UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Desarrollo de un sistema de resúmenes de opiniones en Twitter

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Cerveró Orero, Adrián

Tutores: Hurtado Oliver, Lluis Felip

Pla Santamaría, Ferran

Valencia, junio de 2017

Resumen

En la actualidad gran parte de los contenidos existentes en Internet son creados por usuarios finales y no por organizaciones proveedoras de contenidos y desarrolladores. Esto ha dado lugar a lo que se ha denominado Web 2.0, una Web en la que predominan sistemas como foros, redes sociales, blogs, wikis y sistemas de recomendación, todos ellos caracterizados por una alta interacción de los usuarios. Estos sistemas generan una gran cantidad de información sobre todo tipo de temas de interés.

Es por ello que surge la idea de procesar esta información y mostrarla de una forma más clara y útil para poder sacar conclusiones. Con estos precedentes se plantea el presente proyecto que tiene por finalidad el desarrollo de un sistema de resumen visual de datos que giran en torno a una temática concreta. Este sistema consiste en una aplicación web que facilita el visionado de una extensa colección de tweets extraídos de la red social Twitter. Los tweets serán almacenados en una base de datos MongoDB y la aplicación desarrollada en JavaScript que permite ver su localización en un mapa, así como varías estadísticas como la frecuencia de palabras, los usuarios más mencionados o reproducir en una simulación los tweets de manera cronológica.

Palabras clave: Twitter, data mining, visualización de datos, recuperación de información, redes sociales.

Abstract

Nowadays, much of the content on the Internet is created by end users and not by content providers and developers. This has given rise to what has been called Web 2.0, a Web in which systems such as forums, social networks, blogs, wikis and recomendation systems predominate, all characterized by a high user interaction. These systems generate a great amount of information on all kinds of topics.

That is why the idea arises to process this information and show the most clear and useful way to conclude. With this background the current project is proposed, whose purpose is the development of a data visualization system that revolves around a specific theme. This system is a web application that facilitates the viewing of a collection of tweets extracted from social Twitter. Being able to see their location on a map as well as varies the statistics as the frequency of words, most users test or play in a simulation the tweets chronologically.

Keywords : Twitter, data mining, data visualization, social network.

Índice de contenidos

Índice de figuras

Glosario

Red social Es un medio de comunicación social que permite establecer contacto con otras personas por medio de la Web. Ej.: Facebook, Instagram, Twitter, Youtube…

Microblogging Servicio que permite a sus usuarios enviar y publicar mensajes breves, generalmente solo de texto. Las opciones para el envío de los mensajes varían desde sitios web, a través de SMS, mensajería instantánea o aplicaciones ad hoc.

API *Application Programming Interface*. Librería o conjunto de funciones y procedimientos a ser utilizadas por un programa informático, consiguiendo de este modo una o varias capas de abstracción en la programación de aplicaciones finales.

Crawler Programa informático dedicado a la obtención de datos de páginas Web.

Tweet Unidad de información de Twitter; mensaje textual publicado por un usuario en Twitter constituido por un máximo de 140 caracteres.

Retweet En Twitter, tweet reenviado por un usuario distinto al que envío dicho tweet.

Follower En Twitter, usuario que "sigue" a un usuario concreto, es decir, que le establece como un usuario a cuyos mensajes quiere tener acceso y de los que quiere recibir notificaciones.

Hashtag En Twitter, término precedido por el carácter '#' que se usa para resaltar o etiquetar la temática de la que trata el tweet generado. Ejemplo: '#educacion', referido a la temática de educación, como leyes y reformas del sistema educativo de un país.

Mención En Twitter, término precedido por el carácter '@' que se utiliza para referenciar a un usuario. Ejemplo: '@bancosantander', referido al usuario en Twitter asociado al Banco Santander.

JSON *JavaScript Object Notation.* Formato ligero de intercambio de datos, que se caracteriza por su simplicidad de sintaxis {"objeto": valor}.

Stopword O palabra vacía, es el nombre que reciben las palabras sin significado como artículos, pronombres, preposiciones, etc. que son filtradas antes o después del procesamiento de textos.

1. Introducción

A lo largo de los últimos años las redes sociales han ido creciendo hasta convertirse en un elemento fundamental de nuestra sociedad. Estas interconexiones surgidas desde los propios usuarios sirven para mantener amistades, conocer gente nueva, entretenerse, crear conjuntamente movimientos, compartir fotografías, comentar nuevos estados de ánimo, etc.

Esta nueva situación requiere de herramientas para gestionar, almacenar y analizar todas esas grandes cantidades de información. Partimos de la hipótesis de que las redes sociales no son sólo el escenario, sino un nuevo agente para tener en cuenta en el entorno de la comunicación y que destaca especialmente en la labor de crear tendencias y expresar opiniones.

1.1. Motivación

Originalmente, Internet era un lugar reservado para unos pocos, grandes empresas, mayormente, eran las que generaban el contenido. Los portales Web eran solamente páginas estáticas escritas en HTML, que se actualizaban cada poco tiempo. Su contenido ofrecía muy poca interacción por parte del usuario que apenas eran meros espectadores. La única forma de proporcionar datos era a través de formularios Web, ésta era la llamada Web 1.0.

Con el avance de Internet la Web fue evolucionando hasta el surgimiento de la Web 2.0 en 2004. Este concepto debe su nombre a los equipos de las empresas de O'Reilly y MediaLive, durante una discusión de grupo sobre la Web y su futuro.

La Web 2.0 no supone cambios a nivel arquitectónico, sino más bien, son cambios en la manera en que los proveedores de contenidos, desarrolladores y usuarios utilizan la Web. Todos estos cambios derivan en una mayor interacción por parte de los usuarios finales, que ahora ya no solo se limitan a ver contenidos, sino también a producirlos. Por esto, la Web 2.0 hace referencia a una serie de aplicaciones y servicios que son utilizados por los usuarios para publican contenidos e interactuar con la Web y con otros usuarios. Entre estos servicios se encontraría las redes sociales como Twitter.

En dichos servicios de la Web 2.0, a medida que los usuarios han ido generando contenidos se ha ido produciendo una progresiva y creciente sobrecarga de información. Y de esta sobrecarga nace la necesidad de crear sistemas capaces de gestionar y procesar la información con la finalidad de sacar información clara y que pueda ser de utilidad. Sistemas como el desarrollado a lo largo de este trabajo que permiten resumir de forma visual y sacar conclusiones rápidamente a partir de una gran cantidad de datos.

1.2. Objetivos

El objetivo principal de este trabajo es el desarrollo de una aplicación Web para la visualización de datos extraídos de la red social Twitter.

Esta aplicación tendrá tres sub-objetivos específicos:

- **Recolección de datos**. Implementar un programa Python de tipo *crawler* capaz de extraer los tweets y almacenarlos en una base de datos Mongo.
- **Procesamiento de los datos.** Desarrollar un programa Python que extraerá los tweets de la base de datos y los procesará para dejarlos listos para el servicio web.
- **Visualización de los datos**. Diseñar una Web que permita a los usuarios visualizar e interaccionar con los datos procesados anteriormente.

En este trabajo no se pretende hacer juicios de valor sobre los datos ni someterlos a un análisis exhaustivo, únicamente mostrarlos de forma objetiva, intuitiva, organizada e interactiva.

1.3. Metodología

La metodología seguida durante la realización de este proyecto se podría clasificar como metodología ágil basada en un desarrollo iterativo e incremental donde los requisitos han ido evolucionando junto con el proyecto.

Se partió de un requisito básico (poder ver los tweets representados en un mapa) y a partir de ahí el proyecto fue avanzando creando una aplicación más completa a base de añadir mejoras y nuevas prestaciones.

Para cada nueva iteración o característica que se añade se sigue unas fases básicas de desarrollo donde se parte de la búsqueda de requisitos, preguntándose que podemos añadir al producto para tener un software más completo. A continuación, una pequeña etapa de búsqueda e investigación de posibles herramientas que se pueden utilizar para comenzar con las fases de diseño e implementación. Por último, se realizan pruebas, se corrigen errores que puedan surgir y se comprueba que esté bien integrado con el resto de características.

Concluida la iteración se vuelve al punto de partida y se exploran nuevas posibilidades.

1.4. Estructura del documento

El documento presente se organiza de la siguiente manera:

En el capítulo 2 se da una visión de la materia que nos ocupa: la extracción de información. También se da una visión de la red social de donde se han recolectado los datos, Twitter. Por último, se analizan algunas herramientas similares a la desarrollada en este proyecto.

En el capítulo 3 se muestra la arquitectura del sistema, repasando todas las partes que lo forman desde un punto de vista general. Además, se habla de la función de cada módulo dentro del sistema.

A continuación, hay un capítulo dedicado a cada uno de los módulos que componen este sistema. El capítulo 4 está dedicado al módulo de extracción de información. El capítulo 5 al procesamiento de los tweets extraídos. El capítulo 6 se ve con más detalle la aplicación web donde se visualizarán los tweets procesados.

Por último, el capítulo 7 cierra este documento presentando las conclusiones del proyecto, así como problemas encontrados y se presentan a posibles trabajos futuros.

2. Estado del arte

2.1 Data Mining

En los tiempos actuales, la información es uno de los activos más importantes para las empresas y organizaciones en general, es por eso la relevancia de poder analizarla y generar decisiones, conocer al cliente y su comportamiento para así poder moldear el producto y adaptarlo a sus preferencias.

El *Data Mining* o minería de datos se basa en el estudio y tratamiento de datos masivos para generar información relevante a partir de ellos. Dicho de otra forma, el *Data Mining* es el conjunto de metodologías, aplicaciones y tecnologías que permiten reunir, depurar y transformar datos de los sistemas transaccionales e información no estructurada en información estructurada, para su explotación directa o para su análisis y conversión en conocimiento y así dar soporte a la toma de decisiones sobre el negocio.

Partiendo de esta definición hay varias aproximaciones como la de Piatesky-Shapiro (1991)[1] que defiende que el *Data Mining* comprende todo el proceso completo de extracción de información, desde la preparación de los datos a la interpretación de los resultados obtenidos. Por otro lado, Molina y García (2004)[2] explica que los datos que se almacenan en las bases de datos no son valiosos de por si, su valor real reside en la información que podamos extraer de ellos.

Twitter, así como muchas otras redes sociales, se ha convertido en una enorme fuente de información para el desarrollo y la investigación en muchos campos. Por ejemplo, en áreas como la asistencia humanitaria o el auxilio de catástrofes se utiliza información en tiempo real. Investigadores han usado Twitter para predecir terremotos [3] o identificar que usuarios son más relevantes de seguir en una situación de crisis [4]. Investigaciones como estas también se han realizado en países como Chile [5] o China [6].

Otra de las ramas que ha despertado interés dentro del *Data Mining* es el análisis de sentimientos. Es un gran reto dentro de las tecnologías del lenguaje el clasificar automáticamente un texto escrito, en un lenguaje natural, en un sentimiento positivo o negativo. Incluso es difícil ponerse de acuerdo entre diferentes anotadores humanos sobre la clasificación a asignar a un texto dado. La interpretación de cada uno puede ser diferente, además de verse afectado por diversos factores culturales y experiencias personales. La tarea resulta aún más difícil cuanto más corto y peor escrito esté el mensaje, como suele ser habitual en el caso de los mensajes en redes sociales como Facebook o Twitter.

2.2 Twitter

En este trabajo nos centraremos en el desarrollo de una aplicación para visualizar datos extraídos de una red social exclusivamente: Twitter. Twitter es una aplicación Web de *microblogging* muy popularizada y extendida por todo el mundo. Es una red social que permite que sus usuarios publiquen *tweets* pequeños mensajes de texto de hasta 140 caracteres que pueden ser leídos por cualquiera de los otros usuarios de la aplicación.

En Twitter cada usuario tiene una lista de seguidores (en inglés, *followers*), que son notificados de los *tweets* publicados por dicho usuario. A su vez, cada usuario tiene una lista de usuarios a los que sigue. Al igual que otras redes sociales, Twitter permite realizar menciones escribiendo el nombre del usuario precedido del carácter '@'.

Una de las características más interesantes de esta red social es la posibilidad de clasificar los *tweets* mediante el uso de etiquetas. Estas etiquetas o *hashtags*, van precedidas del símbolo '#'. Lo que permite fácilmente encontrar *tweets* de un tema concreto mediante el uso de estos *hashtags*.

Por último, los tweets permiten dos tipos de interacciones. Por un lado, el *retweet*, que permite al usuario reenviar el *tweet* a todos sus seguidores de esta forma se consigue propagar el mensaje por toda red. Por otro lado, el botón "me gusta" permite valorar positivamente un mensaje. Por tanto, un tweet contiene dos variables numéricas que permiten medir su interacción, el número de *retweets*, que son las veces que un determinado *tweet* ha sido retuiteado, y el número de "me gusta".

2.2 Tecnologías similares

En muchos campos, el estudio de redes sociales para la obtención de datos ha supuesto un gran avance. A continuación, veremos algunas aplicaciones de interés que se situarian en el mismo marco que la aplicación desarrollada a lo largo de este trabajo.

- **Twitter Analytics**[1]: Herramienta que permite ver estadísticas en torno a un usuario o evento concreto, mostrando gráficas y datos numéricos entre otros.

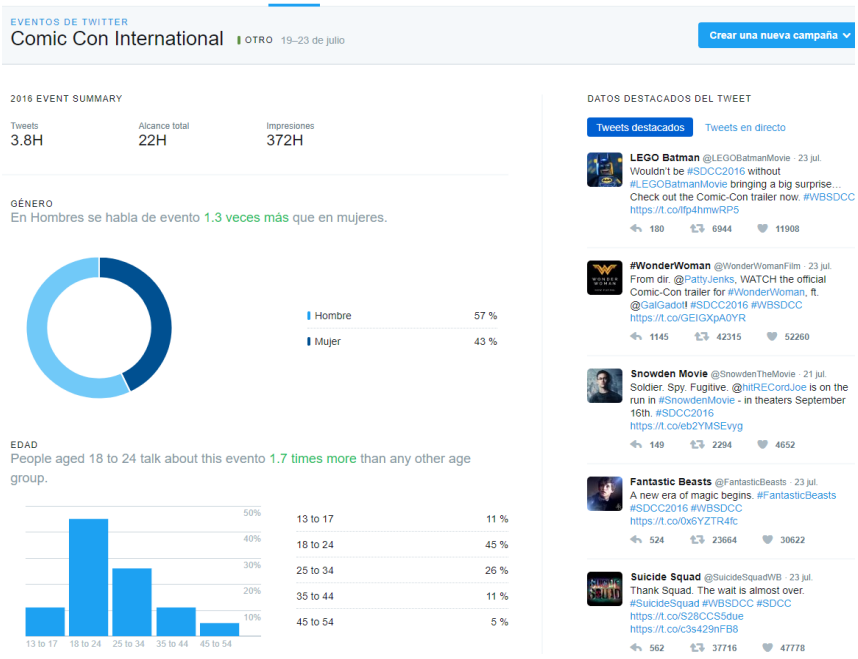

Figura 1: Captura de ejemplo de Twitter Analytics

[1] https://analytics.Twitter.com

- **One Million Tweet Map**[1]: Aplicación que muestra en tiempo real el último millón de tweets publicados y los representa en un mapa. También se puede filtrar usando hashtags.

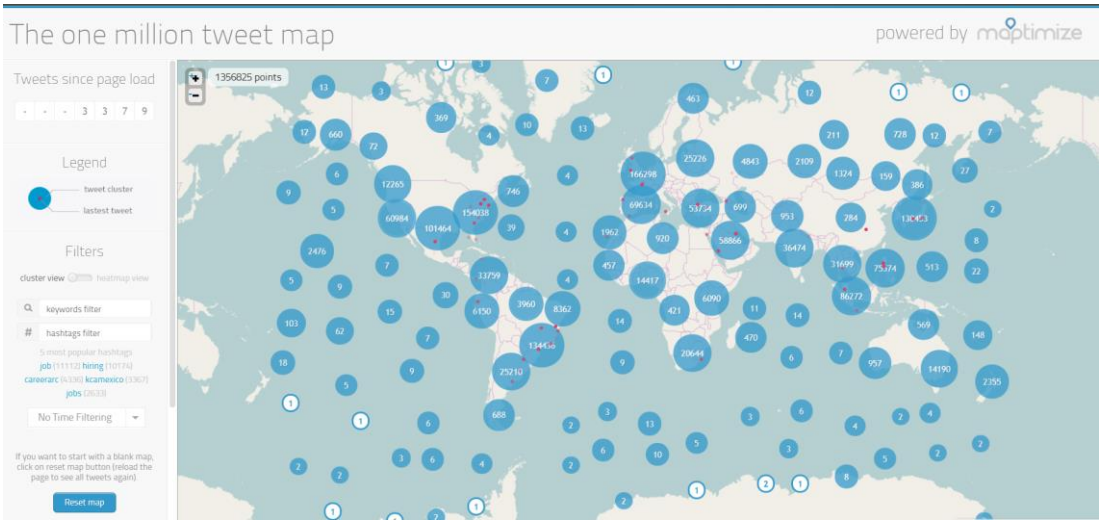

Figura 2: Captura de ejemplo de One Million Tweet Map

- **TweetsMap** [2] : Herramienta que realiza un análisis lingüístico de nuestros seguidores, nos indica el país, provincia y su ciudad de origen, la zona horaria en la que se encuentran, facilitándonos además un mapa y un gráfico interactivo de geolocalización que podemos exportar para realizar estudios y presentaciones.

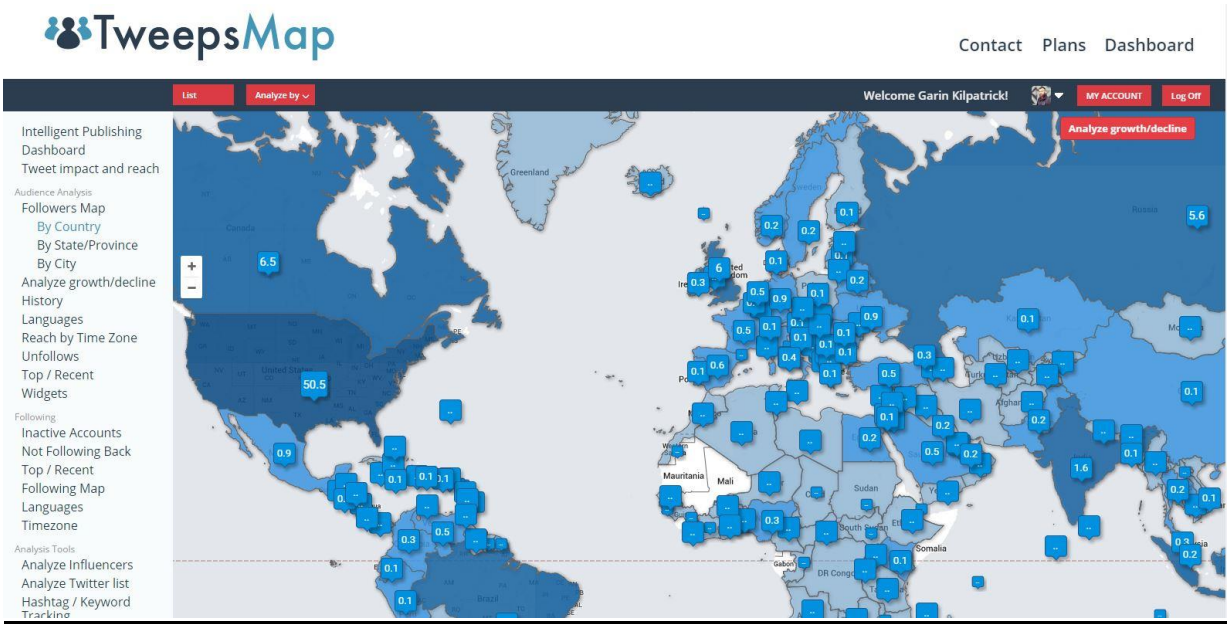

Figura 3: Captura de ejemplo de TweetsMap

[1] http://onemilliontweetmap.com/
[2] https://tweepsmap.com/es/

- **TweetStats** [1] : Es una de las aplicaciones de análisis de cuentas de Twitter que más tiempo lleva en funcionamiento. Se trata de un servicio de gratuito que te permite saber datos estadísticos de cuentas de Twitter a partir de los últimos tweets de la cuenta (hasta un límite).

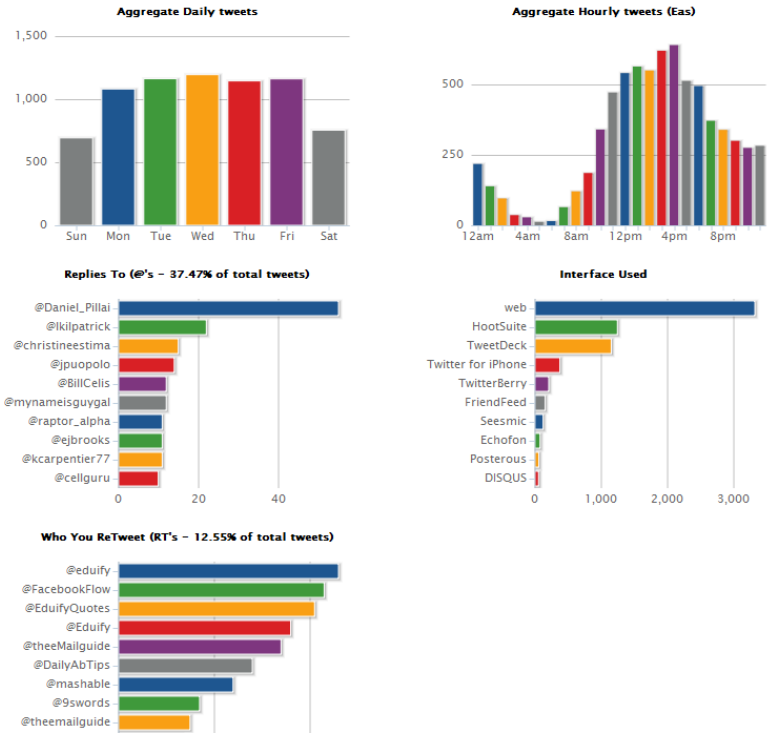

Figura 4: Captura de ejemplo de TweetStats

[1] http://www.tweetstats.com/

3. Arquitectura del sistema

En este apartado iremos viendo las diferentes partes que componen todo el sistema desarrollado en este trabajo. El diagrama de flujo que lo representa sería:

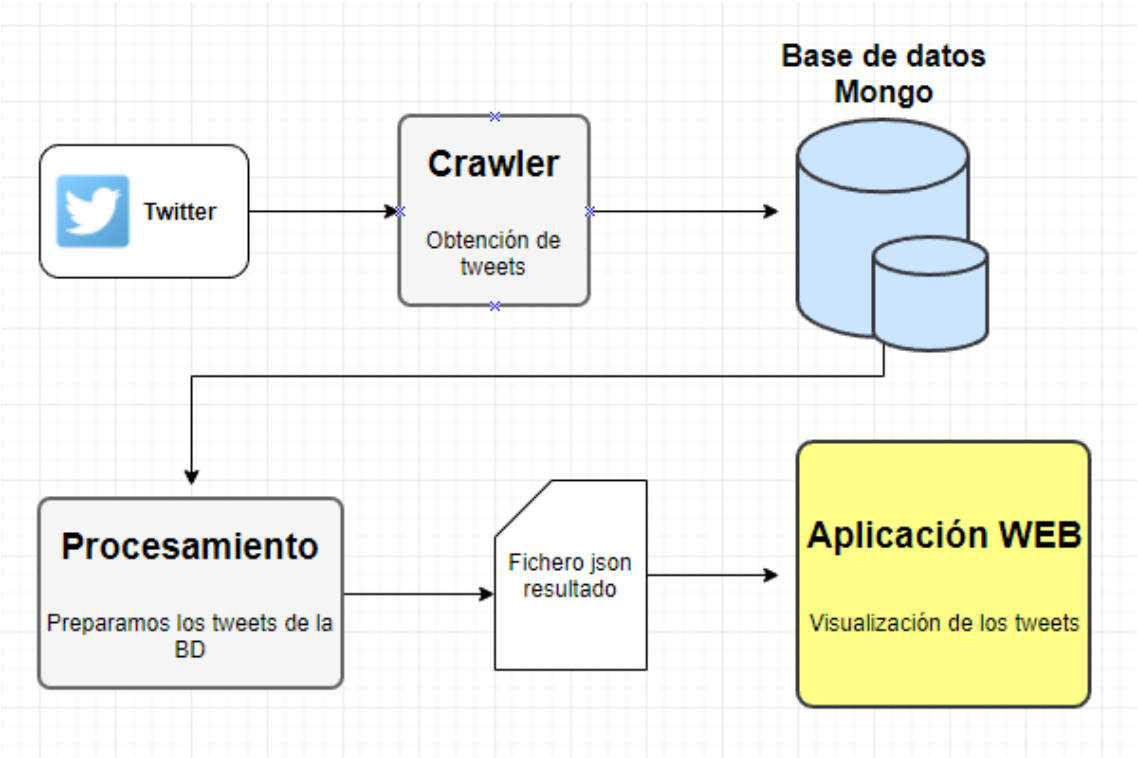

Figura 5: Diagrama de la arquitectura del sistema

Como vemos, los tweets pasan por diversas fases hasta que son visibles por el usuario. A continuación, veremos más detalladamente cada una de estas fases.

3.1 Recolección de los tweets

La primera de estas fases es la obtención de los propios tweets a partir de la API de Twitter y su almacenamiento en una base de datos Mongo.

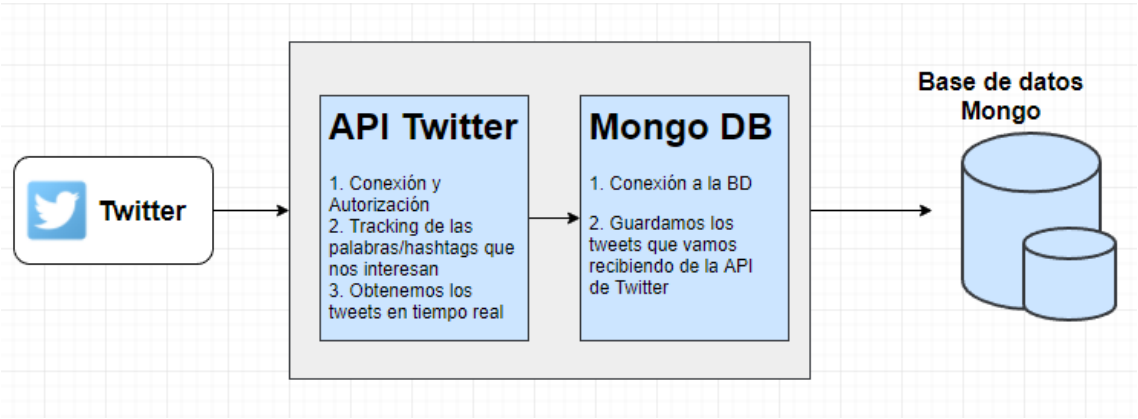

Figura 6: Diagrama del módulo de recolección de tweets

Para llevar a cabo esta fase, se ha desarrollado un módulo de recolección de datos o *crawler* de Twitter, que se encarga de obtener el corpus de tweets rastreando los tweets en tiempo real basándose en una lista de palabras/hashtags/usuarios que queremos rastrear. Esos tweets se van almacenando en una base datos mongo a medida que se van obteniendo, almacenándolos todos en la misma colección.

3.2 Procesamiento de los tweets

En la fase dos extraeremos una colección de tweets de la base de datos y los procesaremos dejando como resultado un fichero JSON como resultado. En este fichero guardaremos solo los tweets con la información procesada dejando solo los datos que nos interesan.

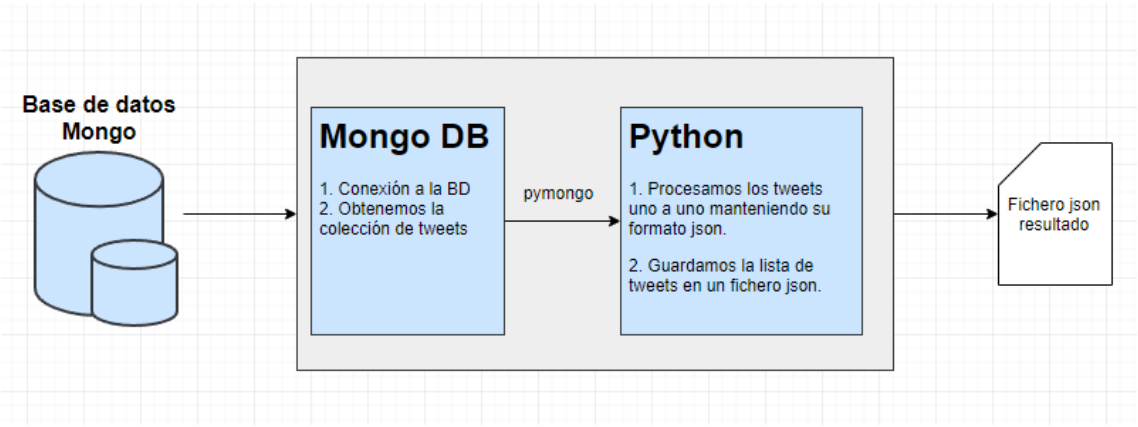

Figura 7: Diagrama del módulo de procesamiento de tweets

En este módulo será necesario tratar cada tweet de manera individual para realizar una tokenización del texto para eliminar *stopwords* y símbolos que no aportan información, clasificar cada token según su tipo (término, hashtag o mención) para luego calcular la frecuencia de palabras, usuarios, hashtags… Este programa permite el uso de parámetros por consola para seleccionar la colección, si queremos solo los tweets geolocalizados y otras opciones.

3.3 Aplicación Web

La aplicación final recibe el fichero JSON resultado del módulo anterior y construye todos los elementos gráficos para la visualización de los tweets.

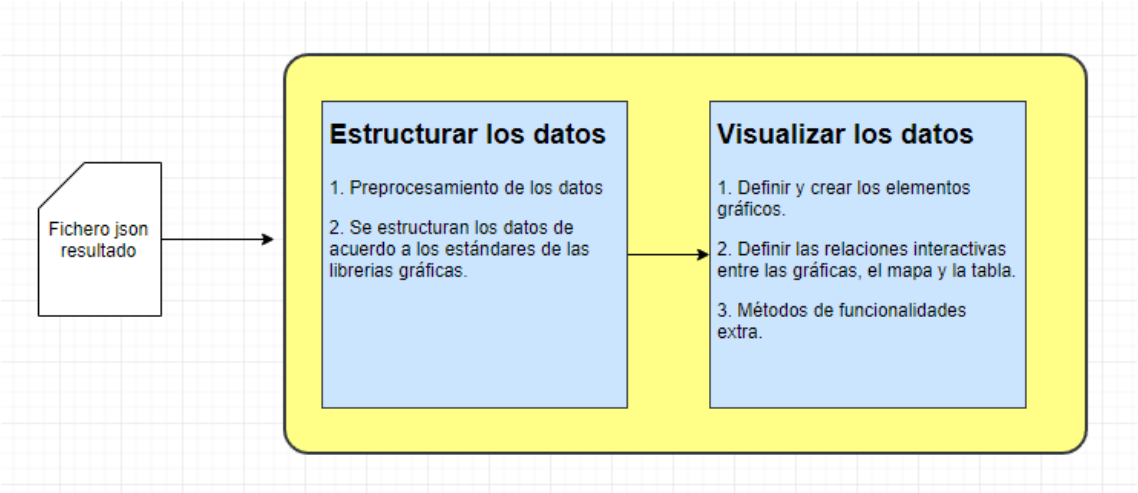

Figura 8: Diagrama del módulo de la aplicación web

El servidor mantiene la aplicación y el usuario puede interactuar con las gráficas, añadiendo y eliminando filtros. Los filtros usados en uno de los gráficos se ven reflejados en el resto.

4. Recolección de los tweets

El módulo de recolección de datos consiste en un programa desarrollado en Python[1] que toma como entrada parámetros que son los términos a rastrear. Haremos uso de dos librerias: Tweepy[2], para la conexión con la API de Twitter y la descarga de los tweets y Pymongo[3], para la conexión con la base de datos Mongo y almacenar los tweets.

Tweepy se encargará de comunicarse con la API Streaming de Twitter mediante la clase StreamListener. Para ello tendremos que realizar un proceso de autentificación usando los tokens correspondientes. A continuación, simplemente se inicia el *listener* con los parámetros de entrada que podrán ser una o varias palabras, normalmente hashtags de una determinada temática. Una vez en marcha, por cada tweet recuperado se imprimirá por pantalla un mensaje de recepción y se procederá a almacenarlo en la base de datos.

Esto se hará estableciendo previamente una conexión con la base de datos mediante Pymongo. Cada vez que se reciba un tweet este automáticamente será guardado en la base de datos.

El programa seguirá en ejecución hasta que se detenga manualmente.

4.2 Conexión a la API de Twitter

4.2.1 REST API y Streaming API

Twitter da soporte a dos API distintas: la REST API y la Streaming API. La primera funciona esencialmente mediante consultas a sus servidores para una petición concreta en la cual se recibe una respuesta (JSON, XML, etc), es la arquitectura típica de cliente-servidor. La Streaming API envía tweets basados en términos de búsqueda o en usuarios determinados que se soliciten, proporcionando información en tiempo real.

Para este proyecto se ha utilizado la Streaming API ya que no tiene las limitaciones que tiene la REST API de número de peticiones por unidad de tiempo, por lo que es más fácil para conseguir grandes cantidades indefinidas de tweets. Además, es la forma que permite cubrir un determinado evento (por ejemplo, unas elecciones o un partido de fútbol). Por otro lado, tiene una desventaja evidente ya que no requiere un tiempo de ejecución mayor puesto que el programa tiene que estar en ejecución durante el tiempo que dure el evento que se quiere cubrir. Además, la API no garantiza que se reciban el 100% de los tweets publicados durante este tiempo.

[1] https://www.python.org/
[2] http://www.tweepy.org/
[3] https://api.mongodb.com/python/current/

4.2.2 Proceso de Autorización

Open Authorization (OAuth)[1] es un protocolo que permite una conexión segura entre una API y una aplicación. Este proporciona a los usuarios un acceso a sus datos al mismo tiempo que protege las credenciales de su cuenta.

Para acceder a la API de Twitter a través del módulo creado es necesario registrar y autenticar la aplicación con los servidores de Twitter utilizando el protocolo OAuth. Para ello, el primer paso es conseguir las claves de autenticación a través del registro de la aplicación en el servicio para desarrolladores que proporciona Twitter[2]. Al final de este proceso se obtendrán la *consumer key* y el *consumer secret* que permiten la autenticación de la aplicación, así como el *access token* y el *access token secret*, que permiten la autenticación del desarrollador.

Una vez conseguidas las claves podemos establecer la conexión con la API fácilmente usando las funciones que proporciona Tweepy.

```
auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
```

4.3 Descarga de los tweets

Para solicitar los tweets de la API de Twitter utilizaremos la clase StreamListener de Tweepy para crear una conexión y esperar que el servidor nos vaya enviando los tweets.

```
listener = StreamListener(api=tweepy.API(wait_on_rate_limit=True))
streamer = tweepy.Stream(auth=auth, listener=listener)
streamer.filter(track=WORDS)
```

La lista de palabras que representa la variable WORDS serán los parámetros de búsqueda. Por ejemplo, si quisiéramos recibir tweets sobre la gala de los Óscars 2017 la variable WORDS podría ser algo como:

```
WORDS = ["#Oscars2017", "#oscars", "#Hollywood", "#bestmovie", "#bestdirector"]
```

De esta forma si ejecutásemos el programa durante la Gala de los Óscars podríamos recibir muchos de los tweets que contengan alguna de esas palabras en su texto en tiempo real.

Finalmente, solo quedaría guardar los tweets en una base de datos Mongo en nuestro caso. Para ello, se utiliza el método "on_data" del que dispone la clase *StreamListener*, que se ejecuta cada vez que se recibe respuesta del servidor, de tal forma que para cada tweet que recibimos nos conectamos a la base de datos y lo insertemos.

5. Procesamiento de los tweets

[1] http://oauth.net/
[2] https://apps.Twitter.com/

Twitter representa los tweets en formato JSON[1], es un formato muy sencillo orientado al intercambio de objetos. Un objeto JSON está constituido por una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un *objeto*, registro, estructura, diccionario, tabla hash, lista de claves o un registro asociativo.

Este módulo al igual que el anterior también es un programa implementado en Python gracias a su facilidad para trabajar en procesamiento de textos y con el formato JSON el cual es muy parecido a los diccionarios de Python.

El objetivo de este programa es procesar los tweets para que sean más manejables y quedarnos únicamente con la información estrictamente necesaria, además de simplificar la información para la aplicación web. Al igual que antes se utiliza la librería Pymongo para la comunicación con la base de datos.

5.1 Estructura de un tweet

Desde el punto de vista de Python un tweet se comporta como un diccionario con aproximadamente 30 entradas, estas entradas algunas tienes valores simples como un valor numérico o un *string* y otras tienen valores que son a su vez otros diccionarios o listas. Los datos que queremos saber de cada tweet son los siguiente: la fecha, el usuario, el texto, coordenadas (si tiene), palabras que aparecen, hashtags, menciones, número de retweets y "me gusta".

Para ahorrar el máximo espacio posible habrá que quedarse solo con la mínima información necesaria. Para ello, para cada tweet seleccionaremos solo las entradas que interesen y se irán añadiendo a un nuevo diccionario auxiliar. Al final de cada iteración, se añadirá este diccionario a una lista resultado que será la que se volcará en un fichero JSON.

5.2 Frecuencias de palabras, hashtags y menciones

Para las entradas que nos servirán para visualizar la frecuencia de términos lo que se hará es para cada tweet preparar tres listas, una para los términos, otra para los hashtags, y por último otra para las menciones a usuarios. Para ello, se tokeniza el texto del tweet usando la librería NLTK[2] y se descartan las palabras que no aportan información (*stopwords*, números, símbolos…). A continuación, solo queda definir tres listas y recorrer cada palabra para clasificarlas en una de estas tres listas: si la palabra empieza con el carácter '#' se considera un hashtag, si empieza por '@' es una mención y si no empieza por alguno de estos caracteres se considera un término.

5.3 Retweets y "me gusta"

Ya que se recuperan tweets en tiempo real es necesario ir actualizando los contadores de retweets y "me gusta" según se van produciendo. El procedimiento a seguir será comprobar si un tweet es un tweet nuevo o una réplica producida por un retweet de un usuario.

[1] http://www.json.org/json-es.html
[2] http://www.nltk.org/

Para comprobarlo se examinará si el tweet contiene el campo “retweet_status”. Este campo solo aparece en un tweet cuando se trata de un retweet y contiene el tweet original. El tweet original que aparece en “retweet_status” contiene el contador de retweets y “me gusta” actualizados hasta ese momento. Por lo que básicamente lo que hará el algoritmo es que cuando se encuentre un retweet actualizará los contadores (retweet_count y favorite_count) del tweet original en la lista resultado, que ya lo habrá procesado anteriormente, y se saltará al siguiente tweet de la colección.

5.4 Ejemplo de tweet procesado


```
tweet = {
        "date" : "Jun 25 16:38:10 +0000 2017",
        "geo" :{
                "type":"Point"
                "coordinates": [24.32452, -23.58375]
                },
        "term_freq" :    ["Castigo", "drive", "through", "trabajar",
                         "auto", "fuera", "fast", "lane"],
        "hash_freq": ["#ForceCheco", "#AzerbaijanGP"],
        "mention_freq": ["@SChecoPerez"],
        "rt_count" : 20,
        "likes_count" : 41,
        "text":  "Castigo a @SChecoPerez con un "drive
                through" por trabajar en el auto fuera del fast
                lane #ForceCheco #AzerbaijanGP",
        "name": "Checo Pérez News"",
        "user": "@ChecoPerezNews"
    }
```

Figura 9: Ejemplo de tweet procesado

6. Visualización de los tweets

En este apartado se verá con más detalle la aplicación donde se visualizará los datos recogidos de los anteriores módulos. Este módulo está desarrollado en JavaScript ya que dispone de gran variedad de librerías gráficas con la que mostrar los datos recogidos al usuario como las que se han utilizado.

6.1 Herramientas

- **Flask**

Es un framework minimalista escrito en Python que te permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código. Está basado en la especificación WSGI de Werkzeug y el motor de templates Jinja2 y tiene una licencia BSD.

- **Data-Driven Documents (D3.js)**

D3[1] (Data-Driven Documents) es una librería JavaScript para presentar y manipular visualmente documentos basados en datos. Con D3 podemos crear desde gráficas sencillas como barras e histogramas, hasta visualizaciones complejas e interactivas. D3 utiliza las tecnologías estándar como HTML5[2], SVG[3] y CSS[4], por lo que funciona en cualquier navegador web moderno.

Algunos ejemplos de lo que es capaz de hacer esta librería se pueden ver en:

https://github.com/d3/d3/wiki/Gallery

- **Crossfilter**

Crossfilter[5] es una librería JavaScript para estructurar grandes conjuntos de datos. Da soporte a interacciones muy rápidas y eficientes. Es capaz de agrupar, filtrar o agregar miles de filas de datos muy rápido y se combina perfectamente con D3 para visualizarlas.

- **Dimensional Charting (dc.js)**

dc.js es una librería JavaScript con soporte nativo con Crossfilter y que utiliza D3 para construir gráficas. Es el punto de unión entre ambas librerías y es la que permitirá poner en marcha la aplicación.

[1] https://d3js.org/
[2] https://www.w3.org/TR/html5/
[3] http://svgjs.com/
[4] https://www.w3.org/TR/CSS1/
[5] http://square.github.io/Crossfilter/

- **Leaflet**

Leaflet [1] es una librería JavaScript ligera que permite la construcción de mapas interactivos. Además, permite la inclusión de plugins para añadir funcionalidades extra.

- **MarkerCluster**

MarkerCluster [2] es un plugin para Leaflet para visualizar los puntos mostrados en el mapa como un clúster según su distancia.

6.2 Preparando los datos

Los tweets vendrán dados como una lista de objetos JSON, formato que Crossfilter admite perfectamente.

Pero antes de empezar con Crossfilter se realiza un preproceso de los datos recorriendo todas las filas y para formatear las fechas para que sean admitidas por D3. También se ha decidido establecer todas las fechas a 0 segundos para reducir la granularidad del histograma.

6.2.1 Configurando Crossfilter

A continuación, se definen los grupos y dimensiones para Crossfilter. Basícamente se crean una dimensión para cada columna de un tweet que necesitemos para construir las gráficas. En total, 5 dimensiones; la fecha, frecuencia de palabras, menciones, hashtags y una última que engloba todas las columnas.

Seguidamente se definen los grupos. Al igual que antes un grupo para cada columna más dos grupos más, uno para agrupar los tweets positivos y otro para los negativos.

Uno vez estructurado los datos en Crossfilter ya podemos empezar a construir las gráficas.

6.3 Interfaz

6.3.1 Histograma

El histograma permite representar los tweets a través del tiempo. Se marcan como fecha inicial el primer tweet cronológicamente y el ultimo como fecha final para acotar la gráfica.

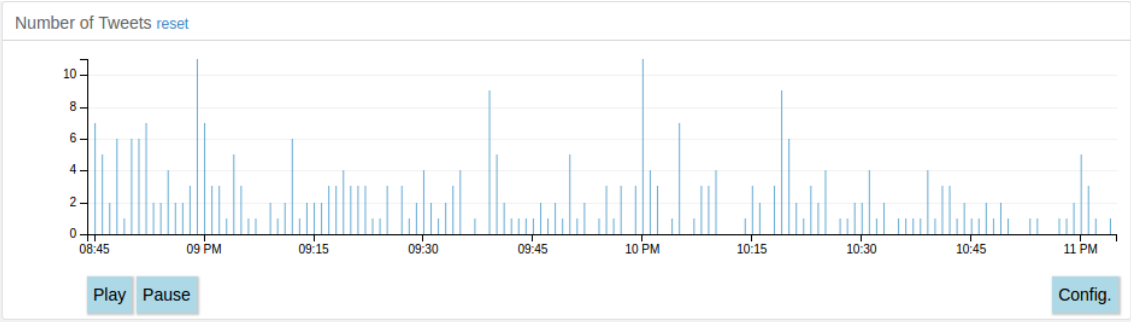

Figura 10: Histograma

[1] http://leafletjs.com/
[2] https://github.com/Leaflet/Leaflet.markercluster

6.3.2 Frecuencia de palabras, hashtags y menciones

Esta gráfica consiste en un diagrama de barras horizontales que representa la frecuencia de palabras/hashtags/menciones en el conjunto de tweets, ordenadas decrecientemente.

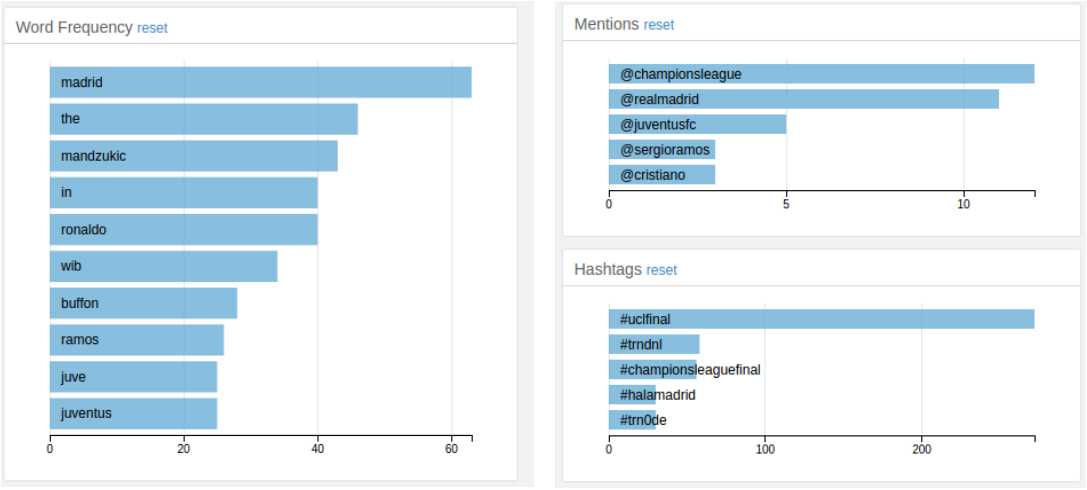

Figura 11: Gráficas de frecuencia de términos

6.3.5 Mapa

En el mapa se pueden ver los tweets geolocalizados por todo el mundo. Cada tweet es representado con un marcador. Los marcadores se aglutinan en clústers según su proximidad para mayor claridad.

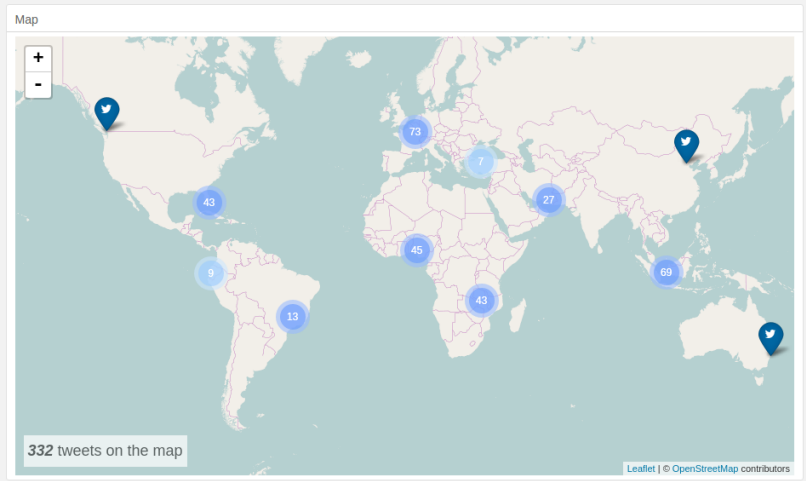

Figura 12: Mapa que muestra la localización de los tweets

Cada marcador se puede presionar para ver más información sobre el tweet en cuestión, como el usuario, el mensaje, la fecha…

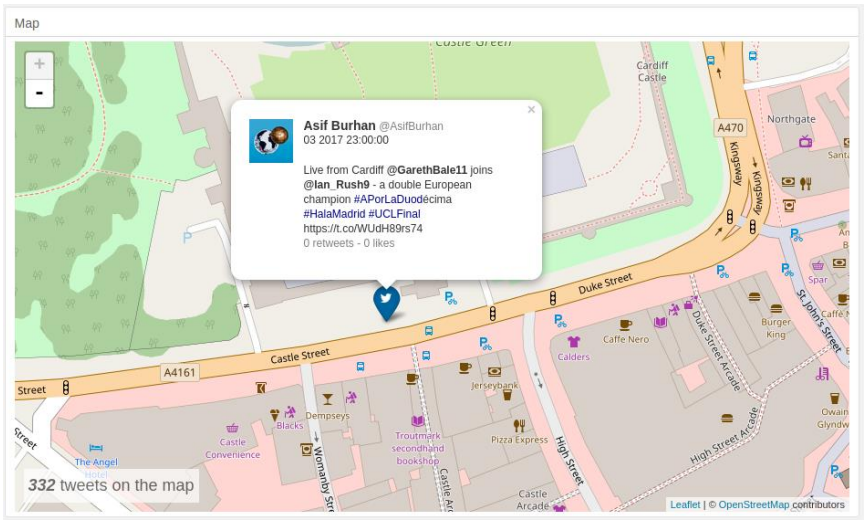

Figura 13: Ejemplo de tweet visto en detalle sobre el mapa

6.3.6 Contador

Indica el número de tweets que hay en la colección o están filtrados es ese momento. Por ejemplo, si en el histograma seleccionamos los tweets de entre las 9pm y las 10pm el contador mostrará el número de tweets en ese intervalo de tiempo.

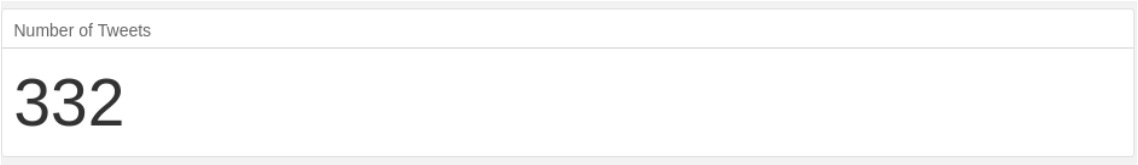

Figura 14: Contador de los tweets

6.3.7 Tabla

La tabla es un listado dividido en columnas donde cada fila representa un tweet de la colección. De izquierda a derecha: fecha, usuario, texto, retweets, likes, y si es positivo o negativo.

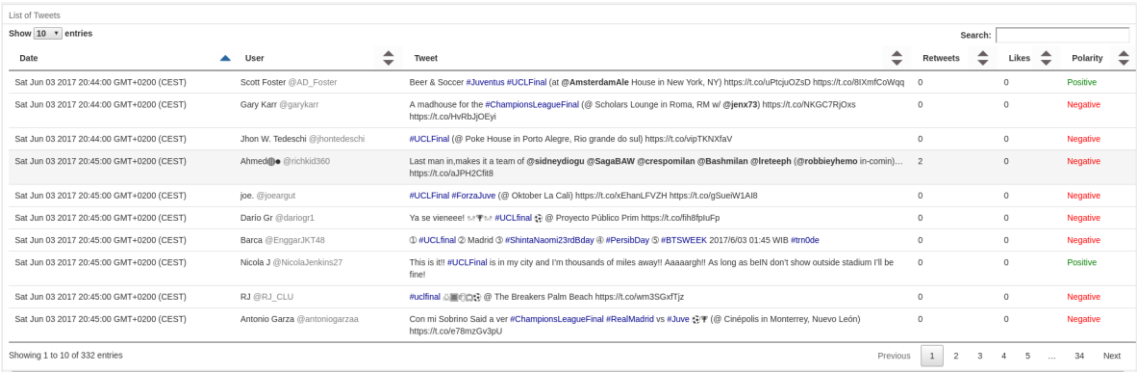

Figura 15: Tabla que contiene la colección de tweets

6.4 Funcionalidades

6.4.1 Filtrado

La característica básica de la aplicación y que se aprovecha de D3 y Crossfilter es la gran interactividad de las gráficas. El usuario puede seleccionar un conjunto de datos de una de las gráficas y automáticamente el resto de gráficas reaccionan filtrando para solo representar esa selección de tweets. Por ejemplo, posibilita seleccionar en el histograma los tweets de una fecha concreta a otra para así ver en el mapa los tweets, las palabras más frecuentes, etc., en ese margen de tiempo concreto.

6.4.2 Histograma animado

El histograma permite la reproducción de los tweets mediante una animación para verlos cronológicamente. De esta forma, se puede recrear la colección de tweets simulando que se representan en el mapa o el resto de graficas en tiempo real.

6.4.3 Visualizar análisis de sentimiento

Aunque en este proyecto no se trabaja el sentimiento de análisis de los tweets, la aplicación da soporte a ver la cantidad de tweets positivos y negativos en el histograma. Representando con verde a los tweets positivos y con rojo a los negativos.

7. Conclusiones

Este capítulo se dedica a conclusiones finales y personales del proyecto, extraídas a lo largo de todo el proceso de desarrollo del mismo.

El objetivo del trabajo era crear una aplicación para resumir de forma visual datos extraídos de Twitter por lo que podemos darlo por cumplido llegados a este punto. Tanto la extracción usando la API de Twitter y su posterior visualización se pueden ver en la aplicación final.

Los principales problemas que me he enfrentado en la realización de este proyecto, ha sido mi poca formación en cuanto a las herramientas que me había planteado utilizar. Puesto que no había utilizado nunca el API de Twitter, ni había trabajado con aplicaciones web. Pese a ello, puedo decir que el resultado del proyecto en general ha sido satisfactorio. Sin embargo, se podría haber mejorado la aplicación, así como reducido el tiempo de desarrollo de este trabajo de haber estado más versado en la materia.

Otro de los problemas fue el tamaño de la colección de tweets. Al principio del proyecto se partía de una colección bastante amplia, 2 millones de tweets aproximadamente. Ha medida que las gráficas y las funcionalidades fueron aumentando la cantidad de tweets en la colección tenía que irse reduciendo para mantener el funcionamiento de la aplicación de manera fluida. La cantidad de tweets utilizado en las últimas versiones corresponden a colección más pequeñas de hasta 3000 tweets, a partir de ese límite el rendimiento empieza a verse afectado.

En cuanto a lo que he aprendido realizando este proyecto: además de haber aprendido a desarrollar un proyecto de principio a fin, pasando por todas sus fases, he adquirido nuevos conocimientos, o ampliado muchos de ellos sobre Python, JavaScript, HTML, CSS, MongoDB, y otros conocimientos necesarios que se han requerido durante todas las fases.

Como trabajo futuro se abre la posibilidad a varios proyectos. Uno de ellos podría haber sido establecer que el sistema funcionase en tiempo real con la con posibilidad de buscar tweets y añadirlos a medida que se publican. Posibilitando, por ejemplo, monitorizar un evento en directo.

Además, por supuesto, se podría añadir más gráficas que muestren aún más información al usuario como el idioma de los tweets, o si se han escrito a través de un dispositivo móvil o desde un navegador web.

BIBLIOGRAFÍA

RUSSELL, Matthew A. (2013), Mining the Social Web Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and more. 2ª Edición. O'Reilly

KUMAR, Shamanth; MORSTATTER, Fred; LIU, Huan (2013), *Twitter Data Analytics*. Springer

KOUL, Anmol (2015) *Interactive Data Visualization using D3.js, DC.js, Nodejs and MongoDB*. Disponible en: https://anmolkoul.wordpress.com/2015/06/05/interactive-data-visualization-using-d3-js-dc-js-nodejs-and-mongodb/ [Documento online] [Consultado 07/2017]

BONZANINI, Marco (2015) *Mining Twitter Data with Python*. Disponible en: https://marcobonzanini.com/2015/03/02/mining-twitter-data-with-python-part-1/ [Documento online], [Consultado 07/2017]

MOUJAHID, Adil (2014*) An Introduction to Text Mining using Twitter Streaming API and Python*. Disponible en: http://adilmoujahid.com/posts/2014/07/twitter-analytics/ [Documento online] [Consultado 07/2017]

MOUJAHID, Adil (2016) *Interactive Data Visualization of Geospatial Data using D3.js, DC.js, Leaflet.js and Python*. Disponible en: http://adilmoujahid.com/posts/2016/08/interactive-data-visualization-geospatial-d3-dc-leaflet-python/ [Documento online] [Consultado 07/2017]

[1] PIATESKY-SHAPIRO, G.; FRAWLEY, W. (1991). *Knowledge Discovery in Databases*.

[2] MOLINA J. y GARCÍA, J. (2004). *Técnicas de análisis de datos*. Disponible en: http://ocw.uc3m.es/ingenieria-informatica/analisis-de-datos/libroDataMiningv5.pdf [Documento on-line], [Consultado 07/2017].

[3] T. Sakaki, M. Okazaki, and Y. Matsuo (2010). *Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors.* Disponible en: http://www.ymatsuo.com/papers/www2010.pdf [Documento on-line], [Consultado 07/2017]

[4] S. Kumar, F. Morstatter, R. Zafarani, and H. Liu. *Whom Should I Follow? Identifying Relevant Users During Crises*. Disponible en: http://www.public.asu.edu/~huanliu/papers/ht2013.pdf [Documento on-line], [Consultado 07/2017]

[5] M. Mendoza, B. Poblete, and C. Castillo. *Twitter Under Crisis: Can we Trust What We RT?* Disponible en http://snap.stanford.edu/soma2010/papers/soma2010_11.pdf [Documento online] [Consultado 07/2017]

[6] Y. Qu, C. Huang, P. Zhang, and J. Zhang. *Microblogging After a Major Disaster in China: A Case Study of the 2010 Yushu Earthquake.* Disponible en: https://www.researchgate.net/publication/220878979_Microblogging_after_a_Major_Di

saster_in_China_A_Case_Study_of_the_2010_Yushu_Earthquake [Documento online], [Consultado 07/2017]

.