produce es la parte previa de selección del umbral de similitud.

5.4.2. spaCy

La librería de spaCy ha servido de gran ayuda a lo largo del proyecto por todo lo que ofrece y las posibilidades de realizar tareas que es capaz de hacer. El ultimo uso que se le va a dar va a ser para la tarea principal del agrupamiento de noticias.

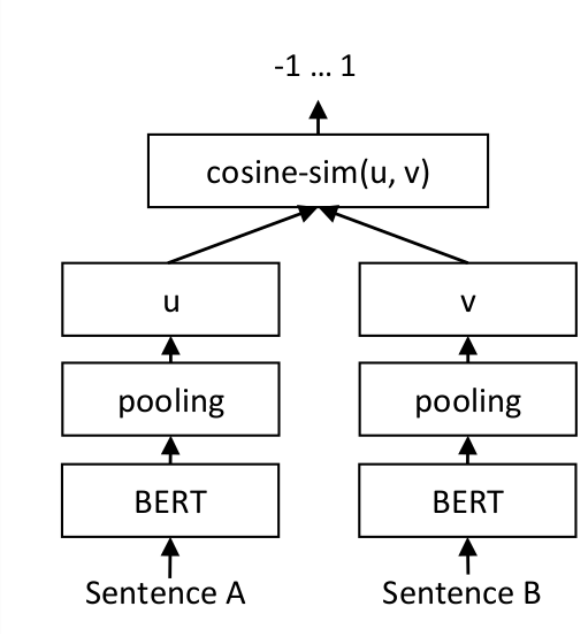
Al procesar el texto por los modelos, este es capaz de tokenizarlo, extraer las entidades nombradas, sus etiquetas gramaticales y, lo que resulta de interés, convertirlo

en vectores. Es decir, estos modelos también transforman el texto en *embeddings* los cuales serán capaces de realizar operaciones matemáticas entre ellos. Además, el propio token que se genera, dispone de una operación capaz de calcular la similitud del token actual con otro. Pero esta última característica únicamente está disponible a partir del modelo mediano, el cual era el que se va a utilizar por esta razón y por los mejores resultados que ofrece como se vio a la hora de las tareas de NER y extracción de palabras clave.

Por lo tanto, esta aproximación se basa en utilizar el *embedding* de spaCy para tener su representación y calcular la similitud con su función predefinida. Una vez se tiene eso calculado, se continúa representando como una matriz igual que en el caso anterior y, los pasos a seguir son idénticos, desde tener que realizar un barrido para seleccionar el mejor umbral, hasta la codificación y posterior cálculo de métrica.

5.4.3. SentenceTransformers

SentenceTransformers [25] es una biblioteca para Python del estado del arte del aprendizaje automático en cuanto a las frases, texto, imágenes y *embeddings*. Ofrece modelos de lenguaje preentrenados que se especializan en la representación semántica de oraciones o fragmentos de texto, en lugar de palabras individuales en una multitud de idiomas. Su entrenamiento consta de la introducción de dos oraciones que mediante su transformación a *embeddings* debe conseguir calcular la similitud deseada, como muestra el Esquema 4.



Esquema 4. Entrenamiento de SentenceTransformers para el cálculo de similitud entre oraciones

En cuanto a la aproximación que se realiza para el cálculo de las agrupaciones de noticias, es similar al caso de spaCy. En este caso se parte de un modelo SentenceTransformer, en concreto distiluse-base-multilingual-cased-v2, ya utilizado en tareas anteriores. Se ha seleccionado este modelo ya que ofrecía mejor rendimiento que el resto. Para su implementación, se carga el modelo y se codifica para pasarlo a un *embedding* el cual posteriormente se compara con el resto para así poder asignarle el valor de la similitud coseno.

De la misma forma que el modelo anterior, se obtiene una matriz con la relación entre documentos y su similitud coseno, se selecciona el umbral optimo, las noticias similares y, finalmente, calcular las métricas.

Capítulo 6

Resultados

6.1. Métrica

Una de las cosas más importantes a tener en cuenta a la hora de desarrollar un proyecto de inteligencia artificial es la elección de la métrica a la hora de evaluar los resultados. Esto resulta primordial debido a que una métrica no significativa obtendrá resultados que no se adecuaran al esperado, ya que no se mide los parámetros correctos.

En este caso, de agrupación de noticias, o traduciendo a una tarea más genérica de *cluster* las métricas usuales suelen ser coeficiente de Silhouette, índice de Calinski-Harabasz, índice de Davies-Bouldin, entre otras. El primero mide la similitud de un objeto con su propio *cluster* en comparación con otros *clusters*. El segundo mide la relación entre la varianza dentro de los *clusters* y la varianza entre los *clusters*. El tercero mide la similitud entre los *clusters* y la distancia entre ellos.

Sin embargo, no se ha podido hacer uso de ninguna de estas métricas ya que el formato de muestras que se dispone no ha llevado posible hacerlo. Debido a ello, y para disponer de una métrica con la cual evaluar el sistema se ha optado por realizar una modificación e implementación propia adaptada sobre la precisión, *recall* y F1 para este caso en concreto.

En el corpus de noticias, se dispone de una columna correspondiente con el grupo al que pertenece. Las noticias que comparten el mismo dígito, significa que corresponden al mismo grupo. Esto se ha traducido a una forma vectorial, como una lista de listas, donde la lista interior coincide con el *cluster* y sus elementos con las noticias. El Esquema 5 es un ejemplo de esta codificación.

$$[\ [0,1,2],[3,4,5,6],[7,8,9],[10,11,12,13,14,15,16],[17,18,19,20],[21,22,23,24],[25,26]\ ]$$

Esquema 5. Codificación de las noticias agrupadas en clustters

Con la codificación se procede al cálculo de los verdaderos positivos, falsos positivos y falsos negativos. Para ello se van recorriendo tanto la lista etiquetada como la lista predicha. Por cada noticia de la lista etiquetada se busca si se encuentra algún *cluster* donde se encuentre, en caso negativo, se incrementan los falsos negativos. Si ha encontrado un *cluster* que incluye la noticia, se recorren los elementos en dirección de ambas listas para poder incrementar los verdaderos positivos (si se encuentra en ambas listas) y los falsos positivos (si solo se encuentra en la lista predicha). Debido a que una noticia puede ser incluida en varios *clusters*, se añade unas listas extras para poder gestionar si la noticia repetida ha sido ya contabilizada de manera correcta y no repetir cálculos.

Finalmente, con estos valores calculados se obtiene la precisión y *recall* y, a partir de esto, el F1.

6.2. Validación

6.2.1. Tf-idf

La Tabla 12 muestra los resultados que se han obtenido en el corpus de validación junto con los datos que se han utilizado para aplicar tf-idf y el mejor umbral. De manera general todos los umbrales son pequeños, entre un 0.30 y 0.35 de similitud. A pesar de ello, es razonable, ya que la mayoría de texto que se tiene en esos datos es limitada, y rápidamente se hace notar la diferencia. En contra posición, véase lo que sucede cuando se añade el cuerpo de la noticia, el texto, este umbral crece pues la cantidad de palabras a discriminar es mucho mayor. En cuanto a la valoración del sistema, con una tokenización y lematización del texto se logra un 0.96 de F1, lo que se traduce en que únicamente ha fallado una noticia, que al tratarse del recall, es una noticia que no ha incluido en algún *cluster*. Respecto a la puntuación obtenida con la información que se ha extraído, *ner* y *keywords*, por separado ofrecen un valor bastante aceptable, pero muy alejado del anterior. No obstante, cuando se realiza la unión entre *ner* y *keywords* se consigue la excelencia, sin ningún tipo de fallo. Al añadir más palabras están adquieren más peso y es capaz de discriminar y seleccionar mejor las noticias similares. Estos son los resultados en el conjunto de validación, pero para los de test se

van a seleccionar las aproximaciones de *Headline+Text+Clean* y *Ner+Keywords* ya que una es perfecta y la otra únicamente ha cometido un error.

Tabla 12. Resultados en la tarea de agrupación de noticias con Tf-idf en validación

Data	Best threshold similarity	Precision	Recall	F1
Headline	0.350	**1**	0.48	0.65
Text	0.700	**1**	0.65	0.78
Headline + Text	0.700	**1**	0.70	0.82
Headline + Text + Clean	0.300	**1**	0.93	0.96
Ner	0.300	0.79	**1**	0.89
Keywords	0.350	0.95	0.69	0.80
Ner + Keywords	0.300	**1**	**1**	**1**

6.2.2. spaCy

Como se ha mencionado se hace uso del modelo mediano y los datos a analizar en las diferentes pruebas son las mismas que en el caso anterior. Con todo ello, se obtienen los resultados de la Tabla 13. En términos generales parece que el método en si ofrezca un resultado bastante aceptable, pero al analizar un poco el *recall* y la precisión, estas muestran que la atribución en los grupos la está errando mucho pues parece ser que únicamente está agrupando en un único grupo. Esto conlleva que suba el F1 ya que se identifican todas las noticias, pero no las agrupa correctamente. En cuanto a la selección de datos que ofrecen mejores resultados, se encuentra nuevamente la unión de titular y texto tokenizado y lematizado, como mejor modelo, y *ner* y su unión con las *keywords* como segundo, obteniendo la misma puntuación. En este segundo caso pareciese que ocurre lo mismo con los anteriores que añade todo a un único *cluster*. A pesar que el mejor modelo obtenido en validación es el tokenizado y lematizado se van a probar los otros en *test*, ya que puede ser que sea problema por la cantidad de datos y con el otro corpus logre funcionar mejor.

Tabla 13. Resultados en la tarea de agrupación de noticias con spaCy en validación

Data	Best threshold similarity	Precision	Recall	F1
Headline	0.385	0.53	**1**	0.69
Text	0.905	0.53	**1**	0.69
Headline + Text	0.900	0.53	**1**	0.69
Headline + Text + Clean	0.955	**0.75**	0.80	**0.77**
Ner	0.980	0.61	**1**	0.76
Keywords	0.995	0.73	0.73	0.73
Ner + Keywords	0.985	0.61	**1**	0.76

6.2.3. SentenceTransformers

La evaluación de esta aproximación es la de la Tabla 14. Aunque los resultados se encuentran bastantes similares, sí que se disponen de tres aproximaciones que ofrecen mejores resultados. Estos son el titular más texto, esto último, pero tokenizado y lematizado y, por último, el uso de las *keywords*. En este caso la variedad entre la precisión y *recall* hace ver que los resultados son mejores que en el caso anterior con spaCy, ya que indica que se estas creando más agrupaciones y por tanto separando las noticias de acuerdo a ello. Sin embargo, debe estar fallando en algunas noticias que las atribuye a *clusters* que no son correctos, y luego algunas noticias que no está incluyendo. Sin embargo, parece un acercamiento correcto y serán estos tres mencionados los que se enfrenten en el corpus de *test*.

Tabla 14. Resultados en la tarea de agrupacion de noticias con SentenceTransformers en validación

Data	Best threshold similarity	Precision	Recall	F1
Headline	0.500	0.77	0.83	0.80
Text	0.450	0.64	**1**	0.78
Headline + Text	0.600	0.86	0.93	0.89
Headline + Text + Clean	0.700	**1**	0.78	0.88
Ner	0.650	0.61	**1**	0.76
Keywords	0.700	0.87	0.93	**0.90**
Ner + Keywords	0.800	**1**	0.68	0.81

6.3. Test

La evaluación de la métrica obtenida respecto al conjunto de validación ha sido en algunos casos sorprendente. Esto es el caso, del uso de la vectorización tf-idf para lograr la perfección o el uso de *embeddings* mediante SentenceTransformers para obtener un buen resultado también. Sin embargo, estos sistemas deben ponerse a prueba con un conjunto de datos que no hayan visto nunca, y es por ello que se ha llevado contra el corpus de test.

Se han seleccionado un total de nueve modelos entre los tres métodos globales presentados para analizarlos en el conjunto de test. En cuanto a los datos a utilizar para hacer la agrupación lo que se ha repetido ha sido el uso de la noticia en su totalidad, es decir, la unión del titular y del texto, pero aplicándole un tokenizado, limpiado y lematizado. También ya sea por separado o junto, las columnas de *ner* y *keywords*.

Tabla 15. Resultados en la tarea de agrupación de noticias en test

Method	Data	Best threshold similarity	Precision	Recall	F1
Tf-idf	Headline + text + clean	0.300	**1**	**1**	**1**
	Ner + keywords	0.300	**1**	**1**	**1**
spaCy	Headline + text + clean	0.955	**1**	0.14	0.25
	Ner	0.980	0.55	0.85	0.67
	Keywords	0.995	**1**	0.60	0.75
	Ner + keywords	0.985	0.55	0.92	0.69
Sentence Transformers	Headline + text	0.600	**1**	0.85	0.92
	Headline + text + clean	0.700	**1**	**1**	**1**
	Keywords	0.700	**1**	0.92	0.96

En cuanto al análisis de los resultados de la Tabla 15 se aprecia como los métodos de spaCy no ofrecen un buen resultado, cosa que ya se vaticinaba con el conjunto de validación. Y que el resto de métodos ofrece un rendimiento excelente. Respecto al método de sk-learn obtiene la perfección de 1 F1 en ambas aproximaciones, las cuales selecciona a la perfección todas las noticias y consigue agruparlas en sus respectivos grupos. Sobre los resultados de SentenceTransformers su mejor modelo con los datos limpios consigue la perfección en comparación con introducir los datos sin manipular que, por su valor parece ser que no ha acertado en dos noticias. Esto hace ver la importancia que ha tenido la eliminación de *stopwords* y diversos elementos, así como su lematización antes de introducirlo en este modelo que calculara su *embedding*. Su otro modelo introduciendo las *keywords* también obtiene un resultado ejemplar, donde solo había errado en una noticia al no incluirla en el *cluster*.

Comparando entre seleccionar NER o las palabras clave, pudiera ser que esta última tenga más relevancia a la hora de agrupar las noticias. Esto es debido a que como propia palabra clave puede haberse introducido alguna entidad tal como un

nombre de organización, persona o algún lugar. Eso añadido a los sustantivos representativos de la noticia puede hacer más relevante la extracción de palabras clave que NER.

Las siguientes ilustraciones muestran los resultados que se han obtenido a modo de ejemplo real y visual de cómo se han agrupado las noticias. En la Ilustración 2 se puede ver lo comentado sobre como spaCy está agrupando todas las noticias en un único grupo, y que, a pesar, de obtener un 0.69 de F1 no debe tenerse en consideración. Sin embargo, y a pesar que supera en cinco puntos el F1, cuando se usa spaCy únicamente con *keywords* ya consigue separar los grupos mejor, Ilustración 1

```
threshold: 0.995
cluster 0
 - Noticia 22: 'Descubren el animal más pesado que ha existido nunca en la Tierra: una ballena de 340 toneladas'
 - Noticia 24: 'El animal más pesado que ha existido' de la sección 'Ciencias' con url 'https://www.publico.es/ci
cluster 1
 - Noticia 0: 'EEUU oculta su programa de ovnis, según un ex funcionario de Inteligencia de la Fuerza Aérea' de l
 - Noticia 3: '"Los avistamientos no son raros ni aislados": el Congreso de EE UU se toma en serio los ovnis' de
cluster 2
 - Noticia 10: 'El empleo crece en 21.900 personas en julio pese a la destrucción de 110.700 puestos en la educac
 - Noticia 12: 'España gana 22.000 afiliados a la Seguridad Social en julio, aunque el ritmo de creación de emple
 - Noticia 13: 'El paro baja en 11.000 personas en julio hasta un nuevo mínimo desde 2008' de la sección 'Economí
cluster 3
 - Noticia 15: 'VOX MURCIA ACUSA AL PP DE QUERER LA REPETICIÓN DE ELECCIONES AL NO DARLES ENTRADA EN EL GOBIERNO'
 - Noticia 16: 'Vox insiste en la vicepresidencia de Murcia y el PP le acusa de mantener pulsado "el botón de la
cluster 4
 - Noticia 18: 'El Festival de Cine Europeo de Sevilla se aplaza hasta 2024' de la sección 'nan' con url 'https:/
 - Noticia 19: 'El PP cancela el Festival de Cine de Sevilla para dar el protagonismo a los Grammy' de la sección
 - Noticia 20: 'El Ayuntamiento de Sevilla retrasa el Festival de Cine a primavera por ser "incompatible" con los
cluster 5
 - Noticia 4: 'SIMEONE MANTIENE EL PULSO A JOAO FÉLIX: "NADIE ESTÁ POR ENCIMA DEL ATLÉTICO"' de la sección 'depor
 - Noticia 5: 'Simeone le pone las cosas claras a Joao Félix: "No hay nadie, pero nadie, por encima del club"' de
 - Noticia 6: 'Palo de Simeone a Joao Félix: "No hay nadie, pero nadie, por encima del Atlético"' de la sección '
cluster 6
 - Noticia 7: 'Hallan los restos del legendario teatro de Nerón durante la excavación de un parking en Roma' de l
 - Noticia 8: 'Roma desentierra a las puertas del Vaticano el legendario teatro de Nerón' de la sección 'Cultura'
 - Noticia 9: 'Los arqueólogos creen haber hallado en Roma el teatro donde ensayaba el emperador Nerón' de la sec
cluster 7
 - Noticia 21: 'Una antigua y colosal ballena desafía el título de "animal más pesado" de la historia' de la secc
 - Noticia 23: 'Hallados los restos de un antiguo y colosal animal que destrona a la ballena azul como el "más pe
Precisión: 1.00
Recall: 0.60
Puntuación F1: 0.75
```

Ilustración 1. Resultado de la agrupación de spaCy con keywords en test

```
threshold: 0.985
cluster 0
 - Noticia 0: 'EEUU oculta su programa de ovnis, según un ex funcionario de Inteligencia de la Fuerza Aérea' de l
 - Noticia 1: 'UN EX OFICIAL DE EE UU ASEGURA EN EL CONGRESO QUE SU PAÍS POSEE NAVES ESPACIALES NO HUMANAS' de la
 - Noticia 2: 'El Congreso de EEUU pide al Gobierno información sobre OVNIS tras escuchar a testigos militares' d
 - Noticia 3: '"Los avistamientos no son raros ni aislados": el Congreso de EE UU se toma en serio los ovnis' de
 - Noticia 4: 'SIMEONE MANTIENE EL PULSO A JOAO FÉLIX: "NADIE ESTÁ POR ENCIMA DEL ATLÉTICO"' de la sección 'depor
 - Noticia 5: 'Simeone le pone las cosas claras a Joao Félix: "No hay nadie, pero nadie, por encima del club"' de
 - Noticia 6: 'Palo de Simeone a Joao Félix: "No hay nadie, pero nadie, por encima del Atlético"' de la sección
 - Noticia 7: 'Hallan los restos del legendario teatro de Nerón durante la excavación de un parking en Roma' de l
 - Noticia 8: 'Roma desentierra a las puertas del Vaticano el legendario teatro de Nerón' de la sección 'Cultura
 - Noticia 9: 'Los arqueólogos creen haber hallado en Roma el teatro donde ensayaba el emperador Nerón' de la sec
 - Noticia 11: 'EL PARO BAJA EN JULIO EN CASI 11.000 PERSONAS POR EL EMPUJE DEL SECTOR SERVICIOS EN VERANO' de la
 - Noticia 12: 'España gana 22.000 afiliados a la Seguridad Social en julio, aunque el ritmo de creación de emple
 - Noticia 13: 'El paro baja en 11.000 personas en julio hasta un nuevo mínimo desde 2008' de la sección 'Economí
 - Noticia 14: 'El paro cae en 10.968 personas en julio y alcanza su cifra más baja desde 2008' de la sección 'Ec
 - Noticia 15: 'VOX MURCIA ACUSA AL PP DE QUERER LA REPETICIÓN DE ELECCIONES AL NO DARLES ENTRADA EN EL GOBIERNO
 - Noticia 16: 'Vox insiste en la vicepresidencia de Murcia y el PP le acusa de mantener pulsado "el botón de la
 - Noticia 17: 'Sevilla aplaza su Festival de Cine para que no coincida con la entrega de los Grammy Latinos' de
 - Noticia 18: 'El Festival de Cine Europeo de Sevilla se aplaza hasta 2024' de la sección 'nan' con url 'https:/
 - Noticia 19: 'El PP cancela el Festival de Cine de Sevilla para dar el protagonismo a los Grammy' de la sección
 - Noticia 20: 'El Ayuntamiento de Sevilla retrasa el Festival de Cine a primavera por ser "incompatible" con los
 - Noticia 21: 'Una antigua y colosal ballena desafía el título de "animal más pesado" de la historia' de la sec
 - Noticia 22: 'Descubren el animal más pesado que ha existido nunca en la Tierra: una ballena de 340 toneladas'
 - Noticia 23: 'Hallados los restos de un antiguo y colosal animal que destrona a la ballena azul como el "más pe
 - Noticia 24: 'El animal más pesado que ha existido' de la sección 'Ciencias' con url 'https://www.publico.es/ci
Precisión: 0.55
Recall: 0.92
Puntuación F1: 0.69
```

Ilustración 2. Resultado de la agrupación de spaCy con ner+keywords en test

```
threshold: 0.700
cluster 0
 - Noticia 10: 'El empleo crece en 21.900 personas en julio pese a la destrucción de 110.700 puestos en la educación'
 - Noticia 12: 'España gana 22.000 afiliados a la Seguridad Social en julio, aunque el ritmo de creación de empleo se
 - Noticia 13: 'El paro baja en 11.000 personas en julio hasta un nuevo mínimo desde 2008' de la sección 'Economía' d
 - Noticia 14: 'El paro cae en 10.968 personas en julio y alcanza su cifra más baja desde 2008' de la sección 'Econom
cluster 1
 - Noticia 15: 'VOX MURCIA ACUSA AL PP DE QUERER LA REPETICIÓN DE ELECCIONES AL NO DARLES ENTRADA EN EL GOBIERNO' de
 - Noticia 16: 'Vox insiste en la vicepresidencia de Murcia y el PP le acusa de mantener pulsado "el botón de la repe
cluster 2
 - Noticia 0: 'EEUU oculta su programa de ovnis, según un ex funcionario de Inteligencia de la Fuerza Aérea' de la se
 - Noticia 1: 'UN EX OFICIAL DE EE UU ASEGURA EN EL CONGRESO QUE SU PAÍS POSEE NAVES ESPACIALES NO HUMANAS' de la sec
 - Noticia 2: 'El Congreso de EEUU pide al Gobierno información sobre OVNIS tras escuchar a testigos militares' de la
 - Noticia 3: '"Los avistamientos no son raros ni aislados": el Congreso de EE UU se toma en serio los ovnis' de la s
cluster 3
 - Noticia 17: 'Sevilla aplaza su Festival de Cine para que no coincida con la entrega de los Grammy Latinos' de la s
 - Noticia 18: 'El Festival de Cine Europeo de Sevilla se aplaza hasta 2024' de la sección 'nan' con url 'https://www
 - Noticia 19: 'El PP cancela el Festival de Cine de Sevilla para dar el protagonismo a los Grammy' de la sección 'Es
 - Noticia 20: 'El Ayuntamiento de Sevilla retrasa el Festival de Cine a primavera por ser "incompatible" con los Gra
cluster 4
 - Noticia 4: 'SIMEONE MANTIENE EL PULSO A JOAO FÉLIX: "NADIE ESTÁ POR ENCIMA DEL ATLÉTICO"' de la sección 'deportes'
 - Noticia 5: 'Simeone le pone las cosas claras a Joao Félix: "No hay nadie, pero nadie, por encima del club"' de la
 - Noticia 6: 'Palo de Simeone a Joao Félix: "No hay nadie, pero nadie, por encima del Atlético"' de la sección 'Depo
cluster 5
 - Noticia 7: 'Hallan los restos del legendario teatro de Nerón durante la excavación de un parking en Roma' de la se
 - Noticia 8: 'Roma desentierra a las puertas del Vaticano el legendario teatro de Nerón' de la sección 'Cultura' cor
 - Noticia 9: 'Los arqueólogos creen haber hallado en Roma el teatro donde ensayaba el emperador Nerón' de la sección
cluster 6
 - Noticia 21: 'Una antigua y colosal ballena desafía el título de "animal más pesado" de la historia' de la sección
 - Noticia 22: 'Descubren el animal más pesado que ha existido nunca en la Tierra: una ballena de 340 toneladas' de l
 - Noticia 23: 'Hallados los restos de un antiguo y colosal animal que destrona a la ballena azul como el "más pesado
 - Noticia 24: 'El animal más pesado que ha existido' de la sección 'Ciencias' con url 'https://www.publico.es/cienci
Precisión: 1.00
Recall: 0.92
Puntuación F1: 0.96
```

Ilustración 3. Resultado de la agrupación de SentenceTransformers con keywords en test

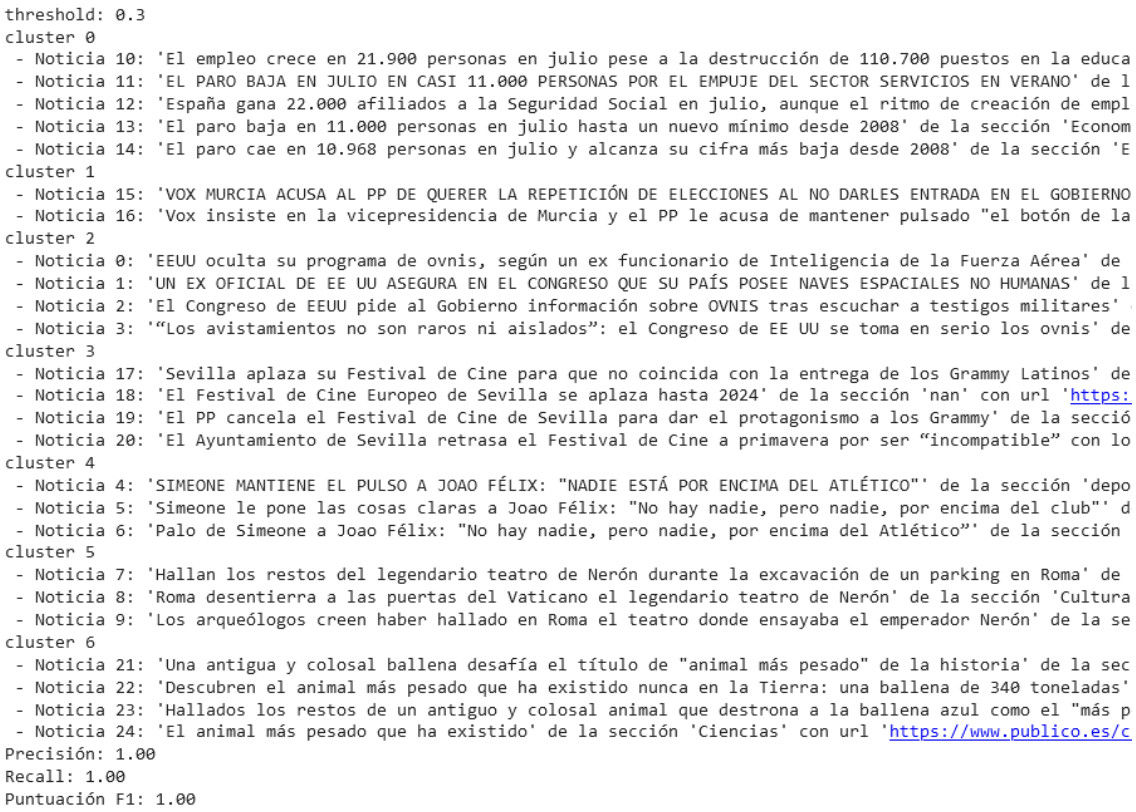


Ilustración 4. Resultado de la agrupación de Tf-idf con ner+keywords en test

Las dos últimas ilustraciones son las que obtienen resultados más que aceptables, donde la Ilustración 3 es SentenceTransformer con *keywords* que, si bien no es el mejor modelo, se comporta realmente bien, puesto que únicamente no incluye la noticia número 11 en las agrupaciones. Y, finalmente, la Ilustración 4 que mediante el uso de tf-idf se obtiene la agrupación perfecta de las noticias del corpus de test.

Capítulo 7

Conclusiones

Los objetivos que se decretaron en el apartado inicial han llegado a cumplirse si bien teniendo en cuenta algunas consideraciones. El primero de ellos era poder extraer noticias de al menos cinco medios digitales de noticias. Este ha sido resuelto dado que además se han podido extraer de un total de ocho. Gracias al uso de la librería Selenium con Python se ha conseguido realizar un *scraper* de la web donde al introducir un rango de fechas se pueden recuperar las noticias comprendidas entre esos días, ambos inclusive. A pesar de ello, cuenta con la propia limitación de las navegaciones de las webs y si las fechas son muy lejanas, algunas de estas no permiten el acceso a esas noticias.

El segundo era la creación de un corpus para la agrupación de noticias digitales. Debido a que no se encontraron corpus dedicados a esta tarea para el español y que el sistema a desarrollar parte con la extracción de noticias se creó el corpus aprovechando esta herramienta. A partir del *scraper* realizado se extrajeron una cantidad de noticias de diferentes días en un rango distinto, se procesaron y limpiaron y, posteriormente, se realizó una anotación manual de las muestras. Esta anotación era necesaria para poder llegar a disponer de datos etiquetados y poder establecer una comparación con los modelos obtenidos. En total se etiquetaron 52 noticias, agrupadas en 14 *clusters* y se extrajeron como datos extra las entidades nombradas, las palabras clave y la agrupación de cada noticia al grupo correspondiente.

El tercer objetivo era conseguir extraer más información de las noticias. Con esto ya en mente, fue la decisión por la cual a la hora de anotar el corpus se decide anotar también las entidades nombradas y palabras clave. Asimismo, esto permite presentar y comparar una serie de modelos que a partir de los datos del titular y/o texto de las noticias sea capaz de extraer estas nuevas características y así aportar más información a la hora de realizar el agrupamiento.

Si bien podría pensarse que la extracción es simple y no aportaría nada nuevo, no ha sido así. Gracias a ello se ha podido presentar más aproximaciones a la hora de realizar las agrupaciones de noticias y tener más variedad para obtener el mejor resultado posible. El hecho de haber invertido un tiempo en la selección de estos modelos y decidir hacer una comparación y selección para obtener estos datos ha conseguido que en la tarea final de agrupación varios de los mejores modelos sean utilizando los datos obtenidos mediante estas técnicas. Este hecho muestra la importancia de realizar un buen análisis previo a la tarea para decidir como manipular los datos, si se puede extraer más información y saber cómo utilizarla. En este contexto se ha querido profundizar en la selección del mejor modelo para estas tareas secundarias en vez de obtener el primero de ellos genéricos, como pudiera ser, mediante spaCy, con el objetivo de obtener aquel que se adecúe más a la tarea con los datos relacionados de las noticias.

Con ello, y teniendo en cuenta que la anotación no ha sido realizada por un experto, se ha logrado un resultado bastante aceptable a sabiendas que no se ha realizado ningún tipo de *fine-tuning* sobre los modelos preentrenados. No obstante, y quizá por estas razones, el mejor modelo para el reconocimiento de las entidades nombradas no ha sido el mediano de spaCy, que tan buen resultado muestra en su ficha. En cambio, el mejor modelo obtenido para esta tarea ha sido Babelscape/wikineural-multilingual-ner, un modelo basado en BERT que presenta un *fine-tuning* sobre la tarea de NER en el corpus WikiNEuRal y multilenguaje. Este modelo ha alcanzado un 0.56 de F1 frente al segundo mejor modelo, el mediano de spaCy, con un 0.49 de F1, ambos en el corpus de test.

Sucede de manera similar con la tarea secundaria de extracción de palabras clave. De nuevo, se ha topado con varias aproximaciones algunas basadas en la parte más clásica con extracción mediante n-gramas y su conteo y otras mediante el uso de *embeddings* gracias a los Transformers. Con ello se deduce que en el caso de uso de n-gramas lo que mejor funciona es el uso de unigramas y, a lo sumo, bigramas pero en ocasiones contadas, pues aunque puede darse el caso, no es habitual tener palabras compuestas como clave. Aunque el uso de los Transformers y sus prestaciones se encuentran en el estado del arte, no se ha logrado combinarlo de manera óptima para obtener un buen rendimiento. No obstante, el modelo de spaCy ya preentrenado sí que ha sido capaz de recuperar las mejores palabras clave y, por tanto, dar como completada la tarea, gracias a la obtención de las palabras mediante su etiqueta gramatical, *post-tag*, y su conteo posterior. Este modelo se ha declarado como ganador

tanto en el corpus de validación como en el test, superando al segundo modelo por más de diez y cinco puntos respectivamente.

Los siguientes y, últimos objetivos consistían en la utilización de modelos preentrenados para conseguir agrupar las noticias según su similitud y poder compararlos para así logran conocer cuál es el mejor obtenido. Esto se ha llevado a cabo utilizando tanto técnicas más clásicas del procesamiento del lenguaje natural como el estado del arte basado en Transformers. Además, para la evaluación de los sistemas y poder compararlos se ha utilizado la métrica F1 la cual se ha implementado un algoritmo para poder calcular la precisión y *recall* ya que el formato del cual se disponía no era compatible con las métricas utilizadas en las tareas de *clsutering*.

La primera aproximación para la tarea de agrupación es la vectorización tf-idf para vectorizar las palabras según su importancia en la noticia y, posteriormente, con la similitud coseno poder seleccionar aquellas noticias iguales. Esta técnica tan básica, sencilla y rápida es la que mejor resultado ha obtenido. La cual, en términos analíticos, puede llegar a ser lógico pues se basa en la importancia relacionada con la frecuencia de la palabra en la propia noticia y en todas ellas. Este concepto es la clave del estudio para establecer esa relación semántica y poder tener una representación de la noticia para establecer los grupos y, de hecho, se repite cuando se usan *embeddings* igual.

A pesar que esta aproximación ya parecía resolver el problema se ha comparado con otros sistemas. Específicamente la manera de transformar la noticia en un vector, ya sea pasarlo a una matriz o un *embedding* mediante Transformers para poder realizar las operaciones matemáticas de distancia entre vectores y entre significados. El modelo de spaCy no ha conseguido un buen resultado con ello ya que no es capaz de asignar un valor de similitud entre noticias lo suficientemente significativo para conseguir separarlas. Únicamente conseguía separar en un par de grupos o agruparlo todo como la misma noticia. No obstante, el ultimo método basado en SentenceTransformers vuelve a obtener resultados inmejorables. Esta librería cuenta con modelos ya preentrenados especializados en la tarea de similitud semántica entre palabras y frases. Por esta razón cuando, se codifica el texto y se comparan entre ellos resulta tan efectivo para poder decidir que noticias están hablando sobre el mismo tema o acontecimiento.

Los mejores modelos que se han obtenido han sido son tf-idf aplicado sobre los datos de *headline+text+clean* y *ner+keywords*, logrando un 1 de F1. A su vez, SenteceTransformer con los datos de *headline+text+clean* y *keywords*, alcanzan un 1 y 0.96 de F1, respectivamente. Este segundo modelo únicamente obtiene un error en una de las noticias al no incluirla en el *cluster*.

Para concluir, comentar que el sistema que se ha conseguido desarrollar resulta de un buen punto de partida para continuar su expansión. El extractor de noticias es capaz de obtenerlas en rangos de fecha y, además, en el caso que se requiera iniciar sesión, es capaz de hacerlo, así como interactuar por toda la página incluso con peticiones y acciones javascript. Esta parte podría tratarse como finalizada pero la parte de aprendizaje y de agrupamiento es la que necesita un poco más de profundidad. La razón son las pocas noticias etiquetadas de las cuales se disponía y que se ha tenido que realizar de manera manual lo cual ha llevado mucho tiempo y no ha sido posible disponer de una cantidad de datos tan grande como se habría querido. A pesar de ese inconveniente, la tarea de agrupamiento de noticias ha conseguido realizarse mediante modelos preentrenados, seleccionando aquellos que pudieran tener más relación e incumbencia con la similitud semántica entre documentos y, obteniendo un resultado muy satisfactorio para este corpus.

A nivel particular, me he encontrado con un reto de proyecto, ya que al realizar un estudio sobre lo que había previamente relacionado con este tema o algo similar sobre este tipo de agrupamiento era muy complejo encontrar algo. Todo ello ha llevado a una investigación más específica y poder modificar y aplicar el resto de algoritmos a esta tarea para poder resolverlo. Sin duda la parte más engorrosa ha sido la anotación manual de todas las noticias, pero una vez realizada ha resultado de gran ayuda para el objetivo final. También comentar que el hecho de hacer la extracción de entidades y palabras clave, aunque a priori pareciese que no sea relevante y que con el texto pudiera valer, ha resultado de gran ayuda en los modelos, pues gracias a estos datos añadidos se ha podido obtener modelos que también consiguen resultados inmejorables.

Capítulo 8

Trabajo futuro

Como ya se ha comentado anteriormente en las Conclusiones, este proyecto ha finalizado como una buena base de partida para poder implementar un sistema completo de extracción y agrupamiento de noticias de principio a fin. Por ello, los siguientes pasos a dar para alcanzar ese producto final pudieran ser los siguientes.

A nivel de extracción de noticias de los medios digitales, realizar un estudio más profundo de las páginas webs para poder llegar a cualquier noticia independientemente del día seleccionado y añadir todos los periódicos que fueran posibles. Para la extracción quizá resultase conveniente también realizar una comparativa con otros sistemas como MyNews [22] o Prensa-e [23] de la Universitat de Valencia. Ambas plataformas disponen de cientos de miles de noticias de cientos de periódicos con la posibilidad de elegir y buscar aquellas que se deseen. El inconveniente es que son plataformas de pago.

En cuanto al apartado de aprendizaje automático y agrupamiento de noticias son varias las tareas que se pueden abordar en un futuro. La primera de ellas es conseguir un conjunto de datos mayor y etiquetado por un experto, el cual además de anotar la información ya conocida sea capaz de extraer conceptos más semánticos y con más significados. Además, incluir noticias que no se encuentren en ningún *cluster* o que solo lo formen ellas, de este modo es la manera de introducir ruido en el corpus. Información adicional que se podría extraer sería el nivel de objetividad o subjetividad de la noticia, la opinión que esa noticia está realizando sobre el suceso, la veracidad de los sucesos que se cuentan, la formulación del titular o un pequeño resumen. Otra de las cosas a nivel de modelo de datos a probar sería un modelo de extracción de resúmenes para la

[22] https://mynews.es/
[23] https://www.uv.es/uvweb/uv-noticias/es/sala-prensa/uv-prensa/acceso-prensa-e-1286017453682.html

noticia y ponerla a evaluación tanto a ella sola como en combinación con las palabras clave, que tan buen resultado han dado. Con estos modelos se tendrían otras aproximaciones y quizá podría obtenerse resultados considerables.

En relación a la extracción de noticias del corpus, si eso se llevase a cabo, puede derivar en la realización de un sistema que una vez haya conseguido agrupar las noticias consiga hacer un análisis en profundidad entre ellas. Por ejemplo, podría ofrecer un resumen a partir de todas las noticias, lo cual podría ser lo más objetivo posible de acuerdo al grado de objetividad/subjetividad que se tenga. Otra función, sería la posible contraposición en argumentos de las noticias entre sí o también indicar cuál de ellas está utilizando un titular más jugoso para una noticia que no llega a tratar de ello.

Las características anteriores para este tipo de sistemas sería el objetivo final a realizar ya que comportaría un análisis completo y extracción de significado avanzado de manera automática, rápida y útil. A pesar de ello, si no se dispone de una plataforma para que se pueda visualizar o utilizar fácilmente carece que uso. Por ello, no cabe olvidar también la realización de una plataforma web o aplicación que permita mediante una interfaz poder acceder a este sistema. Esta se basaría en un filtrado de rangos de fecha y un selector de periódicos para indicar de cuándo y dónde se quiere extraer las noticias y el sistema internamente sea capaz de recuperar las noticias, bien de la web o de su base de datos. Una vez con las noticias poder agruparlas y aplicar las funciones de análisis anteriores para mostrarle al usuario todo ello.

Referencias

[1] D. E. Rumelhart y G. E. W. R. J. Hinton, «Learning internal representations by error propagation,» San Diego, California, 1985.

[2] S. Hochreiter y J. Schmidhuber, «Long short-term memory,» de *Neural computation*, 1997, pp. 1735-1780.

[3] J. Chung, C. Gulcehre, K. Cho y Y. Bengio, «Empirical evaluation of gated recurrent neural networks on sequence modeling,» de *NIPS 2014 Deep Learning and Representation Learning Workshop*, 2014.

[4] Vaswani, «All You Need is Attention,» 2017.

[5] J. Devlin, M. W. Chang, K. Lee y K. Toutanova, «Bert: Pre-training of deep bidirectional transformers for language understanding,» 2018.

[6] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li y P. J. Liu, «Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer,» 2019.

[7] T. B. Brown, «Language Models are Few-Shot Learners,» 2020.

[8] A. Gutiérrez-Fandiño, «MarIA: Spanish Language Models,» Sociedad Española para el Procesamiento del Lenguaje Natural, 2021.

[9] Devlin, «mBERT,» 2018.

[10] J. Cañete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang y J. Pérez, «Spanish Pre-Trained BERT Model and Evaluation Data,» de *PML4DC at ICLR 2020*, 2020.

[11] A. Conneau y D. Kiela, «SentEval: An Evaluation Toolkit for Universal Sentence Representations,» 2018.

[12] H. Jiang, P. He, W. Chen, X. Liu, J. Gao y T. Zhao, «SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization,» 2020.

[13] N. Reimers y I. Gurevych, «Sentence-bert: Sentence embeddings using siamese bert-networks,» 2019.

[14] J. MacQueen, «Some methods for classification and analysis of multivariate observations,» de *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1967.

[15] A. P. Dempster, N. M. Laird y D. B. Rubin, «Maximum likelihood from incomplete data via the EM algorithm,» *Journal of the Royal Statistical Society,* pp. 1-38, 1977.

[16] B. J. Frey y D. Dueck, «Clustering by passing messages between data points. science,» 2007.

[17] dezzai, «Hugging Face,» [En línea]. Available: https://huggingface.co/MMG/xlm-roberta-large-ner-spanish.

[18] E. F. Tjong Kim Sang, «Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition,» de *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[19] N. Garagiola, «Hugging Face,» [En línea]. Available: https://huggingface.co/NazaGara/NER-fine-tuned-BETO.

[20] S. Tedeschi, V. Maiorca, N. Campolungo, F. Cecconi y R. Navigli, «WikiNEuRal: Combined Neural and Knowledge-based Silver Data Creation for Multilingual NER,» de *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic, Association for Computational Linguistics, 2021, pp. 2521-2533.

[21] J. Cañete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang y J. Pérez, «Spanish Pre-Trained BERT Model and Evaluation Data,» de *PML4DC at ICLR 2020*, 2020.

[22] D. Adelani, «Hugging Face,» [En línea]. Available: https://huggingface.co/Davlan/bert-base-multilingual-cased-ner-hrl.

[23] R. Campos, V. Mangaravite, A. Pasquali, A. Jatowt, A. Jorge, C. Nunes y A. Jatowt, «YAKE! Keyword Extraction from Single Documents using Multiple Local Features,» de *Information Sciences Journal*, Elsevier, 2020.

[24] B. DeWilde, «textacy: NLP, before and after spaCy,» [En línea]. Available: https://textacy.readthedocs.io/en/latest/index.html.

[25] N. Reimers y I. Gurevych, «Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,» de *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2019.