UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Estudio y predicción de la evolución e impacto de tweets en trending topics

Alfonso Sebares Mecha
Tutor: Lara Quijano Sánchez
Ponente: Iván Cantador Gutiérrez

Julio 2020

Estudio de la evolución e impacto de los usuarios sobre Trending Topics

AUTOR: Alfonso Sebares Mecha
TUTOR: Lara Quijano Sánchez

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid

Julio de 2020

Resumen (castellano)

Este Trabajo de Fin de Grado consiste en el desarrollo de un sistema de predicción para determinar la probabilidad de impacto de un tweet sobre una tendencia o *trending topic* determinada en la red social de Twitter.

La motivación del proyecto viene dada por la enorme presencia que ha tenido y tiene esta red social a la hora de reflejar las ideas y sentimientos de sus usuarios ante grandes acontecimientos del mundo casi en tiempo real. El hecho de que se trate de un formato puramente en texto hace que dichos sucesos cristalicen en forma de topics o *trends* sobre los que los usuarios expresan sus ideas y compitan de forma indirecta por expresar en pocos caracteres sus ideas con el mayor impacto posible. Esto es especialmente atractivo para los profesionales de las campañas de marketing, campañas políticas o incluso otros investigadores interesados en establecer patrones del comportamiento humano.

Para ello se ha llevado a cabo un proceso de investigación previo en el que se ha recopilado toda la información respecto a dicho algoritmo de popularidad de Twitter, sus revisiones en el tiempo y otros estudios académicos sobre el tema en cuestión, con el objetivo de sacar toda la información posible antes de intentar arrojar algo de luz sobre esta tecnología.

También ha sido necesario el desarrollo de una herramienta capaz de recopilar, procesar y guardar un set de datos en tiempo real directamente desde la fuente sobre el que poder aplicar nuestro análisis.

Finalmente, se ha desarrollado un modelo estadístico utilizando aprendizaje automático capaz de predecir el impacto de un nuevo tweet en un *trending topic* por las características que posee, habiendo hecho previamente un análisis de qué atributos so interesantes a la hora de realizar esta clasificación.

Abstract (English)

This Bachelor Thesis consists in the development of a system capable of predicting the probability of impact of a tweet in a given trend or *trending topic* in the Twitter social network.

Twitter has been historically the most prevalent network when it comes to reflecting in real time the ideas and feelings of its users about significant developments in the world. The fact that the information lives in pure text makes it so that this information is labeled in topics or *trends*, where users can make contributions and where its engagement with other users is proportional to the impact they cause. This is especially attractive for professionals from the marketing sector, politic campaigns or other academic researches interested in developing patterns of human behavior.

To accomplish this task, a preliminary research work has been carried out with the aim of synthesizing in the shape of rules all the available information about the popularity algorithm of Twitter, its revisions over time and other academic work about the matter.

This has also required the development of a tool capable of collecting, processing and storing a dataset in real time directly working on the source of information, which will be used as an object of study in this work.

Finally, a statistical model has been developed using machine learning techniques capable of predicting the impact of a given tweet in a given trending topic based on its features, with a previous analysis of what attributes are interesting to accomplish this task.

Palabras clave (castellano)

PLN, Aprendizaje automático, Minería de datos, RRSS, Red Neuronal, API REST

Keywords (inglés)

NLP, Machine Learning, Data Mining, Social Networks, Neural Network, API REST

Agradecimientos

Me gustaría expresar mi agradecimiento a todas las personas que me han ayudado y apoyado en la realización de este TFG.

En particular, me gustaría agradecer a Lara Quijano, mi tutora, su tiempo y su implicación en el trabajo durante todo su desarrollo.

A todos mis amigos que me han apoyado. A Óscar R. por los ratos de estos meses y todos los años de la carrera, gracias.

A Lourdes por tu apoyo diario.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

Las redes sociales se han convertido en una de las principales herramientas utilizadas por los usuarios para expresar su opinión, obtener información sobre cierta temática, compartir sus vivencias e incluso, como medio mediante el cual contactar con empresas o potenciales clientes.

De acuerdo con Smart Insights [1] en su informe sobre el resumen digital global de 2020 (*Digital 2020 Global overview*), en el mes de enero existían 4.54 billones de usuarios en Internet, de los cuales 3.8 billones contarían con algún perfil en una red social. Comparando estas cifras con el mismo periodo en el año anterior, se observa un aumento de un 7.0% de los usuarios de Internet y un 9.2% en la cifra de usuarios de redes sociales.

Twitter se configura como una de las principales redes sociales, y a pesar de que su popularidad ha ido variando con los años, dejando a plataformas como Instagram o Tik Tok posicionarse en las primeras posiciones en las listas de redes sociales, cada año el número de usuarios de Twitter sigue aumentando.

De acuerdo con su informe para los accionistas publicado el 30 de abril de 2020 [2], en el primer cuatrimestre de este año contaría con 166 millones de usuarios activos, aumentando en un 24% con respecto al año anterior.

Estas características, sumado a la naturaleza pública y abierta de su contenido, hacen que esta red social sea especialmente atractiva como objeto de estudio tanto con fines económicos como con académicos. Se han realizado múltiples estudios desde casi su concepción, enfocados en el *sentiment análisis* [4][5][6][7], detección de campañas de bots [3], detección de *fake news* [8][16][17] y medidores de odio [18] o tweets ofensivos [19] y clasificación de tendencias [11][13][14] como principales familias de estos trabajos.

No obstante, ningún estudio existente se ha centrado en entender cómo moldean los usuarios individualmente las tendencias populares en un momento dado de esta red social. Los usuarios deciden qué conversaciones son relevantes en todo momento y contribuyen a estas conversaciones con más o menos impacto respecto a otros usuarios.

El analizar, medir y predecir el impacto que tienen los usuarios sobre estas tendencias es la principal motivación de este trabajo, al resultar un aspecto muy interesante de esta red social y que ha sido poco explorado. La red social motiva y recompensa a los usuarios a escribir *tweets* con impacto que generen interacciones o *engagement* con otros usuarios. Cuanto más impacto tenga un *tweet* determinado, más visibilidad e influencia tendrá sobre un tema de conversación. La información resultante de este trabajo puede ayudar a los usuarios a optimizar esto.

Otro caso de uso que puede ser más interesante todavía es combinar los resultados de este trabajo con estudios existentes o futuros. De manera que podríamos combinar la clasificación de impacto de un *tweet* con la detección de *fake news* o delitos de odio por ejemplo para estudiar aún más a fondo la naturaleza de este tipo de contenido en la red social.

La segunda de las motivaciones a la hora de realizar este trabajo es construir una herramienta que no existía capaz de recopilar, almacenar, normalizar y entrenar en tiempo real un enorme volumen de información que la red social pone disponible a manos de los investigadores y puede ser una base sólida para futuros estudios académicos.

Para llevar a cabo estas tareas se han estudiado técnicas de predicción usando machine learning (ML), de extracción de datos usando data mining (DM) y de análisis del contexto y del lenguaje usando natural language processing (NLP), aplicado a su vez a grandes volúmenes de información.

1.2 Objetivos

El objetivo de este trabajo es la consecución de un modelo efectivo y novedoso que permita la extracción de los elementos relevantes de un conjunto de información muy grande (en este caso, los tweets) para poder establecer patrones que ayuden a prever qué características debe tener un tweet para que tenga impacto en una tendencia.

Para la consecución de este objetivo será necesario el cumplimiento de dos fases: la investigación y el desarrollo.

En cuanto a la investigación, desde casi los comienzos de Twitter se han realizado estudios analizando distintos aspectos de la red social. También se han analizado los datasets de Twitter existentes, su viabilidad como fuente de datos para llevar a cabo este trabajo y el razonamiento de por qué finalmente no ha sido posible reutilizarlos. En esta fase se han analizado los más relevantes en cuanto a logros y se detallan en la siguiente sección.

Durante la fase de investigación también se ha hecho un análisis exhaustivo de las limitaciones tecnológicas pare generar un dataset desde 0 directamente desde la fuente de información que es Twitter.

Para la fase de desarrollo, ha sido necesario cumplir otra serie de objetivos secundarios hasta obtener el clasificador mencionado. Esto se ha realizado en forma de una herramienta capaz de hacer las siguientes operaciones en tiempo real:

- Descarga periódica de información vía API REST
- Parseo de la información para descartar errores o datos no deseados
- Normalización de la información
- Reentrenamiento de un modelo predictivo
- Almacenamiento de la información en una base de datos en cada una de las fases

Y por último, bajo demanda, la clasificación de un *tweet* o conjunto de *tweets* haciendo uso del modelo predictivo que estamos entrenando.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Estado del arte:** en el capítulo 2 se explican los conceptos sobre los cuales se ha debido aprender para la realización de este Trabajo de Fin de Grado, pasando por los estudios más relevantes y sus logros para continuar con el análisis fundamental del funcionamiento del algoritmo y la API de Twitter, el aprendizaje sobre conceptos como la minería de datos, el aprendizaje automático y los algoritmos de clasificación.
- **Diseño**: en el capítulo 3 detalla la fase del diseño. En este apartado se realiza la descripción del sistema, los pasos seguidos para la consecución y extracción de un data set, así como el diseño del procesamiento y la clasificación de los datos. También en este capítulo se exponen las limitaciones encontradas a la hora de realizar este Trabajo y las soluciones a estas limitaciones.
- **Integración, pruebas y resultados**: En el capítulo 4 se detallan las pruebas de integración realizadas, los problemas encontrados a la hora de realizar la integración y su resolución, así como los resultados obtenidos.
- **Conclusiones y trabajo futuro**: en el capítulo 5 se enumeran las conclusiones obtenidas tras la realización de este Trabajo de Fin de Grado. Asimismo, en este capítulo se explican las acciones fututas que serían necesarias para la mejora de este proyecto.

2 Estado del arte

2.1 Principales logros hasta el momento

En las siguientes dos secciones se detallan los trabajos más interesantes realizados sobre la popular red social y así como los *datasets* publicados como objeto de estudio disponibles en el momento de la elaboración de este trabajo.

2.1.1 Estudios realizados sobre Twitter

Como se ha mencionado anteriormente, podemos identificar distintas familias de trabajos cuyo objeto de estudio ha sido la red social.

En primer lugar, el análisis de los sentimientos, *opinion mining o sentiment analysis* sobre *tweets* de usuario podría considerarse como la categoría más popular. En esencia el *sentiment analysis* es particularmente útil como forma recopilar opiniones o *feedback* de la experiencia de los usuarios respecto a algo. Es particularmente útil en análisis de mercados [19][20] o análisis de experiencia de compradores [9], modelar opiniones sobre marcas [6] o simplemente sobre ideas [5]. Normalmente este tipo de estudio está basado en clasificaciones binarias o ternarias (positivo, neutro o negativo) y en 2020 hasta la propia API de Twitter en su versión profesional aporta *sentiment analysis* sobre los tweets extraídos.

En segundo lugar, tenemos la detección de noticias falsas o *fake news* como tema relevante desde su explosión en popularidad en 2016. La idea principal es clasificar los *tweets* o noticias embebidas en los *tweets* para identificar de otra manera si una noticia es verdadera o no [8][16][17]. Con este tipo de clasificación están muy relacionados los estudios enfocados a identificar campañas de manipulación de la opinión pública vía cuentas falsas o *bots* [3][21][22][23].

Otra categoría de trabajos particularmente interesantes es aquellos dedicados a la extracción del significado semántico de lo que escriben los usuarios en la red social para hacer mediciones del odio [18] o detectar tweets ofensivos [19]. Dado el enorme volumen de información en las redes sociales esto puede ser interesante a la hora de moderar estos espacios y ayudarnos a entender mejor cómo interactuamos en la red.

Por último, tendríamos los trabajos centrados en hacer algún tipo de categorización sobre la información disponible en Twitter. Esto es realmente útil para luego elaborar métricas y comprender qué información es relevante en la red social y cómo de popular es. En este apartado la gran mayoría de los estudios se centra en las tendencias o *trends* de la red social [11][13][14]. La idea consistente en tomar la información en bruto y ser capaces de asignar una etiqueta o categorización semántica de esa información a base de aplicar técnicas de procesamiento del lenguaje natural. De esta manera podríamos por ejemplo tomar un tweet y clasificarlo dentro de la familia “deportes”, “política”, “celebridades”, etc. [11]

La principal motivación de este trabajo es el hecho de que ninguno de los estudios de los últimos años mencionados en este apartado se centra específicamente en entender las características de los *tweets* que escriben los usuarios que hacen que tenga más o menos impacto dentro de una tendencia o *trend* determinada.

2.1.2 Datasets a partir de la información disponible

Como parte de este trabajo ha sido necesaria la elaboración de un *dataset* propio. La principal razón es que tal y como detalla Twitter en su Acuerdo y Política del Desarrollador [24] no está permitida la redistribución a terceros de su contenido, haciendo mención a *datasets* descargados.

Aún así, existen *datasets* publicados, así como algunas herramientas desarrolladas específicamente para extraer datos de la red social usando distintas técnicas.

Para el caso de estas herramientas, tras hacer pruebas se encontraron dos limitaciones. La primera es que o bien se limitaban a categoría específica de datos [25] o bien estaban basadas en *scrapping* y perdemos capacidades como los filtros de las búsquedas que sí nos da la API [26]. La segunda limitación es que al hacer un gran número de búsquedas con estas herramientas para generar un histórico extenso en el tiempo entran en juego las contramedidas de Twitter para evitar el abuso de su servicio con grandes volúmenes de peticiones, limitando drásticamente los datos que podemos extraer.

En cuanto a los *datasets* publicados, se destaca un proyecto de GitHub en el cual se reúnen algunos con el mayor número de publicaciones analizadas [27]. Pero debido a varias limitaciones en los propios *data sets* que se expondrán a continuación, no ha sido posible reusarlos y será necesaria la creación de un *dataset* propio.

Las limitaciones principales encontradas son:

- **Antigüedad:** Gran parte de los estudios sobre tendencias y análisis de sentimiento en Twitter son previos al cambio de algoritmo en esta red social en el año 2017, por lo que el modelo seguido para la consecución de dichos datos, no tiene en cuenta elementos esenciales como la localización geográfica. Además, como consecuencia de los cambios en el algoritmo y el número de usuarios activos en la red social, las métricas de un *tweet* considerado como "popular" ahora mismo difieren con las de hace cinco años. Por lo tanto, la utilización de *data sets* de hace cinco o diez años como, por ejemplo, el llamado Lerman Twitter 2010 Dataset [28] o *Twitter 2010* [29] actualmente no aportará los datos relevantes para la elaboración de este Trabajo de Fin de grado.
- **Enfoque metodológico**: No todos los *data sets* existentes en internet tienen el mismo objetivo que el de este trabajo. Algunos de ellos se enfocan en eventos concretos en el tiempo y una temática específica (como por ejemplo, el realizado para el análisis de la temática de la COVID-19 [30]; o de la Primavera Árabe [31]); pero también se encuentran otros en los cuales se busca entender las propias relaciones entre Tweets y usuarios, como en el caso de *Twitter Social Graph 2012* o *Social Clicks 2016.* [32] La existencia de distintos enfoques metodológicos hará que en la obtención de metadatos y la parametrización del *dataset*, no se encuentre la misma información. Algunos únicamente buscarán el ID del Tweet, el ID del Usuario y el link del Tweet, otros buscan la localización geográfica, la fecha, el sentimiento, etc. En el capítulo siguiente entraremos en detalle de los atributos que utilizamos para construir nuestro modelo y entre ellos se encuentra el campo *is_popular*, esencial para nuestro análisis y ausente en la mayoría de *datasets* al ser un parámetro de la búsqueda a la hora de extraer los *tweets* vía API.
- **Necesidad de llamadas a la API para regenerarlos**: En otros casos, se han observado *data sets* basados en referencias a IDs de Tweets, lo cual requiere llamadas

por API para volver a generar la información real del *dataset*, como es el caso del utilizado para el estudio *72 Hours of #Gamergate* [33]. Al no haber un mecanismo de consulta en Twitter basado en estas IDs por lotes, sería necesario una llamada por cada tweet. Además de esto, los *tweets* son mutables en el tiempo. Muchas de esas IDs corresponden con tweets que ya no están disponibles. Y por último, nos volvemos a encontrar con el problema del enfoque metodológico: las IDs corresponden a una colección de *tweets* que no tiene las características que nosotros buscamos (número de interacciones, *retweet* o no, *tweet* con mención o no...).
Teniendo en cuenta la limitación de llamadas a la API ya explicada anteriormente, esta opción no es apropiada para la realización de este trabajo.

2.2 El algoritmo de Twitter

Twitter se trata de una red social basada en la idea del "*microblogging*" que permite a los usuarios realizar publicaciones cortas (en un principio de 140 caracteres, y actualmente de 280) que son llamados "tweets". Los usuarios de Twitter siguen a otros usuarios para poder ver las publicaciones que realicen las personas que siguen en su "*timeline*". Además, el usuario puede interaccionar a estos tweets, contestándolos, dando "me gusta" y haciendo retweet.

En sus inicios en el año 2006, Twitter tenía una estructura muy sencilla, mostrando los tweets como una línea temporal, en orden cronológico inverso. Es decir, las publicaciones más recientes aparecían las primeras. Pero con el paso del tiempo y el incremento en el número de usuarios de la red social, con esta manera de ver las publicaciones, para un usuario se volvía casi imposible seguir las conversaciones, ya que el volumen de *tweets* a mostrar por minuto era muy elevado.

Por este motivo, Twitter decidió introducir un algoritmo que permitiera mostrar a los usuarios en su *feed* las conversaciones más relevantes o de más interés para ellos. En el año 2017 se produjo uno de los mayores cambios en el algoritmo de Twitter, introduciendo el concepto de "ranking".

Tal como explica Twitter en su blog, el algoritmo de 2017 funciona de la siguiente manera: [36] Tras reunir todos los tweets, cada uno se califica por un modelo de relevancia, atendiendo a características que varían según el usuario, lo que permite establecer qué tweet es más interesante para cada usuario. Tras esto, se muestra un conjunto de tweets con la puntuación más alta en la parte superior de la línea de tiempo, y el resto se muestra directamente debajo.

Este algoritmo además introduce "*In case you missed it" o* "En caso de que te lo hayas perdido", que se trata de un módulo en el que se contienen una pequeña muestra de tweets relevantes ordenados por su puntuación en el "ranking". Este módulo mostrará determinadas publicaciones dependiendo de la cantidad de Tweets disponibles para cada usuario y la cantidad de tiempo desde su última visita.

El algoritmo de 2017, para establecer este ranking tenía en cuenta una serie de parámetros: [36]

- El Tweet en sí: su actualidad, la presencia de tarjetas multimedia (imagen o video), interacciones totales (por ejemplo, número de retweets o Me gusta)
- El autor del Tweet: interacciones pasadas con este autor, la fuerza de la conexión con ellos, el origen de su relación.
- El usuario: Tweets que encontró interesantes en el pasado, con qué frecuencia y con qué frecuencia usa Twitter.

2.2.1 Funcionamiento del algoritmo de Twitter en 2020

A finales de 2019 Twitter realizó otro cambio relevante en su algoritmo, realizando una división entre "Top tweets" y "Temas", además del ya introducido "*In case you missed it*". Top Tweets muestra las publicaciones que más interesarán a un usuario concreto, teniendo en cuenta las cuentas o categorías de tweets con las que interactúa más dicho usuario. El apartado de "Temas" mostrará al usuario publicaciones de una temática de su interés. A finales de enero de 2020, Twitter publicó que tenían 100 temáticas dentro de este apartado de "Temas" en los que englobaban los tweets, y que semanalmente añadirían más temas. [34] Cabe destacar que es el propio usuario el que tiene que establecer los temas que le interesan, en un buscador específico.

Figura 1: Buscador de "Temas" en Twitter

En este apartado, el algoritmo funciona agregando nuevos temas basados en el volumen y el estado de las conversaciones de Twitter, observando cuánta gente está tweeteando, retweeteando, respondiendo y dando "me gusta" a los Tweets sobre un tema. Cuanta más conversación sobre una temática, más probable será que se evalúe para su inclusión en próximo lote de "*Temas*" disponibles. Al contrario que los Trending Topics, el apartado de "Temas" reflejan las conversaciones más amplias y duraderas sobre determinado evento. No obstante, dentro de estos "Temas" el algoritmo también analiza las publicaciones para identificar los tweets más relevantes dentro de esa temática.

2.2.2 Algoritmo de Twitter: limitaciones y dificultades

Los *trending topics* o *trends* son aquellos temas de discusión más populares en un momento dado dentro de la red social y se representan como ideas descritas brevemente por un conjunto de palabras, en notación *hashtag* (palabras unidas sin espacios precedidas por el carácter '#') o sin estructura.

El algoritmo de Twitter determina los *trending topics* combinando el volumen de tweets generados sobre determinada temática y el tiempo que ha sido necesario para crear ese volumen. Por lo tanto, si un determinado *hashtag* o conversación es recurrente en el tiempo y no se han producido picos al alza en las publicaciones de los usuarios sobre dicha temática en las últimas veinticuatro horas, Twitter no lo considera un *trending topic*, sino que el algoritmo lo considera una conversación y podría eventualmente ser incluido en el apartado de "*Topics*".

Si el volumen de *tweets* que tratan una temática o utilizan un *hashtag* determinado experimenta un pico elevado entre las publicaciones, entonces el algoritmo de Twitter lo considera tendencia.

Otro aspecto interesante en el algoritmo de Twitter a la hora de decidir qué mostrar como *trending topic* es que, dependiendo del usuario, lo que se muestra como tendencia varía. El algoritmo se basa en aspectos como la localización geográfica, los gustos o las personas a las que sigue un usuario para mostrarle las tendencias que considera pueden interesar a dicho usuario.

Por lo tanto, si el usuario quiere saber qué está siendo tendencia en Estados Unidos, tendrá que realizar una serie de modificaciones en la configuración de su cuenta. Durante la realización de este trabajo de Fin de Grado, se ha tenido en cuenta este aspecto del algoritmo de Twitter, para lo que se ha decidido configurar las cuentas utilizadas como si el usuario se encontrara en Estados Unidos. Se ha seleccionado esta configuración debido a que la mayor parte de los usuarios de esta red social se encuentra en Estados Unidos (48.65 millones de usuarios) y se trata de una cifra muy amplia que permitirá realizar el estudio de manera más global, evitando el sesgo por país o región geográfica.

Además, las cuentas utilizadas para la extraer información de la red social son cuentas nuevas sin *followers* ni *followees* o cualquier tipo de personalización más allá del idioma y la localización con el objetivo de influir lo menos posible sobre el algoritmo de Twitter a la hora de presentarnos la información.

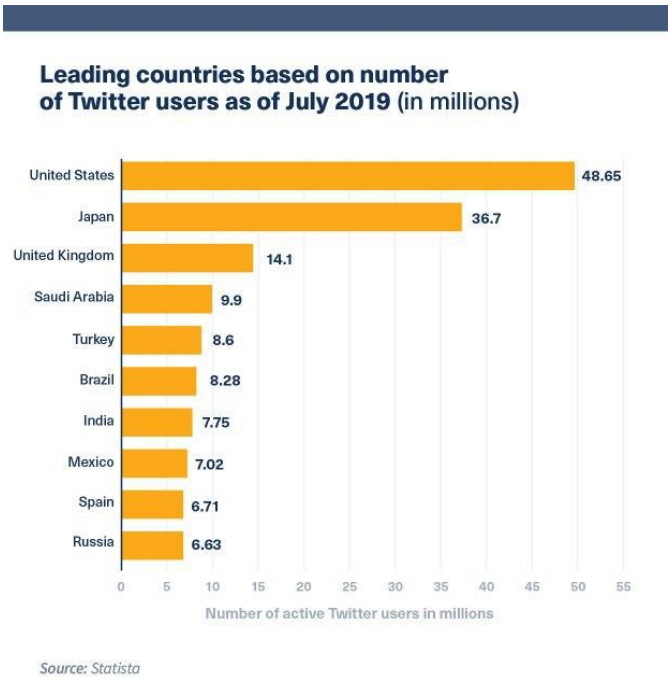

Figura 2: Gráfica de usuarios de Twitter en el mundo

A pesar de que es posible consultar el *ranking* de los temas más populares en tiempo real, Twitter no especifica ni la metodología que utiliza para extraer los *topics* en base al contenido de los tweets ni tampoco las métricas para determinar que en efecto son *trending topic*.

La red social sí especifica que las nuevas tendencias se generan en base a las métricas de las tendencias existentes o anteriores (como el volumen de tweets, por ejemplo). Por tanto, si un *hastag* o conjunto de palabras tiene un gran volumen de tweets recientes en comparación con el pasado, se podría considerar que está siendo popular, y podría convertirse en un *trending topic* o en uno de los Tweets relevantes del apartado de "Topics" en la web.

No obstante, a pesar de la existencia de una serie de "Temas" utilizados en el presente algoritmo de Twitter y podría resultar interesante abordarlo en este trabajo, en la API REST no se pueden extraer los metadatos que ayuden a la clasificación de los *topics* por temática.

Aunque se extrajese dicha clasificación de los *topics* por otras vías, tras haber realizado varias observaciones en el tiempo se aprecia rápidamente que no es ni consistente ni acertada en casi más de la mitad de los casos.

Por una parte, no es consistente ya que los "Temas" son más o menos de manera arbitraria entre dos *topics* determinados que semánticamente tienen el mismo nivel de detalle acotados (e.j.: Deportes vs Tenis).

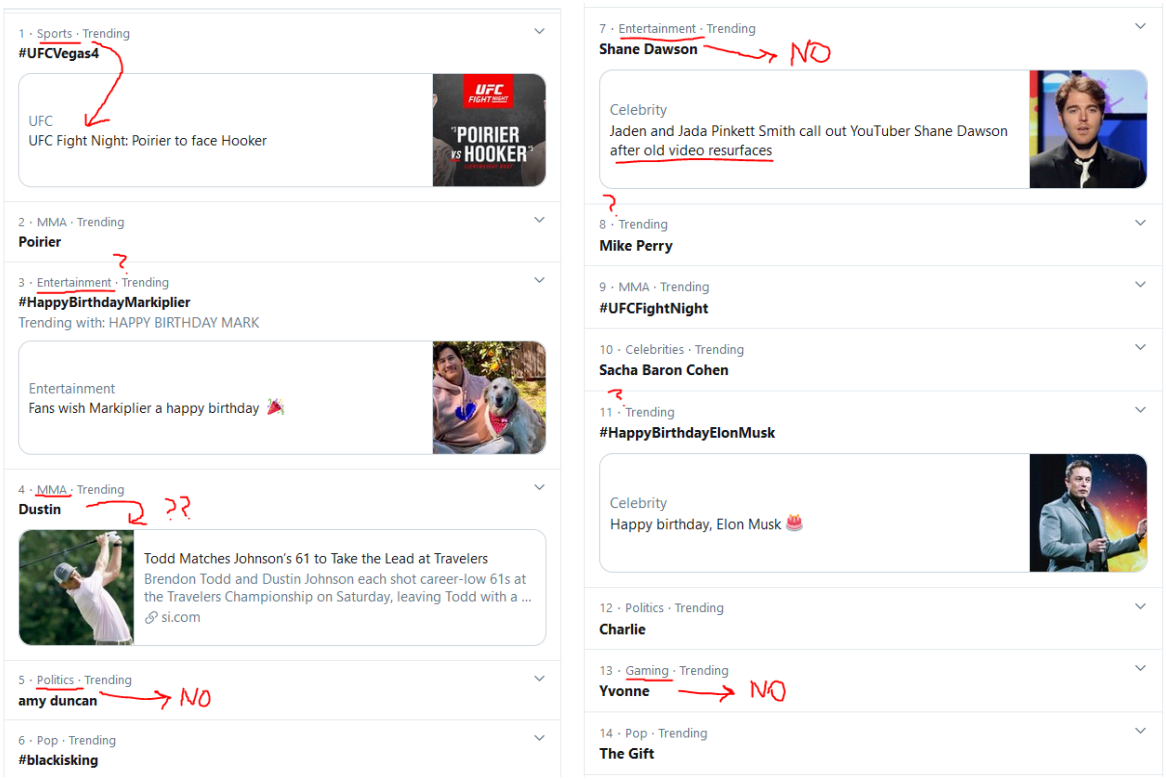

Figura 3: Discrepancias en la categorización en "Temas" de varios *trending topics*

Por otra parte, muchas veces los "Temas" son directamente erróneos. Es decir, la categorización que hace Twitter no tiene sentido en relación al contenido semántico del *topic*. Sumado a esto, muchas veces ni siquiera aparecen con una categorización.

Debido a las razones expuestas no se ha considerado utilizar esta información como una fuente de conocimiento interesante a la hora de realizar nuestro análisis y clasificación.

Por último, aunque no conozcamos específicamente los límites en los que se basa el algoritmo de Twitter para lo que se ha descrito en este apartado, sí se puede extraer otra información útil para este estudio.

En particular, para un tweet determinado se puede extraer si ha llegado a categorizarse como *popular* dentro de un *topic* por el algoritmo. Esta información va a resultar clave para el trabajo en cuestión y se abordará en los siguientes apartados cómo la extraemos y qué hacemos con ella.

2.3 La API de Twitter

Para acceder a la API de Twitter es necesario contar con una cuenta en la red social, para después solicitar una cuenta como "desarrollador", obtener el acceso a la misma previa verificación de Twitter. Para la realización de este trabajo se ha obtenido acceso a la API "estándar", que cuenta con una serie de limitaciones que serán comentadas posteriormente.

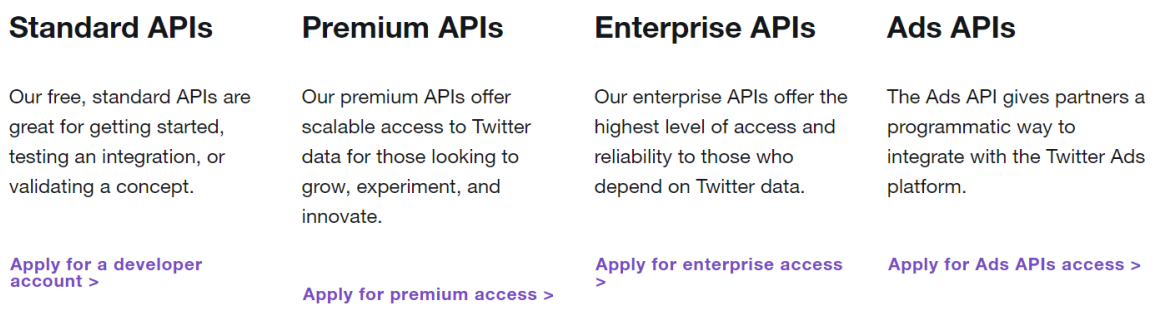

Figura 4: Tipología de APIs de Twitter

Además de la obtención del acceso, para poder realizar solicitudes de API a Twitter es necesario crear una aplicación en la sección de desarrolladores de Twitter [35]. La creación de una aplicación es la forma estándar para que los desarrolladores obtengan acceso a la API y para que Twitter pueda monitorizar e interactuar con desarrolladores de plataformas de terceros según sea necesario. La creación de esta aplicación de Twitter sigue un proceso estándar, en el que simplemente es necesario el acceso de "sólo lectura" a la API. Tras esto y con una conexión API autorizada, será posible emitir una solicitud a la misma.

Durante la elaboración de este Trabajo de Fin de grado se ha utilizado la API REST de Twitter, que permite el acceso de lectura y escritura de los datos de Twitter. Las respuestas de la API REST de Twitter se encuentran en formato JSON.

Las principales características de la API de Twitter son:

- Existen cuatro "objetos" principales: Tweets, Usuarios, Entidades y Lugares.
- La API está basada en HTTP (sobre SSL), por tanto, los procesos que necesiten un método HTTP determinado, si no realiza la solicitud correcta, devolverán un error.
- Existen una serie de parámetros específicos en las solicitudes a la API, así como límites de paginación y bibliotecas, que son generadas específicamente para adaptar el funcionamiento de la API.

Además, Twitter establece una serie de límites de velocidad sobre cuántas solicitudes puede hacer una aplicación a un recurso API determinado dentro de un período de tiempo determinado. Los límites de velocidad de Twitter están bien documentados, y cada recurso API individual también establece sus límites particulares para su conveniencia.
Los límites de velocidad se dividen en intervalos de 15 minutos. Todos los *endpoints* requieren autenticación, por lo que no será posible realizar llamadas sin autenticar o sin límite de tiempo.
Cabe destacar que las limitaciones específicas son por *endpoint*, no por protocolo HTTP. Es decir, el *endpoint* "search/tweets" tiene distintas limitaciones al *endpoint* "trends". Por ende, hay dos categorías iniciales disponibles para las solicitudes GET: 15 llamadas cada 15 minutos y 180 llamadas cada 15 minutos [37].

Durante la elaboración de este Trabajo de Fin de Grado, ha sido necesario realizar un número mayor de llamadas a la API de Twitter para la elaboración del "data set" y la extracción de

los datos. Por tanto, estos límites en la API de Twitter han supuesto un reto para la investigación, que ha sido resuelto mediante la creación de varias cuentas de desarrollador para acceder a la API de Twitter desde distintos usuarios y conseguir así un mayor rango de llamadas.

2.3.1 Los Tweet IDs

Los Tweets son los elementos esenciales de Twitter, que se consideran "actualizaciones de estado" [38] Los tweets tienen numerosos atributos, entre los cuales se encuentran los fundamentales, que serán: ID, Created_at (fecha de creación) y Text (texto).

En este Trabajo de Fin de Grado, el atributo ID se considera fundamental, puesto que es utilizado para la elaboración del *dataset* que permitirá establecer una clasificación de las publicaciones.

El atributo ID se trata de la representación entera del identificador único para este Tweet.
Twitter utiliza el servicio "Snowflake" para generar ID únicos en los objetos dentro de Twitter (Tweets, Mensajes directos, Usuarios, Colecciones, Listas, etc.). Estos ID son enteros únicos (sin signo) de 64 bits, que están basados en el tiempo. [39]
El ID completo del Tweet o del objeto se compone de una marca de tiempo, un número de trabajador y un número de secuencia.
Es necesario tener en cuenta que, al tratarse de un número mayor de 53 bits, algunos lenguajes de programación pueden tener dificultades al interpretarlo. Por ejemplo, al utilizar la API de Twitter usando JSON, será necesario usar campo id_str en lugar de id. Esto se debe a la forma en que Javascript y otros lenguajes que utilizan JSON evalúan enteros grandes.

2.3.2 La API estándar

Como se ha comentado anteriormente, la API de Twitter es una API REST que se utilizará para la consecución de distintos objetivos. Para la elaboración de este Trabajo de Fin de Grado, tan solo es necesaria la API de búsqueda, que permite encontrar una serie de Tweets basados en atributos. Esta API cuenta con distintas modalidades: estándar y "premium".

La API estándar, la usada en este Trabajo, busca en una muestra de Tweets recientes publicados en los últimos 7 días, y devuelve una colección de Tweets relevantes que coinciden con una consulta específica.

La API de búsqueda estándar de Twitter, de igual manera que ocurre en el resto de APIs de esta red social, requiere autenticación y se encuentra limitada. El número de solicitudes por ventana de 15 minutos (autenticación de usuario) es de 180, mientras que el número de solicitudes por ventana de 15 minutos en la autenticación de la aplicación es de 450.

Los parámetros que podrán utilizarse en esta versión de la API son:

Parámetro	**Descripción**
q	Una consulta de búsqueda codificada en URL UTF-8 de 500 caracteres como máximo, incluidos los operadores.

geocode	Devuelve tweets de usuarios ubicados dentro de un radio de latitud/longitud.
lang	Restringe los tweets a un idioma determinado.
result_type	Especifica qué tipo de resultados de búsqueda preferiría recibir: - mixed: Incluye resultados populares y en tiempo real. - recent: devuelve solo los resultados más recientes. - popular: devuelve solo los resultados más populares.
count	El número de tweets que serán devueltos, hasta un máximo de 100.
until	Devuelve tweets creados antes de la fecha dada. En esta API el límite es de 7 días, por lo que no se encontrarán tweets anteriores a una semana.
since_id	Busca el ID mayor que (es decir, más reciente que) el ID especificado.
max_id	Busca el ID menor que (es decir, anterior o igual) al ID especificado.
include_entities	Incluye en la búsqueda las "entidades" (hashtag, URLs, etc)

A parte del atributo ID, comentado en el apartado anterior, para la elaboración del *data set* en el que se basa este Trabajo de Fin de Grado, el parámetro result_type será también fundamental.

2.3.3 La API premium

La API premium de Twitter requiere de una solicitud para su consecución, así como el pago de una cuota mensual. El coste de esta API premium es de 149 dólares al mes y permite 500 peticiones.

Entre las diferencias de la API estándar y la API premium destaca que la API premium de Twitter permite una búsqueda en un periodo de tiempo mayor que la estándar (siete días), existiendo un *endpoint* en el cual se busca en los últimos 30 días, y otro en el cual la búsqueda se realiza desde el año 2006.

Además, en la API premium se encuentra disponible un *endpoint* de "recuento" en el que se proporcionan los volúmenes totales de tweets asociados con una consulta, ya sea diariamente, por hora o por minuto.

También en esta API no existe el límite de consultas por ventana de 15 minutos, sino que contará con un número de solicitudes al mes; y tiene una mayor cantidad disponible de parámetros, como:

Parámetros	**Descripción**
query	El equivalente a una regla/filtro premium de hasta 1,024 caracteres. Este parámetro debe incluir todas las partes del filtro, incluyendo operadores.
tag	Las etiquetas se pueden utilizar para diferenciar las reglas y sus datos coincidentes en diferentes grupos lógicos.
fromDate	Establece la fecha desde la cual se proporcionarán los tweets (hasta 30 días antes).
toDate	Establece la fecha hasta la cual se proporcionarán los tweets.

maxResults	Máximo número que será devuelto en la consulta, con un mínimo de 10 y el máximo será el del límite del sistema.
next	Permite obtener la “siguiente página de resultados”

2.4 Machine learning

2.4.1 Machine learning en RRSS

El aprendizaje automático o *machine learning* se trata de un método de análisis de datos basado en la automatización de la construcción de modelos analíticos. Este método se trata de una rama de la inteligencia artificial, que recoge la idea de que los sistemas son capaces de aprender de los datos, identificar patrones y tomar una serie de decisiones sin necesidad de la implicación humana [40].

El *machine learning* ha sido ampliamente utilizado en estudios sobre Twitter, para realizar distintas clasificaciones como de usuarios, análisis de sentimiento, detección de *bots* o perfiles “*trolls*”, seguimientos de fenómenos naturales o del avance de epidemias, etc.

La utilización de este método requiere la creación algoritmos que estén capacitados para generalizar comportamientos a partir de ejemplos de aprendizaje. Los algoritmos de *machine learning* establecen un modelo de aprendizaje basado los ejemplos y utilizarán este modelo para realizar predicciones o describir patrones [41]. Los algoritmos de aprendizaje automático son fundamentales para el proceso de minería de datos (*data mining*).

2.4.2 Data mining en Twitter

El concepto de *data mining* hace referencia al proceso de la búsqueda de patrones, anomalías y correlaciones dentro de grandes conjuntos de datos, que tras ser tratados permitirán predecir una serie de resultados [42].

Como se comentó en la introducción, Twitter se trata de una de las redes sociales con más tráfico e interacciones de usuarios, de acuerdo con su informe para los accionistas publicado el 30 de abril de 2020, en el primer cuatrimestre de este año contaría con 166 millones de usuarios activos, aumentando en un 24% con respecto al año anterior [2]. Este incremento ha venido de la mano de la pandemia mundial de COVID19, que ha llevado a muchos usuarios a buscar en redes sociales como Twitter su fuente de información de manera "instantánea". De acuerdo con el propio informe de la empresa, los picos de los usuarios activos y sus interacciones son estacionales.

Es por ello que sería imposible analizar la gran cantidad mensajes intercambiados entre los usuarios en uno de estos picos sin realizar un proceso de *data mining*. Para la consecución del objetivo de este Trabajo de Fin de Grado será necesario conseguir una normalización de los datos de un *tweet* que ayuden a su análisis, para poder así entender los patrones que se repiten en los Tweets que han tenido impacto sobre un *trending topic*, lo que ayudará a prever qué publicaciones serán más propensas a convertirse en "populares" en el futuro.

Esta normalización y parametrización será necesaria realizarla a través de un proceso, que tal como mencionan Nancy Verma y Gaurav Gupta en su investigación "*Sentiment analysis of real time social tweets*", constará de varias fases [43].

La primera de estas fases será la selección de los datos con el correspondiente filtrado de los mismos, seguida por el procesamiento y la transformación de los datos. La transformación hace referencia a la normalización de la información obtenida para extraer y generar los atributos necesarios que permitirán definir los individuos (*tweets*) de la población (*data set*). La transformación es un paso clave para realizar el proceso de *data mining* y generar modelos de predicción.

2.4.3 El problema del volumen de datos en el aprendizaje automático

Existen tres componentes esenciales en un modelo de aprendizaje automático: los datos, las características extraídas de los datos y el modelo. La experiencia acumulada durante la última década ha demostrado que el tamaño del *dataset* es el factor más importante [44]. Por este motivo, una de las partes esenciales de este Trabajo de Fin de Grado es la elaboración de un *data set* amplio y completo.

Los estudios sobre aprendizaje automático han demostrado repetidamente que los modelos simples basados en enormes cantidades de datos son mejores que los modelos más sofisticados basados en menos datos [45].

Pero, el análisis y la utilización de grandes volúmenes de datos en el *proceso de machine learning* tiene también una serie de limitaciones.

2.5 Modelos del comportamiento humano basados en Twitter

Para realizar una selección de los datos extraídos, es necesario entender que detrás de los Tweets se encuentran personas, y que éstas actúan o tienen un comportamiento a la hora de realizar publicaciones que también será parametrizable. Por tanto, encontrar un modelo apropiado del comportamiento humano en Twitter permitirá acotar y llevar a cabo una mejor selección de los datos.

Durante la realización de este Trabajo de Fin de Grado se ha observado que la mayor parte de las investigaciones y estudios que tratan el comportamiento humano y el análisis de sentimiento en Twitter se basan en dos modelos: el léxico y el del aprendizaje automático. Ambos están a su vez basado en un modelo denominado "*bag-of-words"* o "bolsa de palabras", que se trata de una representación que convierte texto arbitrario en vectores de longitud fija, en los cuales se cuenta cuántas veces aparece cada palabra.

2.5.1 Modelo Léxico

En este primer modelo será esencial seleccionar los elementos léxicos de un mensaje que deben considerarse en el análisis de sentimiento. Algunos autores, como Benamara et al. Proponen el método de Combinación Adverbio-Adjetivo (AACs) para detectar la polaridad en los sentimientos [46]. Otros autores se han basado en la utilización de emoticonos o expresiones para asociarlos a un sentimiento (positivo, negativo o neutro).

También, entre los modelos basados en el léxico encontrados en diferentes investigaciones, se destaca el estudio de Lee et al. llamado *Twitter Trending Topic Classification*, en el cual proponen dieciocho categorías para la clasificación de los temas o *topics* [47].

En este estudio, el modelado de los datos se realiza de dos maneras: modelado basado en el texto (lenguaje) y modelado basado en la red. En el modelado del lenguaje utilizaron una serie de términos, vocabulario y abreviaturas comúnmente utilizadas en Twitter debido a las limitaciones de espacio de las publicaciones (140 caracteres en el momento del estudio); y eliminaban las denominadas *stop words* (palabras comunes en el lenguaje como conjunciones, artículos y pronombres). Tras esto, transformaban las palabras en una medida denominada "tf-idf" (*term frequency–inverse document frequency*), es decir, la importancia de un término en un documento. En este modelo, es posible diferenciar las siguientes categorías: arte y diseño, libros, caridad y ofertas, moda, comida y bebida, salud, humor, música, política, religión, vacaciones y fechas, ciencia, deportes, tecnología, negocios, televisión y películas, otras noticias y otros.

2.5.2 Modelo de Aprendizaje automático

En los estudios observados, el enfoque de aprendizaje automático en Twitter utilizado para la clasificación de texto y análisis de sentimiento, se basa en algoritmos que analizan datos que anteriormente se etiquetaban como positivos, negativos o neutros. Estos algoritmos extraen características que modelan las diferencias entre diferentes clases e infieren una función que puede usarse para clasificar nuevos ejemplos no vistos antes. [48]

2.6 Procesamiento del lenguaje natural

El procesamiento del lenguaje natural o *NLP (Natural Processing Language)* es la tecnología utilizada para trasladar el lenguaje humano al lenguaje máquina, ya que no es un proceso directo y sencillo conseguir que un ordenador comprenda el lenguaje natural.

Aunque los seres humanos entendemos fácilmente las sutilezas del lenguaje natural, esto no es fácilmente transmitible a una máquina. A la hora de derivar el significado de una frase, existen reglas sencillas como puede ser el concepto de propiedad de un objeto que puede derivarse a partir de los posesivos “mío”, “tuyo”, “suyo” ... Otras reglas no son tan sencillas y pueden llegar a ser muy abstractas, como el concepto del sarcasmo o la ironía.

El *NLP* implica una serie de algoritmos para identificar y extraer estas reglas del lenguaje natural y utilizarlas para convertir texto no estructurado, como puede ser en nuestro caso un *tweet*, en un formato de texto que sí sea entendible por una máquina.

Las principales técnicas de las que haremos uso en este trabajo caen dentro del análisis sintáctico de una frase. El análisis sintáctico nos permite determinar cómo se alinea el lenguaje natural con las reglas gramaticales. Después mediante algoritmos aplicamos dichas reglas a un conjunto de palabras y derivamos el significado en conjunto de éstas [49].

La normalización del lenguaje es un proceso complejo, existiendo diversas técnicas que tienen en cuenta distintas características para realizar dicha normalización. En este Trabajo

de Fin de Grado se hará uso de varias técnicas actuales, léxicas y sintácticas que se exponen a continuación.

2.6.1 Tokenization

Uno de los primeros requerimientos para realizar procesamiento del lenguaje natural es el *corpus*. El *corpus* es una colección de textos estructurado. En nuestro caso consistirá en una gran colección de tweets ya existente que utilizaremos como nuestro vocabulario a modo de modelo preentrenado. El plural del *corpus* es el *corpora*. Un ejemplo de un *corpora* sería juntar 2 *corpus* como un dump de todos los artículos de la Wikipedia en inglés junto con un dump de Tweets.

Los *tokens* son los bloques de construcción del *NLP* y son la manera más común de interpretar el lenguaje natural. La *tokenization* o tokenización es el proceso de partir el texto en trozos llamados *tokens*, ignorando caracteres como los signos de puntuación, caracteres especiales y espacios. Estos

El objetivo de la *tokenization* es formalizar el texto en lenguaje natural en una lista de palabras, formando un vocabulario común entre los distintos *tweets*. Cada una de estas palabras de nuestro vocabulario tiene una representación en nuestro *corpus*. El principal reto de esta técnica de normalización es cómo representar las palabras que están fuera de nuestro *corpus.* [50]

Aunque es un proceso aparte, cuando hablemos de *tokenization* incluiremos como paso previo la eliminación de *stopwords* o palabras que no tienen significancia para ser incluidas dentro de nuestro vocabulario. Por lo general siempre es interesante realizar este paso previo para reducir nuestro vocabulario total.

2.6.2 Lemmatization

La lematización es una manera de lidiar con las diferentes derivaciones de una sola palabra que tienen una raíz común. Por ejemplo, en inglés palabras como *connect*, *connection*, *connecting*, *connected*, etc tienen todas un mismo significado léxico: *connect* (hacemos uso de ejemplo en inglés dado que nuestro trabajo se centra en analizar texto en lengua inglesa).

Con esta técnica reducimos considerablemente el vocabulario total sobre el que trabajamos.

2.6.3 Part of speech (POS) Tagging

El *POS* de una palabra nos da la información de qué función tiene una palabra dentro de una frase. Es decir, consiste en etiquetar las palabras con su función.

Si por ejemplo tomamos la frase “This program is beautiful”, al aplicar *POS Tagging* etiquetaremos que “This” es un determinante, “program” es un nombre, “is” un verbo y “beautiful” un adjetivo. Esta información puede sernos útil para derivar el significado de esa frase.

2.6.4 Bag of words (BOW)

A partir de un texto tokenizado, *bag of words* es una manera de representar texto en forma de vectores de tamaño fijo a base de contar cuantas veces aparece cada palabra. Este proceso

se conoce también como vectorización o *vectorization*. El tamaño máximo del vector siempre será igual al número total de palabras existentes.

Su virtud principal es la simplicidad en la computación frente a otras formas de representar texto. [51]

2.6.5 Word embeddings

Nos centraremos en esta sección específicamente en los *neural word* embeddings, la forma más sofisticada utilizada actualmente en el *NLP* [53].

Los *word emeddings* (sin una traducción al español práctica) son otra manera de representar texto en formato numérico que una máquina puede entender. En esencia permite tener una representación similar de palabras que son similares, haciendo uso de una fase de aprendizaje. Cada palabra es mapeada a un vector y los valores de cada vector son la salida de un aprendizaje similar al de una red neuronal.

La potencia de estos modelos es que son capaces de capturar el contexto de una palabra en un texto en cuanto a su similitud semántica y sintáctica con otras palabras [54]. Los modelos más interesantes como objeto de estudio son Word2Vec [55] y Glove [56].

2.6.6 Contracciones en inglés

Una de las técnicas particularmente interesante para nuestro estudio a la hora de reducir las dimensiones del vocabulario es eliminar de una manera informada las contracciones de la lengua inglesa antes de realizar la tokenización.

En inglés, construcciones como “ain’t” pueden expandirse de distintas maneras, en función del contexto de la frase: “am not”, “are not”, “is not”, “has not”, “have not”. Estas pequeñas diferencias pueden cambiar el significado semántico de una frase.

Haciendo uso de modelos pre-entrenados podemos extrapolar el contexto de la frase analizar a la de uno de esos modelos y expandir la contracción a las palabras originales.

Como parte de esta técnica también es interesante aprovechar los modelos pre-entrenados y realizar una corrección de faltas orográficas de cada palabra analizada [52].

3 Diseño y desarrollo

3.1 Descripción del sistema

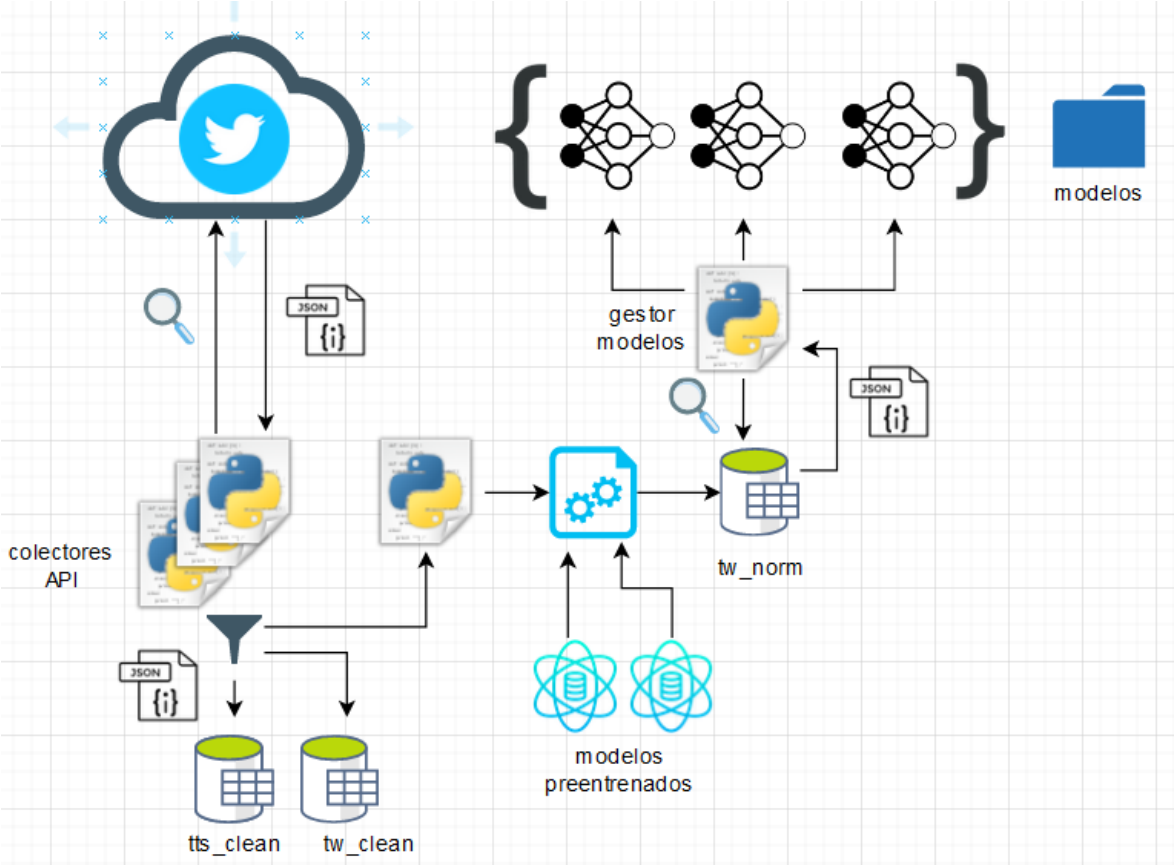

Figura 5: Overview de arquitectura

La figura 5 consiste en una vista lógica de la arquitectura con todas las piezas que componen el sistema desarrollado.

A continuación, se explica cada uno de los componentes etiquetados en el diagrama, detalles sobre su implementación relevantes y operaciones que se realizan.

3.1.1 Colector de datos

Con el fin de simplificar la explicación, se asume que se utiliza un único colector de datos con una única pareja de claves asociadas a una app de Twitter. La app de Twitter hace uso de la *standard API* con las limitaciones que ello conlleva. En el apartado 2.3 se han descrito

ya estas limitaciones. Partiendo de un colector así, se intentará siempre optimizar el número de peticiones por ventana para extraer el mayor volumen de datos posible.

```
REQ_INTERVAL    = 900.0
SEARCH_COUNT    = 100
TOP_TTS         = 12
POPULAR_ITERS   = 5
RECENT_ITERS    = 10
SEARCH_MODE     = 'extended'
LANG            = 'en'
```

Figura 6: Parámetros de un colector "standalone

El colector de la API es el script que corre bajo un proceso independiente ininterrumpido. El colector hace uso de una *API key* y una *API secret key* para autenticarse con la API REST de Twitter. Cada 15 minutos (duración de una ventana de peticiones) se realizan las siguientes peticiones a la API de Twitter siguiendo las limitaciones descritas en el apartado 2.3:

- `GET trends/place x1`
- `GET search/tweets x5 with:`
  - `result_type="popular" x5`
  - `count=100`
- `GET search/tweets x10 with:`
  - `result_type="recent"`
  - `count=100`

Primero se extraen los 12 *trending topics* o TTs más populares siguiendo el orden dado por Twitter de más a menos popular. Los topics son parseados de la respuesta de la petición y almcenados en una colección llamada "tts_clean" en la base de datos. Se indexa por el campo "topic".


```
_id: ObjectId("5efbdbbf435238a45a8b9912")
topic: "#AmericaStrongerWithBiden"
as_of: 2020-07-01T06:30:21.000+00:00
location_name: "United States"
location_woeid: 23424977
promoted_content: null
query: "%23AmericaStrongerWithBiden"
tweet_volume: 155234
url: "http://twitter.com/search?q=%23AmericaStrongerWithBiden"
```


Figura 7: Esquema de un trending topic almacenado

Se toma cada uno de los 12 "topics" más destacados y se procede a extraer los tweets relevantes para cada topic.

Por una parte, se realizarán 5 consultas a la API para extraer los *tweets* marcados como `is_popular = True` con un máximo de 100 tweets por consulta (máximo permitido por la API). El máximo potencial de tweets que se pueden extraer son 500 de este tipo por ventana.

En la API de Twitter no existe explícitamente el concepto de la paginación así que el número de peticiones que se realizan lo determina el cursor que devuelve la API en forma del objeto con metadatos *search_metadata* [57]. Las querys sucesivas que se realizan hacen uso de estos metadatos para seguir extrayendo tweets hasta que la API deja de devolver resultados, por lo que no se sabe a priori cuantos resultados nos va a devolver en una ventana de peticiones.

Por otra parte, realizamos 12 consultas a la API esta vez para los *tweets* marcados por la API como `is_popular = False`. El máximo potencial de tweets que se pueden extraer son 1200 por ventana. Esta vez extraemos todos los tweets que han sido creados recientemente y pueden ser o no populares en un momento dado.

Una vez se tienen los dos conjuntos de Tweets, se comprueba que los tweets en la suma de ambos conjuntos sean únicos. Para cada uno de ellos se realiza una primera fase de pre-procesado, donde se eliminan los metadatos que no nos interesan. Esto lo hacemos debido a la limitación descrita en el punto 4.1.2. A continuación se actualizan en una primera colección llamada "tw_clean" en la base de datos. El índice en este caso es el *tweet* ID.

Figura 8: Esquema de tweet parseado

Dado que estas operaciones se realizan cada 15 minutos, muchos tweets ya existirán en esta colección y se producirá una actualización de sus atributos en su lugar. Si alguno de los tweets actualizados trae el valor `is_popular=True` diremos que el tweet en cuestión ha tenido impacto sobre el `relevant_topic` que tiene asociado.

La ventaja de tratar los tweets con este enfoque único y asíncrono es que podemos parar y arrancar la colecta de tweets en cualquier momento, añadir más colectores o modificar el esquema si fuese necesario.

Esta primera collección llamada "tw_clean" conserva los tweets tal y como los hemos obtenido de la fuente con un mínimo de modificaciones, de manera que nuestros datos son reutilizables si decidimos cambiar el caso de uso.

A continuación, se pasa la lista de tweets al pipeline de procesamiento.

3.1.2 Pipeline NLP

En el pipeline de procesamiento realizamos todas las modificaciones necesarias a los tweets para utilizarlos en el entrenamiento de nuestro modelo posteriormente.

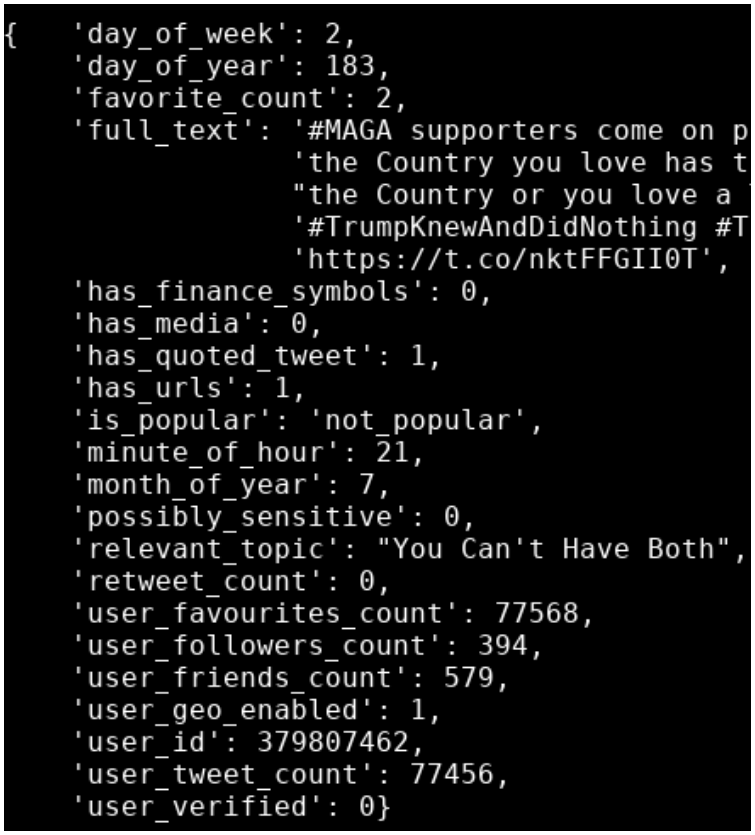

Figura 9: Campos de un tweet previo a modificaciones del pipeline

Hasta este punto, los tweets que van a procesarse para ser añadidos a nuestro dataset contienen una serie de atributos de tipo texto, numérico, array o booleano.

3.1.2.1 Normalización de atributos del tweet

Como fase preliminar se aplicará una función a cada tweet que haga una primera normalización de los atributos que no sean numéricos o texto, quedando cada tweet como se muestra en la figura 10. No obstante, tras haber realizado distintas pruebas, esta lista no representa todos los atributos definitivos que tendrá cada tweet del dataset como se explicará en el apartado 4 pero sí representa los atributos esenciales.

Los atributos de tipo array como has_urls o has_media por ejemplo contienen arrays con el contenido correspondiente si el tweet lleva esos datos embebidos y se normalizan al número de elementos contenido en el array.

Los atributos de tipo booleano se normalizan también a un valor numérico, 0 o 1 en función de lo que corresponda.

3.1.2.2 Corrección de faltas ortográficas

El primer procesado del texto se hace sobre el campo que contiene el texto del tweet `full_text` y consiste en la corrección de las faltas ortográficas. Para ello hacemos uso del primer modelo estadístico pre-entrenado en el pipeline. Este modelo nos da datos estadísticos sobre un vocabulario formado por dos grandes *corpus*: artículos de la Wikipedia en inglés y una colección de 330.000 tweets escritos en inglés [58]. En esencia lo que hacemos en esta parte del pipeline es utilizar estas estadísticas de palabras para buscar el candidato más probable para realizar la corrección [60]. La modificación se hace *in-place* sobre el propio campo.

3.1.2.3 Expansión de contracciones

El siguiente procesado del texto del tweet consiste en realizar el proceso descrito en el apartado 2.6.6 y de nuevo haremos uso de un modelo estadístico pre-entrenado, al que nos referiremos como *GloVe.Twitter.100d*. En este caso usaremos *word embeddings* generados con GloVe [56]. Los vectores de palabras usados se han generado a partir de un gran *corpus* de tweets en inglés con 27 billones de tokens y 1.2 millones palabras de vocabulario.

Las modificaciones de nuevo se harán *in-place* sobre el campo que contiene el texto del tweet.

3.1.2.4 Word segmentation, annotation & tokenization

La siguiente fase de procesamiento sobre el texto está formada por dos operaciones. De nuevo se hará uso de las estadísticas de palabras *GloVe.Twitter.100d* para llevarlas a cabo.

La primera es la que nos referiremos como segmentación de palabras o *word segmentation* y en nuestro caso consiste en aprovechar las estadísticas que tenemos sobre textos en tweets para descomprimir o expandir ciertos patrones de texto. Esto es particularmente útil para interpretar los *hashtags* que forman parte del tweet.

Un *hashtag* consiste en un conjunto de palabras precedidas por el carácter '#` y escritas en formato *PascalCase* o directamente escritas todas juntas sin delimitación. Los *hashtags* nos dan información sobre qué se está hablando en un *tweet* y pueden existir varios, por eso es importante extraer su significado.

En la práctica lo que ocurrirá es que construcciones en inglés como por ejemplo '#Notmypresident' las interpretamos como 'not my president' y conseguimos dos cosas: reducir el número de palabras distintas de nuestro vocabulario a procesar (porque hemos reducido el *hashtag* a palabras convencionales del inglés) y extraer el significado.

```
In: #NotMyPresident
Out: <hashtag> not my president </hashtag>
```

Figura 10: Segmentación y anotaciones en la tokenización

Pero si nos limitásemos a sustituir los *hashtags* por su forma expandida en lenguaje natural dentro del texto, estaríamos perdiendo una pieza importante: el contexto dentro del tweet. Es decir, para mantener el significado semántico de una frase, tenemos que anotar de alguna manera que esas palabras provienen de un *hashtag*. Aquí es donde entran las anotaciones

previas a la tokenización. En nuestra implementación, los siguientes objetos son anotados con los tokens especiales:

- Hashtags: como se ilustra en la Figura 11
- palabras en mayúsculas: "DIFFERENT" → <allcaps> different </allcaps>
- palabras "alargadas": "noooooooo" → no <elongated>
- repetición de caracteres: "yes!!!!!!!" → yes ! <repeated>

La segunda operación consiste en tokenizar el texto teniendo en cuenta que tenemos otros objetos en el texto que no son palabras de la lengua inglesa, pero tienen información semántica dentro de la frase. Estos objetos son normalizados con un token especial. Pertenecen a este grupo:

- URLs, *handles* de usuario
- direcciones de email, teléfonos
- onomatopeyas, emojis
- símbolos de bolsa, símbolos de dinero, porcentajes
- fechas, horas

```
In: The new #johndoe movie suuucks!!! WAISTED $10 #badmovies :/
Out: The new <hashtag> john doe </hashtag> movie sucks <elongated> !
<repeated> wasted <allcaps> <money> . <repeated> <hashtag> bad
movies </hashtag> <annoyed>
```

Figura 11: Ejemplo de transformaciones de anotación y normalizacion con tokens

Tras aplicar ambas operaciones descritas, aplicamos una tokenización simple que consiste en extraer cada palabra o token especial del texto a una lista.

Esta lista de tokens reemplazará al campo full_text en cada uno de los tweets que pasan por el pipeline y finalmente son almacenados en una nueva colección de la base de datos llamada "tw_norm" que contendrá los tweets con todos los atributos normalizados.

3.1.3 Gestor de modelos

Siguiendo la filosofía de un diseño modular en el que se separa la colecta de datos, el entrenamiento y la clasificación, el gestor de modelos es un componente independiente que sirve a modo de interfaz con el motor de TensorFlow a través de su API de alto nivel [61], así como de gestor de los modelos utilizados. Principalmente realiza las funciones de:

- **Definición, guardado y carga de los modelos**
  - Durante la realización de este trabajo se han definido y probado distintos modelos de entrenamiento y clasificación. Con el fin de poder reproducir los resultados obtenidos, una vez definidos y/o entrenados los modelos vía API se etiquetan y almacenan en formato serializado en disco.
- **Entrenamiento**
  - Dado que el proceso de normalización de los datos es independiente del entrenamiento, el gestor de modelos puede en cualquier momento tomar una muestra de la colección de datos normalizados de la colección "tw_norm" y entrenar o reentrenar uno de los modelos almacenados.
- **Clasificación**

- Teniendo en mente la lista de modelos disponibles en disco, la función de clasificación se comporta como una caja negra: pasamos por parámetro el conjunto de tweets que queremos clasificar junto con el modelo almacenado y a continuación se genera la batería de resultados o *insights* sobre la clasificación que ha tenido lugar, tales como la salida de log o las gráficas de resultados.

3.2 Limitaciones de las tecnologías utilizadas

A la hora de implementar el diseño descrito, se han encontrado algunas limitaciones u observaciones que es importante recalcar tanto para justificar algunas decisiones de diseño como de cara a la reutilización de este trabajo.

Las más destacables son las que se describen a continuación:

- A la hora de desplegar el colector de datos en una instancia se hizo el primer intento de utilizar instancias en la nube EC2 de AWS de tipo "Free tier" [62]. La principal ventaja de este tipo de instancias es que proporcionan 750 horas de uso gratuito al mes (con ciertas restricciones). El problema es que ya hay otros usuarios con la misma idea y Twitter parece haber filtrado directamente las peticiones a su API desde el rango de IPs que se asignan a este tipo de instancias de AWS. En cualquiera de las regiones que se probó se obtenía el mismo resultado.

```
(env) [ec2-user@ip-172-31-6-144 twitter-scraper]$ python
Python 3.7.6 (default, Feb 26 2020, 20:54:15)
[GCC 7.3.1 20180712 (Red Hat 7.3.1-6)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from twitter_scraper import get_trends
>>> get_trends()
<Response [200]>
{}
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/ec2-user/Code/twitter-scraper/twitter_scraper/modules/trends.py", line 20, in get_trends
    html = html.json()["module_html"]
KeyError: 'module_html'
>>> exit()
(env) [ec2-user@ip-172-31-6-144 twitter-scraper]$ cur^C
(env) [ec2-user@ip-172-31-6-144 twitter-scraper]$ curl https://twitter.com/i/trends
Too Many Requests(env) [ec2-user@ip-172-31-6-144 twitter-scraper]$
```

Figura 12: Timeouts peticiones a la API de Twitter

- Aunque no se especifica en los límites de la API [37], se ha comprobado que al realizar un gran volumen de peticiones a mayor velocidad de una petición HTTP por segundo, la API devuelve *timeouts* al cabo de un tiempo indeterminado. Manteniendo la velocidad a una petición por segundo no existe ese problema.
- La API ofrece el *endpoint "*trends/place" para extraer los *trending topic* actuales de la red social, pero permite extraer un histórico de los *trending topic* en los últimos N días. Es por esta misma razón que se ha optado por sacar los tweets asociados a un *topic* en tiempo real frente a utilizar el histórico que sí permite el *endpoint* de tweets.
- Existe un límite no documentado del número efectivo de tweets que se pueden extraer del endpoint "search/tweets" cuando se pasan por parámetro filtros en la búsqueda. Por esta razón no se extrae nunca el número teórico de tweets que podríamos extraer por ventana de tiempo.
- No es posible extraer el número de respuestas que ha tenido un tweet determinado. Esta métrica que mide parte del *engagement* de un *tweet* podría ser muy interesante como parte del vector de atributos.

- Uno de los atributos que compone cada uno de los objetos *trending topic* que extraemos es el `tweet_volume`. Este atributo nos dice cuántos tweets contienen embebidos en su texto el *topic* en cuestión. Por alguna razón no documentada, cuando un *trending topic* está escalando posiciones rápidamente en el ranking de popularidad su valor es nulo. Como extraemos los *trending topic* cada 15 minutos por defecto, en cuanto ese *topic* se "enfría" en cuanto a interacciones, nos muestra el volumen acumulado.

3.3 Clasificación

En nuestra colección de tweets normalizados tenemos atributos de dos tipos, numéricos y texto. Los atributos numéricos contienen todas las métricas acerca del tweet más allá del contenido. Los atributos de texto, el `full_text` y `relevant_topic`, nos dan toda la información semántica del tweet.

Para comprobar cuánto influyen estos dos conjuntos de datos a la hora de determinar la popularidad de un tweet, se han definido 3 topologías de modelo sobre las que haremos pruebas de entrenamiento y clasificación.

Existen una serie de consideraciones y parámetros que serán comunes a todos los modelos:

- El paso de convertir el texto normalizado de cada tweet tal y como sale de nuestro *pipeline NLP* a una representación numérica que entienda nuestro clasificador se realiza en el momento previo al entrenamiento. De esta manera, podemos probar a utilizar distintos modelos estadísticos para generar los *word embeddings* sin tener que actualizar en nuestra base de datos de *tweets* los vectores.
- Para generar la matriz de *word embeddings* utilizaremos de nuevo vectores de palabras entrenados con GloVe. En concreto, usaremos un modelo [63] entrenado con:
    - 2 billones de tweets
    - 27 billones de tokens
    - 1.2 millones de palabras en el vocabulario
    - Vectores de 200 dimensiones
- Tomaremos 200 como el máximo número de palabras por tweet
- La matriz de *word embeddings* tendrá 200 filas, una por cada palabra (dimensión) del vocabulario del modelo de GloVe que estamos cargando.
- La función de activación [66] de la capa de salida de nuestro clasificador será "softmax".
- La función de pérdida [65] o *loss function* será "categorical_crossentropy" y nuestra función de optimización será "adam".

3.3.1 Modelo basado en texto del tweet

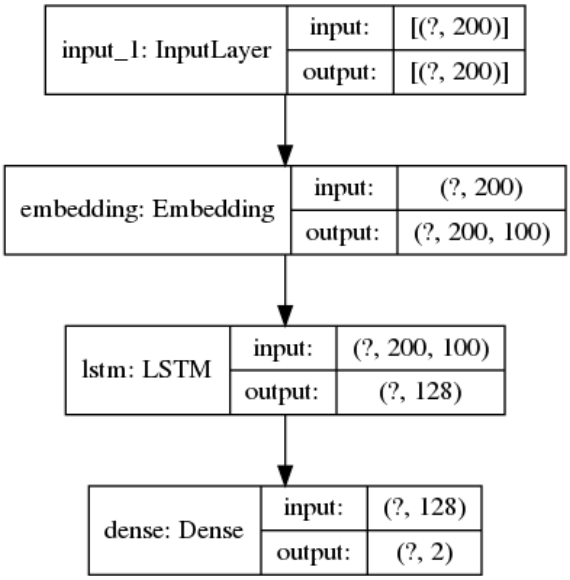

Figura 13: Modelo para clasificación de tweets basado en texto únicamente

Este modelo constará de:

- Una capa de entrada o *embedding layer*
- Una capa LSTM con 128 neuronas
- Una capa de 2 salidas de clasificación

3.3.2 Modelo basado en metadatos del tweet

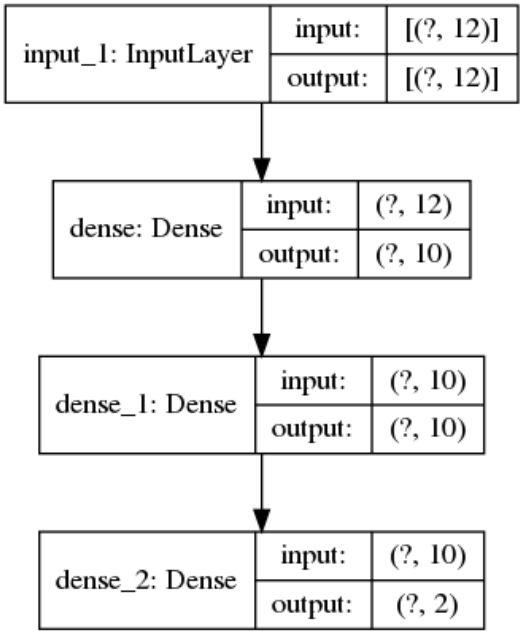

Figura 14: Modelo para clasificación de tweets basado en metadatos únicamente

Este modelo constará de:

- Una capa de entrada
- Dos capas ocultas de 10 neuronas cada una usando la función de activación “ReLU” [66]

- Una capa de salida de 2 neuronas y función de activación “softmax” [66].

3.3.3 Modelo mixto

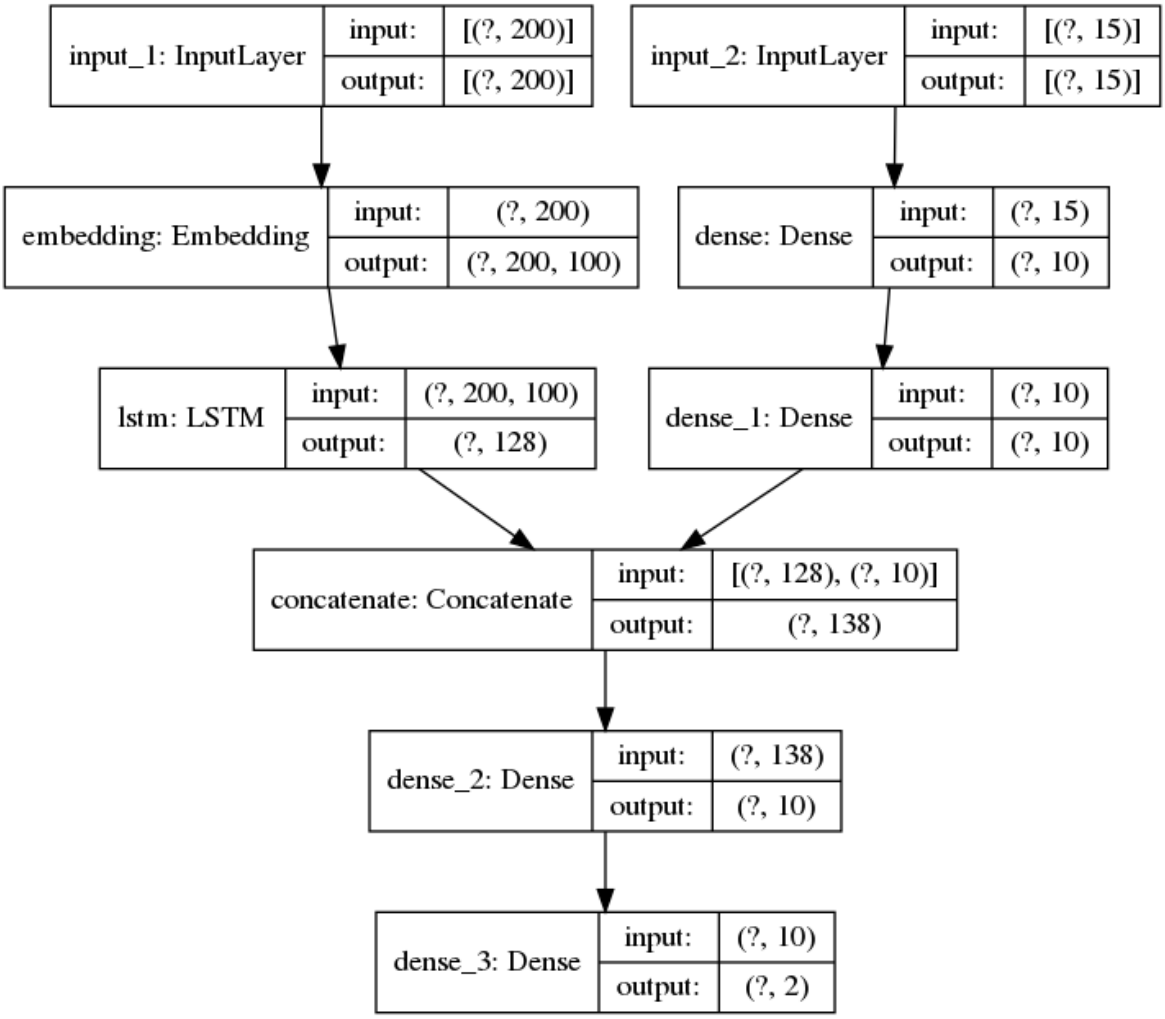

Figura 15: Modelo mixto para clasificación de tweets

Para combinar el texto con los atributos numéricos, usaremos dos submodelos:

- El primer submodelo toma como entrada los *embeddings* de los tweets. Estará formado por una capa de entrada, una capa de *embedding* y una LSTM de 128 neuronas.
- El segundo submodelo toma como inputs los distintos atributos numéricos del tweet que hemos normalizado. Consta de una capa de input y dos capas ocultas.
- La salida del primer submodelo y la salida del segundo modelo son concatenadas y el resultado se usará como un input conjunto de una capa oculta de 10 neuronas.
- Finalmente, tenemos una capa de salida de 2 neuronas con la clasificación final.
- Las capas ocultas utilizan una función de activación “ReLu”.

4 Integración, pruebas y resultados

4.1 Problemas encontrados y resolución

En esta sección se detallan los principales problemas encontrados durante la fase de pruebas y obtención de resultados de este trabajo, más allá de las limitaciones específicas de la tecnología del apartado 3.2.

4.1.1 Pérdidas de información semántica de los tweets

Debido al creciente uso de objetos multimedia en Twitter y las herramientas que la red social pone a disposición de los usuarios en constante evolución, se ha encontrado un gran volumen de *tweets* de los cuales no podemos extraer un significado semántico cuando están compuestos únicamente de vídeo o imágenes. El tratamiento de este tipo de contenido se salía del alcance de este trabajo, pero se ha tenido en cuenta filtrándolos del dataset de entrenamiento y test: si tienen multimedia y el texto son sólo URLs de multimedia, se descartan.

4.1.2 Espacio de almacenamiento para recopilar dataset

La API especifica que el tamaño máximo de un tweet en formato JSON son 5KB/tweet. Si se toma como referencia la API estándar descrita anteriormente en esta memoria y se asume que queremos extraer el mayor número posible de tweets en el tiempo: con un límite de 180 peticiones cada 15 minutos y 100 tweets por petición máximos, esto suma 18000 tweets cada 15 minutos. Es decir, el volumen máximo son 360 MB/h. Durante una semana de recopilación de datos esto suma unos 60GB aproximadamente.

Como los recursos hardware para desarrollar este trabajo eran limitados y se ha tenido en mente la reusabilidad de lo desarrollado, no se almacenan todos los datos y metadatos asociados a un tweet. La principal fuente de los 5KB de tamaño máximo por tweet son otros tweets embebidos para los casos en los que el tweet en sí es un *retweet*, es decir un tweet nuevo generado a partir de exactamente la misma información de otro. Esto también ocurre con los tweets que son respuesta a otros tweets. Estos dos conjuntos de tweets son descartados desde la ingesta de datos.

4.1.3 Codificación UTF-8 del texto

El texto contenido en los tweets extraídos a través de la API viene codificado en UTF-8, la codificación de caracteres más extendida en la red. Así mismo, el entrenamiento de los modelos con estadísticas de palabras utilizados en este trabajo ha sido realizado por sus mantenedores con la misma codificación. Sin embargo, se ha observado que a la hora de la tokenización y expansión de contracciones las librerías diseñadas para ello [59][68] tienen problemas si los delimitadores como las apostrofes, los puntos o los guiones no están en el rango de los 8 bits de la tabla ASCII.

Tras realizar muchas observaciones se ha integrado en el pipeline *NLP* una función que realiza una sustitución de estos caracteres si se encuentran presentes en codificación UTF-8 para poder procesar correctamente el texto.

4.2 Entrenamiento y test

Algunas observaciones sobre la metodología seguida para la fase de entrenamiento y test:

- A día de 07/07/2020 la base de datos de tweets normalizados contenía 1.225.415 tweets de los cuales 110.543 pertenecen a la categoría de “populares”.
- Para los sets de tweets de entrenamiento y test tomamos un 80% de los tweets de *train* y un 20% de *test*.
- Se han tomado muestras aleatorias de 100.000 tweets formadas por 50.000 tweets “populares” y 50.000 tweets “no populares”.
- Cada día se almacenaban tweets nuevos en la base de datos y se tomaba de nuevo una muestra de 100.000 tweets.
- Para comprobar la consistencia de los modelos en un periodo continuado de tiempo, durante una semana se han realizado 210 clasificaciones en total, 30 clasificaciones por modelo.
- Sacando la media, desviación típica y coeficiente de variación de los resultados de cada día por clasificador hemos comprobado la variación en la precisión de los clasificadores a la hora de clasificar los tweets de test como “populares” o “no populares”.
- Los parámetros y atributos de los tweets reflejados en el diseño componen la versión definitiva del modelo de clasificación que hemos desarrollado pero también se han realizado pruebas con variaciones en ellos y se describen los resultados en el apartado de conclusiones.

A continuación, se muestran resultados de clasificaciones donde el *model accuracy* y *model loss* se aproximan más a la mediana de los resultados obtenidos durante este periodo de pruebas.

4.2.1 Resultados modelo basado en texto del tweet

```
Epoch 96/100
100/100 [==============================] - 28s 277ms/step - loss: 0.6932 - acc: 0.4977 - val_loss: 0.6931 - val_acc: 0.5109
Epoch 97/100
100/100 [==============================] - 28s 277ms/step - loss: 0.6932 - acc: 0.4934 - val_loss: 0.6932 - val_acc: 0.4891
Epoch 98/100
100/100 [==============================] - 28s 277ms/step - loss: 0.6932 - acc: 0.4966 - val_loss: 0.6932 - val_acc: 0.4891
Epoch 99/100
100/100 [==============================] - 28s 281ms/step - loss: 0.6932 - acc: 0.4970 - val_loss: 0.6932 - val_acc: 0.4891
Epoch 100/100
100/100 [==============================] - 28s 281ms/step - loss: 0.6932 - acc: 0.5013 - val_loss: 0.6932 - val_acc: 0.4891
125/125 [==============================] - 5s 41ms/step - loss: 0.6931 - acc: 0.5048
```

Figura 16: Output entrenamiento-test de modelo basado en texto

En la figura 17 se puede apreciar que el *training accuracy* final es de un 50.13% y el *validation accuracy* es de un 48.91%, lo que nos da que pensar que nuestro modelo no está haciendo *overfitting* sobre los datos de entrenamiento.

4.2.2 Resultados modelo basado en metadatos del tweet

```
Epoch 96/100
150/150 [==============================] - 34s 227ms/step - loss: 0.4475 - acc: 0.8353 - val_loss: 0.4481 - val_acc: 0.8350
Epoch 97/100
150/150 [==============================] - 34s 227ms/step - loss: 0.4476 - acc: 0.8353 - val_loss: 0.4479 - val_acc: 0.8350
Epoch 98/100
150/150 [==============================] - 34s 227ms/step - loss: 0.4477 - acc: 0.8353 - val_loss: 0.4479 - val_acc: 0.8350
Epoch 99/100
150/150 [==============================] - 34s 230ms/step - loss: 0.4477 - acc: 0.8353 - val_loss: 0.4482 - val_acc: 0.8350
Epoch 100/100
150/150 [==============================] - 34s 229ms/step - loss: 0.4477 - acc: 0.8353 - val_loss: 0.4479 - val_acc: 0.8350
188/188 [==============================] - 7s 38ms/step - loss: 0.4628 - acc: 0.8258
Test Score: 0.46275627613067627
Test Accuracy: 0.8258333206176758
```

Figura 17: Output entrenamiento-test de modelo basado en metadatos del tweet

En la figura X se puede apreciar que el *training accuracy* final es de un 83.53% y el *validation accuracy* es de un 83.50%, lo que nos da que pensar que nuestro modelo no está haciendo *overfitting* sobre los datos de entrenamiento.

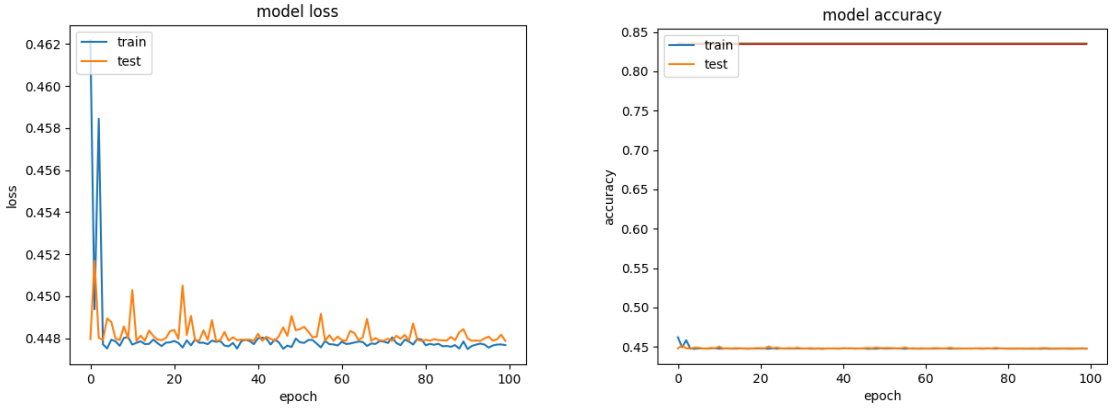

Figura 18: Model loss y model accuracy para el modelo basado en metadatos

4.2.3 Resultados modelo mixto

```
Epoch 96/100
1140/1140 [==============================] - 1s 910us/step - loss: 0.1067 - acc: 0.9622 - val_loss: 0.0881 - val_acc: 0.9686
Epoch 97/100
1140/1140 [==============================] - 1s 898us/step - loss: 0.1226 - acc: 0.9546 - val_loss: 0.1035 - val_acc: 0.9621
Epoch 98/100
1140/1140 [==============================] - 1s 910us/step - loss: 0.1083 - acc: 0.9617 - val_loss: 0.0980 - val_acc: 0.9636
Epoch 99/100
1140/1140 [==============================] - 1s 901us/step - loss: 0.1059 - acc: 0.9628 - val_loss: 0.0903 - val_acc: 0.9671
Epoch 100/100
1140/1140 [==============================] - 1s 896us/step - loss: 0.1034 - acc: 0.9626 - val_loss: 0.0902 - val_acc: 0.9689
225/225 [==============================] - 0s 781us/step - loss: 0.0996 - acc: 0.9661
Test Score: 0.09963607788085938
Test Accuracy: 0.9661111235618591
```

Figura 19: Output entrenamiento-test de modelo mixto

En la figura se puede apreciar que el *training accuracy* final es de un 96.26% y la *validation accuracy* es de un 96.89%, lo que nos da que pensar que nuestro modelo no está haciendo *overfitting* sobre los datos de entrenamiento.

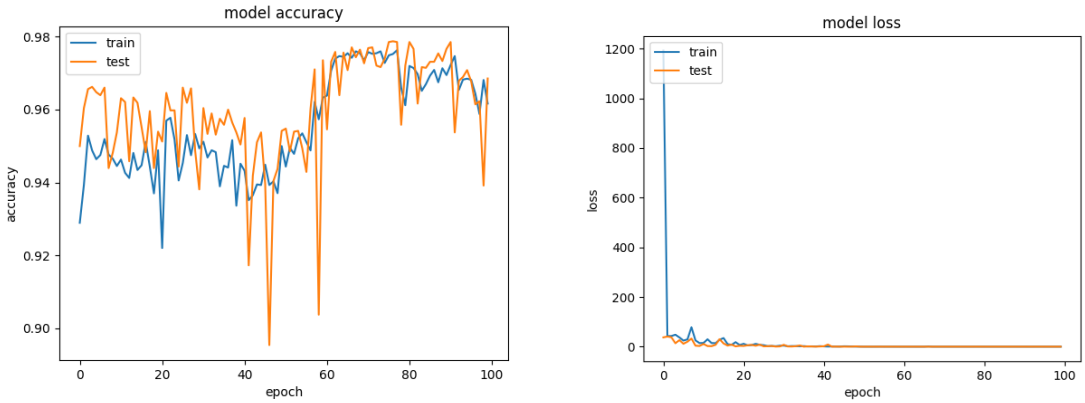

Figura 20: Model loss y model accuracy para el modelo mixto

5 Conclusiones y trabajo futuro

5.1 Conclusiones

Sobre los resultados de las pruebas realizadas se pueden sintetizar las siguientes conclusiones:

- El modelo de clasificación basado en atributos numéricos de los tweets tiene una tasa de acierto mayor que el modelo de clasificación basado puramente en texto. Esto se debe probablemente a que es mucho más fácil inferir el impacto en popularidad de un tweet mirando métricas como los favoritos, *retweets* o seguidores que por el contenido del tweet.
- Si se eliminan del vector de atributos de un tweet los atributos `retweet_count` y `favorite_count` para la clasificación de su impacto, apreciable sobre la clasificación a la hora de medir su impacto.
- El modelo de clasificación mixto que combina métricas sobre el usuario y el *tweet* con el significado semántico del texto mejora la tasa de acierto del clasificador basado puramente en métricas. Esto significa que es posible modelar una proporción del impacto de un *tweet* por su significado semántico.
- Aunque en esta memoria se han plasmado las transformaciones definitivas del pipeline *NLP*, se ha llevado a cabo un estudio previo del estado del arte de la formalización del lenguaje natural y a base de pruebas preliminares se han descartado transformaciones como la lematización o el *PoS tagging* puro en el pipeline final en favor a una anotación y tokenización más específica y entrenada con estadísticas para nuestro caso de uso.
- A la hora de elegir vectores de palabras preentrenados para las distintas fases de la normalización, se ha comprobado que se obtienen mejores resultados si el *corpus* de origen se adecúa a nuestro caso de uso (collecciones de Tweets) que, si están formados por colecciones de texto más ricas, pero más genéricas (Google News o Wikipedia como *corpus* único).
- El entrenamiento sobre los atributos numéricos del tweet es de media 30 veces más rápido que el entrenamiento sobre el texto de los tweets.

Sobre el trabajo desarrollado, se ha conseguido definir e implementar un framework para generar un dataset de tweets normalizados de manera modular. Futuros trabajos académicos que se hagan sobre esta red social podrán hacer uso del trabajo de análisis de normalización que se ha llevado a cabo para generar datasets de manera sencilla.

Se ha definido un pipeline de normalización específico para Twitter que recopila las técnicas más novedosas del procesamiento del lenguaje natural haciendo uso de modelos estadísticos probados.

Por último, se ha definido un modelo de clasificación mixto que, combinando las métricas y metadatos de un tweet con el significado léxico y semántico de su texto, es capaz de predecir su impacto sobre un *trending topic*. Esta clasificación en combinación con otras clasificaciones realizadas por anteriores trabajos académicos, nos puede permitir entender mejor los patrones de comportamiento humano en las redes sociales y adelantarnos a los cambios de tendencias con fines sociales, como la prevención de delitos de odio. También resultará interesante a la hora de detectar campañas que tengan el fin de manipular el discurso político en las redes.

5.2 Trabajo futuro

5.2.1 Escalar recopilación datos para obtener un dataset más rico

La principal ventaja del enfoque se ha seguido para la recopilación de datos es que está totalmente desacoplada de el pipeline *NLP* y el clasificador. Las pruebas realizadas para su desarrollo han tenido lugar en una instancia única donde todos los procesos corrían en la misma, pero existe la posibilidad de lanzar varios colectores de datos con distintas API keys apuntando a la misma base de datos y multiplicar el volumen diario de datos que podemos sacar de la red social.

El poder generar rápidamente un dataset genérico y desacoplado de un caso de uso particular puede solucionar fácilmente la gran problématica con los datasets existentes de Twitter a la de realizar un nuevo trabajo académico.

5.2.2 Desarrollar una API de clasificación en tiempo real

Dentro de los términos de uso de Twitter está contemplado el construir aplicaciones de terceros que hagan uso de sus datos.

Teniendo en cuenta siempre dichos límites, una evolución interesante sería construir una API REST entorno al clasificador de manera que usuarios que puedan predecir el impacto de un tweet antes de enviarlo o a posteriori para predecir si tendrá impacto o no sobre un trending topic. Este caso de uso es particularmente interesante a la hora de realizar sondeos del interés público en algún tema en particular, por ejemplo.

5.2.3 Análisis en profundidad de los parámetros que influyen en el impacto

Una vez desarrollado un primer modelo para predecir el impacto de un tweet en un trending topic, sería interesante como trabajo posterior realizar un estudio exhaustivo de qué atributos son los que más influyen en dicho impacto. Este potencial trabajo contaría de dos tareas de interés.

Por una parte, expandir todavía más el vector de atributos de un tweet que hemos desarrollado en este trabajo con otros atributos calculados. Una posible vía de esto podría ser comenzar a hacer análisis sobre los elementos multimedia embebidos en los tweets y completar aún más el significado semántico. Hay mucho trabajo por hacer en este aspecto y puede ser cada vez más relevante a medida que se haga uso más extendido de estos elementos para comunicarnos.

Por otra parte, realizar un análisis estadístico sobre el vector de atributos que se ha desarrollado y optimizarlo para eliminar los atributos que no influyan sobre el impacto y expandir aquellos que sí lo hacen.

Referencias

[1] KEPM, S. (2020) *Digital 2020: Global Digital Overview*, DATAREPORTAL. https://datareportal.com/reports/digital-2020-global-digital-overview

[2] Twitter IR (2020) *Q1 2020 Letter to Shareholders*. Twitter. https://s22.q4cdn.com/826641620/files/doc_financials/2020/q1/Q1-2020-Shareholder-Letter.pdf

[3] El Azab, A; Idrees, A.M ;Mahmoud, M.A; Hefny, H. (2015) *Fake Account Detection in Twitter Based on Minimum Weighted Feature set*. Zenodo. https://zenodo.org/record/1110582

[4] Nagajothi, B.; Priyadarsini, R.J. (2019) *Sentiment Analysis on Twitter Dataset using R Language*. Zenodo https://zenodo.org/record/3587699

[5] Hedaoo, K. ; Bhoyar, H. ; Gaikwad, S. ; Matey, V (2020) *Twitter Tweets Analysis using Python*. Zenodo https://zenodo.org/record/3793295

[6] Chawla, S.;Anchal Garg, P.; Jha, B. (2019*) Review on Sentiment Analysis of Twitter Data with Some Classifiers Ensembles*. Zenodo. https://zenodo.org/record/2620789

[7] Verma, N.; Gupta, G. (2018) *Sentiment analysis of real time social tweets*. Zenodo. https://zenodo.org/record/1400407

[8] El Azab, A; Idrees, A.M ;Mahmoud, M.A; Hefny, H. (2015) *Fake Account Detection in Twitter Based on Minimum Weighted Feature set*. Zenodo. https://zenodo.org/record/1110582

[9] Anand, N.; Kumar, T. (2017) *Twitter analytics and visualization using R*. Zenodo https://zenodo.org/record/829749

[10] Sentiment Analysis on Twitter Dataset using R Language https://zenodo.org/record/3587699

[11] Lee, K.; Palsetia, D.; Narayanan, R.; Patwary, M.A.; Agrawal, A; Choudhary, A. (2011) *Twitter Trending Topic Classification*. 1th IEEE International Conference on Data Mining Workshops. http://cucis.ece.northwestern.edu/publications/pdf/LeePal11.pdf

[12] Yegin Genc, Y.S.; Nickerson, J.V. (2011) *Discovering context: Classifying tweets through a semantic transform based on Wikipedia*. Proceedings of HCI International. https://link.springer.com/chapter/10.1007/978-3-642-21852-1_55

[13] Kinsella, S.; Passant, A.; Breslin, J.G. (2011) *Topic classificationin social media using metadata from hyperlinked objects*. Proceedings of the 33rd European conference on Advances in information retrieval. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.472.2116&rep=rep1&type=pdf

[14] Zubiaga, A.; Spina, D.; Fresno, V.; Martínez, R. (2014) *Real-Time Classification of Twitter Trends*. Journal of the American Society for Information Science and Technology (JASIST). https://arxiv.org/pdf/1403.1451.pdf

[15] Hadeer, A.*;* Traore, I.; Saad, S. (2017); *Detecting opinion spams and fake news using text classification*. Wiley. https://onlinelibrary.wiley.com/doi/pdf/10.1002/spy2.9

[16] Zhang, J.; Dong, B.; Yu, P.S. (2019) *FAKEDETECTOR: Effective Fake News Detectionwith Deep Diffusive Neural Network* . ArXiv. https://arxiv.org/pdf/1805.08751.pdf

[17] Shao, C.; Ciampaglia, G.L.; Flammini, A.; Menczer, F. (2016) *Hoaxy: A Platform for Tracking Online Misinformation*. Cornell University. https://arxiv.org/abs/1603.01511

[18] Pereira-Kohatsu, J.C.; Quijano-Sánchez, L.; Liberatore, F.; Camacho-Collados, M. (2019) *Detecting and analyzing hate speech in Twitter: HaterNet a system in the Spanish prevention of hate crime office*. MDPI. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6864473/
[19] Mittal, A.; Goel, A. (2011) *Stock Prediction Using Twitter Sentiment Analysis*. Stanford University. https://pdfs.semanticscholar.org/4ecc/55e1c3ff1cee41f21e5b0a3b22c58d04c9d6.pdf
[20] Kordonis , A.; Symeonidis ; S.; Arampatzis, A. (2016) *Stock Prediction Using Twitter Sentiment Analysis*. The 20th Panhellenic Conference on Informatics. https://www.researchgate.net/publication/311843931_Stock_Price_Forecasting_via_Sentiment_Analysis_on_Twitter
[21] Forelle, M.; Howard, P.; Monroy-Hernández, A.; Saiph, S. (2015) *Political Bots and the Manipulation of Public Opinion in Venezuela* . SSRN Electronic Journal https://www.researchgate.net/publication/280538627_Political_Bots_and_the_Manipulation_of_Public_Opinion_in_Venezuela
[22] Badawy, A.; Ferrara, E.; Lerm, K. (2018) *Analyzing the Digital Traces of Political Manipulation: The 2016 Russian Interference Twitter Campaign.* IEEE https://ieeexplore.ieee.org/abstract/document/8508646
[23] Lokot, T. ; Diakopoulos, N. (2015) *Automating news and information dissemination on Twitter*. Taylor and Francis. https://www.tandfonline.com/doi/abs/10.1080/21670811.2015.1081822
[24] Twitter (2020) *Content redistribution. Developer Agreement and Policy.* Twitter Developer. https://developer.twitter.com/en/developer-terms/agreement-and-policy
[25] Vered1986 (2018) *Chirps. Predicate Paraphrases From Twitter*. GitHub. https://github.com/vered1986/Chirps/
[26] Taspinar (2016) *Twitterscrapper*. GitHub. https://github.com/taspinar/twitterscraper
[27] Shaypal5 (2018) *awesome-twitter-data*. GitHub. https://github.com/shaypal5/awesome-twitter-data
[28] Lerman, K. (2010) *Lerman Twitter 2010 Dataset*. ISI. http://academictorrents.com/details/d8b3a315172c8d804528762f37fa67db14577cdb
[29] Lerman, K. (2010) *Lerman Twitter 2010 Dataset*. ISI. http://academictorrents.com/details/d8b3a315172c8d804528762f37fa67db14577cdb
[30] Thepanacelab (2020*) covid19_twitter*. Twitter. https://github.com/thepanacealab/covid19_twitter
[31] Freelon, D. (2012) *Arab Spring Twitter data now available (sort of)*. Dfreelon.org. http://dfreelon.org/2012/02/11/arab-spring-twitter-data-now-available-sort-of/
[32] Twitter Social Graph 2012. Social Clicks 2016. soTweet: Studying Twitter at Scale https://dl.acm.org/doi/10.1145/2591971.2591985
[33] Baio, A. (2014) *72 Hours of #Gamergate*. Medium. https://medium.com/message/72-hours-of-gamergate-e00513f7cf5d
[34] Twitter (2020) *Topics: Behind the Tweets*. Twitter. https://blog.twitter.com/en_us/topics/product/2020/topics-behind-the-tweets.html
[35] Twitter (2020) *Apply for access.* Twitter. https://developer.twitter.com/en/apply-for-access
[36] Twitter (2017) *Using Deep Learning at Scale in Twitter's Timelines*. Twitter. https://blog.twitter.com/engineering/en_us/topics/insights/2017/using-deep-learning-at-scale-in-twitters-timelines.html
[37] Twitter (2020) *Rate limiting.* Twitter API docs. https://developer.twitter.com/en/docs/basics/rate-limiting

[38] Twitter (2020) *Tweet objects.* Twitter API docs. https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object
[39] Twitter (2020) *Twitter IDs (snowflake).* Twitter API docs. https://developer.twitter.com/en/docs/basics/twitter-ids
[40] De, S.S.; Dehuri, S.; Wang, G.N. (2012) *Machine Learning for Social Network Analysis: A Systematic Literature Review*. Research Gate. https://www.researchgate.net/publication/251236864_Machine_Learning_for_Social_Network_Analysis_A_Systematic_Literature_Review
[41] Shalev-Shwartz, S.; Ben-David, S. (2014) *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press. https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf
[42] Layton, R. (2017*) Learning Data Mining with Python*. Packt Publishing. https://www.oreilly.com/library/view/learning-data-mining/9781787126787/
[43] Verma, N. y Gupta, G. (2018) *Sentiment analysis of real time social tweets*, Global Journal of Engineering Science and Researches.
[44] Halevy, A.; Norvig, P.; Pereira, F. (2009) *The unreasonable effectiveness of data.* IEEE Intelligent Systems.
[45] Banko, M.; Brill, E. (2001*) Scaling to very very large corpora for natural language disambiguation*. ACL
[46] Benamara, F., Irit, S., Cesarano, C., Federico, N., and Reforgiato, D. (2007). *Sentiment Analysis: Adjectives and Adverbs are better than Adjectives Alone*. In Proc of Int Conf on Weblogs and Social Media.
[47] Lee, K., Palsetia,D.; Narayanan, R.; Patwary, M.A.; Agrawal, A.; Choudhary, A. (2011) *Twitter Trending Topic Classification*, 11th IEEE International Conference on Data Mining Workshops.
[48] Kolchyna, O.; Souza. T.T.P.; Treleaven, P.C; Aste, T. (2015) *Twitter Sentiment Analysis: Lexicon Method, MachineLearning Method and Their Combination.* Arxiv. https://arxiv.org/pdf/1507.00955.pdf
[49] Garbade, M.J. (2018) *A Simple Introduction to Natural Language Processing*. Becoming Human. https://becominghuman.ai/a-simple-introduction-to-natural-language-processing-ea66a1747b32
[50] Pai, A. (2020) *What is Tokenization in NLP? Here's All You Need To Know*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/
[51] Brownlee, J. (2017) *A Gentle Introduction to the Bag-of-Words Model*. Machine Learning Mastery. https://machinelearningmastery.com/gentle-introduction-bag-words-model/
[52] Ian Beaver (2019) *PyContractions.* GitHub https://github.com/ian-beaver/pycontractions
[53] Goldberg, Y. (2017) *Neural Network Methods for Natural Language Processing*. Morgan and Claypool. https://books.google.es/books/about/Neural_Network_Methods_in_Natural_Langua.html?id=8wklDwAAQBAJ&printsec=frontcover&source=kp_read_button&redir_esc=y#v=onepage&q&f=false
[54] Karani, D. (2018) *Introduction to Word Embedding and Word2Vec*. Towards data science. https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa
[55] Pathmind (2020) *A Beginner's Guide to Word2Vec and Neural Word Embeddings.* Pathmind. https://pathmind.com/wiki/word2vec

[56] Pennington, J. (2014) *GloVe: Global Vectors for Word Representation*. ACL. https://nlp.stanford.edu/pubs/glove.pdf
[57] Russel, M.A. (2014) *Mining the Social Web*. O'Reilly. https://www.webpages.uidaho.edu/~stevel/504/Mining-the-Social-Web-2nd-Edition.pdf
[58] Baziotis,C.; Pelekis,N.; Doulkeridis, C. (2017*) DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis.* ACL. https://www.aclweb.org/anthology/S17-2126/
[59] Cbaziotis (2019) *Ekphrasis.* GitHub https://github.com/cbaziotis/ekphrasis
[60] Peter Norvig. How to Write a Spelling Corrector http://norvig.com/spell-correct.html
[61] What are Symbolic and Imperative APIs in TensorFlow 2.0?. TensorFlow https://blog.tensorflow.org/2019/01/what-are-symbolic-and-imperative-apis.html
[62] Free Tier limits. AWS docs https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/free-tier-limits.html
[63] Glove. Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 200d vectors, 1.42 GB download) http://nlp.stanford.edu/data/wordvecs/glove.twitter.27B.zip
[64] https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/
[65] Jason Brownlee (2019). Loss and Loss Functions for Training Deep Learning Neural Networks https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/
[66] Jason Brownlee (2019). A Gentle Introduction to the Rectified Linear Unit (ReLU) https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/
[67] Keras API reference https://keras.io/api/layers/activations/#softmax-function
[68] spaCy 101: Everything you need to know https://spacy.io/usage/spacy-101

Glosario

API	Interfaz de Programación de Aplicaciones en español. Se trata de un conjunto de rutinas, protocolos y herramientas utilizadas para la creación de aplicaciones de software y especificar cómo deben interactuar los componentes de dicho software.
Bot	Software o programa informático que realiza tareas repetitivas a través de Internet y es capaz de interactuar con sistemas o usuarios.
Emojis	Ideogramas utilizados en mensajes electrónicos y sitios web que representan de manera visual una emoción.
Engagement	En Twitter, se trata del número total de veces que un usuario interactúa con una publicación.
Feedback	Es la transmisión de información evaluativa o correctiva sobre una acción, evento o proceso a la fuente original o de control.
Feedback	Es la transmisión de información evaluativa o correctiva sobre una acción, evento o proceso a la fuente original o de control.
Handle	Conjunto de caracteres precedidos por una almohadilla (@) que sirve para identificar de manera unívoca a alguien en Twitter.
In-place	En el mismo lugar. Una modificación in-place de texto implica que no se genera un nuevo objeto con las modificaciones.
Opinion mining	También conocido como análisis de opinión, se refiere al uso del procesamiento del lenguaje natural, el análisis de texto y la lingüística computacional para identificar y extraer información.
PascalCase	Tipo de escritura en la cual primer carácter de una palabra se encuentra en mayúscula, mientras que el resto de caracteres están en minúsculas.
Scrapping	Técnica utilizada para extraer información de sitios web haciendo uso de software.
Trend	En redes sociales, se trata de una tendencia, de lo que es popular en un periodo de tiempo determinado.
Tweet	Publicación realizada en la red social Twitter.

Anexos

A Manual de instalación

A continuación, se especifican los requerimientos y pasos para poder hacer uso del código desarrollado extraídos del fichero README.

Este trabajo ha sido desarrollado y probado sobre Ubuntu 18.04 LTS

Las dependencias de instalación necesarias son:

- `python 3.6.9`, `3.8.3`
- `mongodb v4.2.0`, `4.2.8`
- `virtualenv 15.1.0`, `20.0.21`

En cuanto a la configuración, se deberán hacer uso de los siguientes ficheros:

- `config/config.yml`
  - Main parameters such as log names, mongodb collections names, etc
- `private/private.yml`
  - Secrets such as API keys
- `twimclassifier/definitions.py`
  - Root paths for the different folders of this project and other params.

Configurados los ficheros, ya se podría generar el entorno virtual para comenzar a recopilar datos:

```
export LANG="en_US.utf8"
virtualenv --python=/usr/bin/python3 env
source env/bin/activate
pip install -r requirements.txt
pip install -e .
```

Por último:

```
python bin/ingest_data.py run
```

B spaCy – librería de Python

Durante las fases preliminares del desarrollo, se hizo uso de librerías y scripts existentes para entender el funcionamiento de un pipeline *NLP*

spaCy se trata de una biblioteca de PNL de código abierto para Python utilizada para la creación de aplicaciones que procesen volúmenes masivos de texto de manera eficiente.
Esta biblioteca proporciona una variedad de anotaciones lingüísticas para dar una idea de la estructura gramatical de un texto. Esto incluye los tipos de palabras, como las partes del discurso, y cómo se relacionan las palabras entre sí [68].

spaCy crea los *tokens* tomando el *input* en forma de texto *unicode* para después generar una secuencia de objetos de *token*. Tras esto, spaCy es capaz de analizar y etiquetar un documento determinado mediante la utilización de un modelo estadístico.

Los modelos estadísticos permiten a spaCy hacer una predicción de qué "etiqueta" se aplica en este contexto: un verbo, un adverbio, un sustantivo, etc. SpaCy actualmente ofrece modelos estadísticos para una variedad de idiomas, que pueden instalarse como módulos individuales de Python.

No obstante, lo interesante de spaCy para este Trabajo de Fin de grado es el proceso de normalización que lleva a cabo. spaCy priero realiza la *tokenización* del texto para producir un objeto Doc. El Doc es procesado siguiendo diferentes pasos (*processing pipeline*).

SpaCy permite la creación de *pipelines* propias, que constan de componentes reutilizables; esto incluye el etiquetador, analizador y reconocedor de entidad predeterminados de spaCy, pero también funciones de procesamiento personalizadas.

Algunas de los pipelines de SpaCy permiten incluir "*stop-words*" que ayudarán en el proceso de normalización, eliminándolas de los datos del *data set;* permiten detectar los "lemas" o unidad autónoma constituyentes del léxico de un idioma; categorizar el texto; o hacer coincidir secuencias de tokens, basadas en reglas de patrones o en documentos.