Extracción, análisis y visualización de información social desde Twitter

MEMORIA DEL PROYECTO

Manuel Artero Anguita
Raúl Marcos Lorenzo

Dirigido por: Guillermo Jiménez Díaz
Codirigido por: Juan A. Recio García

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Junio 2014

Extracción, análisis y visualización de información social desde Twitter

Memoria de Proyecto de Sistemas Informáticos
Ingeniería del Software e Inteligencia Artificial
06/14

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Junio 2014

Autorización

Manuel Artero Anguita y Raúl Marcos Lorenzo, alumnos matriculados en la asignatura de Sistemas Informáticos, autorizan, mediante el presente documento, a la Universidad Complutense de Madrid (UCM), a difundir y utilizar con fines académicos, no comerciales, y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado. Todo ello realizado durante el curso académico 2013 - 2014 bajo la dirección de Guillermo Jiménez y Juan Antonio Recio, profesores del Departamento de Ingeniería del Software e Inteligencia Artificial.

Manuel Artero

Raúl Marcos

Agradecimientos

Nos gustaría agradecer el cariño y la confianza de todas las personas que nos han ayudado a llegar hasta aquí:

Por estar siempre a nuestro lado apoyándonos, a nuestras familias; por guiarnos a lo largo de este proyecto y no perder la esperanza en nosotros, a Guillermo y Juan Antonio; y por último, por aportarnos siempre una luz de esperanza en los momentos de mayores dudas, a StackOverflow.com.

A todos ellos, gracias.

Resumen

Hoy en día, cada vez tiene más importancia que el contenido de la web sea accesible en el mismo momento de su creación. Al mismo tiempo, Twitter es una red social ampliamente utilizada para acceder a información en tiempo real ya que la gran mayoría de su contenido es accesible de forma pública.

El objetivo de este proyecto es la extracción y análisis de información accesible a través de Twitter, así como la investigación de las posibilidades existentes para su procesamiento y posterior visualización.

En este proyecto se hace una revisión tanto de artículos de investigación como de servicios relacionados con el uso de información que provee Twitter, seguida de la definición de un marco teórico que clasifique toda esa información. Se presenta el diseño de un sistema orientado en la extracción y procesamiento de información obtenida desde Twitter en español. Se han determinado tres estrategias de generación de información: la detección de género de los usuarios, la categorización de tweets por contenido y el posicionamiento de tweets por áreas geográficas. Adicionalmente, el sistema ofrece a aplicaciones externas la posibilidad de acceder a la información generada.

Por último, se describe como ejemplo de uso una aplicación web que permite visualizar la información recogida y procesada por el sistema de diferentes formas. En ella se puede tanto interactuar con información en tiempo real como visualizar de forma gráfica la información almacenada.

Palabras clave: Twitter Streaming API, minería de datos, recuperación de información, visualización geoespacial, procesamiento en tiempo real, detección de género, categorización textual.

Abstract

Nowadays, there is an increment on the need of real-time data. As a result of his policy of allowing access to the large majority of its content, Twitter is a social network widely used to obtain real-time information.

This project aims to extract and to analyze information accessible via Twitter, as well as to research on the existing opportunities for its processing and subsequent graphical display.

We have made an extensive review of research papers and Twitter-based services related to the information provided by Twitter, in addition to the definition of a theoretical framework that classifies this information. We have designed a system to digest data extracted and processed from Twitter in spanish. We determined three data generation strategies: detection of user's gender, categorization by content of tweets and geospatial display of tweets. The system also offers to external applications the possibility of accessing this generated data.

Lastly, we have built a web application as an example of how to display information extracted and processed by the system. The application interacts with real-time information in addition to displaying graphically stored data.

Keywords: Twitter Streaming API, data mining, information retrieval, geospatial display, real-time processing, gender detection, text categorization.

x

Índice

Índice de figuras

Índice de tablas

1. Motivación y objetivos

La web siempre ha estado basada en el contenido: cantidades ingentes de información, clasificada y siempre accesible. Si bien esto sigue siendo una constante, cada vez tiene más importancia que dicho contenido sea accesible en todo momento, especialmente en el momento de creación. Los ejemplos más claros de esta tendencia son las redes sociales, y más concretamente Twitter[1] y Facebook[2].

Twitter se caracteriza por tener la gran mayoría de su contenido accesible de forma pública, e incluso provee diferentes métodos para acceder a él. Este contenido puede ser usado como base para dar respuesta a preguntas que han surgido tras la popularización de las redes sociales: las marcas quieren saber qué opinión se tiene sobre ellas, las empresas quieren conocer mejor a sus clientes, los medios de comunicación quieren saber cómo se transmiten sus noticias y la clase política quiere conocer qué dice la opinión pública. Esta información no está siendo ampliamente utilizada para responder a todas estas preguntas, y menos aún en español, idioma que aún siendo el segundo más usado en Twitter [Europa Press, 2013] apenas cuenta con herramientas y utilidades.

Por ello nuestra motivación es aportar un poco de luz en el panorama del análisis de información de Twitter en español, creando una herramienta que extraiga y clasifique contenido, para luego centrarnos especialmente en la categorización de temática de dicho contenido, detección de género de los usuarios que lo crean y visualización de todos estos datos. Creemos que éste puede ser un paso interesante para aportar orden a la información de Twitter, de forma que sea posible entender mejor esta red, y responder a algunas de las preguntas que, como decíamos, están surgiendo en esta nueva tendencia en el mundo digital.

[1] *twitter.com*

[2] *facebook.com*

1.1. Objetivos

El objetivo general del proyecto es la extracción y análisis de la información social accesible a través de Twitter, así como el desarrollo de estrategias para su procesamiento en español y su posterior visualización.

Este objetivo se ha dividido en varias partes. En primer lugar, se ha realizado un estudio para la clasificación de la información extraíble desde Twitter. Como aplicación práctica, se ha creado un sistema de clasificación de información de Twitter en español en un entorno web. Este sistema será capaz de extraer dicha información, convertir parte de los datos recibidos en información procesada y exponerla, todo ello en tiempo real. Por último, se quiere desarrollar una aplicación web que sirva como ejemplo de un posible cliente del sistema, mostrando cómo es posible utilizar nuestro servicio para visualizar información de forma interactiva.

A modo de resumen, el trabajo ha consistido en la realización de las siguientes actividades:

- Búsqueda y revisión de artículos de investigación relacionados con el uso de información disponible en Twitter.
- Investigación y análisis servicios que hacen uso de Twitter para suministrar información.
- Análisis y propuesta de clasificación teórica de la información extraíble de Twitter.
- Diseño de un sistema para la extracción de información desde Twitter.
- Proposición de estrategias de generación de información procesada en español: detección del género del usuario, categorización de *tweets* por contenido y posicionamiento de *tweets* por áreas geográficas.
- Implementación de las estrategias anteriores, siguiendo un diseño lo suficientemente modular para facilitar la integración de nuevos módulos de procesamiento de la información.
- Exposición de la información generada por medio de una API[3] o servicios web que ofrezca la posibilidad a una aplicación externa de usar el servicio abstrayéndose de nuestra implementación.

[3] *http://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones*

- Diseño e implementación de una aplicación web que sirva como ejemplo de uso del sistema desarrollado.

1.2. Organización de la memoria

El contenido de esta memoria se ha estructurado de la siguiente forma:

1. En este primer capítulo se ha expresado la motivación y objetivos que han impulsado la elaboración del proyecto.
2. En el capítulo 2 se describe el estado del arte de extracción análisis y visualización de información desde Twitter realizado mediante el análisis tanto de artículos de investigación como de sistemas o servicios relacionados con la extracción y procesamiento de información extraíble de Twitter.
3. Durante el capítulo 3 se presenta un análisis teórico para la clasificación de la información. Se determina a qué información podemos acceder y se establece una posible clasificación teniendo en cuenta distintos aspectos relacionados con la naturaleza de los datos.
4. En el capítulo 4 se describe la arquitectura de nuestro sistema, siendo analizado desde una perspectiva general hasta una visión más detallada de los diferentes módulos que configuran la implementación del proyecto.
5. En el capítulo 5 se muestra el diseño y la implementación de una posible aplicación que usa como base el sistema desarrollado, demostrando así su modularidad y las posibilidades que ofrece.
6. En el capítulo 6 se enumeran las conclusiones finales del proyecto y se destacan posibles líneas de trabajo futuro.
7. Por último, se incluyen varios apéndices en los que se pueden encontrar información adicional sobre el proyecto como por ejemplo detalles a nivel de código que no se tratan en los apartados anteriores.

2. Estado del arte

En este capítulo se expone la situación actual de los diversos ámbitos de investigación que se manejan en el desarrollo del proyecto.

En primer lugar se expone una breve introducción a la red social Twitter explicando qué es y cómo funciona, acompañada de un glosario de términos comunes que se repetirán a lo largo de toda la memoria. A continuación se expone un breve análisis de la información que podemos tratar, así como la situación actual en la que se encuentra la red socia en relación con la información. Por último se presenta un análisis tanto de artículos de investigación como de sistemas o servicios relacionados con la extracción y procesamiento de información social extraíble de Twitter.

2.1. Introducción a la red social Twitter

2.1.1. Qué es Twitter

Twitter es una red social basada en mensajes cortos de hasta 140 caracteres. Alrededor de estos mensajes, llamados *tweets*, se articula todo lo demás: cuando un usuario entra en Twitter, verá un listado cronológico de *tweets* que han sido creados por aquellos usuarios que él mismo ha decidido seguir. El usuario tiene la opción de retransmitir estos *tweets* a todas aquellas personas que, a su vez, han decidido seguirle. A estos usuarios, que se suscriben a los *tweets* publicados o retransmitidos por otro usuario, se les denomina *followers* o seguidores.

De esta forma, Twitter se ha convertido en una red social ideal para estar informado lo más rápido posible de cualquier noticia, ya que éstas aparecen en Twitter nada más son producidas, y los

propios usuarios las difunden, haciendo que lleguen a casi todos los usuarios en cuestión de minutos. Twitter ha sido ampliamente usado para situaciones de emergencia o catástrofes naturales, e incluso la propia red social lanzó en septiembre de 2013 un servicio de alertas en colaboración con diferentes gobiernos y organismos [Coyne, 2013]. En España, este servicio[4] está activado gracias a una colaboración con el Cuerpo Nacional de Policía [Romero, 2014].

La gran cantidad de información creada en Twitter es pública por defecto, y gracias a su política de proveer formas sencillas de acceder a ésta mediante interfaces, la convierte en la red social más usada para obtención de información y análisis en diversos ámbitos: desde empresas que quieren saber más sobre sus clientes, hasta en investigaciones científicas.

2.1.2. Glosario de términos comunes

A continuación se definen una serie de términos relativos a Twitter:

- ***Tweet*.** Mensaje de texto limitado a 140 caracteres publicado por un usuario de Twitter. Los *tweets* de un usuario llegarán de forma automática a todos los usuarios suscritos (*followers*) a sus publicaciones por medio de sus *timelines*.
- ***Follower*.** Usuario que decide suscribirse al contenido publicado por el usuario. De esta forma recibirá automáticamente los *tweets* que publique el usuario al que sigue. Es importante notar que la relación no es bidireccional.
- **Hacer *follow*.** Acción de seguir o suscribirse a un usuario. No es necesaria la autorización del usuario al que queremos seguir.
- ***Friend*.** Usuario de Twitter al que sigue el usuario.
- ***Retweet*.** Copia de un *tweet* de otro usuario en el perfil propio. Cualquier persona que acceda al perfil puede ver el *tweet* junto con el nombre de la persona que originalmente lo escribió. Un

[4] *https://twitter.com/policia/alerts*

retweet puede entenderse como una cita textual de lo dicho por otro usuario, manteniendo quién es el escritor original.

- **Mensaje directo.** Mensaje enviado de forma privada a un único usuario. La única diferencia con los *tweets* ordinarios es que existirá un único receptor y nadie aparte del emisor y el receptor tendrá acceso al mensaje.
- **Mención.** *Tweet* en el que se incluyen uno o varios nombres de usuarios registrados en Twitter. Para mencionar un usuario, es necesario preceder su nombre del símbolo "@". Si la mención es una respuesta a otro *tweet*, recibe el nombre de *reply.*
- ***Timeline* de un usuario.** Listado cronológico de todos los *tweets* publicados por aquellos usuarios a los que se sigue.
- ***Hashtag.*** Cadena de caracteres precedida por el símbolo "*#*" y formada por una o varias palabras concatenadas. Sirve para etiquetar el *tweet* en el que aparece como perteneciente a un tema específico.
- ***Trending topic.*** Tendencia que muestra de lo que se está hablando dentro de la red representada por una palabra o frase. Las más relevantes se pueden consultar en la página de Twitter y pueden ser filtradas geográficamente.

2.2. Qué información podemos extraer de Twitter

En líneas generales, y antes de realizar un estudio exhaustivo de la información accesible a través de Twitter, se ha realizado un análisis que identifique las cuatro secciones más relevantes en las que se puede dividir un *tweet* en el momento en que un usuario lo publica:

- **Quién.** La persona que lo escribió (o hizo *retweet)* junto con sus datos públicos: nombre completo, localización, lenguaje, etc. Si estudiamos la demografía de Twitter nos encontramos con que se trata de una red social adulta. En concreto, gran parte de los usuarios que usan

Twitter son adultos de más de 35 años que no han utilizado otra red social con anterioridad. Además, solamente el 11% de los usuarios tienen menos de 16 años [Dreyer, 2011].

- **Qué.** El contenido del *tweet,* lo que incluye no solo el texto del mensaje sino también los links, menciones o contenido multimedia que pueda contener. ¿Qué tipo de contenido podemos esperar encontrarnos publicado en la red? Según un estudio de mercado [Kelly, 2009] aproximadamente el 87% de los *tweets* son ruido, conversaciones personales entre usuarios o *spam.* El porcentaje de noticias o contenido de interés general se reduce al 4% y el 9% restante se trata de repeticiones de contenido ya publicado (*retweets*).
- **Cuándo.** Fecha y hora de publicación. Gracias al perfil del usuario es posible determinar el huso horario en el que se encuentra.
- **Dónde.** Coordenadas geográficas de la ubicación desde donde fue publicado. Esta información no está presente en todos los *tweets,* apenas 2,02% de ellos contienen esta información [Leetaru, 2013]. Sin embargo es posible extraer únicamente *tweets* que sí cuenten con estos datos (se explica en el capítulo 3).

2.3. Búsqueda y revisión de artículos de investigación relacionados con Twitter

En esta sección se ha realizado un análisis de artículos relacionados con la extracción de información desde Twitter y el uso de dicha información en diferentes áreas. En primer lugar se han estudiado trabajos de investigación que hacen uso de la información social de Twitter para realizar sistemas de recomendación. En segundo lugar, se han analizado documentos de investigación que se centran en la topología de la red y el análisis de relaciones sociales en Twitter.

2.3.1. Uso de información de Twitter en sistemas de recomendación

Twittomender [Hannon, 2011] es un sistema de recomendación de usuarios de Twitter que usa estrategias de recomendación basadas en contenido, y refinado con filtrado colaborativo. Como herramienta principal emplea Lucene[5], librería de código abierto centrada en la recuperación de información basada en texto. El recomendador de usuarios define los documentos para indexar Lucene como un vector de pesos con información relativa al perfil del usuario**.** Los datos que considera relevantes para el perfilado de usuarios son los últimos *tweets* del propio usuario, de sus seguidores (*followers*) y de las personas a las que sigue (*friends*). Además, a estos datos añade los identificadores de todos sus seguidores y personas a las que sigue (es en este punto cuando recalca la existencia de filtrado colaborativo). Con todos estos datos genera para cada usuario un vector con pesos. El peso de cada término en el documento es el Tf-idf[6] proporcionado por Lucene y por lo tanto proporcional a la frecuencia de aparición en el perfil del usuario e inversamente proporcional a la frecuencia en el resto de perfiles.

Buzzer [Phelan, 2010] es un sistema de recomendación de artículos basado en contenido extraíble desde Twitter. El punto de reflexión principal de Buzzer es lo que llama las "*topical news stories"* que definen como aquello que irrumpe en la red durante un periodo de tiempo determinado. Su hipótesis es que durante este tiempo gran parte del contenido generado girará alrededor de estos temas. La herramienta que emplea también es Lucene. El funcionamiento de Buzzer es el siguiente: parte de un conjunto de artículos *R* y un conjunto de *tweets T*. A continuación, indexa con la herramienta Lucene por separado ambos conjuntos, obteniendo los conjuntos *MR* y *MT* sobre los que hace la intersección *I*. Mediante Lucene, los elementos del conjunto *I* se usan para sacar una serie de artículos recomendados.

2.3.2. Uso de información de Twitter en el estudio de la topología y relaciones sociales de la propia red

[5] *http://lucene.apache.org/*

[6] *http://es.wikipedia.org/wiki/Tf-idf*

En [Teutle, 2010] se describe un estudio teórico de la topología de Twitter que se basa fundamentalmente en la teoría de grafos para definir diferentes métricas aplicables a la red social. Define la topología de Twitter tomando los usuarios como los nodos de la red y las relaciones entre los usuarios como aristas dirigidas. Destaca que la dirección de la arista será contraria al flujo de información ya que si el usuario A está suscrito al contenido publicado por el usuario B, la información fluirá desde B hasta A. El artículo trata dos términos interesantes: lo que denomina la *dinámica **de** la red* (*dynamics of the network*) que define como los cambios topológicos en la estructura de la red y la *dinámica **en** la red* (*dynamics on the network*) definida como "*la interacción entre diversos nodos […] condicionada por los [nodos] vecinos*". El estudio llega a una serie de conclusiones sobre el comportamiento de la red social, sobre las que destacan las siguientes:

- Existe una relación directa entre el *follower ratio* de un usuario con el rol que desempeña el usuario dentro de la red. Se define el *follower ratio* de un usuario como la división del número de *followers* entre el número de personas a las que sigue el usuario. En la Tabla 2.1 se recoge el significado de los diferentes valores de esta relación.

Follower ratio	Significado
Aproximadamente 0	Cuentas no personales orientadas a la recolección de información para un *trending topic* determinado.
Menor que 1	Usuarios que buscan la recolección de información sobre la generación de nuevo contenido.
Cercano a 1	Usuarios ordinarios de Twitter. Aproximadamente el 50% de las cuentas de usuario se encuentra en este rango.
Mayor que 1	Usuario generador de contenido apreciado por su propia comunidad.
Mayor o igual a 10	Prescriptores de contenido (usuarios más influyentes de la red social).

Tabla 2.1. Relación entre el follower ratio *y el tipo de cuenta de usuario definida por [Teutle, 2010]*

- Las cuentas no personales aparecen y desaparecen dependiendo de las tendencias temporales o *trending topics* para las que fueron creados. Se define una cuenta no personal como aquella que no es usada con fines personales como por ejemplo la cuenta de una marca.

- El 30% de los usuarios reciben los nuevos contenidos en el momento en que son creados. Es decir, el desfase entre el momento de creación y el momento de recepción es insignificante.
- El 25% de los usuarios mantiene una red pequeña y balanceada de seguidores y amigos. El tamaño de esta red es de aproximadamente 50 usuarios.

En el artículo [Weng, 2010] se describe un estudio centrado en técnicas para la localización de usuarios influyentes en la red Twitter. Principalmente, trata sobre cómo aplicar la técnica Latent Dirichlet Allocation (LDA)[7] para encontrar los usuarios más influyentes en Twitter. Los dos términos principales en el modelo LDA son los documentos y las categorías. Se definen los documentos como una distribución de probabilidades sobre un conjunto de temáticas. Cada temática es, a su vez, una distribución de probabilidades sobre un conjunto de palabras. A partir de estas definiciones implementa un algoritmo con el objetivo de determinar la localización de usuarios influyentes.

El resultado al que llega el estudio es la existencia de homofilia[8] en la red Twitter. Es decir, la tendencia de los usuarios de seguir a usuarios que les han seguido previamente, quedando en segundo plano si existe o no un interés en el contenido publicado. El artículo afirma que el 72% de los usuarios cumplen que el 80% de sus seguidores se deben a la reciprocidad y no al interés en los contenidos publicados por el usuario.

En [McPherson, 2001] se realiza un análisis teórico de las relaciones existentes en las redes sociales. El artículo no está centrado en la red social Twitter, sino que trata sobre las redes sociales en general. El estudio determina que la información de la red se queda concentrada en ciertos subgrafos determinados por una de las ocho posibles relaciones que define el propio artículo. Las relaciones que establece son sentimental, de amistad, profesional o estudiantil, debido a un foro de discusión, de contacto, de interés, por aparición en un sitio público o de homofilia.

---

[7] *http://jmlr.org/papers/v3/blei03a.html*

[8] *http://fernandosantamaria.com/blog/2012/10/la-homofilia-un-principio-activo-en-la-estructura-de-las-redes-sociales/*

2.4. Servicios relacionados con la extracción y procesamiento de la información facilitada por Twitter

Se ha analizado la funcionalidad que ofrecen los siguientes tres servicios relacionados con la extracción y procesamiento de información:

SecondSync[9] es una empresa británica especializada en el análisis de Twitter enfocada únicamente a la televisión (la llamada "televisión social"). Su servicio consiste en un panel donde las cadenas de televisión o similares pueden ver un análisis detallado de sus programas, ya sea por emisión o en general. En este análisis las empresas pueden no solo consultar datos relativos al volumen de *tweets* que generan sus programas sino también si lo que se dice de ellos es positivo o negativo o el perfil medio de quién lo comenta incluyendo su sexo como se ve en la Figura 2.1

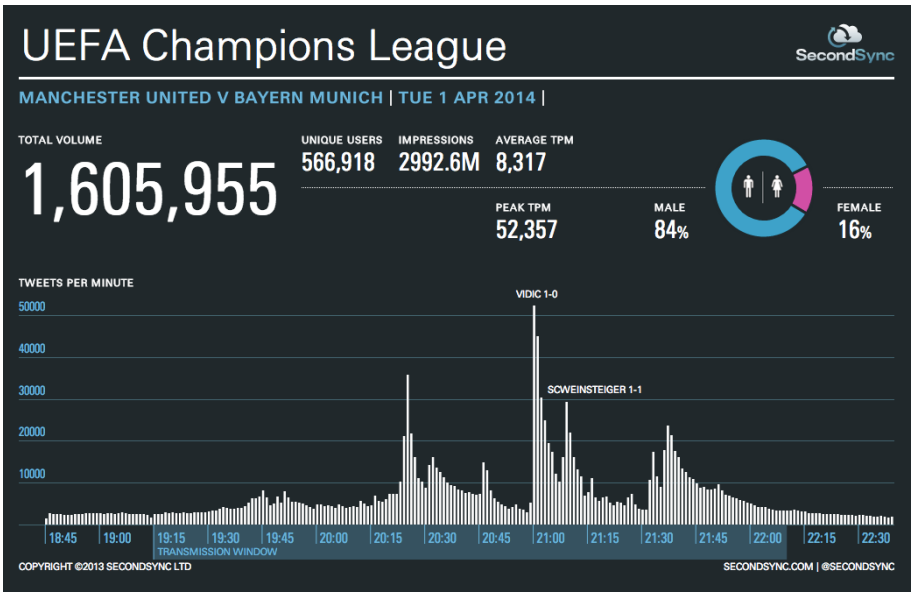

Figura 2.1. Captura de pantalla de SecondSync

[9] *https://secondsync.com/*

BluefinLabs[10] se centra también en la televisión pero con un enfoque diferente al de SecondSync. Si bien SecondSync se focaliza básicamente en el contenido para ofrecer una serie de estadísticas, BluefinLabs busca saber qué dice la gente sobre un programa o un anuncio de la televisión, y cómo afecta a estos. Es decir, no solo ofrecen datos cuantitativos sobre los programas o anuncios, sino que relacionan estos datos con información externa para ofrecer estadísticas sobre el rendimiento de campañas publicitarias, listas de términos más repetidos, relaciones del tipo "a los usuarios que les gusta este programa les gustará este otro", etc. La Figura 2.2 es una captura del panel principal de la aplicación web de BluefinLabs

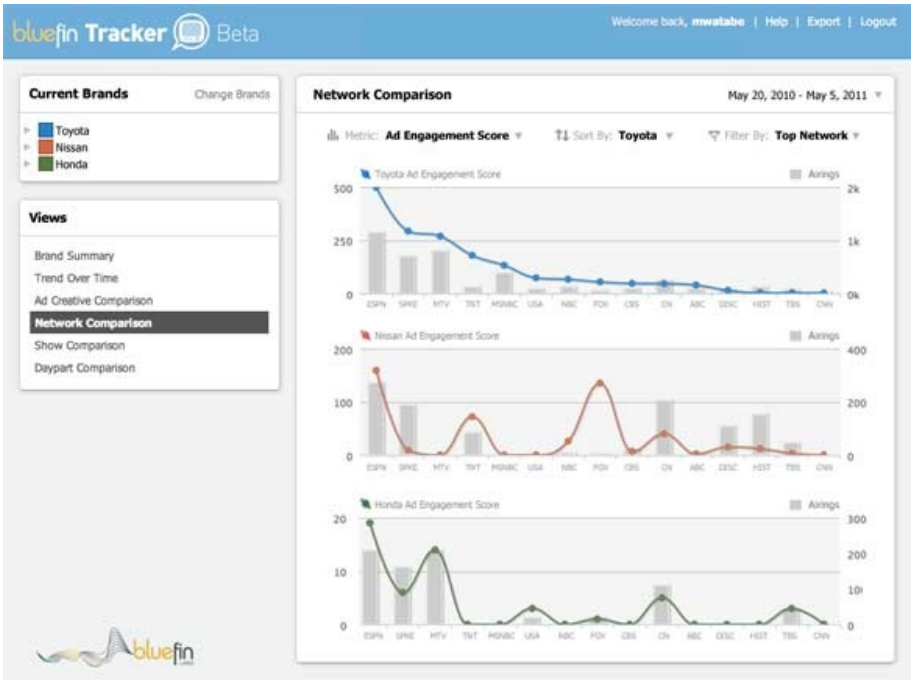

Figura 2.2. Captura de pantalla de BluefinLabs

Follower Wonk[11] es una herramienta que determina la autoridad o influencia de un usuario dentro de la red con un valor entre 0 y 100 llamado ***wonk social authority.*** En la aplicación se pueden ordenar

[10] *http://bluefinlabs.com/*

[11] *http://followerwonk.com/*

usuarios de Twitter en función de esta métrica como se ve en la Figura 2.3. Las variables tenias en cuenta para calcular este valor son:

- **Tasa de *retweets.*** Se utilizan aproximadamente los 100 últimos *tweet*s del usuario que no contengan menciones a otros usuarios. La razón por la que se filtran los *tweets* con menciones es que existe una gran relación entre el número de *retweets* que consigue un usuario con la tasa de menciones que recibe, en torno al 80%. Otro dato interesante es que la relación entre la tasa de *tweets retweteados* y los *tweets* que contienen URLs es aproximadamente del 70%.
- **Lo reciente que es el *retweet.*** Se favorece a los *tweets* a corto plazo contra *tweets* antiguos. Lo que se intenta es destacar el contenido que tiene impacto en Twitter en cada instante. Se define la vida media de un *tweet* en 18 minutos.
- **Número de *followers*.** Este tercer factor es el que menos peso tiene y se optimiza mediante un modelo de regresión basado en el número de *retweets* recibidos por el usuario. El objetivo de este modelo es restar importancia a *tweets* ocasionales de un usuario que hayan recibido una repercusión excepcional teniendo en cuenta que hay una relación directa entre el número de seguidores y el porcentaje de *tweets retweeteados*, hasta el punto que a partir de los 10.000 seguidores se consigue una tasa de *retweets* del 25%.

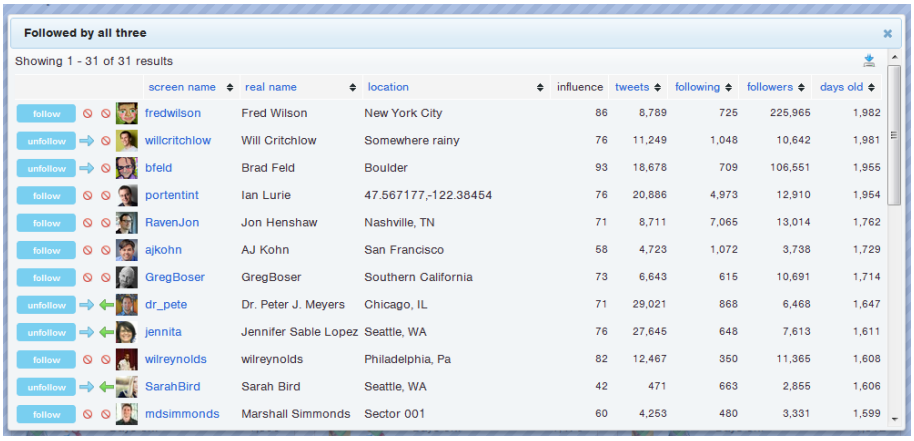

Figura 2.3. Captura de pantalla de Follower Wonk

En el siguiente extracto del blog de los desarrolladores [Bray, 2013] se puede observar su postura defendiendo que los algoritmos para determinar la influencia de los usuarios dentro de una red social basados en *retweets* son mejores que aquellos basados en el número de *followers*:

"@autocorrects ha sido retweeteado *un 7% más que @BarackObama, a pesar de que tiene 14 veces menos* followers. *Como se puede ver,* Social Authority *descubre una serie completamente diferente de* top users*: aquellos que son muy efectivos comunicándose con sus* followers *[…] [estos usuarios] han descubierto contenido que capta la atención de sus audiencias, sea de su agrado o no, y provoca más interacción en términos de* retweets *y tráfico generado."*

2.5. Conclusiones

En este capítulo se han definido los conceptos básicos relativos a la red Twitter y se ha hecho un breve análisis de qué información es accesible. También se ha hecho un repaso de la situación actual tanto de artículos de investigación como de servicios basados en la extracción y análisis de la información disponible en Twitter.

En base a la revisión de estos artículos de investigación y servicios, nos planteamos que Twitter, entendido como fuente de información, está seccionado en las siguientes tres entidades: los **usuarios** que generan contenido, los ***tweets,*** y la **actividad** de la red al publicarse contenido que relaciona a los usuarios que la componen. En el capítulo 3 retomaremos esta división en entidades relacionadas con la información existente en Twitter.

3. Marco teórico de la clasificación de la información

Durante este capítulo se detalla la información a la que se tiene acceso a través de Twitter. Un caudal tan extenso de contenidos que sin un análisis teórico y su necesario tratamiento, resulta inabarcable. Por ello hemos creado un listado de lo que denominamos puntos de información, que permiten el tratamiento y organización de la información.

En primer lugar se especifican los dos métodos de extracción de la información que ha establecido Twitter. A continuación se detalla a qué datos se puede acceder directamente empleando estos métodos, así como la información que se encuentra de manera implícita, ya que Twitter no solo responde a la petición directa sino que también ofrece información adicional. Por ejemplo, si se le pide datos sobre un *tweet* también aportará el perfil completo del usuario que lo creó.

En segundo lugar se define una clasificación teórica de la información atendiendo a diferentes aspectos de la naturaleza de los datos como son a qué categoría pertenecen los datos, a qué modelo pertenece la respuesta que se recibe de Twitter y cuál es el nivel de procesamiento de la información obtenida.

En tercer lugar se clasifica toda la información expuesta atendiendo a la clasificación teórica definida en este mismo capítulo indicando, para cada punto de información, cuales son las categorías asociadas, qué modelo de datos está relacionado y el nivel de procesamiento de la información.

Por último, cotejaremos la clasificación teórica que se ha desarrollado con las aplicaciones que se basan en el análisis y tratamiento de información extraída de Twitter vistas en el capítulo 2. El objetivo de este último punto es mostrar que nuestra clasificación es capaz de englobar desde el punto de vista teórico la información extraíble desde Twitter.

3.1. Información extraíble de Twitter

En primer lugar se va a determinar qué información es accesible desde Twitter de forma pública. El primer paso será entender cómo se puede extraer dicha información. Existen dos formas de consumir información desde Twitter: la API REST[12] y la Streaming API[13], que detallamos a continuación:

- **API Rest.** Permite interactuar con un sistema de forma transparente a la arquitectura de dicho sistema mediante peticiones web. Dicho de otra forma: es posible acceder a la funcionalidad ofrecida por un sistema en forma de caja negra realizando peticiones GET y POST. En el caso de la API ofrecida por Twitter, podemos realizar todas las acciones a las que tenemos acceso desde la página web o sus aplicaciones móviles.
- **Streaming API.** En este modo de acceso, al hacer la petición inicial Twitter abre una conexión entre el solicitante y su servidor, y enviará por ella *tweets* que sean publicados a partir de ese momento que cumplan los filtros que se hayan establecido en el momento de iniciar la conexión. Se recibirán nuevos *tweets* ininterrumpidamente hasta que se decida cerrar la conexión. Este proceso es en tiempo real: desde que alguien crea un *tweet* hasta que llega a los usuarios con una conexión a la Streaming API abierta sólo pasan aproximadamente 350 milisegundos [Vaughan-Nichols, 2012].

Es importante destacar que mientras la API Rest proporciona acceso a la **información ya existente** en Twitter en el momento de hacer la llamada, mediante la Streaming API sólo recibiremos **información creada posteriormente** a la creación del canal de comunicación entre nosotros y Twitter.

De forma añadida, es importante resaltar que al recibir una respuesta por parte de Twitter (independientemente del método que hayamos usado) se obtiene la representación formal de la información o modelo definido por Twitter completo. Corresponderá al solicitante seleccionar qué

[12] *https://dev.twitter.com/docs/api/1.1*

[13] *https://dev.twitter.com/docs/streaming-apis*

información dentro de ese modelo es relevante y cuál no. La información que reside en estos modelos también será objeto de estudio en los siguientes apartados al ser información también extraíble desde Twitter.

Debido a que el objetivo de este apartado es definir un marco teórico global para toda la información accesible desde Twitter, se estudian todas las posibilidades que brinda tanto la Rest API de Twitter, como la Streaming API, como la información contenida en las respuestas. Los detalles técnicos o relativos a temas de implementación serán obviados durante todo el capítulo.

3.1.1. Información extraíble a través de la API Rest de Twitter

A continuación se presenta un listado de las posibles peticiones a la API Rest de Twitter. Sólo incluimos aquellas peticiones relacionadas con la obtención de información, pasando por alto aquellas que estén relacionadas con el manejo de cuentas de usuario o su configuración. Debido a que es una lista moderadamente extensa, se presenta dividida en secciones dependiendo del dato de entrada necesario para hacer la petición, distinguiendo entre un usuario registrado, una lista de usuarios, un *tweet* o una consulta. Dentro de esta última posibilidad, distinguimos entre consultas orientadas a la búsqueda de *tweets*, a la búsqueda de usuarios o consultas de ámbito general.

3.1.1.1. Peticiones que reciben como entrada un usuario

- **Últimos *tweets* publicados por el usuario.** Hasta 100 *tweets*.
- **Últimos *tweets* que contienen una mención al usuario.** Hasta 20 *tweets*.
- **Últimos *tweets* marcados como favoritos por el usuario.** Hasta 20 *tweets*.
- **Últimos *tweets* que han sido *retweeteados*.** Hasta 20 *tweets* sin incluir los *retweets*.
- **Lista de usuarios a los que sigue el usuario.** Lista de amigos.
- **Lista de usuarios que siguen al usuario.** Lista de *followers.*
- ***Timeline* del usuario.** *Tweets* publicados por los usuarios seguidos.
- **Últimos mensajes directos recibidos por el usuario.**

- **Últimos mensajes directos enviados por el usuario.**
- **Lista de usuarios de los que el usuario no quiere recibir *retweets*.**
- **Relación entre el usuario y una lista (máximo 100) de usuarios.** Esta relación puede ser: "ambos usuarios se siguen entre sí", "un usuario sigue al otro", o "no existe ninguna relación"
- **Detalles de configuración de usuario.** Ejemplos de estos detalles son el idioma en el que tiene configurada la interfaz o la zona horaria en la que se encuentra.
- **Lista de usuarios que el usuario ha bloqueado.** Bloquear a un usuario implica dejar de seguir a ese usuario, y fuerza al usuario bloqueado a que deje de seguir al que ha hecho el bloqueo, siempre que sea posible. Además, Twitter no permitirá que el usuario bloqueado vuelva a seguir al usuario que le bloqueó.
- **Listas a las que está suscrito el usuario.** Definimos como una lista a un grupo de usuarios creado por algún otro usuario de la red para organizarlos, y poder ver en el *timeline* de la lista todos sus *tweets*. Dependiendo de la visibilidad, una lista puede ser pública (visible en toda la red) o privada (visible sólo para el usuario que la creó).
- **Lista de usuarios recomendados.** A partir de la clasificación que hace Twitter internamente de cada usuario, Twitter recomienda al usuario otras cuentas a las que puede interesarle seguir. Twitter no revela los criterios usados por este sistema de recomendación.

3.1.1.2. Peticiones que reciben como entrada una lista de usuarios

- ***Timeline* de los *tweets* publicados por cualquier usuario perteneciente a la lista.**
- **Comprobación de si un usuario determinado está suscrito a la lista.**

3.1.1.3. Peticiones que toman como entrada un *tweet*

- **Los últimos *retweets* del *tweet* especificado.** Hasta un máximo de 100.
- **Los últimos usuarios que han *retweetado* el *tweet* especificado.** Hasta un máximo de 100.

3.1.1.4. Peticiones que reciben como entrada una consulta

Consultas orientadas en la búsqueda de *tweets*. Las posibilidades para la construcción de este tipo de consultas son las siguientes, pudiéndose además combinar los criterios de búsqueda.

- **Buscar *tweets* dependiendo del contenido.** Es posible definir una lista de términos que queremos que aparezcan en el texto del *tweet*, cadenas exactas de texto, *hashtags*, nombres de usuarios o enlaces a direcciones de internet. Además, también se puede indicar la actitud en el mensaje diferenciando entre mensajes con una actitud positiva y mensajes con actitud negativa. Este último punto implica un procesamiento interno de lenguaje natural transparente a nosotros.
- **Buscar *tweets* dependiendo de la fecha de publicación.** Es posible definir un rango de fechas para determinar *tweets* que hayan sido publicados antes, durante o después de esas fechas.
- **Buscar *tweets* dependiendo del lenguaje en el que están publicados.** El sistema que sigue Twitter para clasificar el lenguaje de los *tweets* es generalizar a partir del lenguaje que el usuario ha marcado como predeterminado para mostrar la interfaz de la página web *twitter.com*.

Consultas orientadas en la búsqueda de usuarios. La única posibilidad que nos da Twitter es indicar una cadena de texto simple, sobre la que no se permite ningún tipo de operación.

Consultas de ámbito general.

- **Usuarios recomendados para una categoría predeterminada.** Estas categorías están definidas previamente por Twitter y las denomina *slugs.* De nuevo, no tenemos información de cómo ha establecido las categorías, ni cómo funciona el sistema de recomendación.
- **Información conocida sobre un área geográfica.** El tipo de información que podemos encontrar es, principalmente, a qué país corresponde al área definida, y si se trata de una ciudad o de “terreno abierto”.

- **Los 10 *trending topics* en una localización concreta.** Es posible solicitar las tendencias en un área geográfica concreta, usando identificadores. Estos identificadores son o de país o de ciudad (sólo están soportadas las más importantes).
- **Lista de localizaciones de las que Twitter tiene información relativa a algún *trending topic***. Esta consulta puede entenderse como la inversa del punto anterior "top-10 de trending topics para una localización concreta".

3.1.2 Información extraíble a partir de la Streaming API de Twitter

La única información accesible utilizando la Streaming API son *tweets* publicados por cualquier usuario de Twitter. La ventaja respecto a la API Rest reside en que esos *tweets* se reciben en tiempo real. En el momento de establecer la conexión con la Streaming API, es posible establecer una serie de parámetros que definirán qué requisitos (o filtros) queremos que cumplan los *tweets* que se reciben desde Twitter. En el siguiente listado se muestran únicamente las opciones relacionadas con las características de la información recibida en forma de *tweets*, excluyendo las opciones técnicas relacionadas con el tipo, duración o calidad de la conexión.

- **Nivel de filtrado.** Se trata de un atributo incluido en el modelo definido por Twitter de *tweet* pensado para aplicaciones que muestran *tweets* en tiempo real. Sus posibles valores son *ninguno*, *bajo*, o *medio*. Con el nivel de filtrado más permisivo (ningún filtro) recibiremos todos los *tweets* generados. Con el máximo nivel de filtrado (a día de hoy es el nivel *medio*) recibiremos *tweets* que cumplan unos requisitos definidos por Twitter. La documentación habla de contenido que ha tenido cierta relevancia[14] dentro de la red, sin embargo no define cómo ha hecho esta clasificación. En principio, permitiría distinguir tweets con contenido de interés del resto de *tweets* calificados como ruido [Roomann-Kurrik, 2013].
- **Lenguaje del *tweet.*** Permite definir el idioma, o los idiomas, en el que se desea que estén escritos los *tweets* recibidos. Determinar el idioma se hace a nivel de tweet y no de usuario.

[14] Definido por los desarrolladores, se trata de los *tweets* más consultados a través de la página de Twitter.

- **Monitorización de un usuario.** Permite definir el usuario (o la lista de usuarios) de los que queremos recibir los *tweets* publicados, *retweets* y respuestas de otros usuarios a cualquiera de sus *tweets*. No se incluyen *tweets* que mencionen al usuario que se está monitorizando (p. ej., "*Hola @BarackObama*").
- **Monitorización de contenido.** Permite definir contenido sobre el que se quiere recibir *tweets,* de forma similar a la monitorización de usuarios. En este caso, es posible establecer una lista de términos. Los términos deberán estar separados por comas , funcionando como un operador lógico O. Se ignora la diferencia entre mayúsculas y minúsculas, y existen diversos límites, como el máximo de términos, fijado en 400, o la búsqueda de términos dentro de una URL.
- **Localización.** Permite definir la localización de la publicación de los *tweets* que se desea recibir. Para definir la zona geográfica se usa una lista de pares de valores de latitud y longitud que definirán un área. Los *tweets* publicados dentro de ese área se incluirán en la respuesta, no así los posibles *retweets* publicados desde un área distinta. Es importante notar que este parámetro se puede usar sólo en términos positivos (términos de pertenencia) y no es posible excluir áreas.

Por último, destacar que la combinación de estas posibles opciones se hace en forma de producto lógico, y no es posible anidar filtros.

3.1.3. Información disponible en los modelos definidos por Twitter

Al recibir una respuesta de Twitter se recibirá un bloque prefijado de información. Estos bloques o *modelos* contendrán siempre los mismos registros de información. Diferenciamos dos posibles respuestas: respuestas de tipo usuario y respuestas de tipo *tweet*. A continuación se analiza la información disponible en los modelos definidos por Twitter.

3.1.3.1. Información contenida en las respuestas de tipo usuario

- Fecha de creación de la cuenta.
- Descripción provista por el propio usuario.

- Recuento de *tweets* marcados como favoritos por el usuario desde el momento de creación de la cuenta.
- Número de personas que siguen al usuario.
- Número de personas a las que sigue el usuario.
- *Flag* que determina si el usuario ha activado la opción de mostrar su ubicación al resto de la red.
- Idioma marcado como predeterminado por el usuario.
- Número de listas públicas de las que el usuario es miembro.
- Localización física establecida por el usuario en formato texto.
- *Flag* que determina si ésta cuenta está marcada como privada por su dueño.
- Número de *tweets* publicados por el usuario.
- Zona horaria de la cuenta (configuración elegida por el usuario).
- Url dada por el usuario relacionada con su perfil.
- *Entidades* determinadas por Twitter y extraídas tanto de la descripción como de la url relacionada con el perfil del usuario. Estos términos podrían representar una clasificación del usuario [Shen, 2013] ya que tanto la descripción como la url relacionada son datos aportados por el propio usuario. Se trata de información que cuenta con un procesamiento previo y transparente a nosotros.

3.1.3.2. Información contenida en las respuestas de tipo *tweet*

- Texto del *tweet*.
- Fecha de creación del *tweet.*
- Coordenadas geográficas del *tweet,* indicadas mediante latitud y longitud.
- Identificador del usuario que ha escrito el *tweet,* que es único en la red.
- Número de veces que este *tweet* ha sido marcado como favorito.
- Número de veces que el *tweet* ha sido *retweeteado* por otros usuarios.

- Flag que determina si el *tweet* contiene algún enlace y/o contenido multimedia.
- Flag que determina si el *tweet* fue creado como respuesta a algún otro tweet. Si este fuera el caso, la respuesta contendrá el identificador del *tweet* concreto al que se está respondiendo, así como el nombre del usuario que lo publicó.
- Idioma en el que está escrito el *tweet*, según el procesamiento propio de Twitter.
- Identificador de lugar con al que el tweet está asociado. Esta información podría no estar presente y no corresponde necesariamente con el lugar donde se publicó.
- *Entidades* determinadas por Twitter que están presentes en el contenido del *tweet*. Al igual que ocurre con el modelo de usuario descrito anteriormente, se trata de información preprocesada por Twitter.

3.2. Definición de la clasificación teórica de la información

Una vez revisada la información que es posible extraer desde Twitter, se define tres ejes de clasificación para la información:

1. Categoría de la información.
2. Nivel de procesamiento.
3. Modelo de la respuesta obtenida.

3.2.1. Clasificación por categoría de la información

Se definen las siguientes cinco categorías de información. Estas categorías no son excluyentes entre sí por lo que se permite que la información extraída de Twitter pertenezca a la vez a varias de estas categorías:

1. **Contenido.** Aquella información que es introducida por los usuarios voluntariamente, ya sea textual o multimedia. El ejemplo principal de contenido son los *tweets*, pero también pertenecerán a esta categoría los mensajes directos o la información de perfil relativa a la biografía del usuario.
2. **Social.** Aquella información que está relacionada de alguna forma con otra información, generando una relación social alrededor de un contenido o de un usuario. Por ejemplo, un *tweet* que contiene una mención a un usuario, los miembros de una lista, etc.
3. **Localización.** Aquella información de la que disponemos de datos geográficos. Habitualmente estos datos serán relativos al lugar de creación de dicha información, pero también está la posibilidad de obtener diferentes datos procesados según la ubicación. Por ejemplo: *tweets* localizados en un área, *trending topics* de una ubicación.
4. **Temporal.** Aquella información que puede cambiar según el momento en el que es consultada. Obviamente, la gran mayoría de la información es susceptible de cambiar: los *tweets* pueden ser borrados, las listas modificadas, un usuario puede dejar de seguir a otro, etc., pero sólo categorizamos la información como temporal si este cambio es por definición. Por ejemplo, las recomendaciones de Twitter en un instante determinado o el listado de *trending topics,* ya que son diferentes en cada momento.
5. **Interés.** Aquella información que lleva implícitamente datos sobre la relevancia de ésta, ya sea dentro de toda la red o para cierto número de personas. Por ejemplo, el número de *retweets* para un *tweet* o el número de veces marcado como favorito.

3.2.2. Clasificación por nivel de procesamiento

La gran mayoría de los datos que se extraen de Twitter son datos que no tienen ningún procesamiento por parte de Twitter antes de ofrecerlos. Sin embargo, ciertos datos son resultado de algún tipo de procesamiento interno por parte de Twitter, por lo que se define la siguiente clasificación:

1. **Información en bruto.** Información que Twitter nos ofrece sin que haya un procesamiento interno previo. Constituye la gran mayoría de la información que nos ofrece. Por ejemplo, la lista de usuarios a los que sigue un usuario o el número de *retweets* que ha tenido un *tweet.*
2. **Información procesada.** Se trata de información que Twitter ha generado a partir de la información en bruto. A diferencia de la información sin procesar, no se sabe de qué forma la obtiene, y por lo tanto no es posible confiar en su fiabilidad o precisión. Ejemplos de está información son la lista de usuarios recomendados a un usuario o la búsqueda de *tweets* que contienen unos términos concretos.

3.2.3. Clasificación según el modelo al que pertenece la respuesta

También se puede clasificar la información dependiendo del modelo al que pertenezca la respuesta de Twitter. Es importante entender que estos modelos no equivalen a los modelos que ha definido Twitter y que se han analizado en el apartado 3.1.3. Se definen de forma abstracta tres modelos de información. Esta clasificación es no excluyente por lo que una información podrá estar relacionada al mismo tiempo con varios modelos:

1. **Modelo usuario**. Integrante de la red que tiene una identidad, y que puede generar contenido y relacionarse con el resto de usuarios. El usuario no tiene porqué ser una persona física: un usuario puede ser la representación de una empresa o de un producto, por ejemplo.
2. **Modelo *tweet***. Unidad básica de contenido de Twitter, formada por hasta 140 caracteres, y que puede contener texto, información audiovisual y/o enlaces.
3. **Modelo de actividad.** Definimos como actividad aquella información que establece una relación dentro de Twitter. Por ejemplo, una mención, un *retweet* o marcar un *tweet* como favorito.

3.3. Clasificación de la información extraíble

En este apartado se van a recorrer los puntos de información que han sido descritos en el apartado 3.1 y a clasificarlos según las definiciones establecidas en el apartado 3.2:

1. ***Tweets* publicados por el usuario.** Pertenecen exclusivamente a la categoría de **contenido.** No se tiene en cuenta el hecho de que sea posible que existan *retweets*, menciones y *tweets* marcados como favoritos en la lista de *tweets* (determinando así que no pertenecen a las categorías social o de interés). Tampoco se tiene en cuenta que el volumen de *tweets* cambie a lo largo del tiempo (no pertenecerá a la categoría temporal) ni la posibilidad de que estos *tweets* sean geolocalizados, puesto que ya existe una llamada para tratar expresamente este punto (no pertenecerá tampoco a la categoría localización). El modelo de datos con el que trabaja es el **modelo *tweet***. Se trata de **información en bruto.**

2. **Timeline de un usuario.** El *timeline* de un usuario es ha definido como un conjunto de *tweets,* por lo que es en primer lugar información de **contenido.** Está formado por las publicaciones de las personas a las que sigue el usuario más las suyas propias. Es decir, son los *tweets* que el usuario ha decidido recibir, por lo que pertenece también a la categoría de **interés**. El modelo de datos relacionado es el **modelo *tweet.*** Se trata de **información en bruto**.

3. **Mensajes privados entre usuarios.** El hecho de mandar o recibir un mensaje directo de otro usuario establece implícitamente una relación social entre ellos. El mensaje en sí mismo lo categorizamos como **contenido**, y la relación entre usuarios implica la categorización **social.** Salvando unos pequeños detalles relacionados con la visibilidad del mensaje, los mensajes privados pertenecen al **modelo *tweet***. Se trata de **información en bruto**

4. ***Tweets retweeteados.*** A pesar de que el objeto sobre el que se está trabajando es un *tweet* (y por tanto se habla de contenido), se considera que un *retweet* no genera nuevo contenido a la red, sino que establece una relación de interés o actividad. Siguiendo el mismo proceso que en el punto 1, abstraemos una serie de características: la posibilidad de menciones a otros usuarios, el cambio de volumen temporal y la geolocalización. Por ello pertenece únicamente a la categoría **interés**. Existen dos modelos actuando sobre esta relación: el **modelo *tweet*** y el **modelo de actividad**. Se trata de **información en bruto.**

5. ***Tweets* marcados como favoritos por el usuario.** Se aplican los mismos criterios que el punto anterior: el objetivo central de este punto es la relevancia con la que ha dotado un usuario a un contenido concreto. Por ello, la categoría a la que pertenece esta información es la de **interés**. Los dos modelos que actúan son el **modelo *tweet*** y el **modelo de actividad**. Se trata de **información en bruto**.

6. ***Tweets* con mención al usuario.** De forma análoga al razonamiento en los puntos 3, 4 y 5, se destaca que la información principal es la relación entre el usuario que publica el *tweet* y el usuario mencionado en él. Es decir, se considera al *tweet* en sí como un medio, por lo que no pertenece a la categoría de contenido y sí a la **social**. Entran en juego los tres modelos de datos de los que disponemos: el **modelo de usuario**, el **modelo *tweet***, y el **modelo de actividad.** Se trata de **información en bruto**.

7. **Lista de usuarios a los que sigue el usuario.** Por definición, la acción de seguir a una persona en Twitter indica un interés potencial en lo que la persona pueda publicar en la red. Parece lógico por lo tanto asignar la categoría de **interés**. Además, el hecho de que una persona siga a otra establece también una relación unidireccional entre ellas. Dado un usuario, se podría definir el grafo de "personas a las que sigue" creando una red de información con el propio usuario como centro. Esto hace que pertenezca también a la categoría **social.** Existen dos modelos de información relacionados: el **modelo de usuario** y el **modelo de actividad**. Se trata de **información en bruto**.

8. **Lista de *followers.*** El razonamiento seguido es el mismo que para el punto anterior: un *follower* es una persona potencialmente interesada en las publicaciones de a quien sigue, así que pertenece a la categoría de **interés**. También aquí es posible considerar un grafo entre el usuario y la lista de usuarios que siguen al usuario; de nuevo, información **social.** Trabaja con el **modelo de usuario** y el **modelo de actividad**. Se trata de **información en bruto**.

9. **Relación entre usuarios.** Este punto guarda una estrecha relación con el punto 7 (lista de usuarios a los que sigue el usuario) y 8 (lista de *followers*), aportando el mismo tipo de información, por lo que también pertenece a las categorías **social** e **interés**. La respuesta está relacionada con el **modelo de usuario** y el **modelo de actividad**. Se trata de **información en bruto**.

10. **Número de *followers.*** Repitiendo el razonamiento del punto 8 (lista de *followers*), las categorías a las que pertenece son **interés** y **social**. Los modelos, igual que antes, son el **modelo de usuario** y el **modelo de actividad**. Se trata de **información en bruto**.

11. **Usuarios bloqueados** y **usuarios sin posibilidad de *retweet***. Se han unificado ambos puntos ya que ambos puntos aportan la misma información. La diferencia entre una y otra está a nivel de definición de Twitter, entre bloquear total o parcialmente a un usuario. Pertenece a la categoría **social** porque se está definiendo una relación entre usuarios, y a la categoría de **interés**, aunque en este caso la información obtenida es la negación de interés de un usuario por otro. Los modelos implicados son el **modelo de usuario** y el **modelo de actividad**. Se trata de **información en bruto**.

12. **Biografía** y **descripción de un usuario**. Ambos puntos aportan el mismo tipo de información, en este caso información que ha sido introducida por el propio usuario relacionada con su perfil. La categoría a la que pertenece es **contenido**. El modelo al que pertenece es el **modelo de usuario**. Se trata de **información en bruto**.

13. **Url publicada por el usuario.** Se trata, al igual que el punto anterior (biografía y descripción de un usuario), de información de perfil introducida por el propio usuario. El motivo por el que se ha separado la información extraíble de un usuario (usuario entendido como entidad definida por Twitter) es que es posible identificar distintos campos con diferentes categorías. En concreto, la url publicada por el usuario pertenece a la categoría **social** ya que está creando una pequeña red de información externa en torno a sí mismo. El modelo de información al que pertenece la respuesta es el **modelo de actividad**. Se trata de **información en bruto**.

14. **Número de veces marcado un *tweet* como favorito.** Esta información puede ser entendida como un reflejo de la relevancia o interés que este usuario suscita dentro de la red. Debido a esto se le asigna la categoría de **interés**. Pertenece el **modelo *tweet*** y el **modelo de actividad**. Se trata de **información en bruto**.

15. **Listas a las que está suscrito el usuario.** las categorías involucradas son **social** ya que partiendo del usuario es posible establecer una red de información alrededor del usuario, e **interés** al tratarse de una relación dentro de Twitter. Actúa sobre el **modelo de actividad**. Se trata de **información en bruto**.

16. **Usuarios que han *retweeteado* un *tweet.*** A pesar de que los objetos sobre los que se interactúa son *tweets* y usuarios, la información a destacar sobre este punto es de **interés** por parte de un número usuarios sobre un contenido concreto. Se considera el **modelo de usuario** y el **modelo de actividad.** Se trata de **información en bruto**.

17. ***Retweets* de un *tweet.*** Igual que en el punto anterior, se ignora el contenido en sí, y destacando el movimiento o **interés** que ha generado el *tweet*. Los modelos que involucra, y ésta es la única diferencia con el punto anterior, son el **modelo de actividad** y el **modelo *tweet***. Se trata de **información en bruto**.

18. **Cadena de respuestas a un *tweet.*** De nuevo no se tiene en cuenta el contenido que ha podido generar la cadena de *tweets* y se determinan como principales las categorías de **social** e **interés**. Se destacan dos modelos involucrados: el **modelo *tweet*** y el **modelo de actividad**. Se trata de **información en bruto**.

19. **Localización de un *tweet.*** Las coordenadas geográficas de los *tweets* es información que pertenece a la categoría de **localización**. El único modelo implicado en la respuesta es el **modelo *tweet***. Es **información procesada.**

20. **Búsqueda de usuarios.** La categorización es de **contenido** ya que el nombre de usuario no deja de ser información que ha aportado un usuario a la red. El modelo involucrado es el de **modelo de usuario**. Aunque mínimo, existe un procesamiento por parte de Twitter por lo que es **información procesada**.

21. **Búsqueda de *tweets*** y **búsqueda avanzada de *tweets*.** Twitter permite un abanico muy grande de criterios de búsqueda en la búsqueda de *tweets*. Es posible realizar consultas orientadas a la obtención de *tweets* en función del **contenido**, de la interacción **social** con otros usuarios, de la **localización** o del instante de tiempo (categoría **temporal**) en que se publicó el *tweet*. Los modelos involucrados son el **modelo *tweet*** y el **modelo de actividad** en caso de que se trate de búsqueda de *tweets* en función de la interacción social con otros usuarios. Es **información procesada**.

22. **Buscar usuarios recomendados.** Pertenece a la categoría **interés** ya que las recomendaciones están hechas en función de los intereses mostrados por los usuarios. Tal y como explica en la documentación de Twitter, esta información tiene una fuerte dependencia

temporal, así que también pertenece a la categoría **temporal**. Se considera al **modelo usuario**. La **información está procesada.**

23. **Buscar usuarios recomendados para una categoría.** Twitter determina ciertos usuarios claves para una serie de categorías definidas previamente por el propio Twitter. Pertenece principalmente a la categoría **interés** ya que se supone que estos usuarios son "líderes" en su ámbito. Sin embargo esta vez se determina que pertenece también a la categoría **temporal** ya que este "status de liderazgo" es completamente temporal, tal y como se expresa en la documentación de Twitter. Pertenece únicamente al **modelo usuario**. Es **información procesada**.

24. ***Tweets* con contenido multimedia.** Es importante recordar que el contenido multimedia identifica tanto urls en el *tweet*, las cuales crean información alrededor del contenido publicado (categoría **social**)**,** como imágenes o archivos de vídeo (**contenido**)**.** El único modelo presente en la respuesta es el **modelo *tweet*.** Aunque mínimo, sí que se considera que hay procesamiento por parte de Twitter ya que se ha aplicado un filtrado a un conjunto de *tweets*, por lo que es **información procesada**.

25. **Lista de *trending topics* por localización.** La naturaleza de los trending topic hacen que este punto sea clasificado de las categorías de **interés**, **temporal** y **localización.** El modelo al que pertenece la respuesta es el **modelo de actividad**. Como se ha comentado en el punto anterior existe un procesamiento previo por parte de Twitter, por lo que es **información procesada**.

La Tabla 3.1 condensa toda la información expuesta durante este apartado. Se han usado las siguientes abreviaturas: el modelo usuario se identifica con la letra U, el modelo *tweet* con la letra T, y el modelo de actividad con la letra A. Además, en la columna que muestra el nivel de procesamiento de la información, “no” significa que son datos en bruto y “sí” que es información procesada.

	Categorías	Model	Procesamient
Tweets del usuario	Contenido	T	No
Timeline del usuario	Contenido, Interés	T, A	No
Mensajes directos	Contenido, Social	T	No
Tweets retweeteados	Interés	T, A	No
Tweets favoritos	Interés	T, A	No
Tweets con mención	Social	U, T, A	No
Lista de amigos	Social, Interés	U, A	No
Lista de *followers*	Social, Interés	U, A	No
Relación entre usuarios	Social, Interés	U, A	No
Número de *followers*	Social, Interés	U, A	No
Usuarios bloqueados	Social, Interés	U, A	No
Biografía de un usuario	Contenido	U	No
Url publicada por el usuario	Social	A	No
Veces marcado como favorito	Interés	T, A	No
Listas suscritas	Social, Interés	U, A	No
Usuarios que han *retweetado* un	Interés	U, A	No
Retweets de un *tweet*	Interés	T, A	No
Cadena de respuestas a un *tweet*	Social	T, A	No
Localización de un *tweet*	Localización	T	Sí
Buscar usuarios	Contenido	U	Sí
Búsqueda de *tweets* por contenido	Contenido	T	Sí
Búsqueda de *tweets* por interacción	Social	T, A	Sí
Búsqueda de *tweets* por temporalidad	Temporal	T	Sí
Búsqueda de *tweets* por localización	Localización	T	Sí
Usarios recomendados	Interés, Temporal	U	Sí
Usuarios recomendados para	Interés, Temporal	U	Sí
Tweets con contenido multimedia	Contenido, Social	T	Sí
Trending topics por localización	Interés, Localización, Temporal	A	Sí

Tabla 3.1. Resumen para la clasificación de la información extraíble

Por último, se muestra de forma visual la distribución de puntos de información de la información extraíble según los tres ejes de clasificación de la información. Fijándonos en la Figura 3.1, en la que se puede ver la distribución de los puntos de información para cada una de las cinco categorías definidas, es posible notar que existe una gran inclinación hacia las categorías social y de interés. La distribución de puntos de información relacionados con cada uno de los modelos posibles es muy regular y se recoge en la figura 3.2. Por último, los porcentajes de distribución de puntos de información que implican información en bruto e información procesada se muestran en la Figura 3.3.

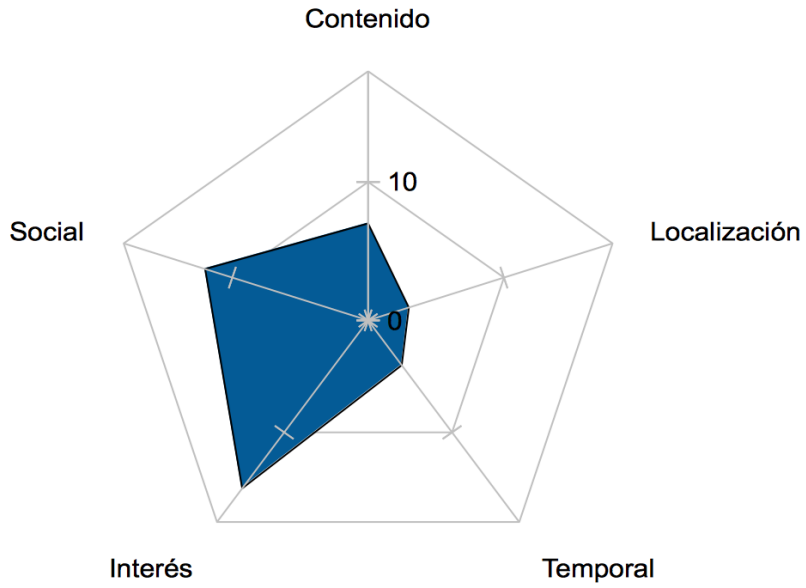

Figura 3.1. Distribución del número de puntos de información para cada una de las cinco categorías definidas

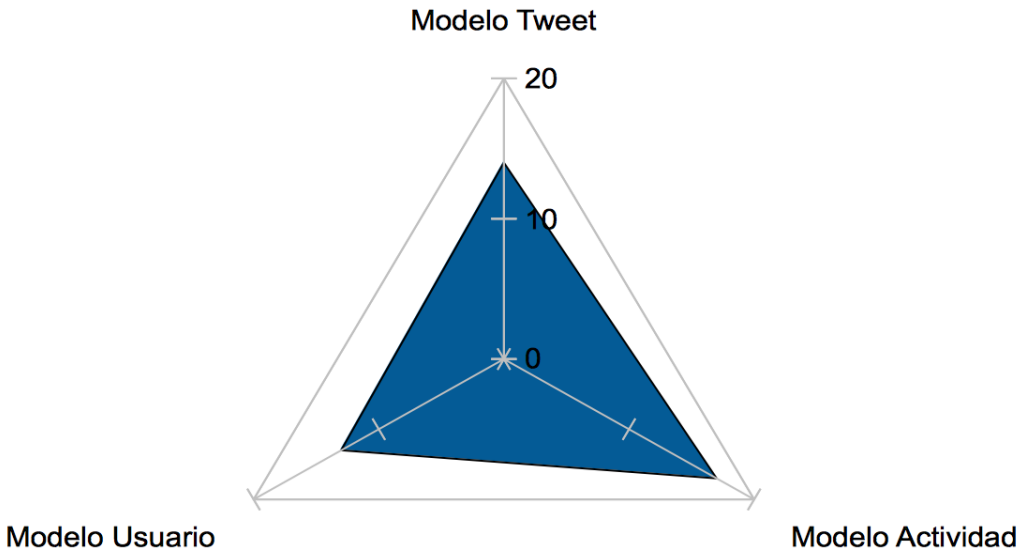

Figura 3.2. Distribución del número de puntos de información para cada uno de los modelos de la respuesta

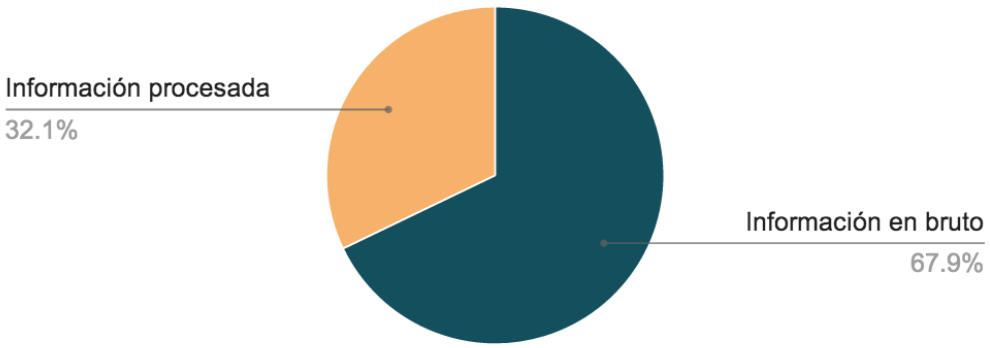

Figura 3.3. Porcentajes de distribución de puntos de información que implican información en bruto y procesada

3.4. Cotejo de la clasificación teórica definida con servicios basados en el análisis y tratamiento de información extraída de Twitter.

Se ha analizado la información ofrecida por los servicios relacionados con la extracción y procesamiento de la información facilitada por Twitter estudiados en el apartado 2.4 para clasificarla según el marco teórico definido en este capítulo con el fin de demostrar que nuestra clasificación es capaz de englobar, desde el punto de vista teórico, la información extraíble desde Twitter.

SecondSync ofrece un análisis del impacto de programas de la televisión sobre la red social Twitter. Dicho análisis está basado principalmente en la categoría de **contenido** y **temporalidad**, ya que se centra en la cantidad de *tweets* que se han generado sobre los programas de la televisión durante las franjas horarias en las que estos se emitían. Además de contabilizar los *tweets* que se generaron, el servicio estudia el sentimiento del contenido del *tweet* y por ello determinamos que trabaja directamente con el **modelo tweet**. La aplicación también detecta el género de los usuarios, por lo que el segundo modelo con el que trabaja es el **modelo usuario**. La herramienta no emplea información perteneciente a la categoría de social, localización o interés ni tampoco trabaja con el modelo que se ha definido como modelo de actividad.

BluefinLabs ofrece un servicio similar a SecondSync generando información clasificada en nuestro modelo en las categorías de **contenido** y **temporalidad**. La diferencia fundamental entre SecondSync y Bluefin se encuentra en que el segundo sí hace uso de información perteneciente a la categoría de **social:** el servicio relaciona los datos cuantitativos con información externa a Twitter para ofrecer estadísticas sobre la percepción de campañas publicitarias. Es decir, a partir de información de Twitter busca generar una red de conocimiento alrededor de un contenido. Se determina que los modelos de datos sobre los que trabaja son el **modelo *tweet*** y el **modelo de actividad.**

Por último, Follower Wonk establece lo que denomina "wonk social authority" o influencia del usuario dentro de la red. En el apartado 3.2.1 se ha definido la información perteneciente a la

categoría de interés como información que conlleva implícitamente datos sobre la relevancia de ésta por ello, la categoría principal sobre la que se centra este servicio es la categoría de **interés**. Para calcular la influencia del usuario dentro de la red hace uso del número de *retweets* que han tenido los últimos *tweets* del usuario y del número de *followers* del usuario. Es decir, emplea también información perteneciente a las categorías **social** y **temporal**. Sin embargo no se determina que haga uso de información categorizada como contenido ni localización. El modelo más relevante para este servicio es el **modelo de actividad**.

En la Tabla 3.2 se recoge un resumen del cotejo de la clasificación teórica definida con los servicios estudiados.

	Categorías de la información empleadas	Modelo de datos con el que trabaja	Nivel de procesamiento de la información
SecondSync	Contenido, temporalidad	Tweet, usuario	Información procesada
Bluefin	Contenido, temporalidad, social	Tweet, actividad	Información procesada
FollowerWonk	Interés, social, temporal	Actividad	Información procesada

Tabla 3.2. Tabla resumen del cotejo de la clasificación teórica con servicios basados en información de Twitter

3.5. Conclusiones

Durante este capítulo se ha especificado cuál es exactamente la información que se puede conseguir usando de Twitter. Se han especificado cuáles son los métodos para acceder a esta información y se ha detallado tanto la información que se obtiene de forma explícita como la que se consigue de forma implícita al usar estos métodos.

Una vez realizado un análisis que aportase una visión general de toda esta información se emprendió la tarea de clasificarla. Para ello, primero se ha definido un marco teórico que diera cabida a toda la información atendiendo a diferentes aspectos de la naturaleza de la misma. A continuación

se ha realizado una clasificación basada en el marco teórico definido y se ha expuesto una representación gráfica de ésta que facilitase su comprensión.

Por último, se ha realizado un cotejo de la información ofrecida por los servicios relacionados con la extracción y procesamiento de la información facilitada por Twitter estudiados en el apartado 2.4 con la clasificación teórica de la información definida en el capítulo.

4. Arquitectura del *backend*

Una vez realizado el estudio teórico de la información extraíble de Twitter, se ha diseñado e implementado un *backend* con la funcionalidad necesaria para responder a los objetivos planteados en el apartado 1.1 relativos a la extracción, análisis y visualización de información social. Este *backend* ha de ser capaz de extraer y almacenar información de Twitter, implementar varias estrategias de generación de información procesada en español y por último aportar las facilidades necesarias para que toda esta información almacenada pueda ser visualizada.

En primer lugar es necesario determinar cuáles serán las estrategias de generación de información procesada en español. En la definición del marco teórico para la clasificación de información, establecimos que en Twitter existían tres modelos abstractos de datos que interactúan entre sí generando la información: los **usuarios** publican contenido en forma de ***tweets***, generando así **actividad** (relaciones) dentro de la red. Se busca que la información generada por nuestro sistema sirva para enriquecer esta conexión: los usuarios, que tienen **género**, publican contenido, que pertenece a una **categoría**, en forma de *tweets* generando así actividad, que se produce en un **lugar.** Por lo tanto, el análisis de información de Twitter se establece como la detección de género de los usuarios, la categorización de *tweets* en función del contenido y la geolocalización de publicaciones de contenido en tiempo real.

En este capítulo se detalla la funcionalidad definida para el *backend* así como el diseño en módulos adoptado. A continuación se explica cuál es el flujo de ejecución al iniciar el sistema, destacando cuáles son las tareas que se ejecutan concurrentemente. Una vez que se han establecido las funcionalidades deseadas y el flujo de ejecución, se explican las decisiones tecnológicas adoptadas y el modelo de datos adoptado para cubrir cada una de las categorías que se definieron en el marco teórico de la clasificación de la información. A continuación se detallan cada uno de los módulos que configuran el *backend* de este proyecto: el extractor de tweets que abre la conexión con Twitter y almacena en base de datos la información recibida, el detector de géneros para usuarios basado en la información de perfil de Twitter, el categorizador de *tweets* basado en contenido y el

traductor de áreas que facilita la tarea de consultar áreas geográficas. Por último se explica la API implementada para consultar la información almacenada en la base de datos.

4.1. Funcionalidad y diseño de la arquitectura del backend

Se ha diseñado e implementado un *backend* capaz de realizar las siguientes funciones:

- **Conectarse con la Streaming API de Twitter para extraer información.** Como se vio en el apartado 3.1.2, esta conexión permitirá extraer *tweets* publicados en tiempo real que cumplan una serie de requisitos fijados en el momento de crear la conexión. Estos filtros pueden ser de diverso género, pero en el proyecto se limitan a solicitar *tweets* que presenten información geográfica y que estén escritos en español. Esta funcionalidad es realizada por el módulo denominado "extractor de *tweets",* que ha sido detallado en mayor profundidad en el apartado 4.5.
- **Desgranar la información en entidades y almacenarla en una base de datos.** Al recibir un *tweet* se identificará toda la información que contiene, clasificándola dentro de las entidades que componen el modelo de datos definido en el apartado 4.4. Una vez creadas estas entidades se almacenarán en una base de datos, de forma que en el futuro éstas puedan ser consultadas.
- **Detectar el género de un usuario de Twitter.** Mediante el análisis de los *tweets* recibidos de Twitter se podrá asignar un género al usuario que lo ha publicado. Para ello se han diseñado varios algoritmos con enfoques diferentes que posteriormente han sido evaluados para elegir el más preciso, como se puede ver en el apartado 4.6. El módulo que realiza esta funcionalidad se ha denominado "detector de género".
- **Categorizar *tweets* por contenido.** Partiendo del texto que contiene cada *tweet,* se ha construido un clasificador de *tweets* que sea capaz de asignar una categoría a éste. De nuevo se han implementado varios enfoques que han sido evaluados para poder elegir el más eficaz. Se pueden ver más detalles del proceso en el apartado 4.7. El módulo que se encarga de esta tarea es el llamado "categorizador de *tweets*".

- **Traducir peticiones restringidas geográficamente.** Debido a la poco intuitivo que resulta hacer consultas restringidas a un área a la vez que se trabaja con datos expresados en latitud y longitud, se ha desarrollado un módulo que permita usar identificadores de área. De esta forma es posible buscar contenido en una ciudad, por ejemplo, sin tener que conocer las coordenadas de los puntos que la delimita. Todos los detalles de este apartado se encuentran en el apartado 4.8. El módulo que realizar esta función se denomina “traductor de localizaciones”.
- **Exponer toda la información almacenada en la base de datos.** Se ha desarrollado una API que permita acceder desde el exterior a la información presente en la base de datos del sistema, ya sea la recogida inicialmente o la generada por los diferentes módulos. Toda la información relativa a esta API se encuentra en el apartado 4.9.

En la Figura 4.1 se puede ver la distribución de la arquitectura en módulos independientes según las funcionalidades comentadas.

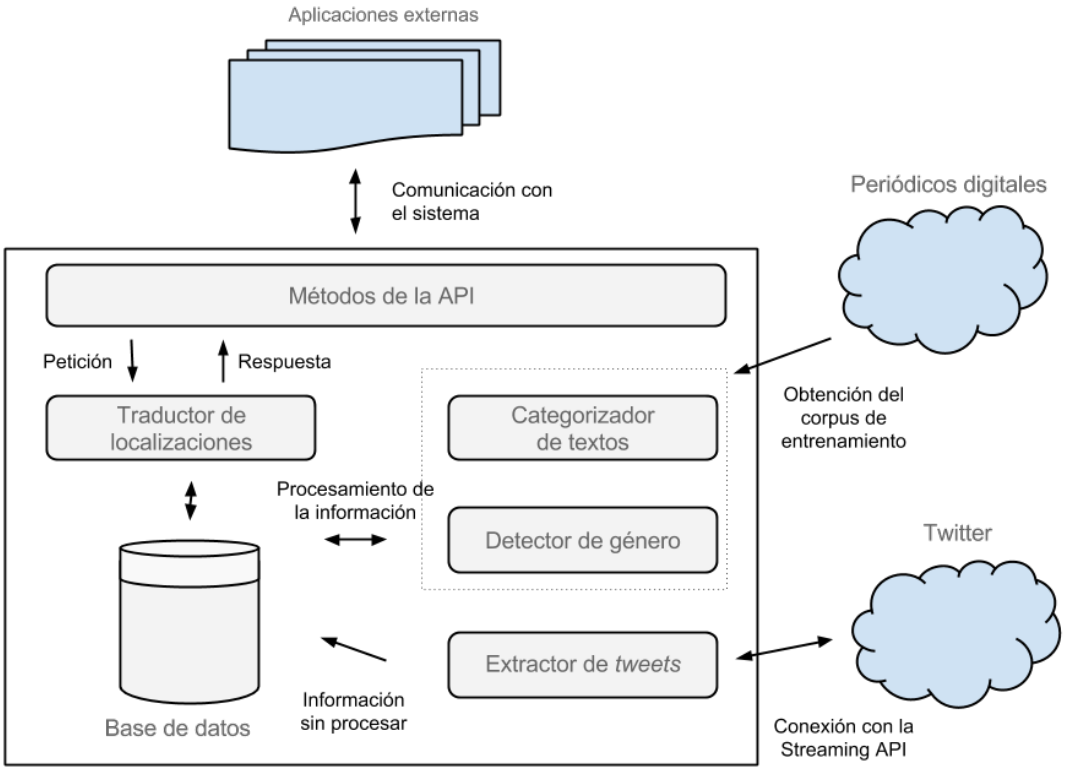

Figura 4.1. Distribución de los diferentes módulos del backend

4.2. Flujos de ejecución

Una de las decisiones esenciales en el diseño de la arquitectura del *backend* es la división del trabajo en tareas que puedan ser ejecutadas concurrentemente por el sistema. Si no se plantease un diseño concurrente, no sería posible extraer datos, almacenarlos, procesarlos y exponerlos para ser consultados por clientes externos consiguiendo tiempos de respuesta aceptables. La tecnología empleada para facilitar la creación, gestión y sincronización de hilos se explica en el apartado 4.3.1.

Al iniciar el sistema, el proceso principal crea y lanza a ejecución tres hilos a los que asigna diferentes tareas. Estos tres hilos se mantienen en ejecución siempre que el sistema esté en funcionamiento. Las tareas asignadas a los hilos de ejecución son las siguientes:

- **Hilo extractor de *tweets***. Este hilo establece y gestiona la conexión con Twitter. Es el encargado de realizar la petición de conexión con Twitter, una vez aceptada mantener esa conexión y gestionar los datos recibidos por ella. El hilo posee la lógica necesaria para capturar cada *tweet* recibido de Twitter, dividirlo según el modelo de datos que definiremos en el apartado 4.4 y guardar sus datos en la base de datos.
- **Hilo detector de género**. Se trata de uno de los dos hilos encargados del procesamiento de información. Al lanzarse a ejecución, el hilo accede a la base de datos buscando usuarios a los que no se haya intentado detectar su género y los procesa. Este hilo cuenta con la siguiente suposición de diseño: *el tiempo de almacenamiento de usuarios que hayan publicado tweets en la base de datos por el extractor de tweets será menor que el tiempo necesario para detectar el género a un usuario, por lo que el hilo de detector de género siempre encontrará usuarios sin procesar en la base de datos.*
- **Hilo categorizador de *tweets.*** Conforma, junto con el detector de género, el segundo hilo encargado del procesamiento de información. Al lanzarse a ejecución cuenta con un proceso de inicialización relativamente complejo ya que realiza una conexión con periódicos digitales para obtener un corpus de entrenamiento (se explica en el aparado 4.5). Una vez inicializado el módulo categorizador, el hilo accede a la base de datos buscando *tweets* que no hayan sido categorizados y los procesa. Al igual que el hilo detecto de género, este hilo cuenta con una suposición de diseño: *el tiempo de almacenamiento de tweets en la base datos por el extractor*

de tweets será menor que el tiempo necesario para categorizar el contenido de un tweet, por lo que el hilo categorizador de tweets siempre encontrará usuarios sin procesar en la base de datos.

Finalmente, en el momento en el que el extractor de *tweets* ha realizado la conexión y está guardando datos provenientes de Twitter en la base de datos, y los dos hilos de procesamiento de la información están se han inicializado, el backend está listo para procesar una petición proveniente de la API. En la figura 4.2 se ve el diagrama de secuencia que modela este funcionamiento. Los sistemas externos al *backend* se han reflejado en azul.

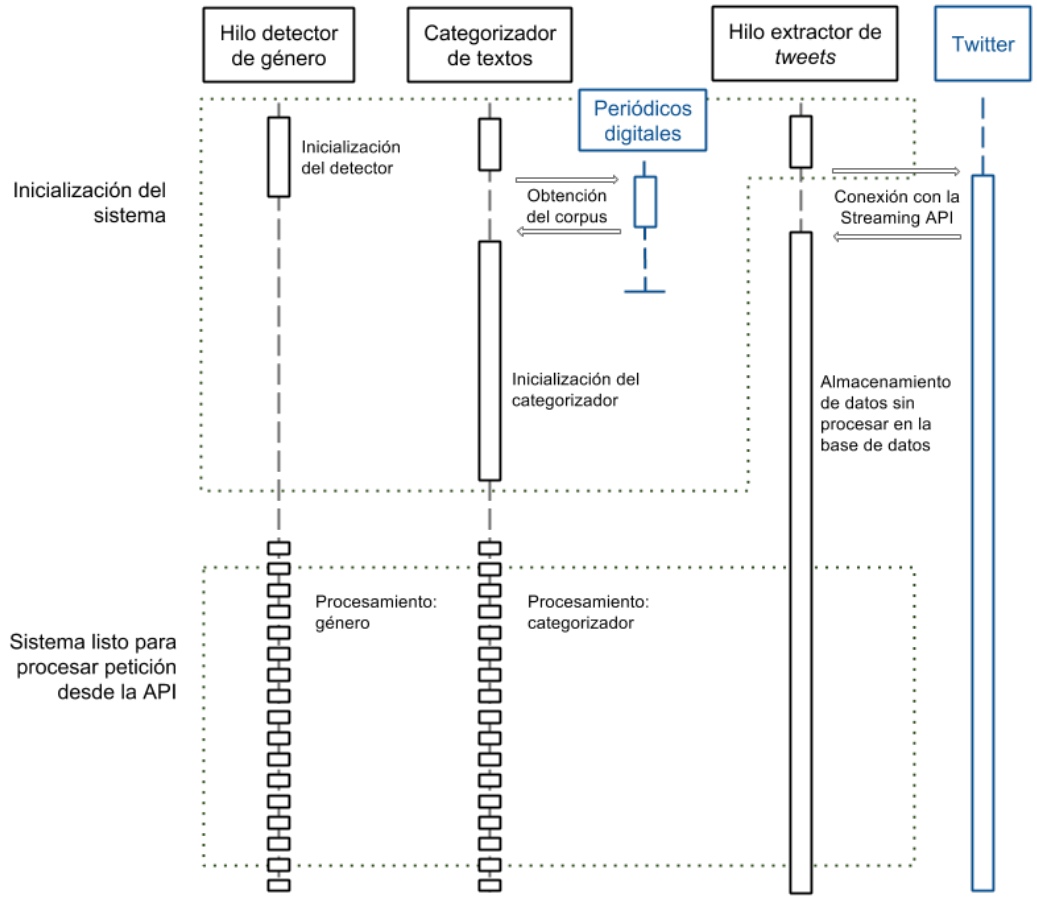

Figura 4.2. Flujo de ejecución del backend

4.3. Tecnologías usadas

4.3.1. Tecnologías usadas en la implementación del *backend*

Nuestro proyecto consiste en un *backend* sólido, ya que debe ser capaz de recibir, procesar y guardar alrededor de cien de *tweets* al segundo. Estas tareas deben hacerse sin que esto afecte a la API Rest que expone los datos almacenados para que puedan ser consumidos por servicios externos. Para que cumpla con estos objetivos se ha optado por en un backend programado en Java, usando Play! Framework 2[15], que delega la parte de concurrencia en Akka[16], y que está conectado a una base de datos MongoDB[17]. Todo ello desplegado en Amazon Web Services[18]. Las razones de cada una de las elecciones son las siguientes:

- **Play! Framework 2** y **Akka**. Juntos son la pareja perfecta para proyectos con problemas de concurrencia. Con Play! se consiguió tener en poco tiempo un servidor que respondiese a las peticiones a la API Rest, además de crear el *websocket* para la visualización en tiempo real que se detallará en el apartado 5.1.1, mientras que Akka permitió abstraer el proyecto de la programación concurrente, ya que su gestión de hilos, en Akka denominados *actores*, aporta toda la potencia de la programación concurrente en Java pero sin apenas problemas. Akka se encarga de gestionar y de auto-reiniciar hilos que han terminado inesperadamente, gestiona transparentemente una cola de tareas para cada hilo y se encarga del paso de mensajes entre hilos.
- Base de datos en **MongoDB**. Se decidió que se quería trabajar con una base de datos no relacional porque el sistema se basa en un hilo escribiendo en la base de datos (el extractor de *tweets*) y luego muchas peticiones leyendo esos datos de la base de datos. Por lo tanto, parecía una solución natural usar un tipo de base de datos que premiase la velocidad de lectura

[15] *http://www.playframework.com*

[16] *http://akka.io*

[17] *http://www.mongodb.org*

[18] *http://aws.amazon.com/es*

sacrificando un poco la atomicidad de las operaciones, ya que no se estaba trabajando con información sensible. Además, como se tiene el control la base de datos y la API Rest, hemos podido optimizar MongoDB mediante índices, haciendo que algunos cuellos de botella como peticiones a la API Rest muy costosas, fuesen eliminados.

- **Amazon Web Services**. Todo el sistema está desplegado en Amazon Web Services, más concretamente en una instancia m3 *medium*, conectado a un disco externo EBS de 100gb. La razón de esto es que se quería un sistema que permitiese ampliar la capacidad de procesamiento y de almacenamiento sin problema, ya que en un principio no se sabía cual iba a ser el alcance de nuestros datos, ni lo que iban a ocupar una vez procesados. Además, ha permitido ahorrar costes, ya que en Amazon tú puedes decidir pagar por periodo de facturación o por horas, y durante casi todo el periodo de desarrollo la máquina estaba apagada, por lo que no tenía coste, pero nuestros datos seguían disponibles. Hacia el final de este periodo se contrató la máquina actual, mucho más potente, para poder poner el servidor a pleno rendimiento guardando *tweets*. Este proceso de migración fue muy sencillo, ya que Amazon te permite hacer un *snapshot* (copia) de la instancia vieja, y copiar dicho *snapshot* en la instancia nueva.

4.3.2. Herramientas utilizadas para la categorización de textos

La categorización de textos es un área de la recuperación de la información que cuenta con mucho trabajo ya realizado. Sin embargo, estos trabajos no se han concentrado en general en las particularidades que presentan los textos cortos [Rosas, 2010]. Para este proyecto se han utilizado dos herramientas orientadas a la categorización de textos y aprendizaje automático con el fin de determinar cuál se adapta mejor en la categorización de textos cortos: Weka y Lucene

Weka[19] (acrónimo de Waikato Environment for Knowledge Analysis) es una herramienta software bajo licencia GPL para el análisis de datos; en concreto aporta una serie de librerías útiles para trabajar con aprendizaje automático sobre cualquier conjunto de datos.

[19] *http://www.cs.waikato.ac.nz/ml/weka/*

Apache Lucene[20] es una API orientada en la recuperación de información, distribuida bajo licencia Apache Software License. Aporta una serie de funciones útiles para el indexado y la recuperación de textos.

4.4. Modelo de datos

Para cumplir nuestro objetivo de extraer y analizar información de Twitter es necesario definir un modelo de datos mediante el cual seamos capaces tanto de desgranar la información presente en lo que nos envía Twitter como de almacenarla para su posterior uso. Por otro lado, para ser capaces de permitir la visualización de la información, nuestro modelo debe estar dividido en entidades de relevancia similar, de forma que sea sencillo extraer información de él.

Observando el análisis realizado en el marco teórico llegamos a la conclusión de que nuestro modelo de datos debe centrarse en los *tweets*, ya que analizando la información que estos contienen tendremos disponible información de todas las categorías que se han definido previamente (contenido, social, interés, temporalidad y localización). Por otro lado, nuestro modelo de datos debe ser capaz en todo momento de proveer información sobre dónde y cuándo fue creada cualquier información que contenga. De esta forma, es posible dotar de contexto a la información lo que permite, por ejemplo, su visualización en tiempo real.

Una vez decidido que nuestro enfoque se basa en los *tweets*, nuestro siguiente paso ha sido definir cómo desgranamos la información en diferentes entidades dentro del modelo de datos para poder guardarlas de forma independiente. De esta forma se da la posibilidad de consultarlas mediante nuestra API Rest. Las entidades que finalmente componen el modelo de datos de este proyecto son las siguientes:

1. ***Tweet***. Datos de un mensaje de texto escrito por un usuario en Twitter. Contiene los mismos datos que el original. En un principio se planificó guardar solo la información relativa al contenido que íbamos a usar, pero más adelante nos dimos cuenta que podría ser

[20] *http://lucene.apache.org/*

interesante tener todos los datos, ya que podríamos extraer más información a posteriori si fuese necesario, así como guardar una copia de los datos originales. La decisión se basó en que el aumento de tamaño por cada elemento no es significativo.

2. **Usuario**. Datos de un usuario de Twitter. Contiene los datos del usuario que publicó el *tweet* que recibimos en el momento que lo escribió. Los datos principales que contiene son su nombre real, su identificador dentro de Twitter y su biografía.
3. **Hashtag.** Datos de un hashtag de Twitter, entendiendo por ello las palabras de un *tweet* que comienzan por #. Contiene la palabra en sí, una relación al *tweet* en el que apareció e información geográfica de dónde fue publicado ese *tweet*.
4. **Links.** Hipervínculo que puede enlazar tanto a contenido multimedia dentro de Twitter como a páginas externas. Contiene la dirección y, al igual que los hashtags, una relación al *tweet* en el que apareció e información geográfica.

La relación completa de atributos para cada una de las entidades que conforman el modelo de datos se puede ver en el Anexo A.

Por último destacar que con este modelo de datos nos asegurarnos de que cubrimos cada una de las categorías que se definieron en el apartado 3.2.1: los *tweets* aportan **contenido**, los links aportan información **social** y los hashtags **interés**, además de **localización** y todos ellos contienen información **temporalidad.** De esta forma el modelo de datos propuesto cubre las necesidades de nuestro proyecto así como está preparado para dar soporte a futuras ampliaciones.

4.5. Extracción y almacenamiento de *tweets* en la base de datos

Para las necesidades de información de este *backend* se decidió usar un enfoque basado completamente en las oportunidades que nos ofrece Twitter mediante su Streaming API, pues ofrece una gran cantidad de información en tiempo real. Es importante notar que como se ha visto en el marco teórico del capítulo anterior, Twitter Streaming provee todos los *tweets* que cumplan una serie de requisitos fijados por el solicitante (en este proyecto, *tweets* en español con información

geográfica), pero con un máximo del 1% del total del tráfico en todo Twitter. En este caso, esa limitación no debería suponer un problema, puesto que sólo el 2.02% de los *tweets* tienen información geográfica [Leetaru, 2013], y a dichos *tweets* aún habría que aplicarle el filtro de idioma.

Al iniciar el sistema se crea un hilo, en Akka llamado actor, que solicita la conexión a la Streaming API de Twitter mediante una petición POST a la Streaming API de Twitter. Hay varios posibles parámetros por los que filtrar, como se vio en el apartado 3.1.2, de los que se usan los siguientes:

- **Localización**. Se usa este parámetro para limitar los *tweets* que recibimos a aquellos que contienen datos sobre la ubicación dónde fueron creados.
- **Idioma**. Se usa este parámetro para solicitar solamente *tweets* escritos en español.

Una vez definidos los parámetros, el último paso consiste en definir un *listener* que será invocado cuando se reciba un *tweet.* Finalmente, se crea la conexión con Twitter, qué permanecerá abierta hasta que se decida cerrarla. Cada *tweet* que cumpla las condiciones impuestas y las limitaciones internas de Twitter será enviado mediante esta conexión. Es en el *listener* dónde podemos controlar qué acciones realizamos sobre la información recibida. En este caso, este *listener* invoca a las funciones necesarias para separar el *tweet* en las diferentes entidades de nuestro modelo de datos definido en el apartado 4.4 y almacenarlo en la base de datos. Para ello, cada una de las entidades está representada en el sistema mediante una clase Java en la que se indican sus campos y su tipo de datos asociado así como su relación con la base de datos. Para esta base de datos se ha elegido MongoDB, ya que como se vio en el apartado 4.3.1, parecía una elección natural debido a su gran rendimiento en lectura de datos poco relacionados entre sí.

4.6. Detección de género de los usuarios

Si bien Twitter provee mediante su API diversos datos personales de los usuarios como foto de perfil, nombre, descripción o idioma, no ocurre lo mismo con el género de éste. Por ello, y porque no fuimos capaces de encontrar nada parecido, nos propusimos crear nuestro propio detector de género partiendo de los datos de los que disponíamos.

Tras un primer análisis, constatamos que las cuentas de Twitter se clasificaban no sólo en personas masculinas y personas femeninas, sino que también hay un cierto porcentaje de cuentas impersonales, tales como las cuentas de empresas o marcas. Se decidió obviar estas cuentas asumiendo que cuando nuestro algoritmo no fuese capaz de identificar al usuario como masculino o femenino, estaría identificando correctamente a ese usuario como impersonal. Esto fue debido a que los datos disponibles en la API de Twitter más prometedores eran el nombre y la descripción, y ninguno de los dos aportaba luz sobre las cuentas impersonales.

En rasgos generales, los pasos que se han seguido para incorporar la detección de género de los usuarios de Twitter a nuestro proyecto han sido los siguientes:

1. Se ha obtenido una muestra representativa de usuarios de Twitter y sus datos asociados: nombre, biografía, lenguaje y situación geográfica.
2. Se han diseñado tres algoritmos diferentes para detectar el género usando diferentes técnicas, que se detallarán en el apartado 4.6.2
3. Se ha implementado un módulo de pruebas dentro de Play! Framework en el que podemos conectarnos a Twitter y extraer una muestra de usuarios con sus respectivos datos. Además, una vez implementados los algoritmos, este módulo nos permite ejecutar dichos algoritmos sobre una muestra de usuarios a elegir, por lo que podemos evaluar la eficacia de cada una de las implementaciones sobre datos de usuarios reales.
4. Se ha marcado un pequeño porcentaje de la muestra para poder evaluar la eficacia del algoritmo. Definimos como usuario marcado todo aquel usuario del que disponemos de los datos necesarios por los diferentes algoritmos, así como un género asignado manualmente por una persona.
5. Se ha evaluado la precisión de los algoritmos, comparando las predicciones hechas por las diferentes versiones de los algoritmos sobre diferentes muestras y el género que había sido marcado a mano para cada usuario.
6. Se ha añadido esta nueva funcionalidad a nuestro proyecto, usando la implementación que se ha demostrado más eficaz en las pruebas anteriores.

4.6.1. Obtención de la muestra

Para poder sacar ideas, reconocer patrones, y probar los algoritmos desarrollados, nuestro primer paso fue obtener una muestra de usuarios de Twitter. Definimos como muestra de usuarios a una colección de datos de usuarios reales que publican *tweets* en español, extraídos de Twitter. Además se ha intentado que la muestra fuese lo más homogénea posible, extrayendo tweets a diversas horas para tener datos tanto de España como de Hispanoamérica y de un tamaño suficientemente representativo.

Para tomar la muestra se ha aprovechado que en el proyecto ya teníamos implementada la conexión a Twitter Streaming. Anteriormente ya éramos capaces de recibir *tweets* en tiempo real con las características de palabras, lenguaje o situación geográfica que quisiéramos, por lo que para obtener la muestra lo lanzamos con un filtro de lenguaje en español para así recibir sólo aquellos *tweets* escritos en español en ese momento.

La API de Twitter está diseñada de forma que cualquier *tweet* que te manden siempre contenga información básica del usuario que ha creado ese *tweet*, o lo ha retuiteado, por lo que cada vez que nos llegaba un *tweet* nosotros podíamos almacenar en nuestra base de datos los datos referentes al usuario que nos interesaba. De esta forma, conseguimos tener una muestra de 50.000 usuarios con su nombre, su biografía, su lenguaje, su imagen de perfil y la localización geográfica del *tweet* por el cual le detectamos (imprescindible para segmentar a los usuarios por país).

4.6.2. Diseño de algoritmos

Una vez obtenida la muestra nos centramos en implementar varias versiones del algoritmo de detección de género. Todos estos algoritmos tendrían como objetivo ser capaces de extraer el género de un usuario cualquiera de Twitter que usase el español como su idioma en la red, partiendo de la información de usuario proporcionada por la API de Twitter.

Estudiando los datos disponibles determinamos que los campos más prometedores para esta tarea eran el nombre elegido por el usuario y su biografía. Es importante tener presente que en

Twitter se distingue entre el identificador del usuario dentro de la red y el nombre asociado a éste. Las diferencias radican en que el identificador tiene que empezar por “@”, está restringido a caracteres alfanuméricos sin espacios y debe de ser único en toda la red, mientras que el nombre permite mayor libertad al usuario, autorizando espacios y caracteres UTF-8 dentro de un límite de 20, y no tiene porqué ser único.

El nombre del usuario es un campo prometedor para detectar el género del usuario ya que es un campo que la gran mayoría de usuarios de Twitter rellena con su nombre y apellido reales. Y quiénes no lo hacen, suelen usar pequeñas variaciones sobre éste, como diminutivos o derivaciones. Esto hizo que nuestros primeros intentos se centrasen en detectar el género mediante la comparación del nombre con diccionarios de nombres en español.

Por otro lado, el usuario tiene la opción de escribir una breve biografía para su perfil. En esta biografía notamos que debido a la restricción de tamaño, son muy abundantes aquellas que consisten en una concatenación de adjetivos con género, como por ejemplo “madre”, “profesora”, “salmantina" o "enamorado de mi trabajo". Esto nos hizo pensar que para aquellos usuarios que no habíamos podido detectar el género gracias a su nombre, se podrían usar ciertas palabras como los adjetivos con género presentes en la biografía.

4.6.2.1. Algoritmo A

Esta primera versión del algoritmo toma el nombre del usuario y lo busca en un diccionario de nombres separados por género, creado a partir de los 2.000 nombres más comunes en español según el Instituto Nacional de Estadística [INE, 2014].

Este diccionario tiene dos versiones, una para cada género, y los nombres que lo forman están guardados igual que son proveídos por el INE: en minúsculas y sin tildes. Antes de buscar el nombre del usuario en ellos, se realiza una tarea que denominamos normalización sobre todo el texto del nombre que consta de los siguientes pasos:

1. Paso a minúsculas.
2. Substitución de tildes por su correspondiente sin tilde. Ejemplo: María por Maria.

3. Substitución de caracteres que la gente usa como vocales por su correspondiente. Ejemplo: "*Carmɛn"* por "Carmen".
4. Normalización de todos los caracteres que no sean ASCII. Ejemplo: "Manu " por "Manu".
5. Eliminación de todos los caracteres no alfabéticos. Ejemplo: "…Marina." por "Marina".

Una vez normalizado el nombre, se divide en partes usando como delimitador cualquier carácter en blanco. Estas partes, almacenadas en un *array*, son buscadas una por una en los dos diccionarios de nombres. En caso de encontrar esa parte en el diccionario, el algoritmo termina y se le asigna el género del diccionario, y si no, sigue buscando más partes del nombre. Gracias a que esta búsqueda se realiza de forma secuencial, el algoritmo acaba cuando detecta un nombre masculino, aunque luego venga un nombre femenino. De esta forma se evitan posibles fallos, como por ejemplo es el caso de "José María" y similares.

A continuación, si el algoritmo no ha sido capaz de asignar un género al usuario, se toman estas partes y se combinan en parejas de partes adyacentes. Por ejemplo, de "Ana María Matute" saldrían las parejas "Ana María" y "María Matute". Con estas parejas se repite de nuevo la parte de búsqueda en diccionarios con el objetivo de buscar en ellos nombres compuestos que podrían no haber sido detectados al estar antes en partes separadas.

Finalmente, si no se detecta ningún género en ninguno de los candidatos, el propio algoritmo busca y elimina vocales que aparezcan consecutivas, y repite el proceso anterior con los mismos diccionarios. Este paso fue necesario porque un porcentaje significativo de los usuarios de Twitter alteran sus nombres añadiendo letras al final como por ejemplo "Anaaa".

4.6.2.2. Algoritmo B

Para esta segunda versión quisimos aprovechar el trabajo hecho en la parte de normalización y creación de diccionarios, pero intentando un nuevo enfoque. En lugar de obtener candidatos y buscarlos en los diccionarios se intentó lo contrario: iterar sobre los diccionarios y buscar en el nombre del usuario. Como más adelante se puede ver en los resultados, este enfoque finalmente resultó ser erróneo porque si bien aumenta muchísimo el porcentaje de detecciones, también aumenta mucho el porcentaje de nombres que detectaba como ambos géneros. Esto es debido a

nombres como Adriana, que en esta implementación sería detectado como hombre al buscar desde los diccionarios Adrián y Adriana.

4.6.2.3. Algoritmo C

Esta versión fue fruto de que percibimos que la gente pone como nombre de Twitter modificaciones de su nombre real y aunque la normalización estaba siendo efectiva, hacía falta algo más. Por ese motivo construimos un analizador de la muestra: cada vez que no éramos capaces de detectar el género de un candidato, se guarda el nombre en una lista. Esto nos permitió ver cuáles eran los nombres más comunes que nuestros algoritmos no eran capaces de detectar. De esta forma fuimos capaces de crear cuatro nuevos diccionarios adicionales para analizar los nombres: dos con nombres que no estaban en la lista del INE (p. ej., Mari o Manolo), y otros dos con palabras que denotan género (p. ej., princesa o rey).

Como mejora adicional nos planteamos analizar también la biografía del usuario. Esto fue debido a que notamos que los usuarios usaban en gran medida adjetivos con género gramatical en este campo, por lo que usando una modificación del analizador anterior, fuimos capaces de obtener una lista de las palabras más frecuentes en las biografías de los usuarios. Gracias a esto pudimos crear otros dos diccionarios específicos para la biografía, compuestos de palabras como "orgullosa" o “actor". El proceso de detección usado para la biografía fue el mismo que con el nombre: normalización del texto, obtención de candidatos, y búsqueda en los diccionarios.

4.6.3. Evaluación de los algoritmos

4.4.3.1. Objetivo

Una vez desarrollados tres algoritmos diferentes para la detección de género, nos propusimos averiguar cuál era el que presentaba mejor precisión bajo unas condiciones de funcionamiento reales.

4.6.3.2. Metodología

El proceso que seguimos para la evaluación de rendimiento consistió en obtener una muestra de datos, clasificar el género de esta muestra a mano, ejecutar los tres algoritmos sobre esta muestra y con los resultados obtenidos, decidir cuál es el algoritmo que mejor se adapta a nuestras necesidades.

Fue muy importante asegurar que la muestra de datos tomada era una muestra adecuada. Para ello, tomamos una muestra diferente a la usada durante el desarrollo de los algoritmos para evitar que los datos fuesen adulterados, ya que usar la misma muestra para crear los diccionarios manuales que para evaluar los algoritmos habría dado mejores resultados que los reales. Es importante notar que los datos de esta muestra contienen información geográfica siempre que sea posible porque nos planteamos la hipótesis de que la precisión del algoritmo sería mayor con usuarios en España que con usuarios sin datos geográficos (que incluiría usuarios de todo el mundo) . Esta hipótesis se basa en que todas las versiones del algoritmo toma como base los nombres más comunes en España, que pueden no coincidir con los nombres más comunes en países latinoamericanos. Además, esta muestra fue tomada a diferentes horas para asegurar una buena homogeneidad de usuarios españoles y latinoamericanos.

De la muestra creada para la evaluación tomamos una porción y mediante una herramienta creada específicamente para ello, clasificamos manualmente el género de estos usuarios en cuatro posibles valores: género masculino, género femenino, usuarios sin género (empresas, asociaciones, marcas) y género imposible de determinar. A esta tarea le denominamos marcado de la muestra.

Una vez marcados los datos, evaluamos los algoritmos ejecutándolos sobre los usuarios a los que ya habíamos asignado un género manualmente. Para la evaluación de los algoritmos hemos usado métricas de recuperación de información [Manning, 2009], en concreto *precision* y *recall*. Se denomina *precision* a la relación entre el número de documentos relevantes recuperados entre el número de documentos recuperados y *recall* como la proporción de documentos relevantes recuperados y el total de documentos relevantes existentes. Para calcular dichas métricas hemos definido los siguientes casos posibles en la ejecución:

- **Positivo verdadero (*true positive*).** El usuario tenía género marcado y el algoritmo lo ha detectado correctamente.

- **Negativo verdadero (*true negative*).** El usuario no tenía género marcado y el algoritmo no ha detectado ninguno.
- **Falso positivo (*false positive*).** El usuario tenía género marcado y el algoritmo lo ha detectado erróneamente, o el usuario no tenía género y el algoritmo ha detectado alguno.
- **Falso negativo (*false negative*).** El usuario tenía género marcado y el algoritmo no ha detectado ninguno.

Las fórmulas empleadas para calcular la precisión y el *recall,* así como sus definiciones, son las siguientes:

- **Precisión.** Porcentaje de las predicciones que son correctas. Se calcula mediante la división entre las predicciones correctas (positivos verdaderos) y todas las predicciones realizadas (positivos verdaderos + falsos positivos).
- **Recall.** Porcentaje de los casos totales que han sido detectados. Se calcula mediante la división entre las predicciones correctas (positivos verdaderos) y todos los casos positivos que había (positivos verdaderos + falsos negativos).

4.6.3.3. Experimentación

Para el marcado de los usuarios se implementa un pequeño sistema de marcado de imágenes en Ruby on Rails[21] ya que esta tecnología permite desarrollar aplicaciones web rápidamente. La herramienta se diseña con el propósito de facilitar y agilizar la tarea de etiquetar manualmente el género de usuarios. Como dato de entrada, la herramienta recibe una conjunto de usuarios de los que se conocen el nombre elegido por usuario, el identificador de Twitter, la biografía y el enlace a la foto de perfil. Para cada usuario muestra en una vista html estos datos junto con un pequeño formulario con cuatro posibles botones. Mediante estos botones podemos decidir si el usuario mostrado es hombre, mujer, no tiene género (es una asociación, empresa, marca…) o nos es imposible determinar su género. Cuando se decide el género del usuario que se está mostrando, la

[21] Ruby on Rails es un *framework* de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby. Más información en *http://rubyonrails.org/*

herramienta almacena el género escogido y muestra automáticamente el siguiente usuario. En la figura 4.3 se puede ver la interfaz de la herramienta de marcado de usuarios implementada.

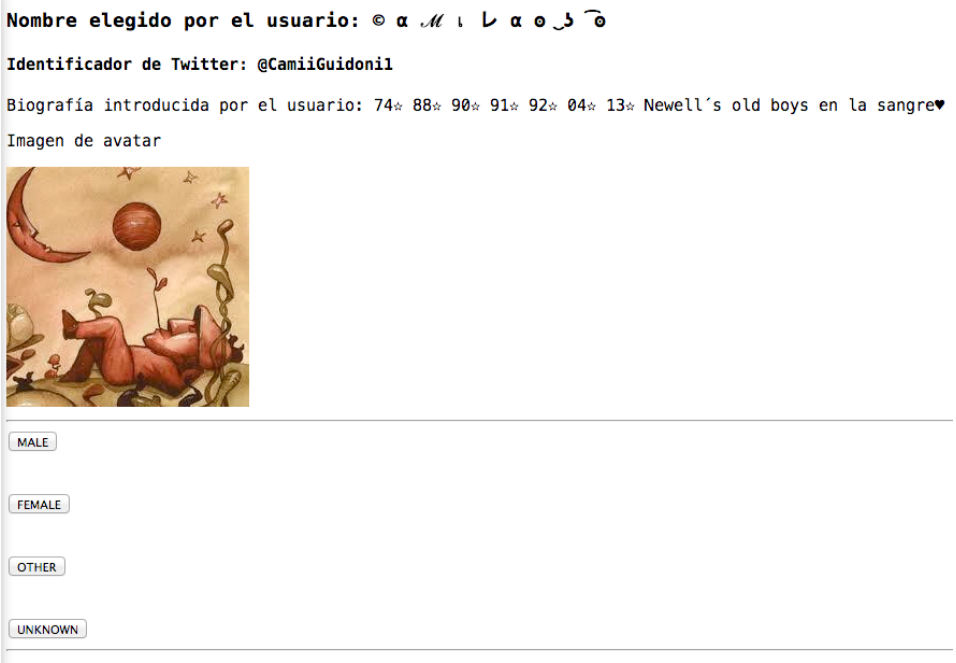

Figura 4.3. Herramienta de marcado de usuarios implementada con Ruby on Rails

Aunque cada usuario estaba marcado como una de estas cuatro opciones, finalmente solo sólo se han usado tres: se unió “desconocido” y “otro” porque el tener dos categorías no aportaba valor, puesto que este detector está centrado en las cuentas personales, y además dificultaba la evaluación del rendimiento de los algoritmos.

Finalmente **se marcaron manualmente 1.200 usuarios** usando este sistema de marcado. Estos usuarios fueron marcados a partir de una base de datos de usuarios de Twitter nueva, extraída a diferentes horas para intentar que fuese lo más homogénea posible. La razón de que sea una nueva es que la primera se usó para desarrollar los algoritmos, extrayendo candidatos a marcadores de género que fueron añadidos manualmente, por lo que evaluar los algoritmos sobre esa misma base de datos daría resultados adulterados.

Es importante mencionar que de estos 1.200 usuarios marcados había 300 que contenían información sobre su ubicación en el momento de escribir el *tweet* por el que Twitter envío sus datos. Esta información geolocalizada nos permitió saber si el usuario era español o latinoamericano, con lo que pudimos evaluar los algoritmos sobre diferentes bases de usuarios confirmando así que el rendimiento mejoraba cuando solo eran considerados usuarios españoles, que era una de nuestras hipótesis iniciales.

4.6.3.4. Resultados

Partiendo de la muestra marcada a mano, se ha evaluado la precisión de los tres algoritmos. Los resultados se recogen agrupados en la Tabla 4.1.

		Sin geolocalización	Geolocalización en España	Sin filtro de geolocalización
Algoritmo A	Precisión	88,41%	**94,25%**	90,36%
	Recall	47,66%	50,31%	51,44%
Algoritmo B	Precisión	63,28%	78,75%	74,62%
	Recall	37,90%	40,38%	41,64%
Algoritmo C	Precisión	86,73%	88,55%	87,37%
	Recall	66,41%	**72,96%**	68,98%

Tabla 4.1. Resultados de la evaluación de los algoritmos de detección de género

Como se puede ver en los resultados, el mejor algoritmo es la versión C, el que incluía diccionarios de marcadores de género para el nombre y la biografía. Este algoritmo tiene un recall muy alto, lo que significa que de los usuarios con género detecta un gran número de ellos, y una precisión que si bien es menor que la del algoritmo A, sigue siendo alta, lo que asegura que a esa tasa de detección alta le acompaña una tasa también muy alta de detecciones correctas.

Por otro lado, también se puede ver cómo queda demostrado todas las versiones mejoran sus datos al restringir los usuarios analizados a aquellos usuarios que están geolocalizados en España

debido a que el diccionario de nombres usado fue extraído del INE (Instituto Nacional de Estadística). Esto hace más adecuada si cabe la elección de usar este algoritmo en el proyecto, ya que nos estamos centrando en *tweets* geolocalizados en tiempo real en España o una zona de ella.

4.7. Categorización de *tweets* por contenido

Uno de los objetivos del proyecto es el procesamiento en tiempo real de *tweets* para obtener información procesada. En el apartado anterior se ha visto la detección del género del usuario que publicó el *tweet*. El segundo punto que se trata es la categorización de *tweets* por contenido. Definido un conjunto de categorías, nuestro objetivo ahora es clasificar cada *tweet* en una de las categoría de dicho conjunto basándonos en su contenido.

Este apartado trata de cómo se ha abordado el problema de la categorización de textos cortos mostrando no sólo la solución final adoptada, sino la evolución y las decisiones adoptadas en el proceso de desarrollo. A rasgos generales, los pasos que se han seguido para incorporar la categorización de textos a nuestro proyecto han sido los siguientes:

1. Se ha creado un conjunto de entrenamiento que sirva como base para clasificar textos cortos.
2. Se han implementado dos algoritmos categorizadores que se basan en herramientas especializadas en la recuperación de información.
3. Se ha implementado un sistema evaluador que fuese capaz de determinar la precisión de los categorizares creados partiendo de un conjunto de *tweets* clasificados manualmente.
4. Se ha realizado un análisis de los resultados obtenidos para determinar la versión del categorizador de textos que se ajusta mejor a las necesidades del proyecto.

4.7.1. Obtención del conjunto de entrenamiento

Los sistemas basados en la recuperación de la información requieren ser entrenados. Para poder entrenar un sistema de categorización de textos, es necesario obtener un conjunto de textos clasificados por categorías. A este conjunto de entrenamiento le llamamos corpus. Escoger un corpus de textos hace plantea la siguiente cuestión: ¿Qué textos aportarían un contenido representativo de términos que son mencionados en Twitter? Simplificando en una frase más concisa: *¿Sobre qué se habla en Twitter*?

Para tratar de encontrar los temas de conversación de interés, obviando todas aquellas conversaciones banales o de carácter privado, se examinaron los *trending topic* que hubo durante 3 días diferentes entre marzo y abril del 2014 en España.

1. *Trending topics* principales en España el 2 de marzo del 2014[22]:
    - Putin
    - Rusia
    - Ucrania

La principal noticia internacional de ese día es el movimiento de tropas rusas en la región ucraniana de Crimea [23]. En el momento en que se redacta este proyecto, Vladimir Putin es el presidente ruso y estos movimientos de tropas podían suponer el inicio de una invasión militar rusa en Ucrania.

2. *Trending topics* principales en España el 24 de marzo del 2014[24]:
    - *DEPAdolfoSuarez.*
    - Aeropuerto Adolfo Suárez.
    - Transición.
    - Barajas.
    - Undiano.
    - Madrid.
    - Ronaldo.

[22] *http://www.trendinalia.com/twitter-trending-topics/spain/spain-140302.html*

[23] *http://elpais.com/elpais/2014/03/02/media/1393799131_328984.html*

[24] *http://www.trendinalia.com/twitter-trending-topics/spain/spain-140324.html*

- Barça.
- Busquets.
- Messi.
- Neymar.

Se aprecian dos temáticas completamente diferentes. Sin embargo, ambas hacen eco de dos hechos ocurridos el día anterior, 23 de marzo del 2014, en el que murió el político español Adolfo Suárez y se disputó el partido de fútbol entre en el Real Madrid C.F. y el F.C. Barcelona. Para situar al lector en contexto, tras la muerte de Suárez, se renombró oficialmente el aeropuerto de Madrid-Barajas como "Aeropuerto Adolfo Suárez Madrid-Barajas[25]". Por otro lado, fue muy criticado el árbitro de la liga española Undiano Mallenco a raíz del partido del 23 de marzo. En ese momento, Cristiano Ronaldo es jugador del Real Madrid y Sergio Busquets, Lionel Messi y Neymar da Silva son jugadores del Barcelona.

3. *Trending topics* principales en España el 7 de abril del 2014[26]:
   - Mickey Rooney.
   - Undertaker.
   - *DíaMundialdelaSalud.*

Buscando el motivo de estos *trending topics*, aparece la muerte del actor estadounidense Mickey Rooney el 6 de abril del 2014, la derrota del luchador de lucha libre conocido como *The Undertaker* tras 21 victorias consecutivas (el mismo día 7) y que, efectivamente, es el día mundial de la salud declarado por la Organización Mundial de la Salud.

Viendo estos ejemplos, ¿es posible hablar de un tema que aúne todo, o al menos la inmensa mayoría del contenido de interés generado en Twitter? Sí, de actualidad, ya sea actualidad política, deportiva, cultural etc. Se determina que el contenido que se genera en Twitter, aparte de las conversaciones privadas o la conversación sin sentido, guarda una relación directa con algún tema que o está ocurriendo, o ha ocurrido en el mundo en un periodo de tiempo muy cercano.

---

[25] Boletín Oficial del Estado, código 3275

[26] *http://www.trendinalia.com/twitter-trending-topics/spain/spain-140407.html*

Volviendo al objetivo de determinar un corpus de textos que sirva como conjunto de entrenamiento para un clasificador de textos se ha llegado a la conclusión de que debido a que los temas de los que principalmente se habla en Twitter guardan una relación directa con temas de actualidad, la mejor opción será utilizar textos que contengan términos que mantengan esa relación directa con las noticias del momento. Más concretamente, noticias que ocurren durante el mismo día o como máximo un día antes.

La solución adoptada ha sido aprovechar los **artículos periodísticos publicados digitalmente** por medio de los sistemas de difusión (*feeds)*[27] provistos por los propios periódicos[28]. Al utilizar este sistema es posible recoger los titulares publicados cada día para construir nuestro corpus inicial. Además, para cada titular se puede saber bajo que categoría se que se publicó originalmente en el periódico (internacional, economía, cultura, sociedad, deportes, etc.)

Las ventajas al adoptar esta solución son que los textos obtenidos:

1. Se actualizan cada día.
2. Pertenecen de forma implícita a una categoría definida por el periódico que lo publicó.
3. Contienen los términos que potencialmente se usarán con más frecuencia en Twitter al identificar un tema de actualidad concreto.

4.7.1.1. Definición del conjunto de categorías

Una vez que se han elegido los textos que van a conformar el conjunto de entrenamiento, se define el conjunto de categorías que se va a tratar para la categorización. Inicialmente se planteó la siguiente cuestión. ¿Es posible cubrir la totalidad del contenido que se genera en Twitter? Dado que el proyecto no se centra exclusivamente en la categorización de textos sino que es sólo una de las áreas en las que se quiere trabajar, se consideró que no será necesaria una categorización exhaustiva de todo el contenido generado.

[27] Feeds: Sistema de difusión de un contenido web. Se utiliza para suministrar información actualizada sin la necesidad de notificar al usuario que se suscribe al sistema.

En un principio, se barajó manejar cuatro categorías diferentes: actualidad política, actualidad economía, actualidad cultural y actualidad deportiva. Esta idea tenía su base en que es, a grandes rasgos, la división de los periódicos de donde la información es extraída. Por lo tanto la correspondencia entre un artículo de periódico y su categoría en la clasificación sería directa. Sin embargo, se percibió que al categorizar textos cortos de 140 caracteres como máximo había muchas posibilidades de que fuese muy difícil discernir, incluso para un ser humano, entre un *tweet* relacionado con la actualidad económica y un *tweet* relacionado con la actualidad política.

Los siguientes dos *tweets* de ejemplo (Figuras 4.4 y 4.5) corresponden a personas ajenas totalmente al proyecto y pueden servir para ilustrar esta situación:

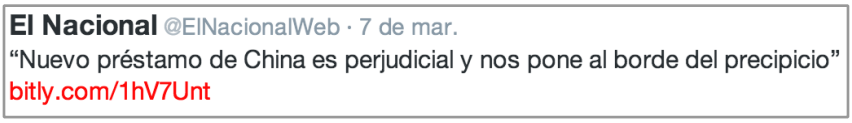

Figura 4.4. Tweet *de ejemplo del 7 de marzo del 2014 por @ElNacionalWeb*

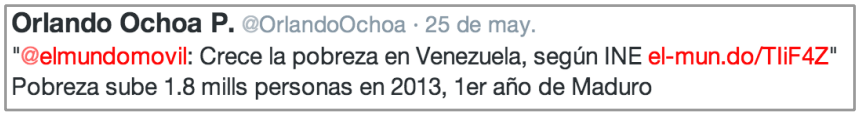

Figura 4.5. Tweet *de ejemplo del 25 de mayo del 2014 por @OrlandoOchoa*

Ambos *tweets* tienen un fuerte contenido político y a la vez están tratando sobre temas económicos. Se decidió que esta división no iba a dar categorías completamente excluyentes.

Por otro lado, analizando los artículos etiquetados como culturales en los periódicos fue fácil notar que habría muchas posibilidades de que el porcentaje de *tweets* etiquetados como cultura fuese extremadamente bajo.

La solución adoptada fue reducir las categorías a dos grandes grupos bien diferenciados: actualidad política y actualidad deportiva. Estas dos categorías no abarcan todo el universo, por que un *tweet* podrá no pertenecer a ninguna de estas categorías. Para simplificar, todo lo que quede fuera

de estas dos categorías pertenece a una tercera categoría que fue denominado contenido no categorizable o ruido.

La Figura 4.6 muestra, en forma de diagrama de conjuntos, qué conjunto de *tweets* se desea categorizar. Sea U el universo de temas de conversación y N el subconjunto de *tweets* que son calificados interesantes, U - N es todo el contenido calificado como ruido. Los dos subconjuntos de N utilizados son P (contenido relacionado con la actualidad política) y D (contenido relacionado con la actualidad deportiva). Se considera que la intersección de estos dos conjuntos es vacía. El sistema determinará dos conjuntos P' (contenido categorizado como actualidad política) y D' (contenido categorizado como actualidad deportiva). El objetivo será acercar lo máximo posible P' a P y D' a D, minimizando el número de falsos positivos y falsos negativos.

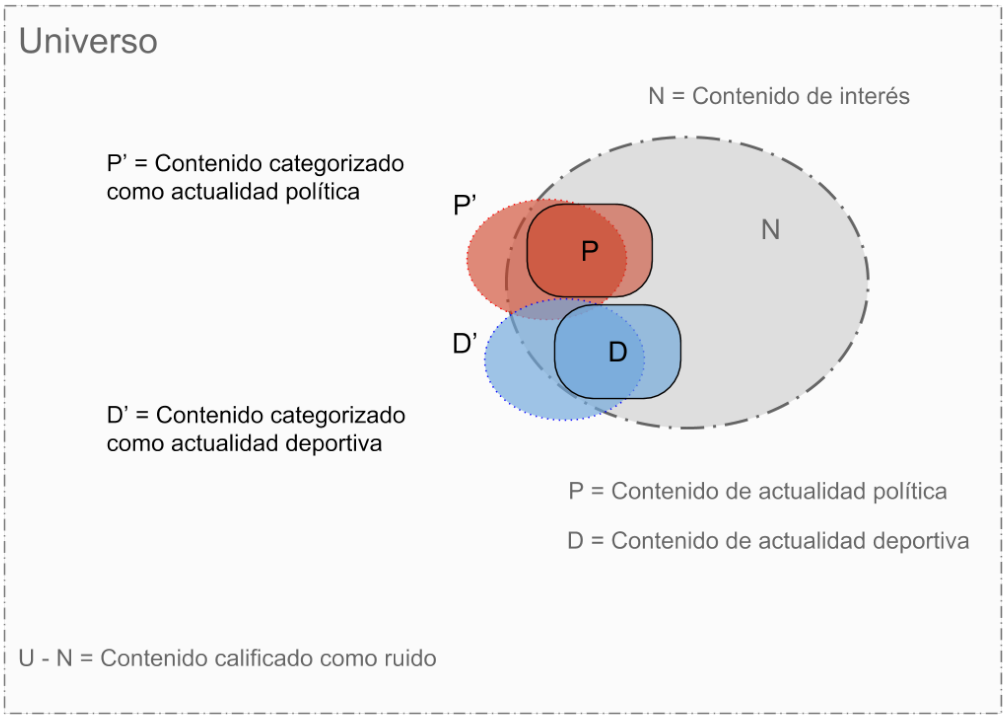

Figura 4.6. Situación abstracta de la categorización del contenido publicado en Twitter

4.7.1.2. El extractor de feeds

Para crear el corpus de artículos clasificados de acuerdo a las categorías descritas en el apartado anterior se ha diseñado un módulo capaz:

1. Conectarse a los *feeds* de la versión digital de los periódicos El País[29] y El Mundo[30].
2. Procesar la respuesta de los periódicos digitales aplicando reglas de normalización al texto obtenido.
3. Guardar la información en un formato que se pueda usar en el proyecto.

Dado un archivo de configuración en el que se indican las URLs (agrupadas por categorías) de las que se quiere extraer la información y una carpeta destino donde se desea guardar el archivo de salida, el extractor de *feeds* recorrerá cada enlace y hará un procesamiento[31] de cada entrada obtenida convirtiéndola a un formato lo suficientemente flexible como para que se pueda utilizar como modelo por cualquier herramienta en la categorización de texto. En el Anexo B se puede encontrar más información relacionada con la utilización de este módulo.

El contenido obtenido se guarda en texto plano siguiendo el siguiente formato:

1. Título, enlace al artículo en formato digital y autor precedido por el carácter “%”.
2. Contenido obtenido (normalizado) entre dobles comillas seguido de una coma y de la categoría (en letras mayúsculas) a la que pertenece.

En resumen, para cada entrada del XML perteneciente a la categoría C se extrae su meta información (título, enlace y autor) y el contenido, al que se le aplican una serie de normalizaciones y se le asigna la categoría correspondiente C (Figura 4.7).

---

[29] *http://servicios.elpais.com/rss/*

[30] *http://rss.elmundo.es/rss/.*

```
%% [Título del artículo] - [Enlace al artículo en formato digital]: [autor]
"texto del artículo normalizado y procesado con lista de parada",CATEGORÍA
```

Figura 4.7. Formato en el que se imprimen los titulares obtenidos

4.7.2. Categorizador de textos basado en Weka

El primer acercamiento para implementar el categorización de textos cortos emplea la herramienta Weka. El objetivo es conseguir un módulo, absolutamente independiente, en el que se pudiese construir un modelo de Weka a partir de un corpus de entrenamiento conseguido de manera transparente. Una vez entrenado el sistema, se puede preguntar a qué categoría pertenece un texto. Las respuestas dadas por el sistema pueden ser de dos tipos: indirecta, en la que obtendremos un vector de valoraciones entre 0 y 1 que representa el grado de pertenencia del texto a cada categoría, o directa, en la que el sistema evalúa el vector de pesos para obtener una de las siguientes respuestas:

- El texto pertenece a la actualidad política.
- El texto pertenece a la actualidad deportiva.
- El texto no pertenece a ninguna categoría.

En el módulo de Weka construido se ofrece estas tres diferentes alternativas para construir el modelo interno.

- Definir de manera dinámica (en código) los datos en forma de tuplas (mensaje, clase).
- Leer un documento de texto con extensión de texto que contiene los datos.
- Leer N documentos de texto que contienen los datos. Internamente se añaden los datos de todos ellos al modelo final. Esta alternativa fue la escogida para realizar las pruebas de precisión realizadas que se detallarán en el apartado 4.7.3.

Un vez construido el modelo usando los corpus creados ya es posible de realizar la categorización de los *tweets*. Al pedir la clasificación de un texto, se le pide a Weka que lo clasifique no con uno, sino con tres clasificadores diferentes. El hecho de usar hasta tres clasificadores de Weka apenas representa coste en ejecución a la hora de clasificar; no así a la hora de construir los modelos a partir del corpus de palabras. Sin embargo, el coste en tiempo al construir los índices no afecta al rendimiento del sistema, ya que ese cálculo sólo hay que hacerlo una vez al iniciar el sistema.

A cambio, se dispone de **tres veredictos para la clasificación del texto**. Cada uno de ellos determinado por un algoritmo diferente:

1. Clasificador basado en reglas. Se ha optado por el clasificador *NNge*[32], implementación del algoritmo KNN (los K vecinos más cercanos).
2. Clasificador basado en la construcción de una tabla de decisión. Se ha optado por el clasificador *DecisionTable*[33].
3. Clasificador basado en la construcción de un árbol de decisión. Se ha optado por en concreto el clasificador *FT*[34].

Al concluir la clasificación disponemos de tres vectores, uno para cada clasificador, de valores diferentes de la forma que se puede ver en la Figura 4.8.

{categoría 1: valor 1, categoría 2: valor 2, …, categoría i-ésima: valor i}

Figura 4.8. Vector obtenido por cada clasificador de textos.

Para determinar los pesos finales asociados a cada categoría, se calcula la media aritmética de los tres vectores, obteniendo de esta forma el vector solución.

---

[32] *http://weka.sourceforge.net/doc.packages/NNge/weka/classifiers/rules/NNge.html*

[33] *http://weka.sourceforge.net/doc.dev/weka/classifiers/rules/DecisionTable.html*

[34] *http://weka.sourceforge.net/doc.packages/functionalTrees/weka/classifiers/trees/FT.html*

Por último, el modulo implementado para solicitar la categoría asignada por Weka a un texto dispone de dos opciones: una que devuelve el vector solución y otra en la que el propio módulo procesa el vector solución y determina la categoría más probable. Para ello, el módulo compara los valores de cada categoría en el vector solución con su valor umbral predefinido. Si algún valor supera el valor umbral asignado a esa categoría, ésta será la categoría asignada. En caso de haber varias categorías que superen sus umbrales, predomina la de mayor valor. Se puede encontrar más información sobre la inicialización y uso de este módulo en el Anexo C.

4.7.3. Validación del categorizador basado en Weka

Una vez construido el categorizador de textos, el siguiente paso fue realizar la validación de los resultados. Los siguientes pasos seguidos fueron los siguientes:

1. Obtención de un conjunto de *tweets*.
2. Clasificación de este conjunto, asignándole de forma manual una de las categorías posibles.
3. Implementación de un evaluador de precisión del sistema que compare la categoría asignada al *tweet* por el clasificador con la categoría real asignada manualmente. Según difieran o coincidan estas categorías, definimos los siguientes resultados:
    - **Positivo verdadero (*true positive*).** El clasificador asigna una categoría, y ésta coincide con la asignada de forma manual.
    - **Negativo verdadero (*true negative*).** El clasificador no asigna ninguna categoría, y manualmente tampoco se le asignó ninguna.
    - **Falso positivo (*false positive*).** El clasificador asigna una categoría, pero ésta no coincide con la asociada, ya sea porque no se asignó ninguna o porque es diferente.
    - **Falso negativo (*false negative*).** el clasificador no asigna una categoría, y manualmente no se le asignó ninguna.
4. Análisis de los resultados obtenidos mediante el evaluador implementado. Este análisis se centra en las mismas métricas que el evaluador del detector de género del apartado 4.4.3. (precision y recall [Manning, 2009]).

En concreto, se seleccionaron un total de 40 tweets de actualidad política y 30 tweets de actualidad deportiva publicados por diferentes personas y sobre temas diversos. Los resultados para esta primera prueba mostraron valores muy prometedores acertando 59 casos de 70 posibles. Sin embargo, se sometió al sistema a una segunda prueba más exigente que la anterior introduciendo ruido: se añadieron al conjunto de *tweets* de validación otros 40 *tweets* que no pertenecían a ninguna categoría. Es decir, *tweets* que quedan fuera de lo que nuestro sistema es capaz de identificar. El comportamiento esperado es que el sistema descarte ninguna clasificación para estos *tweets* (internamente se permitió el valor vacío como una categorización válida, que indica que no pertenece a ninguna de las categorías).

Los resultados para la prueba que incluye ruido mostraron que el sistema no era capaz de determinar cuándo un *tweet* no pertenecía a ninguna categoría. La precisión del módulo empleando los categorizadores dados por Weka es del **60,2%** y el recall del **84,3%** En la Tabla 4.2 se pueden ver los resultados de la prueba para el categorizador de textos basado en Weka. A partir de estos resultados se puede construir la tabla de contingencia mostrada como Tabla 4.3.

		Casos de actualidad política	Casos de actualidad deportes	Casos sin categoría
Predicción de casos	Actualidad política	37	8	36
	Actualidad deportiva	3	22	3
	Sin categoría	0	0	1

Tabla 4.2. Resultados de la prueba con ruido para el categorizador de textos basado en Weka.

	Caso negativo	Caso positivo
Predicción negativa	1	11
Predicción positiva	39	59

Tabla 4.3. Tabla de contingencia para el categorizador de textos basado en Weka

A pesar de que el sistema cuenta con un valor de recall aceptable, consideramos que la precisión con la que ha respondido es preocupante, y no sólo porque tenga un valor bajo. Es importante recordar que el conjunto de *tweets* se componía de 40 *tweets* que no pertenecían a ninguna categoría. Sin embargo, el sistema sólo ha determinado 1 *tweet* como no perteneciente a ninguna categoría. Esto lleva a suponer que si el conjunto de validación del sistema fuese modificado introduciendo más tweets que no pertenecen a ninguna categoría, la precisión descendería. En el apartado 2.2 se ha visto que aproximadamente el 87% de los tweets se consideran ruido [Kelly, 2009], el hecho de que el módulo no sea capaz de determinar cuándo un *tweet* no pertenece a ninguna categoría, hace que el categorizador de textos implementado sea prácticamente inservible.

4.7.4. Categorizador de textos basado en Lucene

Debido a los resultados que se determinaron en la evaluación del categorizador de textos basado en Weka se optó por reintentar la tarea, esta vez empleando como base la herramienta Lucene. El objetivo vuelve a ser el mismo que antes: conseguir un módulo independiente en que se pueda obtener, a partir de un conjunto de entrenamiento, a qué categoría pertenece un texto.

El núcleo de Lucene es el índice. Al solicitar a Lucene que se guarde un documento, primero lo indexa y después lo almacena en un directorio interno**.** Una vez que se ha poblado el índice es posible inicializar un buscador para recuperar los documentos con mayor grado de pertenencia a alguna de las categorías. Siguiendo el mismo esquema que para el categorizador anterior, es posible solicitar la respuestas de dos formas distintas: la llamada respuesta indirecta en la que se espera un vector de pesos para cada categoría y la respuesta directa en la que el sistema evalúa el vector de pesos para obtener una única solución. En el caso de la respuesta directa, las soluciones posibles son “el texto pertenece a actualidad política”, “el texto pertenece a actualidad deportiva”, y “el texto no pertenece a ninguna categoría”. Se pueden encontrar más detalles del módulo en el Anexo D.

4.7.5. Validación del categorizador de textos basado en Lucene

Para validar los resultados de esta segunda implementación del categorizador, se ha empleado las misma prueba que aplicamos la segunda vez a la categorización basada en Weka. Es decir, se ha

tomado una base de casos compuesta por *tweets* reales categorizados manualmente como *tweets* de actualidad política, *tweets* de actualidad deportiva o *tweets* que no pertenecen a ninguna de las categorías. Se le ha pedido al sistema que evaluase cada uno de estos *tweets* comparando la respuesta del sistema con la categoría real. La precisión del sistema fue del **69,33%** y el recall tuvo un valor de **74,29%.** Los resultados se recogen en la Tabla 4.4, en la Tabla 4.5 se puede ver la tabla de contingencia obtenida a partir de estos resultados utilizada para obtener la precisión y el recall del módulo.

		Casos de actualidad política	Casos de actualidad deportiva	Casos sin categoría
Predicción de casos	Actualidad política	32	8	17
	Actualidad deportiva	1	20	6
	Sin categoría	7	2	17

Tabla 4.4. Resultados del categorizador de textos basado en Lucene

	Caso negativo	Caso positivo
Predicción negativa	17	18
Predicción positiva	23	52

Tabla 4.5. Tabla de contingencia obtenida a partir de los resultados del categorizador basado en Lucene

4.7.6. Comparación de los resultados

En la Tabla 4.6 se exponen a modo de comparativa los resultados obtenidos por los dos categorizadores implementados. En vista de estos resultados, queda claro que la clasificación de textos cortos es un problema difícil de resolver: ninguna de las versiones es capaz de categorizar correctamente todos los *tweets*, como era nuestro objetivo inicial. Por este motivo se decidió usar el categorizador para poder determinar el porcentaje de tweets que pertenecen a una categoría respecto del total. Teniendo esto en mente, la versión que mejor se adapta a nuestras decisión es la

versión que usa Lucene, ya que si bien categoriza menos *tweets* que la que usa Weka (el *recall* es menor), los que sí categoriza los categoriza con mayor precisión (*precision* mayor). Debido a esto, la versión finalmente utilizada es ésta última.

		Resultados
Módulo basado en Weka	Precisión	60,20%
	Recall	**84,3%**
Módulo basado en Lucene	Precisión	**69,33%**
	Recall	74,29%

Tabla 4.6. Comparación de la precisión y el recall de los dos categorizadores implementados

4.8. Traductor de localizaciones

Una de las funcionalidades que se ha planificado para el backend del proyecto es la posibilidad de solicitar información almacenada en la base de datos aplicando filtros de restricción geográfica. Toda la información geográfica del modelo de datos está expresada en términos de latitud y longitud, ya que es el estándar de facto para localizar un punto. Sin embargo, no es intuitivo a la hora de filtrar geográficamente ya que requiere la definición de un área mediante coordenadas, y casi siempre es necesario usar herramientas externas para calcularlas. Para facilitar el uso de estos filtros se decidió desarrollar un módulo muy sencillo capaz de traducir identificadores de área definidos previamente a su correspondiente tupla de coordenadas. Así para restringir búsquedas a una ciudad no será necesario que el usuario indique las coordenadas que la delimitan, sino que podrá usar simplemente el nombre de ésta.

En el proyecto solo están incluidos como identificadores de área las ciudades de Madrid y Barcelona, de forma que solo se pueden acotar los datos para estas ciudades. Sin embargo, este módulo fue diseñado para que fuese fácilmente ampliable: solo sería necesario registrar un nuevo identificador, y definir su traducción a la tupla de coordenadas que defina su área. También es importante notar que en el proyecto solo se usan áreas definidas por dos coordenadas, es decir, un

rectángulo, pero sería posible añadir tantas coordenadas como hiciese falta, definiendo de esta forma el polígono deseado.

Por otro lado, este módulo también es capaz de traducir un segundo tipo de filtros geográficos. En esta ocasión, los datos de entrada son una única coordenada geográfica y un *offset* que se va a aplicar sobre ésta. El traductor de localizaciones consigue a partir de los datos de entrada dos coordenadas que, de nuevo, definen un área rectangular. Para ello, toma la coordenada dada como parámetro como centro de un rectángulo imaginario. Calcula la coordenada superior izquierda restando el *offset* a la latitud y sumándoselo a la longitud; la coordenada inferior derecha se obtiene sumando el *offset* a la latitud y restándoselo a la longitud.

4.9. Interfaz de Programación de Aplicación (API) implementada

Se ha diseñado e implementado una API que permita al acceso a la información almacenada en nuestra base de datos mediante llamadas HTTP. De esta forma, posibles solicitantes externos, como la aplicación web desarrollada en el capítulo 5, pueden emplear esta información. De la misma forma que Twitter, que ofrece tanto información procesada como información en bruto como hemos visto en el capítulo 3, nuestra API ofrece la posibilidad de acceder a la información procesada por nuestro sistema y a la información en bruto a partir de la cual obtuvimos la información procesada. Todas las funciones definidas en la API del proyecto están diseñadas siguiendo las directrices REST (Representational State Transfer).

4.9.1. Directrices REST

REST es una forma de interactuar con la información de un servidor mediante llamadas HTTP. Fue definida en el año 2000 por Roy Fielding en su tesis doctoral [Fielding, 2000] como una alternativa a mecanismos de interacción más complejos como SOAP o RPC. Actualmente goza de gran popularidad aplicada a servicios web, dónde cada modelo de información puede ser creado

(petición PUT), leído (petición GET), modificado (petición POST), o borrado (petición DELETE), mediante una llamada a una misma url con diferente método HTTP. En el proyecto, cada una de las entidades definidas en el punto 4.1 que forman nuestro modelo de datos solo puede ser consultada, es decir, solo admite peticiones GET.

4.9.2. Diseño de la API

Como se ha detallado en el punto 4.1, cada una de las entidades del modelo de datos (*tweets*, usuarios, hashtags y links) puede ser consultada mediante su propia colección de llamadas a la API, que se encuentra en *www.itafy.com/api.* Esta API recibe únicamente peticiones GET con parámetros opcionales en la cabecera que podrán ser usados para especificar filtros (detallados en el apartado 3.1.2). El formato de la respuesta será JSON, ya que la equivalencia entre los resultados extraídos de la base de datos en MongoDB y JSON es casi trivial para el sistema. En la figura 4.9. se puede ver un ejemplo de una respuesta JSON de la API.


```
[
  - {
        id: "53a22623300477601ae667de",
        latitude: -33.430664,
        longitude: -70.594902,
        title: null,
        url: "http://m.publico.es/528660",
        category: null,
        tweetId: "53a22623300477601ae667dd",
        createdAt: 1403135523908,
        updatedAt: 1403135523908
    },
  - {
        id: "53a22623300477601ae667e3",
        latitude: 9.940081,
        longitude: -84.145746,
        title: null,
        url: "http://instagram.com/p/pZ43BlnBlD/",
        category: null,
        tweetId: "53a22623300477601ae667e2",
        createdAt: 1403135523916,
        updatedAt: 1403135523916
    },
```


Figura 4.9. Extracto de respuesta a una petición a itafy.com/api/links

4.9.3. Rutas de la API implementadas

Se han implementado rutas de primer y segundo nivel para cada una de las entidades del modelo de datos. Se denominan rutas de primer nivel a aquellas que solamente indican el indicador de la entidad (p. ej., */api/tweets*) y rutas de segundo nivel a aquellas que además del indicador de entidad especifican alguna restricción (p. ej., */api/tweets/53a226233004776 01ae667de*). Las únicas rutas de segundo nivel que se han implementado son aquellas que restringen los resultados mediante un identificador para poder hacer búsquedas de elementos por la clave primaria de la base de datos. De esta forma es sencillo hallar el elemento relacionado desde otra entidad. En la Tabla 4.7 se puede consultar un listado de las rutas implementadas para cada entidad.

Entidad	Indicador	Ruta relativa	Descripción
Tweet	*tweets*	*/api/tweets*	Devuelve un listado de tweets.
		/api/tweets/:id	Devuelve información del tweet con el id especificada en la ruta.
Usuario	*users*	*/api/users*	Devuelve un listado de usuarios.
		/api/users/:id	Devuelve información del tweet con el id especificada en la ruta.
Link	*links*	*/api/links*	Devuelve un listado de enlaces.
		/api/links/:id	Devuelve información del enlace con la id especificada en la ruta.
Hashtag	*hashtags*	*/api/hashtags*	Devuelve un listado de hashtags.
		/api/hashtags/:id	Devuelve información del hashtag con la id especificada en la ruta.

Tabla 4.7. Listado de las rutas de entidades

Por otro lado, se han creado una serie de rutas más concretas para dar soporte a la aplicación de uso que se explicará en el capítulo 5. La mayor parte de estas rutas proveen de la información necesaria a las gráficas del *dasboard*, detallado en el apartado 5.1.2 usando el protocolo Rest igual que las explicadas anteriormente mientras que hay una ruta especial que no usa el protocolo Rest. Esta ruta sirve para conectarse al *websocket* que envía *tweets* en tiempo real, por lo que debe ser llamada mediante dicho protocolo. En la Tabla 4.8 se pueden consultar las rutas implementadas para dar soporte al mapa en tiempo real y al dashboard de gráficas.

Ruta relativa	Descripción
/live	Se conecta al websocket que envía tweets en tiempo real, ya procesados.
/api/tweets/per-minute	Devuelve un listado de los tweets recogidos en el *backend* por minuto.
/api/tweets/categories-percentage	Devuelve datos de la distribución de las categorías por porcentaje y género.
/api/tweets/genders-percentage	Devuelve datos de la distribución de los géneros por porcentaje.

Tabla 4.8. Listado de las rutas especiales

4.9.4. Filtros disponibles para las peticiones a la API

A continuación se detallan los diferentes filtros que se le pueden poner a estas llamadas:

- **Filtro geográfico.** Permite limitar los resultados a los creados dentro del área geográfica especificada. Hay dos posibilidades para especificar este área: mediante una tupía de latitud, longitud y *offset*, o mediante un identificador. Ejemplos: */api/links?area=madrid*. */api/links?latitude=0.5&longitude=0.35&offset=0.2*.
- **Filtro temporal.** Permite acotar los resultados especificando un intervalo de tiempo. Ejemplo: */api/users?after=10-06-2014* devolverá usuarios introducidos en la base de datos posteriormente al 10 de junio de 2014.
- **Filtro de género.** Permite limitar los resultados a elementos creados por un usuario del género establecido. Ejemplo: */api/hashtags?genre=female* devolverá hashtags creados por mujeres.
- **Filtro de categoría.** Permite acotar los resultados a elemento que pertenezcan a la categoría solicitada. Ejemplo: */api/tweets?category=deportes* devolverá *tweets* que hayan sido detectados como *tweets* que hablan de actualidad deportiva.
- **Filtro cuantitativo.** Permite solicitar un número específico de elementos. Ejemplo: */api/tweets?limit=20* devolverá 20 *tweets*.

4.10. Conclusiones

Durante este capítulo se ha explicado el sistema de clasificación de información de Twitter que se ha desarrollado. En concreto las estrategias que se determinaron para la generación de información procesada en español han sido la detección de género del usuario, la categorización de *tweets* basada en el contenido y la geolocalización de publicaciones en tiempo real. En primer lugar se ha descrito la funcionalidad analizando la división en módulos que configuran la arquitectura del sistema.

Se ha explicado cómo este sistema es capaz de extraer la información conectándose a la Streaming API de Twitter, desgranarla en una serie de entidades definidas acorde con la clasificación del marco teórico del capítulo 3 y almacenar la información en una base de datos no relacional. A continuación se han detallado los diferentes módulos que conforman el procesamiento de la información obtenida. En primer lugar se ha explicado el diseño, la implementación y validación de los diferentes algoritmos propuestos para el módulo detector de género del usuario así como las herramientas diseñadas e implantadas para facilitar la validación de los algoritmos. En segundo lugar se ha detallado el diseño, implementación y validación de los dos módulos categorizadores de texto desarrollados. En tercer lugar se ha explicado cómo funciona el módulo traductor de localizaciones diseñado para facilitar la tarea de consultar áreas geógraficas. Por último se ha explicado la API implementada para acceder a toda la información almacenada por el sistema.

5. Aplicación de visualización de información como ejemplo de uso del *backend*

Una vez diseñada e implementada la arquitectura del *backend* el siguiente objetivo fue desarrollar un ejemplo de uso que pidiese información al backend y, sin procesarla, fuese capaz de mostrarla de forma visual y atractiva para el usuario, para así demostrar la utilidad del *backend* diseñado.

Se ha desarrollado una aplicación web que muestra de manera visual la geolocalización de la información existente en el *backend* por medio de un mapa en el que muestran localizados *tweets* por género, así cómo representa mediante gráficas la información procesada por el sistema.

En este capítulo se detalla la funcionalidad de esta aplicación. A continuación, se detallan qué métodos de la API implementada en este proyecto han sido necesarios para conseguir la aplicación de ejemplo. Por último se exponen las tecnologías empleadas.

5.1. Funcionalidad

Esta aplicación de ejemplo consta de dos secciones independientes:

- **Un mapa interactivo** en el que van apareciendo localizados los *tweets* que se crean en tiempo real, con un identificador del género de la persona que los creó.
- **Un *dashboard*** formado por diferentes gráficas que ilustran estadísticas relativas a los diferentes datos que se pueden extraer de la información almacenada, especialmente aquellos

relacionados con género y categoría (actualidad política o deportiva) a la que pertenecen los *tweets* recibidos.

5.1.1. Mapa interactivo

La interfaz muestra en la parte izquierda una gráfica y un contador de *tweets*, y a la derecha un mapa (Figura 5.1). En el momento en el que la aplicación se conecta al *backend* del proyecto empieza a recibir *tweets* mediante un *websocket.* Cada *tweet* se posicionará en el mapa en el lugar exacto en el que ha sido creado, identificando el género del usuario que lo creó mediante colores (azul para género masculino, rosa para género femenino, y gris para aquellos usuarios a los que no se ha podido determinar su género). Cada *tweet* permite hacer *click* sobre él, de forma que se pueda ver en detalle toda la información de la que disponemos: usuario que lo creó con su nombre y su foto

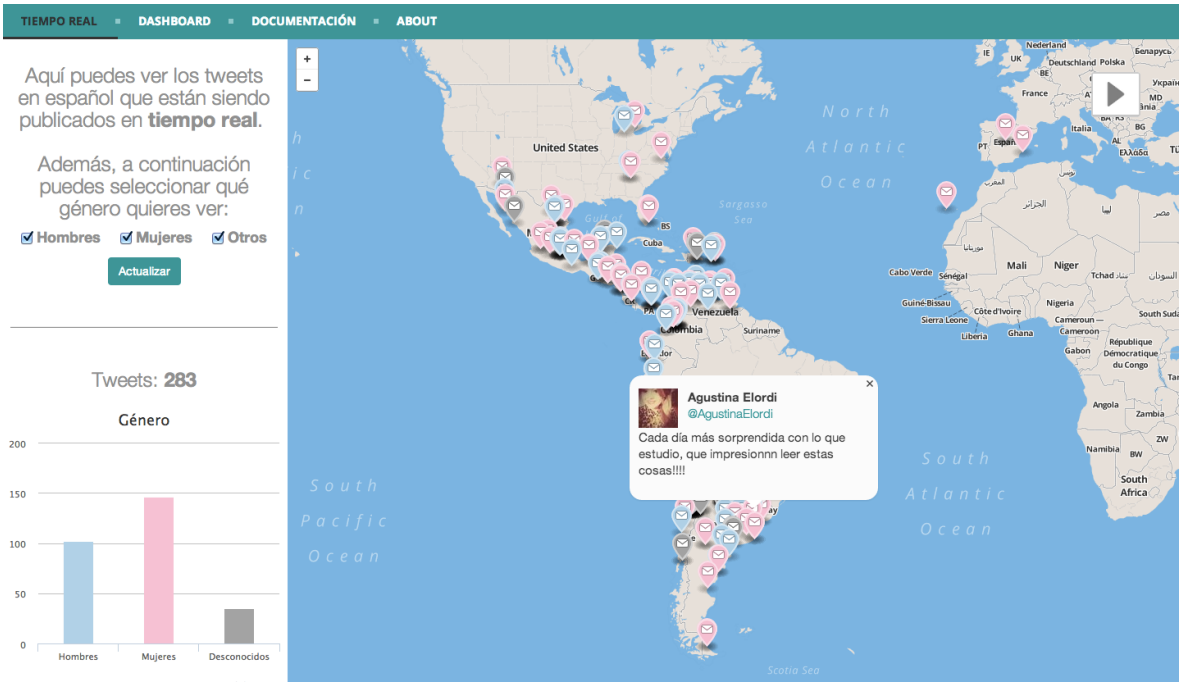

Figura 5.1. Vista general del mapa interactivo de la aplicación de ejemplo de uso del backend

de perfil, el texto del *tweet* con enlaces funcionales, y la imagen adjuntada en el *tweet* (si la hubiese)

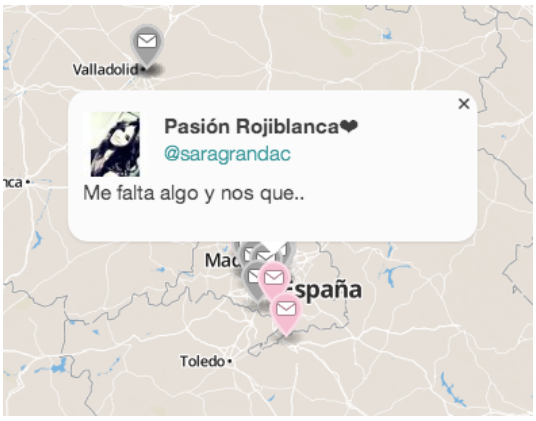

Figura 5. 2. Vista en detalle de un tweet *sobre el mapa interactivo*

como muestra la Figura 5.2. Por otro lado, por cada *tweet* recibido la gráfica y el contador son actualizados. La gráfica muestra la distribución de género de los usuarios que están tuiteando. además indica el número exacto de *tweets* publicados por usuarios de un género al pasar el ratón por encima de la barra correspondiente (Figura 5.3). El contador muestra el número total de *tweets* que han sido recibidos y que están mostrándose en el mapa.

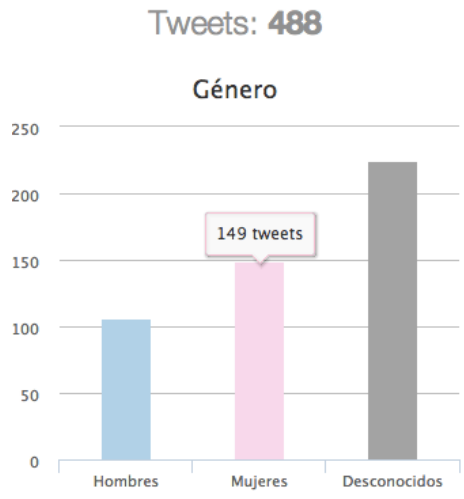

Figura 5.3. Contador de tweets y gráfica comparativa de géneros

El mapa también permite “filtros". En el menú de la izquierda se pueden seleccionar de qué géneros se quiere recibir *tweets* (Figura 5.4). Por defecto se mostrarán los *tweets* publicados por usuarios de cualquier género, pero mediante este menú podemos especificar si queremos recibir datos de algún género concreto. Esto se puede hacer incluso una vez que ya se ha puesto en marcha la visualización de *tweets*. En este caso, se borrarán los *tweets* mostrados que no cumplan los

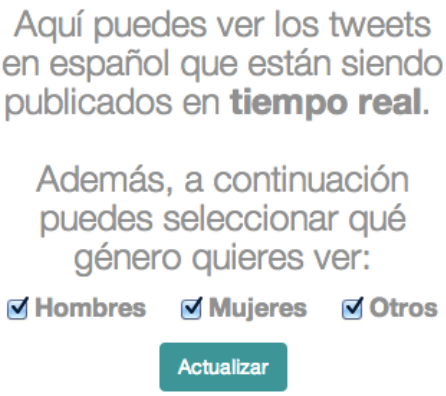

Figura 5.4. Menú del mapa interactivo

nuevos requisitos y se actualizarán la gráfica de distribuciones de géneros y el contador. En la Figura 5.5. se pueden ver dos capturas del mismo área geográfica aplicando el filtro de género, sexo masculino y femenino a la izquierda, y solo sexo femenino a la derecha.

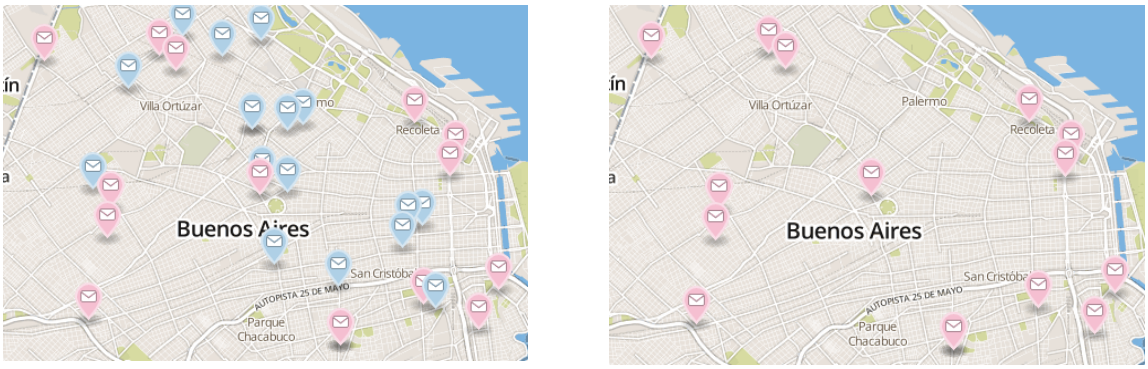

Figura. 5. 5. Comparación visual del mismo área geográfica en el mismo instante de tiempo aplicando el filtro disponible para la visualización de tweets *por género.*

Además, arriba a la derecha se encuentra un botón que detener la aplicación (Figura 5.6). Si se pulsa, los *tweets* seguirán llegando a través del *websocket*, pero la aplicación simplemente los ignorará, por lo que no se mostrarán en el mapa, ni modificarán la gráfica ni el contador. Al volver a pulsarlo, la aplicación volverá a tener en cuenta cada *tweet* que reciba, pero no analizará los que ignoró anteriormente.

Figura 5.6. Botón para detener la aparición de nuevos tweets.

5.1.2. Panel informativo

En este panel se pueden visualizar estadísticas extraídas de los datos recopilados por el extractor de *tweets* en las últimas 24 horas. Estas estadísticas se muestran mediante diferentes gráficas, cada una con un objetivo diferente, y unas opciones de interacción también diferentes. Mientras que la gráfica relacionada con el mapa muestra información en tiempo real, en esta sección se muestra información estática. La sección consta de las siguientes gráficas:

- **Número de *tweets* por minuto**. Gráfica de línea en la que los puntos indican el número de *tweets* registrados para cada minuto del intervalo. Esta gráfica es interactiva, permitiendo definir intervalos de tiempo menores, adaptándose ésta y mostrando más detalle, además de permitir ampliar la información de cada punto representado. También cuenta con un botón para regresar el nivel de zoom a su valor original. Figura 5.7.

- **Porcentaje de *tweets* categorizados respecto al total de *tweets* almacenados**. Gráfica de barras que muestra la relación entre los *tweets* que pertenecen a actualidad deportiva o a la

actualidad política en relación con el número total de *tweets*. A diferencia de la anterior, esta gráfica no es interactiva, aunque también permite ver información detallada de cada sección. Figura 5.8.

Figura 5.7. Número de tweets *por minuto*

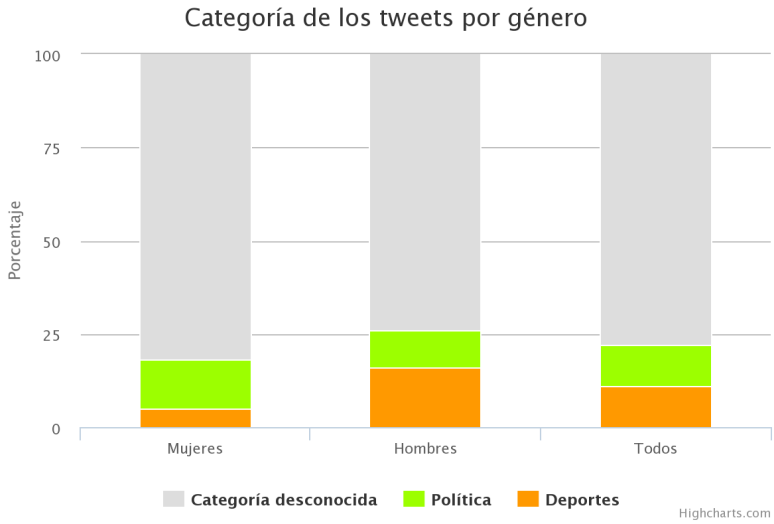

Figura 5.8. Porcentaje de tweets categorizados respecto al total de tweets almacenados

- **Proporción de género por usuario**. Gráfica de barras que índica la proporción existente entre los tres géneros posibles: “masculino”, “femenino”, o “sin determinar”. La diferencia con la gráfica que muestra la distribución de género en el mapa de la Figura 5.1 es que en el panel informativo estamos haciendo referencia a los datos disponibles en el backend referentes a las últimas 24 horas. Figura 5.9.

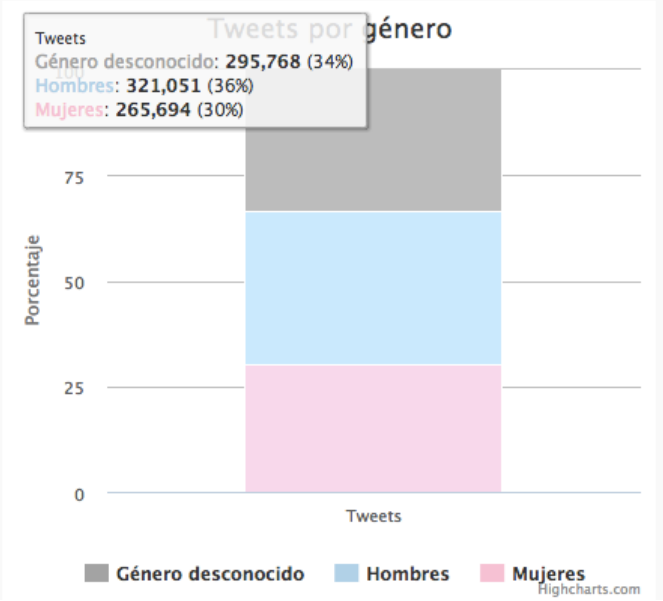

Figura 5.9. Proporción de género por usuario

5.2. Comunicación con el *backend*

Todas las gráficas presentes en la aplicación de ejemplo consiguen sus datos a través de llamadas a la API del *backend*, que fue descrita en el apartado 4.9. Estas llamadas se hacen

mediante peticiones AJAX[35]. En cada una de estas peticiones se define un *callback*, una función que se llamará cuando la petición finalice con éxito, en la que se define cómo se tratarán los datos recibidos. De esta forma, cada gráfica hace una petición al método de la API necesario y utiliza los datos incluidos en la respuesta sin necesidad de procesarlos.

En la Tabla 5.1 se puede ver una relación de las llamadas que cada uno de los componentes de la aplicación de ejemplo hace al *backend.*

Componente	Petición al *backend*	Protocolo
Mapa interactivo en tiempo real	/live	websocket
Gráfica de tweets en tiempo real	/live	websocket
Gráfica de número de *tweets*	/api/tweets/per-minute	Rest (GET)
Gráfica de categorías por *tweets*	*/api/tweets/categories-percentage*	Rest (GET)
Gráfica de *tweets* por género	*/api/tweets/genders-percentage*	Rest (GET)

Tabla 5.1. Relación de llamadas al backend

5.3. Tecnologías empleadas

Para esta aplicación de ejemplo se han usado tecnologías web actuales(HTML5, CSS3 y Javascript), junto con librerías específicas para cada una de las funcionalidades desarrolladas:

- **Mapbox.js**[36]**.** Librería encargada de gestionar el mapa, lo cual incluye la navegación por el mismo, la posibilidad de añadir o quitar marcadores (los *tweets*) y la visualización del código asociado a cada uno de ellos (visor detallado de cada uno de ellos).

[35] AJAX (Asynchronous JavaScript And XML) es una tecnología para el desarrollo web. Se puede encontrar documentación en *http://api.jquery.com/category/ajax/*

[36] *https://www.mapbox.com/mapbox.js/api/v1.6.4/*

- **Highcharts**[37]**.** Librería responsable de las gráficas, de las que ofrece multitud de variedades, además de una API con la que interactuar con ellas. De esta forma, nos es posible actualizar las gráficas con cada petición o en cada recepción de un *tweet*.
- **jQuery**[38]**.** Librería que, además de simplificar la manera de interactuar con los elementos de la propia página y hacer peticiones a nuestro *backend*, es necesaria para ejecutar Highcharts.
- **Autolinker.js**[39]**.** Librería muy ligera que busca enlaces dentro de un texto y los convierte en hipervínculos, por lo que es muy útil para el visor detallado de *tweets* en el mapa.
- **Less.js**[40]**.** Preprocesador de CSS que permite añadir a éste funcionalidades útiles, como declaración de variables, funciones y anidación de cláusulas.
- **Bootstrap**[41]**.** Framework frontend que simplifica el diseño de las páginas permitiendo la gestión sencilla de espacios, además de proveer de componentes editables ya diseñados.

Finalmente, la aplicación de ejemplo ha sido añadida al *backend* existente, para que éste sea el encargado de servir todos los archivos estáticos. Si bien esto se ha hecho así por comodidad, tanto el *backend* como la aplicación de ejemplo son independientes y podrían estar desplegados cada uno en un sitio diferente sin ningún problema.

5.4 Conclusiones

En este capítulo se ha explicado la aplicación desarrollada para la visualización de información procesada por el *backend* detallado en el capítulo 4. Esta aplicación se ha desarrollado como ejemplo de una aplicación externa capaz de usar los métodos provistos por la API desarrollada.

---

[37] *http://www.highcharts.com/component/content/article/2-news/139-highmaps-released*

[38] *http://jquery.com/*

[39] *https://github.com/gregjacobs/Autolinker.js*

[40] *http://lesscss.org/*

[41] *http://getbootstrap.com/*

En primer lugar se ha detallado el funcionamiento planificado para la aplicación y se han mostrado capturas de pantalla para ilustrar las diferentes funcionalidades. A continuación se ha explicado qué métodos de la API fueron necesarios para obtener la información necesaria en los diferentes componentes. Por último se han detallado las tecnologías web empleadas en la implementación de la aplicación.

6. Conclusiones y líneas de trabajo futuro

En este proyecto se ha presentado tanto un estudio teórico de la información que está disponible en Twitter como aplicaciones prácticas de cómo se puede analizar y visualizar esta información.

En el apartado teórico se ha estudiado qué trabajo previo existía relacionado con el uso de Twitter como fuente de información, se ha puesto sobre la mesa a qué información es posible acceder exactamente y se ha emprendido la labor de clasificarla. Para ello, se ha definido una clasificación basada en tres ejes que atiende a diferentes aspectos de la naturaleza de los datos, y sobre las definiciones, se ha clasificado toda la información extraíble de Twitter.

Partiendo de este marco teórico, se han determinado tres estrategias de generación de información procesada en español: la detección de género de los usuarios, la categorización de *tweets* por contenido y el posicionamiento de *tweets* por áreas.

Al llevar este trabajo a la práctica, se ha diseñado e implementado un sistema capaz de:

1. Conectarse con la Streaming API de Twitter para recibir *tweets* en tiempo real.
2. Almacenar en una base de datos no relacional los *tweets* geoposicionados en español en tiempo real.
3. Detectar en tiempo real el género de los usuarios basándose en su nombre y biografía del perfil.
4. Conectarse al RSS de periódicos digitales para extraer los artículos relacionados con la actualidad política y deportiva del día.
5. Categorizar en tiempo real textos cortos utilizando un corpus creado a partir de los artículos extraídos a través de los *feeds* de periódicos digitales.

6. Exponer por medio de peticiones a una API la información conseguida.
7. Gestionar concurrentemente la extracción, el procesamiento de grandes volúmenes de información y la respuesta a peticiones externas.

Además, como prueba de concepto del *backend* implementado se ha realizado una aplicación web que por medio de peticiones a la API desarrollada muestra información por medio de diferentes opciones de visualización. Más concretamente esta aplicación permite:

1. Ver mediante un mapa interactivo los *tweets* recibidos en tiempo real clasificados por género del usuario que lo ha publicado.
2. Visualizar mediante diferentes gráficas los datos extraídos de la información recibida: cantidad de *tweets*, categorías a las que pertenecen y género de sus escritores.

6.1. Líneas de trabajo futuro

A continuación, se expone lo que se considera es la continuación natural del trabajo desarrollado en este proyecto. Se trata de diferentes aspectos de mejora a funcionalidades ya presentadas.

1. **Mejorar la precisión del categorizador de *tweets* por contenido.** Dados los resultados presentados en el apartado 4.7.9, se observa que existe un gran margen de mejora en este aspecto. En concreto, conseguir determinar y separar los *tweets* que se pueden calificar como ruido o conversación vacía de los *tweets* que contienen información de interés.
2. **Ampliación del categorizador de *tweets* por contenido definiendo y tratando un mayor número de categorías.** Un objetivo a largo plazo podría ser cubrir un universo mayor de temas de conversación de actualidad en Twitter, y no solo actualidad deportiva y política.

3. **Ampliación de funcionalidad para el módulo de traducciones de localización geográfica.** El conjunto de áreas que pueden consultarse se encuentra predefinido actualmente; una posible mejora sería añadir la posibilidad al usuario de definir áreas geográficas para poder hacer consultas al sistema en base a esas áreas.

4. **Mejorar el módulo de detección de género.** El comportamiento actual del módulo está recogido en el apartado 4.6.3. Si bien se considera que en principio no hay una gran margen de mejora en la precisión del algoritmo, se podría profundizar en la distinción entre "marcas y empresas" y "personas", así como en añadir campos a analizar, como por ejemplo el texto de los *tweets* publicados por ese usuario o enlaces publicados en estos.

Por otro lado, existen múltiples líneas de trabajo futuro aparte de la mejora o ampliación del trabajo ya realizado. A la hora de diseñar la arquitectura se ha procurado hacer un código modular que facilitase su compresión y extensión por parte de terceros. También se ha puesto un énfasis especial para que todas las clases y métodos públicos cuenten con un Javadoc correcto y explicativo. Posibles líneas de trabajo podrían ser:

5. **Usar la API Rest de Twitter.** De nuevo, el proyecto se ciñe únicamente al Streaming de Twitter, dejando sin explotar las posibilidades que puede ofrecer este segundo método de acceso a la información en Twitter.

6. **Usar la información procesada por Twitter.** El proyecto emplea estrictamente información provista por Twitter sin procesar, pero como se vio en el apartado 3.1, Twitter también da acceso a información procesada por ellos mediante su API Rest. Sería posible utilizar estos datos, crear una nueva estrategia, diseñarla e implementarla.

7. **Añadir un analizador de sentimiento basado en contenido al procesamiento de *tweets* en tiempo real.** Actualmente los tres ejes sobre los que se cimienta el proyecto son el género del usuario, la categorización por contenido y la geolocalización, por lo que se podría añadir un analizador de sentimientos como cuarto eje de clasificación de la información.

8. **Profundizar en el tratamiento de Big Data.** Este proyecto puede suponer la puerta de inicio para proyectos que empleen grandes volúmenes de información como punto de partida para

el análisis de datos y estadísticas. En las condiciones actuales, el sistema recibe y procesa unos 1,6 millones de *tweets* al día, con lo que creemos que tiene potencial para proyectos que requieran de grandes cantidades de información.

9. **Añadir el análisis y visualización de redes para los datos procesados por el sistema.** Partiendo de los datos recogidos de Twitter, se puede establecer diferentes definiciones de redes aprovechando la gran cantidad de información respecto a las interacciones sociales entre los usuarios de la red. De esta forma, se podrían calcular diversas redes, e implementar su cálculo y visualización mediante grafos.

7. Anexos

Anexo A. Modelo de datos

Atributo	Tipo
id	ObjectId
rawTweet	String
text	String
hashtagEntities	List<String>
urlEntities	List<String>
userMentionEntities	List<String>
inReplyToScreenName	String
inReplyToStatusId	long
inReplyToUserId	long
latitude	double
longitude	double
source	String
userMentionEntities	List<String>
place	String
isPossiblySensitive	boolean
category	String
latitude	double
longitude	double
createdAt	Date
updatedAt	Date

Tabla 7.1. Atributos de la entidad tweet

Atributo	Tipo
id	ObjectId
userName	String
userId	long
verified	boolean
followersCount	int
friendsCount	int
genre	String
latitude	double
longitude	double
createdAt	Date
updatedAt	Date

Tabla 7.2. Atributos de la entidad usuario

Atributo	Tipo
id	ObjectId
name	String
latitude	double
longitude	double
category	Categor y
tweetId	ObjectId
createdAt	Date
updatedAt	Date

Tabla 7.3. Atributos de la entidad hashtag

Atributo	Tipo
id	ObjectId
url	String
title	String
latitude	double
longitude	double
category	Category
tweetId	ObjectId
createdAt	Date
updatedAt	Date

Tabla 7.4. Atributos de la entidad link

Anexo B. Detalles de implementación del lector de *feeds*

Todo el código relacionado con la extracción de contenidos provenientes de un conjunto de RSS, así como su empaquetado en archivos de texto con la extensión deseada, se encuentra en el paquete `utils.feedReader`. En este paquete se pueden encontrar cinco clases que implementan la funcionalidad descrita más una clase llamada `FeedReaderExample` que muestra cómo usar las clases del paquete.

Al usar este módulo es posible cargar la configuración por defecto como muestra el código de la Figura 7.1:

```
FeedReader feedReader = new FeedReader();
feedReader.createModel();
```

Figura 7.1. Código de ejemplo para la creación de un lector de feeds *con la configuración por defecto*

También se permite indicar un archivo de configuración diferente al de por defecto y/o una localización diferente para la carpeta destino del archivo generado (Figura 7.2):

```
FeedReader feedReader = new FeedReader(
    "ruta del archivo de configuración");
feedReader.createModel("ruta a la carpeta destino");
```

Figura 7.2. Código de ejemplo para la creación de un lector de feeds *con un archivo de configuración*

Por último, si no se desea leer un archivo de configuración sino únicamente obtener los textos tal cual se hayan publicado, es posible haciendo uso de las clases `Feed` y `FeedMsg` como muestra el código de la Figura 7.3.

```
FeedParser parser = new FeedParser();
Feed feed = parser.readFeed("dirección del feed");
for (FeedMsg message : feed.getMessages()) {
      System.out.println(message);
}
```

Figura 7.3. Código de ejemplo para imprimir por consola los textos leídos de una dirección

Anexo C. Detalles de implementación del clasificador de textos basado en Weka

El categorizador de textos basado en Weka explicado en el apartado 4.7.2 se puede encontrar en el paquete `utils.textClassifier.weka`. Puede recibir un único o varios archivos como entrada para construir el conjunto de entrenamiento como muestra las Figura 7.4.

```
// creación de un clasificador con un único archivo de datos
TextClassifier classifier = new TextClassifier(
    "dirección del archivo”);

// creación de un clasificador con varios archivos
TextClassifier classifier = new TextClassifier();
// para una serie de archivos de texto {
     // …
     classifier.updateDataset("dirección de archivo”);
// }
```

Figura 7.4. Código para la inicialización del módulo categorizador de textos basado en Weka

Las respuestas dadas por el categorizador de textos pueden ser una respuesta directa de la categoría escogida como muestra la Figura 7.5. o un vector de valoraciones asociadas a cada categoría como muestra la Figura 7.6.

```
// obtención del nombre de la categoría predicha
TextClassifier classifier = new TextClassifier();
// inserción de datos en el clasificador
String response;
response = classifier.classifyAndEvaluateMsg("texto a clasificar");
```

Figura 7.5. Código de ejemplo para obtener una respuesta directa por el categorizador de textos basado en Weka.

```
// obtención de un vector con valoración para cada categoría
TextClassifier classifier = new TextClassifier();
// inserción de datos en el clasificador
HashMap<String, Double> response;
response = classifier.classifyMessage("texto a clasificar");
```

Figura 7.6. Código de ejemplo para obtener el vector de pesos por el categorizador de textos basado en Weka.

Anexo D. Detalles de implementación del clasificador de textos basado en Lucene

La implementación del módulo para la categorización de textos basado en Lucene descrito en el apartado 4.5.4. se encuentra en `utils.textClassifier.lucene`. La clase principal de este paquete es `LuceneEvaluator`. Para poder usarla, es necesario entrenar primero el sistema con un corpus de documentos ya categorizados. Se ofrecen dos opciones para ello.

- Añadir cada texto usando el método `addText(String texto, String categoria)`
- Recoger todos los textos de un archivo de texto: `trainWithFile(String ruta)`

Una vez que se ha rellenado el *index* con documentos, es posible realizar consultas para categorizar un texto. Se ofrecen dos posibilidades dependiendo de la respuesta esperada.

- Nombre de la categoría escogida: método `queryAndEvaluate(String texto)`
- Vector de pesos asignados a cada categoría: método `query(String texto)`

La Figura 7.7. es un ejemplo de uso de este módulo.

```
// ejemplo de uso del categorizador de textos basado en Lucene
LuceneEvaluator clasificadorDeTextos = new LuceneEvaluator();

// entrenar el sistema recorriendo los textos de un archivo
clasificadorDeTextos.trainWithFile("ruta al archivo de texto");

// categorizar un texto
String categoria;
categoria = e.queryAndEvaluation("texto que queremos categorizar");

// comprobar la puntuación de la última respuesta
ArrayList vectorDePesos = e.getEvaluableData(categoria);

// categorizar un texto consultando los resultados intermedios
HashMap resultadosIntermedios;
resultadosIntermedios = e.query("texto que se quiere categorizar");
```

Figura 7.7. Código de ejemplo de uso del módulo de categorización de textos basado en Lucene

Bibliografía

[Bray, 2013] Peter Bray, Matt Peters, “Social Authority: Our Measure of Twitter Influence,” Moz, The Moz Blog, 13 de febrero del 2013. [Documento web]. Disponible: http://moz.com/blog/social-authority. [Accedido: 16 de junio de 2014].

[Coyne, 2013] Bridget Coyne, “Introducing Twitter Alerts,” Twitter Blogs, 25-Sep-2013. [Online]. Disponible: *https://blog.twitter.com/2013/introducing-twitter-alerts*. [Accedido: 9 de mayo de 2014].

[Dreyer, 2011] Kate Dreyer, “U.S. Demographic Profiles at Facebook and Twitter.” 9 de febrero del 2011 [Documneto web]. Disponible: *http://www.comscore.com/Insights/Data_Mine/U.S._Demographic_Profiles_at_Facebook_and_Twitter* [Accedido: 1 de junio de 2014].

[Europa Press, 2013] Europa Press, “Ya somos 500 millones de hispanohablantes,” ABC.es, 14 de enero 2013. [Documento web]. Disponible: h*ttp://www.abc.es/sociedad/20130114/abci-espanol-lengua-hablada-mundo-201301141519.html*. [Accedido: 19 de junio del 2014].

[Hannon, 2011] J. Hannon, K. McCarthy, y B. Smyth, “Finding Useful Users on Twitter: Twittomender the Followee Recommender,” en “Advances in Information Retrieval”, P. Clough, C. Foley, C. Gurrin, G. J. F. Jones, W. Kraaij, H. Lee, y V. Mudoch, Eds. Springer Berlin Heidelberg, 2011, pp. 784–787.

[INE, 2014] Instituto Nacional de Estadística, 2014. Apellidos y nombres más frecuentes. [Documento web]. Accesible: http://www.ine.es/daco/daco42/nombyapel/nombyapel.htm [Accedido el 4 de abril de 2014].

[Kelly, 2009] R. Kelly, “Twitter Study - August 2009.” 12 de agosto del 2009 [Documento web]. Disponible: http://web.archive.org/web/20120503013715/http://www.pearanalytics.com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009.pdf. [Accedido: 24 de mayo de 2014].

[Leetaru, 2013] K. Leetaru, S. Wang, G. Cao, A. Padmanabhan, and E. Shook, “Mapping the global Twitter heartbeat: The geography of Twitter,” First Monday, vol. 18, no. 5, abril de 2013. Disponible: http://firstmonday.org/ojs/index.php/fm/article/view/4366 [Accedido: 18 de junio de 2014].

[Manning, 2009] Manning, Christopher D., Prabhakar Raghavan, y Hinrich Schütze, “Introduction to information retrieval”, vol. 1. Eds. Cambridge: Cambridge University Press, 2009, pp. 117-129. Disponible: http://www.informationretrieval.org/ [Accedido: 18 de junio de 2014].

[McPherson, 2001] M. McPherson, L. Smith-Lovin, y J. M. Cook, “Birds of a Feather: Homophily in Social Networks,” Annual Review of Sociology, vol. 27, pp. 415–444, enero de 2001. Disponible: http://www.jstor.org/stable/2678628. [Accedido: 3 de abril de 2014].

[Phelan, 2010] O. Phelan, K. McCarthy, y B. Smyth, “Buzzer – Online Real-Time Topical News Article and Source Recommender,” in Artificial Intelligence and Cognitive Science, L. Coyle and J. Freyne, Eds. Springer Berlin Heidelberg, 2010, pp. 251–261.

[Romero, 2014] P. Romero, “Las alertas de Twitter se activan en España,” EL MUNDO, 6 de mayo del 2014. [Documento web]. Disponible: http://www.elmundo.es/tecnologia/2014/05/06/5368a250ca47410e0f8b456e.html. [Accedido: 1 de junio de 2014].

[Roomann-Kurrik, 2013] Arne Roomann-Kurrik, “Introducing new metadata for Tweets,” Twitter Blogs, 13-Feb-2013. [Documento web]. Disponible: https://blog.twitter.com/2013/introducing-new-metadata-for-tweets. [Accedido: 16 de junio de 2014].

[Rosas, 2010] M. V. Rosas, M. L. Errecalde, y P. Rosso, “Un Análisis Comparativo de Estrategias para la Categorización Semántica de Textos Cortos,” Procesamiento del Lenguaje

Natural, Revista no 44, marzo de 2010, pp 11-18. Disponible: http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/273. [Accedido: 10 junio de 2014].

[Shen, 2013] W. Shen, J. Wang, P. Luo, and M. Wang, “Linking Named Entities in Tweets with Knowledge Base via User Interest Modeling,” en “Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining”, Nueva York, NY, USA, 2013, pp. 68–76.

[Teutle, 2010] A. R. M. Teutle, “Twitter: Network properties analysis,” 2010 en “20th International Conference on Electronics, Communications and Computer (CONIELECOMP)”, 2010, pp. 180–186.

[Vaughan-Nichols, 2012] Steven J. Vaughan-Nichols, “How Twitter tweets your tweets with open source,” ZDNet, 30 de agosto del 2012. [Documento web]. Disponible: *http://www.zdnet.com/how-twitter-tweets-your-tweets-with-open-source-7000003526/*. [Accedido: 15 de abril de 2014].

[Weng, 2010] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He, “TwitterRank: Finding Topic-sensitive Influential Twitterers,” en “Proceedings of the Third ACM International Conference on Web Search and Data Mining”, Nueva York, NY, USA, 2010, pp. 261–270.