UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Análisis de redes sociales y minería de textos aplicadas a caracterizar comunidades de usuarios en Twitter

Máster Universitario en Ingeniería Informática

Autor: Blázquez Pardo, Roberto

Tutor(es): A. Haya Coll, Pablo

Departamento de Ingeniería Informática

FECHA: Septiembre, 2019

Análisis de redes sociales y minería de textos aplicadas a caracterizar comunidades de usuarios en Twitter

AUTOR: Roberto Blázquez Pardo
TUTOR(ES): Pablo A. Haya Coll

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Septiembre de 2019

Resumen

El contenido publicado en las redes sociales se ha convertido en uno de los grandes objetos de estudio en la actualidad, debido a que la información obtenida resulta de gran utilidad para multitud de propósitos, como pueden ser temas comerciales o detección de radicalización en las personas.

Este Trabajo de Fin de Máster (TFM) se basa en la elaboración de un sistema informático distribuido, que analiza los tweets publicados durante un periodo de tiempo en concreto, extrae la información de ellos y agrupa a los usuarios en comunidades, con el objetivo de obtener los temas que se comentan en cada comunidad.

En primer lugar, se agruparán y filtrarán todos los tweets de las distintas fuentes. Una vez tengamos el conjunto de datos, se generarán las comunidades de usuarios con respecto a los retweets y menciones, y se procesará cada uno de los tweets para obtener la información sobre la que el usuario escribe.

Una vez se hayan generado las comunidades y procesado los tweets, se insertan los resultados en una base de datos NoSQL y se visualizan mediante distintas gráficas, para poder saber que está comentando cada una de las comunidades.

Para que el sistema sea posible, se han elegido una serie de herramientas para apoyar al desarrollo. Apache Kafka como sistema de transmisión distribuida, Apache Spark para el procesamiento en paralelo, Spacy para el procesamiento de lenguaje natural, Elasticsearch como base de datos donde alojar los resultados, y Kibana como herramienta de visualización de los datos resultantes del sistema.

El TFM está englobado en dos de los temas más actuales dentro del mundo de la informática, Big Data, debido a la cantidad de tweets que se generan y debemos de analizar, y el procesamiento del lenguaje natural (NLP).

Palabras clave

Big Data, Twitter, tweet, Apache Kafka, NLP, Spacy, Apache Spark, comunidades, Elasticsearch, Kibana

Abstract

Nowadays, most of the news published on social networks are one of the great object of study, due to that information is very useful for many purposes, such as commercial issues or detect radicalization in people.

The Master's Thesis (TFM) is based on the development of a distributed computer system. Its function is to analyze the tweets published during a specific period of time, extract the information from the tweets and group users into communities. The objective is to obtain the topics discussed in each community.

First, all tweets will be grouped and filtered from different sources. Once we have the data set, user communities will be generated with respect to retweets and mentions, and each tweet will be processed to obtain the information about which the user writes.

Once the communities have been generated and the tweets processed, the results will be inserted into a database and visualized through different graphs, to be able to know what each of the communities is commenting on.

To make the system possible, a number of tools have been chosen to support development. Apache Kafka as a distributed transmission system, Apache Spark for parallel processing, Spacy for natural language processing, Elasticsearch as a database to host the results and Kibana as a tool for displaying the resulting data of the system.

The TFM is encompassed in two of the most current issues within the world of computing, Big Data, due to the amount of tweets that are generated and analyzed, and natural language processing (NLP).

Keywords

Big Data, Twitter, tweet, Apache Kafka, NLP, Spacy, Apache Spark, communities, Elasticsearch, Kibana

Agradecimientos

En primer lugar, quiero agradecer a mi tutor Pablo, todo el apoyo que me ha dado y toda la ayuda que me ha aportado durante la elaboración del TFM.

También quisiera agradecer a mis compañeros del Máster entre los que nos hemos apoyado para poder terminar con éxito esta etapa. A los profesores del Máster por compartir sus conocimientos y apoyo durante la realización de este.

Por último, me gustaría agradecer a mi familia y amigos todo el apoyo que me han ofrecido durante esta etapa.

Índice

Índice de figuras

Índice de tablas

1 Introducción

1.1 Motivación

La realización de este TFM está motivada por la necesidad de agrupar a los usuarios y analizar sus ideas mediante sus tweets publicados en España en español durante un periodo concreto en Twitter. Por lo que se construye un sistema capaz de realizar dicha tarea, empleando técnicas de Big Data y de procesamiento del lenguaje natural (PLN).

Una vez dispongamos del resultado de los datos analizados a través del sistema creado, podemos observar que tema comenta y discute cada comunidad durante un periodo en Twitter. Dichos resultados nos aportan información para muchas aplicaciones en las distintas áreas, desde temas comerciales y de publicidad, como puede ser la decisión de a qué personas dirijo una campaña publicitaria, hasta temas policiales, como la detección de radicalización de colectivos.

1.2 Objetivos

El objetivo principal del TFM es desarrollar un sistema informático distribuido que realice el análisis de las conversaciones que se producen en Twitter. El sistema extraerá grupos de conversación mediante algoritmos de detección de comunidades, analizará la actividad de publicación, y los temas para cada comunidad, y realizará una adecuada visualización de los resultados accesible en línea.

Para validar el TFM se ha utilizado un conjunto de aproximadamente 3.900.000 de tweets, en formato JSON, publicados en España durante 3 periodos en concreto el día 17/11/2018.

1.3 Estructura del trabajo

La memoria consta de los siguientes capítulos:

- **Estado del arte**

En este capítulo se realiza un estudio de las tecnologías actuales en relación con el proyecto, y en base a los resultados se escogerán las más adecuadas para su desarrollo.

- **Diseño**

En este apartado se describe la arquitectura propuesta del sistema informático distribuido.

- **Desarrollo**

En esta sección se describe la implementación de cada módulo de la arquitectura propuesta para el sistema informático.

- **Integración, pruebas y resultados**

En esta parte se describen las pruebas realizadas y los resultados obtenidos sobre el sistema informático distribuido.

- **Conclusiones y trabajos futuros**

Por último se describen los argumentos y afirmaciones relativos a los resultados obtenidos. También se describe una serie de tareas para mejorar nuestro sistema.

2 Estado del arte

En este capítulo se realiza un estudio de las tecnologías actuales en relación con el proyecto.

2.1 Sistema de intermediación de mensajes

Debido a la necesidad de desacoplar los datos y distribuir las tareas entre diferentes nodos, se ha decidido implementar un sistema de intermediación de mensajes, por ello se ha escogido la plataforma Apache Kafka para su desarrollo, sobre otras como pueden ser RabbitMQ o Active MQ.

Figura 1. Logo Apache Kafka

Apache Kafka es una plataforma de transmisión distribuida, donde una serie de productores insertan mensajes, también llamados registros, en una cola, y a continuación, los llamados consumidores leen mensajes de la cola. Dichos mensajes son almacenados en disco de forma duradera y tolerante a fallos, gracias a la posibilidad de poder crear un clúster de servidores [1].

Los registros son almacenados dentro de categorías llamadas *topic*, las cuales disponen de varias particiones, los registros de entrada se introducen en las particiones asignándoles un identificador único por partición. Una gran característica, es que los registros pueden mantenerse en la partición, hasta un cierto tiempo, aunque sean leídos por un consumidor. Las particiones se distribuyen entre los servidores del clúster, y cada una de ellas se replica en un número considerable de servidores para soportar tolerancia a fallos.

Kafka dispone de 4 APIs:

- **Producer**: permite introducir los registros en las *topic*.
- **Consumer:** permite leer los registros disponibles en los *topic*.
- **Streams:** es una librería para construir aplicaciones y microservicios, donde los datos de entrada y salida son almacenados en un clúster Kafka.
- **Connector:** nos permite construir *producer* y *consumer* que se conectan a un *topic*, y nos permiten mover grandes cantidades de datos dentro y fuera de un clúster Kafka.

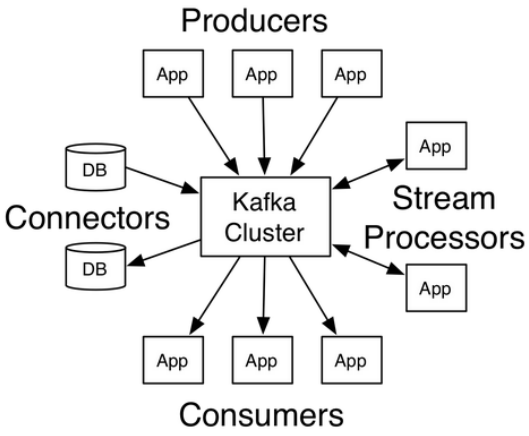

Figura 2. Arquitectura Apache Kafka

2.2 Sistema de computación a gran escala

Para el procesamiento de los datos, se ha decidido investigar varias herramientas de computación a gran escala, debido a que el tamaño de la fuente de datos en muy grande. Una vez analizadas las herramientas, se ha escogido el framework Apache Spark, sobre otras herramientas, como pueden ser Hive, Impala o Presto, debido a su gran expansión y facilidad para implementar.

Figura 3. Logo Apache Spark

Apache Spark es un framework de computación en clúster de grandes tamaños de datos y de código abierto. Está integrado con Apache Hadoop y dispone de APIs para Java, Scala Python y R.

Spark trabaja principalmente en memoria, lo que nos proporciona una mayor velocidad de procesamiento, aunque en el caso necesario de que la información no quepa en memoria, Spark empieza a escribir en disco, sacrificando velocidad de procesamiento.

Apache Spark permite el procesamiento en tiempo real con el módulo Spark Streaming, que con la ayuda de SparkSQL nos permite transformar y volcar los datos según se vayan inyectando.

Una de las mayores características de Apache Spark, es que trabaja con una evaluación perezosa, es decir, todas las operaciones que se realizan sobre los RDD, no se realizan hasta que forcemos su ejecución, mientras se van almacenando en un grafo acíclico dirigido (DAG) [2].

Apache Spark está basado en cinco componentes principales:

- **Spark Core:** el núcleo de Spark, donde se apoyan el resto de los módulos.
- **Spark SQL:** módulo diseñado para el procesamiento de datos estructurados y semi-estructurados.
- **Spark Streaming:** este módulo nos permite la ingesta de datos en tiempo real.
- **Spark MLLib:** librería que dispone de un gran número de algoritmos de Machine Learning.
- **Spark Graph:** módulo que permite el procesamiento de grafos (DAG).

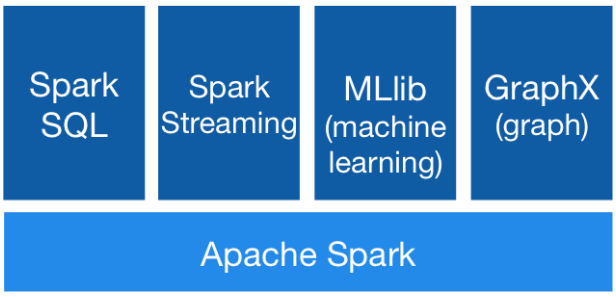

Figura 4. Combinación de módulos de Apache Spark

2.3 Generación de comunidades

Para generar las comunidades de usuarios hemos escogido el algoritmo *Label Propagation Algorithm (LPA)*, el cual está basado, como su propio nombre indica, en la propagación de etiquetas a lo largo de la red.

A continuación describimos los pasos que sigue el algoritmo para generar las comunidades:

1. Etiquetar cada nodo con un identificador único.
2. Reetiquetar cada nodo con la etiqueta que más veces aparezca entre sus vecinos.
3. Repetir el paso 2 tantas veces hasta conseguir las comunidades (5 iteraciones es lo óptimo, pero cada problema puede necesitar un numero de iteraciones distinto)

Nota: En caso de que dos etiquetas aparezcan el mismo número de veces entre los vecinos, se escogerá aleatoriamente.

2.4 Procesamiento de Lenguaje Natural (NLP)

El Procesamiento de Lenguaje Natural (NLP) es una rama de la inteligencia artificial, que ofrece ayuda a los ordenadores para comprender y utilizar el lenguaje humano. Su objetivo es acercar el lenguaje de las computadoras con el lenguaje humano.

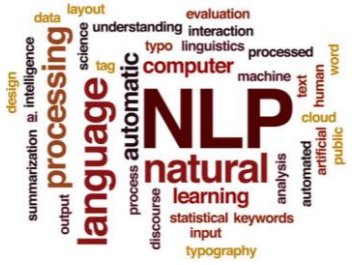

Figura 5. NLP

Utiliza diversas técnicas para comprender el lenguaje humano, desde métodos estadísticos y el aprendizaje basado en maquina hasta enfoques basados en reglas y algorítmicos. Se necesita variedad de métodos, debido a que los datos de texto y voz que proporcionan los humanos, difieren mucho unos de otros, a pesar de tener un denominador común, que es el idioma y cada uno de ellos tiene un conjunto único de reglas gramaticales y de sintaxis, términos y palabras coloquiales.

Las tareas de NLP engloban la simbolización y el análisis sintáctico, lematización/derivación, etiquetado de la parte del habla, detección del lenguaje e identificación de relaciones semánticas [3].

De todas las herramientas de NLP, se ha escogido la librería de código abierto Spacy, debido a su integración con Spark y Python. A continuación vemos algunas de sus características:

- **Tokenization:** segmentación de texto en palabras, signos de puntuación, etc…
- **Etiquetado del texto:** asignar una clase a las palabras, nombre, verbo, etc…
- **Análisis de dependencia:** asignar etiquetas de dependencia sintácticas, describir las relaciones entre tokens individuales.
- **Lemmatization:** obtener la forma básica de la palabra. Por ejemplo, para “casamos” su lemma es “casar”. Lo que nos permite identificar distintas formas verbales como un único verbo.
- **Detección de límite de oración (SBD):** encontrar y segmentar oraciones individuales.
- **Reconocimiento de entidad con nombre (NER):** etiquetar los nombres propios. Por ejemplo, “Juan”.
- **Semejanza:** comparar elementos del texto e identificar similitudes.
- **Clasificación de texto:** asignar etiquetas a un documento o parte de él.
- **Emparejamiento basado en reglas:** encontrar secuencias de tokens basadas en el texto y anotaciones lingüísticas.
- **Entrenamiento:** actualización y mejora de las predicciones de los modelos estadístico.
- **Serialización:** guardar objetos en archivos o cadenas de bytes.

Figura 6. Logo Spacy

Algunas de las anteriores características requieren de modelos estadísticos, que apoyan a Spacy para descubrir anotaciones lingüísticas. Spacy dispone de modelos para los idiomas: inglés, alemán, francés, español, portugués, italiano, holandés y griego. También incluye un modelo multi-lenguaje por si no tenemos claro el idioma del texto a analizar. Dichos modelos suelen incluir los siguientes componentes:

- **Pesos binarios:** para el etiquetador del texto, el analizador de dependencias y el reconocedor de nombres propios para predecir esas anotaciones en contexto.

- **Entidades léxicas en el vocabulario:** las palaras y sus atributos independientes del contexto.
- **Vectores de palabras:** representaciones de significado multidimensionales de palabras que le permiten determinar las similitudes entre ellas.
- **Opciones de configuración:** como pueden ser el idioma y la configuración de la canalización del procesamiento [4].

2.5 Sistema de búsqueda de texto completo

Para guardar los datos resultantes del sistema, se necesita una base de datos en donde lo primordial se centre en la lectura, ya que únicamente vamos a introducir los datos, y a partir de dicho momento lo único que se ejecutará contra la base de datos, serán consultas. Después de analizar varias bases de datos, se ha escogido Elasticsearch debido a su rapidez en la búsqueda, flexibilidad y su integración con el sistema de visualización escogido, explicado en la siguiente sección.

Figura 7. Logo Elasticsearch

Elasticsearch es el motor de búsqueda y análisis distribuido en el core de Elastic Stack. Provee búsquedas en tiempo real y es capaz de manejar cualquier tipo de dato, ya sea estructurado o no estructurado. Elasticsearch es capaz de almacenar e indexar los datos de tal manera, que permite las búsquedas rápidas. Y a medida que su volumen de datos crece, elasticsearch está preparado para crecer junto a ellos, gracias a la posibilidad de formar un cluster [5].

2.6 Sistema de visualización de datos

Por último, el sistema precisa de una herramienta para poder visualizar los datos obtenidos del análisis. Por ello, después de investigar varias herramientas, se ha escogido Kibana debido a su interfaz amigable, su gran variedad de gráficos y la integración con la herramienta de búsqueda de texto escogida, Elasticsearch.

Figura 8. Logo Kibana

Kibana permite visualizar los datos guardados en Elasticsearch y navegar por Elastic Stack, por lo que se puede realizar cualquier tarea, desde observar la carga de consultas hasta como fluyen los datos entre las aplicaciones. A continuación vemos las distintas categorías de cómo podemos visualizar los datos [6]:

- **Basics:** las gráficas más clásicas, como son los histogramas o gráficas lineales.
- **Location analysis:** con Elastic Maps podemos representar datos de localizaciones en un mapa.
- **Time series:** nos permite representar series temporales con datos en formato fecha.
- **Machine learning:** detecta las anomalías en los datos y busca las propiedades que influyen en ellas con funciones de aprendizaje automático sin supervisión.
- **Graphs and networks:** nos permite visualizar las relaciones de nuestros datos en forma de grafo.

3 Diseño

En esta sección se describe el diseño completo de la arquitectura del sistema de creación de comunidades y caracterizarlas según los tweets publicados por los usuarios.

3.1 Arquitectura del sistema

En la Figura 9, vemos la arquitectura del sistema de una forma global.

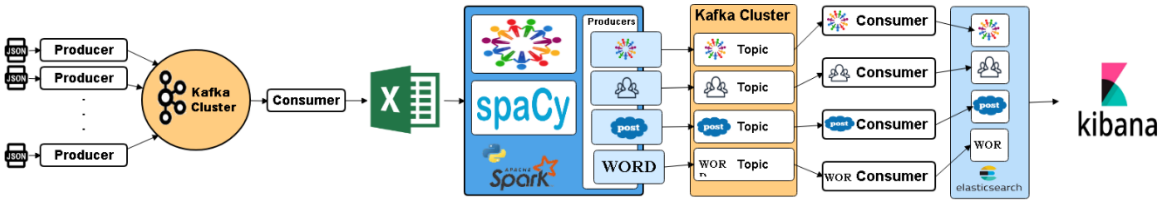

Figura 9. Arquitectura del sistema

En la arquitectura anterior podemos distinguir 4 grandes bloques, los cuales podemos ver en la Figura 10, cada uno con una o varias funciones.

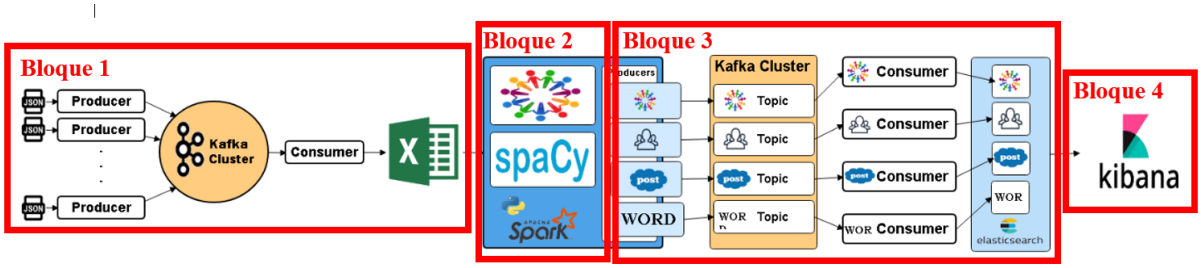

Figura 10. Arquitectura del sistema dividida en bloques

3.1.1 Bloque 1 - Ingesta de datos

Este bloque se encarga de la ingesta de datos para unificarlos en un único fichero CSV, que nos permite tener un fichero que se puede ampliar según se vayan recogiendo datos de otras fuentes. Para unificar los datos, nos hemos ayudado de un clúster de Apache Kafka, que nos permite un desacoplamiento de la ingesta gracias a sus colas, y nos asegura que el sistema no se bloquea en caso de que alguno de los productores falle. En el proceso también se realiza un filtrado para guardar únicamente los datos útiles para el análisis.

3.1.2 Bloque 2 - Procesamiento de datos

Este bloque podemos definirlo como el core del sistema y agrupa varias funcionalidades: generación de la red, detección de comunidades y procesamiento del lenguaje natural de cada tweet.

Todo el bloque está basado en Apache Spark para proporcionar una computación en distribuida. En primer lugar se guardan los datos, obtenidos del fichero CSV, en formato Parquet, lo que nos aportará una mayor velocidad de procesamiento. Para la detección en comunidades nos hemos apoyado en el algoritmo Label Propagation Algorithm (LPA), método sencillo y simple, lo que nos ayuda a generar comunidades para grandes cantidades de nodos. Para obtener el grafo de una manera gráfica, se ha utilizado Graph-Tool, la herramienta más rápida en python para pintar grafos. Por último, para procesar el texto de cada tweet se ha escogido Spacy, una de las herramientas más rápidas en el procesamiento de texto.

3.1.3 Bloque 3 - Persistencia de datos

Este bloque se encarga de guardar los datos resultantes del Bloque 2 e insertándolos en el sistema de búsqueda Elasticsearch, que nos permite realizar búsquedas de una forma rápida, además de alojar los datos de una manera distribuida. Como paso intermedio entre el resultado del Bloque 2 y el sistema de búsqueda, se implementa un clúster Kafka para que el proceso de ingesta se elabore de una manera distribuida y desacoplada, para asegurarnos de que la caída de uno de los consumidores no bloquee la ingesta.

3.1.4 Bloque 4 - Visualización de datos

Este bloque se encarga de representar los datos de una manera amigable y útil para el usuario final con la ayuda de gráficos representados con el complemento de visualización Kibana, herramienta que dispone de una interfaz y multitud de gráficos predefinidos, donde únicamente nosotros debemos de indicar que datos mostrar con que gráfico. Dichos datos se consultan sobre el sistema de búsqueda Elasticsearch.

4 Desarrollo

En esta sección se describe la implementación de cada uno de los bloques mencionados en el anterior capítulo.

4.1 Bloque 1 - Ingesta de datos

Este bloque está dividido en dos módulos básicos y escalables, *dataConsumer* y *dataProducer*, y un servidor de cola de mensajes implementado con la herramienta Apache Kafka, denominado Kafka Cluster.

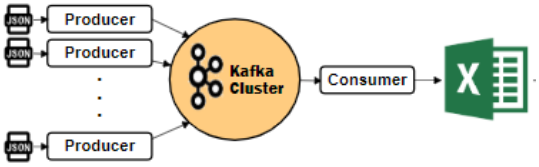

Figura 11. Arquitectura Bloque 1

4.1.1 Entrada de datos

A continuación, se describe el formato de entrada de cada registro y sus atributos más útiles, dentro de las fuentes de datos en formato JSON.

- **id:** identificador del tweet.
- **actor:** usuario que publica el tweet.
  - **objectType:** tipo de usuario.
  - **id:** identificador del usuario.
  - **summary:** descripción del usuario sobre sí mismo.
  - **friendsCount:** número de usuarios seguidos.
  - **followersCount:** número de seguidores.
  - **location:** lugar donde vive el usuario
- **postedTime:** fecha de publicación.
- **body:** mensaje publicado.
- **object:** si el mensaje es un retweet en este atributo esta toda la información del mensaje retuiteado con los atributos mencionados en esta sección.
- **twitter_entities:** objetos destacados mencionados en el tweet.
  - **user_mentions:** usuarios mencionados en el tweet.
- **twitter_lang:** idioma del tweet.
- **retweetCount:** número de veces que el tweet ha sido retuiteado.

4.1.2 DataProducer

El módulo DataProducer se encarga de leer los tweets de un fichero en formato JSON e introducirlos en la cola de mensajes del servidor desplegado para ello, desarrollado con la clase KafkaProducer.

Por cada fichero en formato JSON, se debe de ejecutar una instancia del módulo apuntando al mismo servidor de cola de mensajes, consiguiendo así un módulo distribuido y escalable.

4.1.3 DataConsumer

El módulo DataConsumer se encarga de leer tweets de la cola de mensajes, con la clase KafkaConsumer. Una vez leídos, son procesados para guardar únicamente la información que vamos a utilizar. Dicha información se guarda en un fichero con la siguiente estructura.

- **tweets.csv:** en este fichero se guardan los tweets con la conformación que hemos escogido de utilidad para realizar el análisis. A continuación, vemos su estructura.
  - **id:** identificador del tweet.
  - **actor:** usuario que publica el tweet.
  - **body:** mensaje publicado.
  - **retweetedUser:** usuario a quien se ha retuiteado.
  - **mentionUser:** primer usuario mencionado en el tweet.
  - **postedTime:** fecha de publicación.

Además de obtener únicamente la información útil para el análisis, también se descartan los tweets que no son retweets, ni han sido retuiteados ni han mencionado a usuarios, debido a que son tweets solitarios y a la hora de construir comunidades, dichos nodos solitarios no nos proporcionan ninguna información, es decir, guardar dicha información es inútil.

4.2 Bloque 2 - Procesamiento de datos

Como hemos mencionado en el apartado anterior, podemos decir que este bloque es el core del sistema. A pesar de ello, solo cuenta con un único modulo que se encarga de realizar varias funcionalidades.

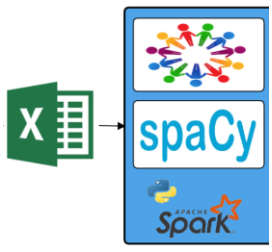

Figura 12. Arquitectura Bloque 2

4.2.1 DataProcessor

El módulo está desarrollado completamente con Apache Spark. En las siguientes secciones se describe cada funcionalidad del módulo por orden de ejecución.

Generación de la red

Una vez cargado el fichero `tweets.csv` en Spark, guardamos los datos en un fichero con formato Parquet, con el cual se realizará todo el procesamiento. Después debemos de generar el grafo, sobre el cual identificaremos las comunidades posteriormente.

Para generar dicho grafo, hemos cogido la lista de usuarios no repetidos, para que cada uno de ellos sea un nodo en el grafo. Para generar los enlaces entre los nodos hemos utilizado los retweets y las menciones de cada tweet. A continuación, lo vemos representado de forma gráfica, en la Tabla 1 donde podemos ver la información que nos llega del tweet. En la Figura 13, vemos cómo se generan los enlaces a partir del tweet.

id	actor	retweetedUser	mentionUser

Tabla 1. Información tweet para grafo

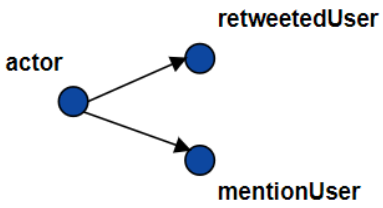

Figura 13. Ejemplo grafo tweet

En caso de que un usuario mencione o retuite a otro usuario varias veces, se generará un enlace por cada una de las menciones o retweet indicando que la conexión entre dichos usuarios es más fuerte que con otros.

Como hemos mencionado antes, cada usuario es representado por un único nodo, si el cual ha publicado tweets que han sido retuiteados por varios usuarios, dicho nodo tendrá varios enlaces, como muestra la Figura 14.

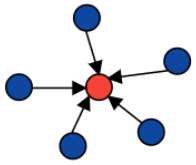

Figura 14. Ejemplo grafo de usuario

Por último, vemos las dos tablas que se introducen en el paquete `GraphFrame` para generar la red, también denominado grafo.

Users

- **id:** identificador de usuario

Edges

- **src:** usuario origen del enlace
- **dst:** usuario destino del enlace

Detección de comunidades

Una vez tenemos el grafo ya formado, identificamos las comunidades mediante el algoritmo Label Propagation algorithm (LPA), explicado en la sección 2.3, con un máximo de 5 iteraciones.

Una vez hemos generado las comunidades, se descartan las de menos de 20 usuarios. Después se etiqueta cada uno de los tweets con la misma comunidad a la que pertenece el usuario que ha generado el tweet.

Por último, obtenemos las cinco comunidades con más usuarios y los usuarios relacionados con estas comunidades en primer nivel, y los pintamos con ayuda de la herramienta `graph_tool`, en la sección 5.3.1 podemos ver un ejemplo.

Procesamiento de lenguaje natural (NLP)

El siguiente paso consiste en procesar el texto de cada tweet para comprender su contenido. Para ello utilizamos la biblioteca Spacy, basada en Procesamiento de lenguaje natural (NLP), explicado en la sección 2.4, con el modelo estadístico para el idioma español "es_core_news_sm".

Una vez procesado el texto, guardamos el lemma de cada palabra que nos aporta información, es decir, descartamos las palabras vacías (stop words), el nombre de los usuarios, por motivos de privacidad, y los signos de puntuación.

4.3 Bloque 3 - Persistencia de datos

Este bloque se encarga de obtener los datos resultantes de Spark e ingestarlos en la base de datos Elasticsearch. El bloque dispone de cinco módulos: dataProcessor, que es compartido con el bloque 2 y encargado de introducir los resultados en Kafka, y un módulo para cada una de las tablas que se encarga de sacar los datos de Kafka e insertarlos en la base de datos.

Para transferir los datos, se dispone de un clúster de Kafka con 4 topic, uno para cada tabla.

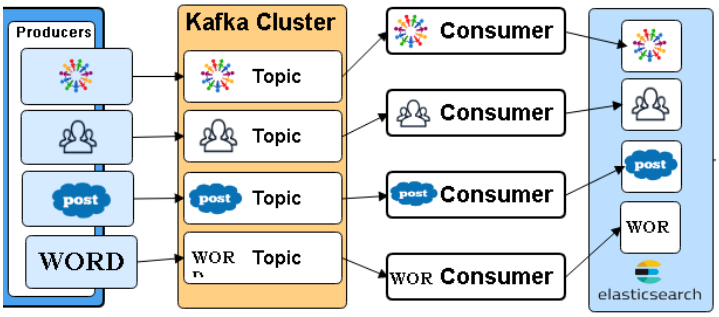

Figura 15. Arquitectura Bloque 3

4.3.1 DataProcessor

Una vez el módulo DataProcessor ha procesado todos los datos, tarea correspondiente al bloque 2, procede a enviar, con la clase `KafkaProducer`, las siguientes estructuras de datos a sus respectivos topic, alojados en el clúster de Kafka dedicado a ello.

Comunities

- **id:** identificador de la comunidad.

Users

- **id:** identificador del usuario.
- **comunity:** identificador de la comunidad a la que pertenece el usuario.

Tweets

- **id:** identificador del tweet.
- **actor:** identificador del usuario que creó el tweet.
- **comunity:** identificador de la comunidad a la que pertenece el tweet.
- **postedTime:** fecha de publicación del tweet.

Words

- **comunity:** identificador de la comunidad a la que pertenece la palabra.
- **word:** un string sacado del cuerpo de un tweet.
- **count:** número de veces que se repite la palabra en la comunidad.

4.3.2 Ingesta de datos

Cada uno de estos módulos tiene un topic asociado. Conforme se vayan añadiendo datos a los topic correspondientes, el objeto `KafkaComsumer` de cada módulo, recogerá

los registros uno por uno y los irá insertando en la tabla correspondiente de Elasticsearch. A continuación podemos ver una tabla con la relación, nombre-topic-tabla.

Nombre	Topic	Tabla
Comunities	comunities_elastic	comunities
Users	users_elastic	users
Tweets	tweets_elastic	tweets
Words	words_elastic	words

Tabla 2. Relación nombre-topic-tabla

4.4 Bloque 4 - Visualización de datos

El bloque 4 se encarga de presentar los datos de una manera gráfica y amigable para el usuario final, permitiendo realizar estadísticas sobre los datos. Para ello usamos el complemento de visualización Kibana, el cual obtiene los datos desde las tablas creadas en el bloque 3 en Elasticsearch.

Lo primero que se realiza es crear un índice por cada tabla: **users, communities, tweets y words.**

Una vez tenemos los índices creados, generamos cada gráfico individualmente con el apartado *Visualize*, los cuales obtienen los datos a través de los índices. Dichos gráficos los distribuimos en dos *Dashboard*, elemento donde se recogen varias visualizaciones, para presentarle al usuario final una mejor visibilidad de las distintas gráficas relacionadas. A continuación se describen los distintos tableros (*Dashboard*) y sus respectivos gráficos (*Visualize*) que hemos creado.

General Data

- **#Users:** número de usuarios analizados.
- **#Tweets:** número de tweets analizados.
- **#Comunities:** número de comunidades obtenidas.
- **Users by Comunity:** histograma de número de usuarios en cada comunidad.
- **Tweets by Comunity:** histograma de número de tweets en cada comunidad.

En la Figura 16 vemos el tablero con una serie de datos de ejemplo.

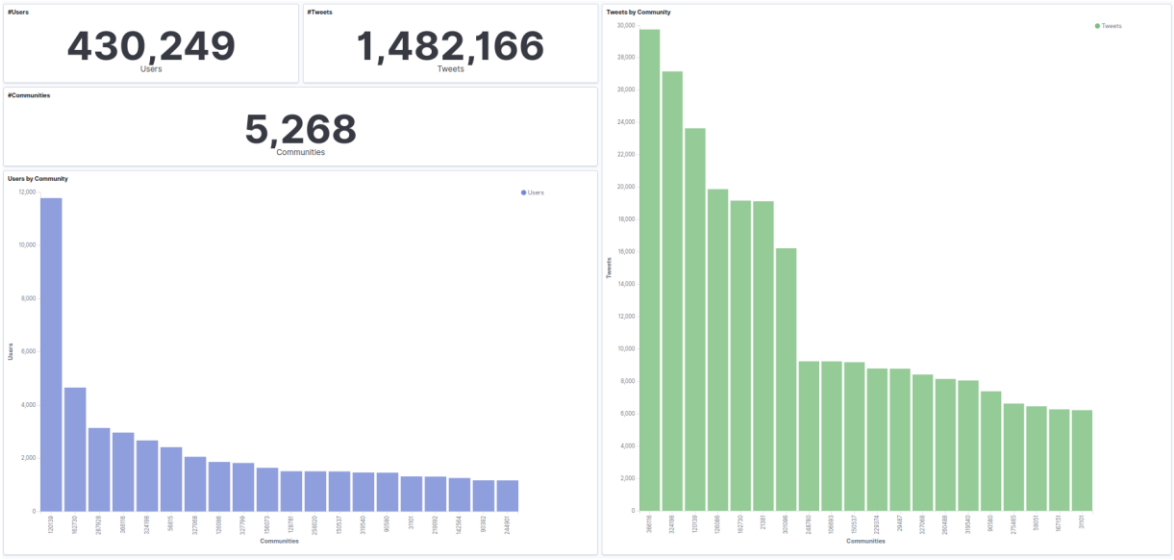

Figura 16. Tablero General Data

Comunity Data

- **Control Community:** formulario donde el usuario elige por la comunidad o comunidades que desea filtrar. El resultado se aplicará al resto de gráficas en este tablero.
- **#Users:** número de usuarios.
- **#Tweets:** número de tweets analizados.
- **Tweets By User:** histograma de número de tweets por cada usuario.
- **Words Cloud:** nube de palabras, donde las palabras más frecuentes aparecen con una fuente más grande y representadas en el centro de la nube.
- **Time Tweets:** herramienta para visualizar la serie temporal de la publicación de los tweets durante el periodo que introducimos en el filtro.

En la Figura 17 vemos el tablero con una serie de datos de ejemplo.

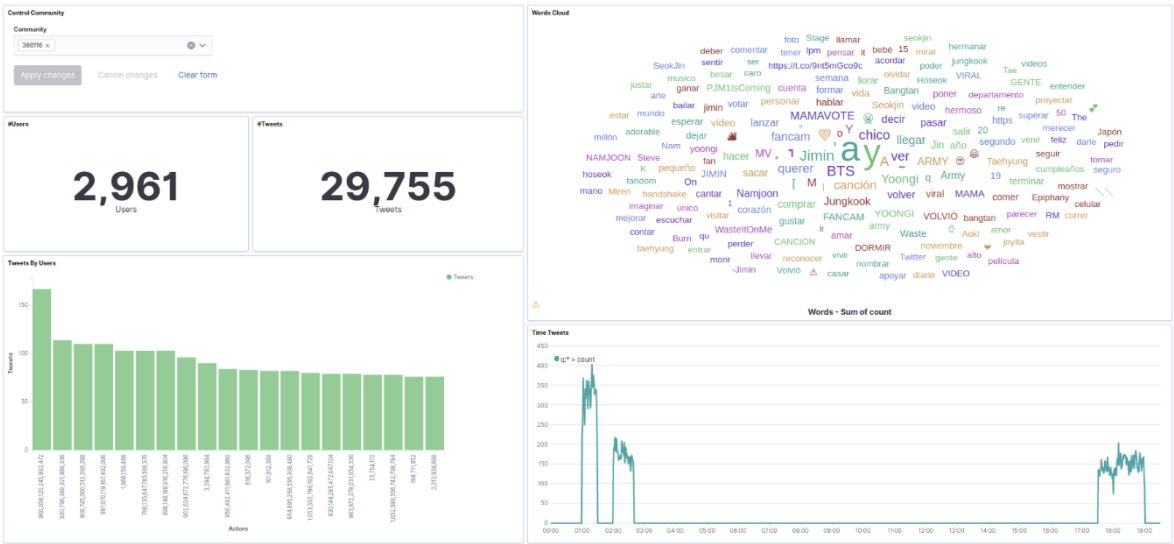

Figura 17. Tablero Community Data

5 Integración, Pruebas y Resultados

En esta sección se describen los recursos en los que se ha apoyado el desarrollo del sistema, las pruebas a las que se ha sometido y los resultados obtenidos de dichas pruebas.

5.1 Recursos Hardware y Software

Para la realización y ejecución del proyecto se ha utilizado el siguiente PC:

- **Modelo:** Inspiron 15 7000 Series -7570 (2017)
- **CPU:** 8th Generation Intel(R)Core(TM) i7-8550U Processor (8MB Cache, up to 4.0 GHz)
- **RAM:** 16GB, DDR4, 2400MHz
- **Disco:** 512GB PCIe NVMe M.2 Solid State Drive

El TFM se ha realizado en el SO Ubuntu 18.04, instalado en una de las particiones del PC mencionado.

Por último, para el desarrollo del código me he apoyado en notebooks de Jupyter [7] para ir probando los resultados de una manera dinámica.

5.2 Datos de Entrada

A continuación se describen los distintos datos de entrada que se utilizan para realizar la prueba completa de nuestro sistema.

5.2.1 Fuente de datos

Para realizar las pruebas, se ha utilizado un conjunto de datos compuesto por 20 ficheros, de los 517 del conjunto total de la fuente, con un tamaño aproximado de 850 MB cada uno, lo que suma un total aproximado de 16,6 GB. Dichos datos representan los tweets publicados en España durante 3 periodos de un día concreto, desde la 01:00 a la 1:30, desde las 2:00 a las 2:45 y desde 17:30 hasta las 19:00 del día 17 de noviembre de 2018. Se han escogido tres periodos para simplificar los datos y tener variedad en las horas del día, esta simplificación se debe a que no se dispone de los recursos de procesamiento necesarios durante la realización del TFM. El conjunto de datos tiene aproximadamente 3.900.000 y 1.450.000 usuarios.

5.2.2 Datos a procesar

Una vez filtrados los datos y agrupados en un mismo fichero CSV, nos queda un conjunto de datos de 577 MB que son guardados en formato Parquet, obteniendo un fichero de 314 MB para procesar.

5.2.3 Diccionario de etiquetas

Para realizar el procesamiento del lenguaje natural, se ha utilizado el modelo de Spacy, en idioma español, **es_core_news_sm,** ya que se han recogido únicamente tweets publicados en España**.**

5.3 Resultados

A continuación se describen los resultados obtenidos de procesar los datos mencionados en la sección 5.2.

5.3.1 Análisis de comunidades

En este apartado vemos el grafo, en la Figura 18, de las cinco comunidades con más usuarios. En la Tabla 3 vemos los identificadores de cada una de las cinco y el color con el que se representa en el grafo, en orden de cantidad de usuarios que engloban. En la sección 5.3.2 veremos aproximadamente la cantidad de usuarios de cada una de las comunidades.

Comunidad	Usuarios	Tweets	Color
120139	11.774	23.642	Azul oscuro
162730	4.653	19,171	Verde
287928	3,136	4,849	Azul claro
366116	2,961	29,755	Rojo
324198	2,667	27,159	Violeta

Tabla 3. Cinco comunidades con más usuarios

- **120139:** el tema comentado son videos de diferentes videojuegos.
- **162730:** los usuarios hablan sobre diferentes sentimientos dentro de las relaciones de pareja.
- **287928:** en esta comunidad se alaba un video divertido subido a Twitter de un grupo de chicos bailando en el metro.
- **366116:** los tuiteros conversan sobre el grupo de música surcoreano BTS.
- **324198:** los individuos escriben sobre el G20 celebrado entre el 30 de noviembre y el 1 de diciembre de 2018 en Buenos Aires.

Los nodos en color gris representan los usuarios con los que están relacionados las cinco comunidades y no pertenecen a ninguna de las cuales.

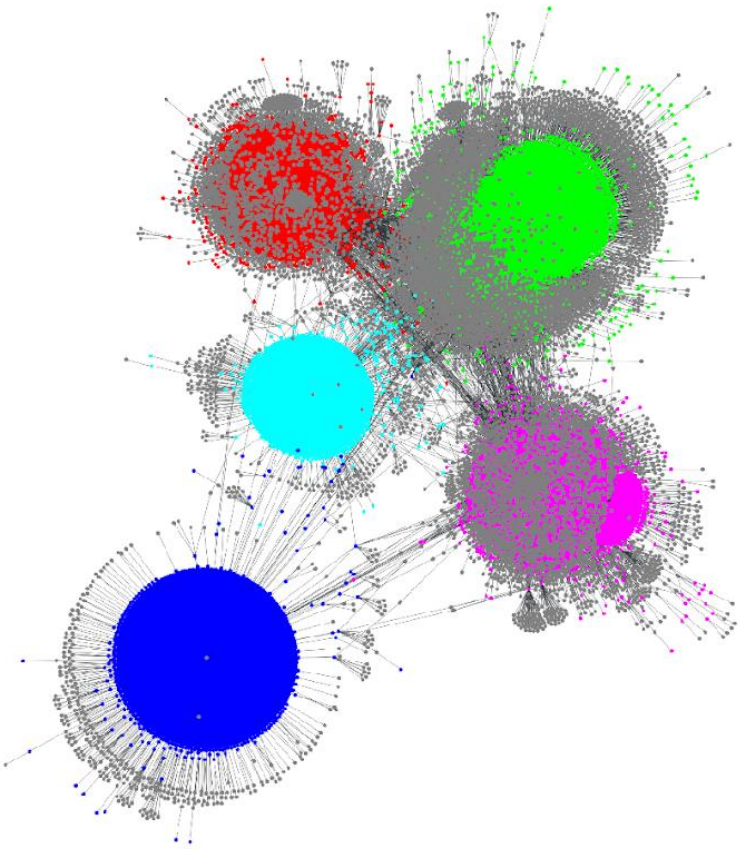

Figura 18. Grafo de las cinco comunidades con más usuarios

En el anterior grafo, podemos observar varios tipos de comunidades. En las representadas en azul oscuro y claro, vemos que sus usuarios están muy relacionados entre ellos y tienen pocas relaciones con otras. En la representada en verde, observamos que la mayoría de sus usuarios están cohesionados entre sí, pero hay una gran parte de ellos que tienen relación con otras. Por último, analizando las representadas en rojo y violeta, vemos que la mayoría de sus usuarios están relacionados con otras, y no están tan cohesionados entre ellos como el resto, es decir, muchos de sus usuarios podrían encajar en comunidades vecinas.

Si observamos el grafo, conociendo la temática de cada una de las comunidades, podemos ver que las comunidades representadas con color rojo, violeta y verde, tienen muchas relaciones entre ellas, lo que nos indica que es común que usuarios de una de estas comunidades, también le guste los temas de las otras dos comunidades. Por otro lado, la comunidad representada en azul, tiene pocas relaciones con otras comunidades, lo que nos indica que raramente la gente que le gusta los videojuegos, le gusta hablar sobre los sentimientos de pareja, música surcoreana o de política.

5.3.2 Análisis detallados

En esta sección se analizan los resultados obtenidos de nuestro sistema, representados en gráficas, dentro de los tableros de Kibana, explicados en la sección 4.4.

General Data

En la Figura 19 observamos el tablero con los datos generales.

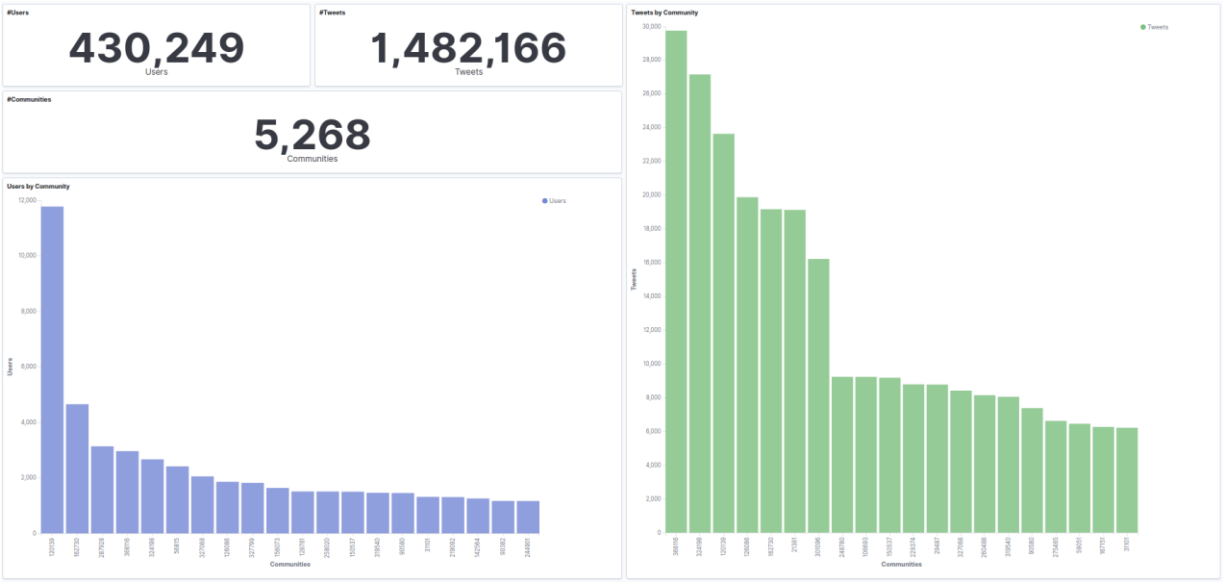

Figura 19. Resultados General Data

Después de descartar los usuarios sin relaciones y los que pertenecen a una comunidad con menos de veinte individuos, podemos observar que el resultado final nos arroja un total de 430.249 usuarios, 1.482.166 tweets y 5.268 comunidades.

A continuación, hacemos zoom sobre las gráficas representadas en el tablero, *Users By Community* en la Figura 20 y *Tweets By Community* en la Figura 21.

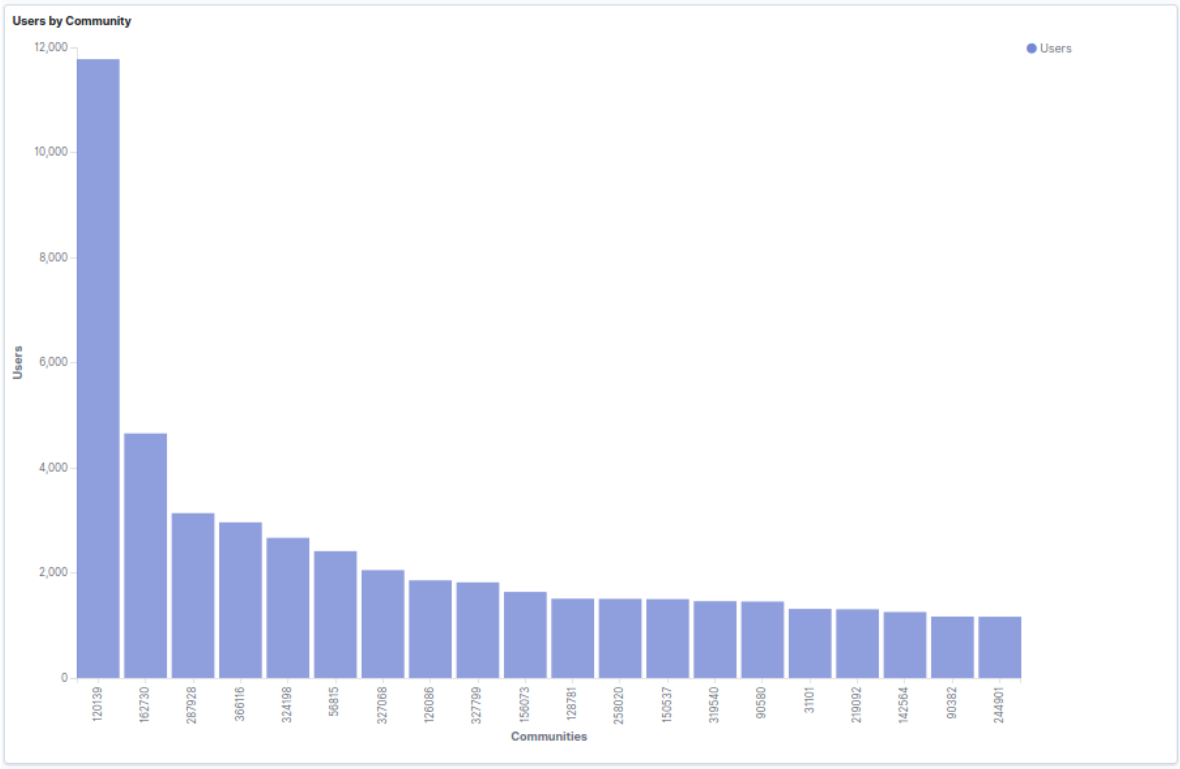

Figura 20. Users By Community

En la gráfica *Users By Community* vemos las veinte comunidades con más usuarios. Observamos que la primera comunidad destaca sobre el resto, teniendo casi 12.000 usuarios, más abajo analizaremos dicha comunidad en profundidad. A partir de la tercera comunidad vemos que la cantidad de usuarios es similar, cada una de ellas posee alrededor de 2.000 usuarios.

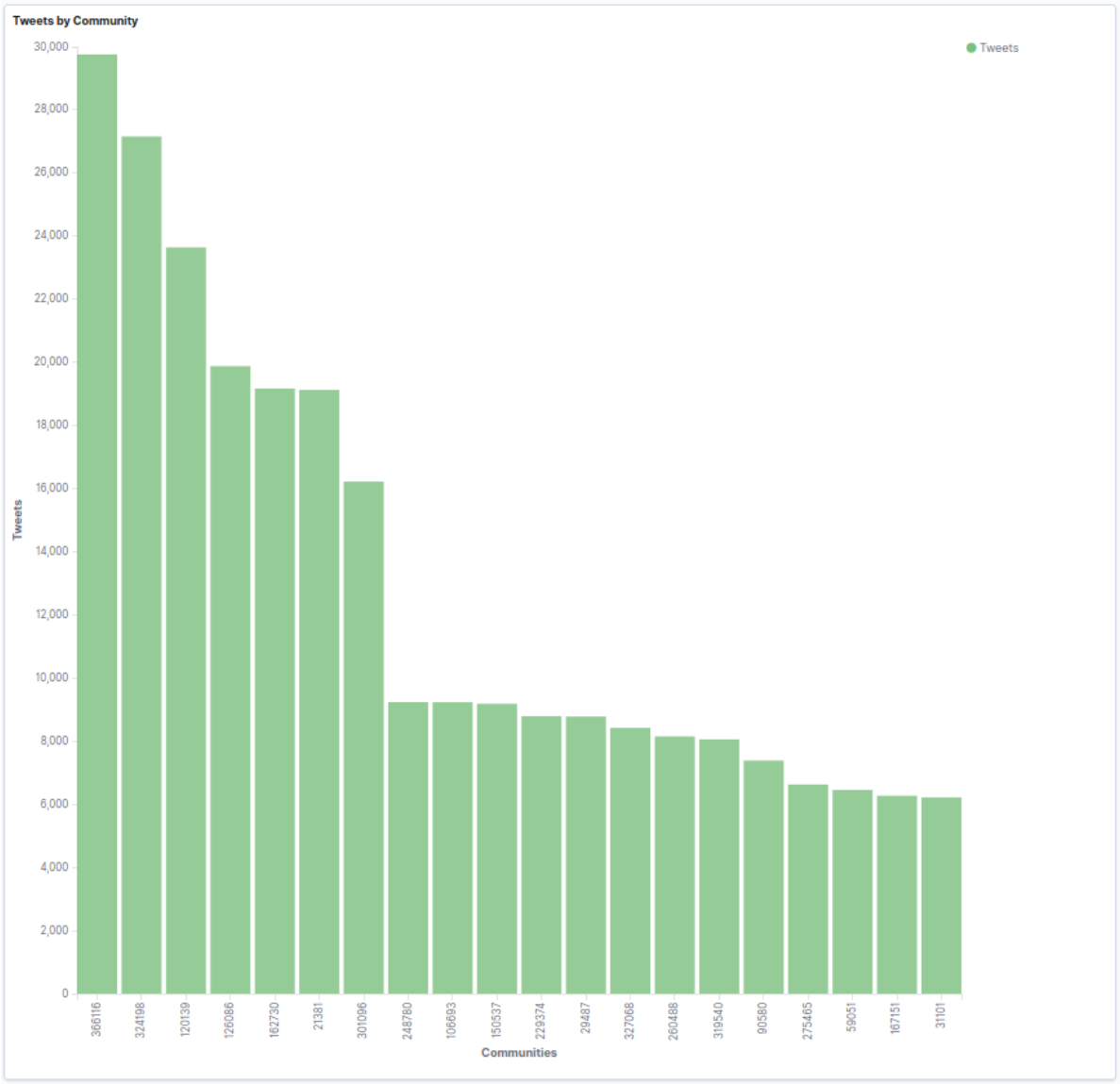

Figura 21. Tweets By Community

En la gráfica *Tweets By Community* vemos las veinte comunidades con más tweets. Podemos observar que las siete primeras comunidades tienen considerablemente más tweets publicados que el resto, yendo de casi 30.000 tweets a aproximadamente 16.000 tweets. A partir de la octava comunidad inclusive, la cantidad de tweets es similar, entre 8.000 y 6.000 tweets. La primera comunidad la analizaremos más en profundidad en esta misma sección.

Community Data

En este apartado analizaremos dos comunidades, la comunidad con más usuarios y la comunidad con más tweets.

Comunidad con más usuarios

La comunidad con más usuarios tiene el identificador 120139, por lo que lo ponemos en el filtro y analizamos los resultados, los cuales podemos ver de forma general en la Figura 22.

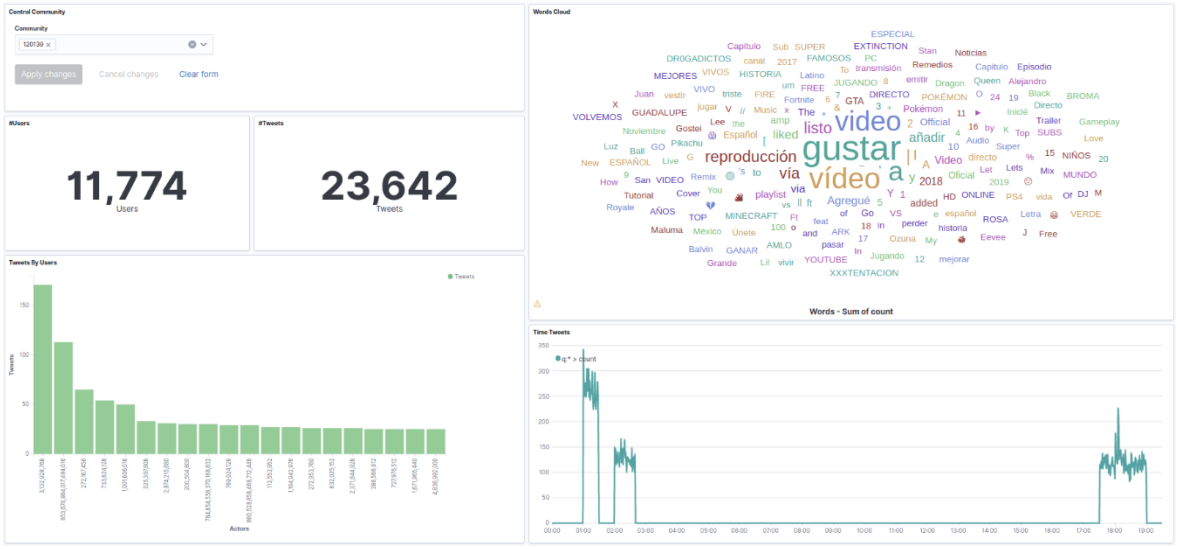

Figura 22. Tablero de la comunidad con más usuarios

En un primer lugar vemos que esta comunidad tiene 11.774 usuarios y 23.642 tweets, lo que implica una media de aproximadamente dos tweets por usuario. Esto nos indica que la mayoría de los usuarios han actuado esporádicamente y no de forma constante con las publicaciones durante el día del conjunto de datos. A continuación, vamos a hacer zoom sobre el resto de las gráficas.

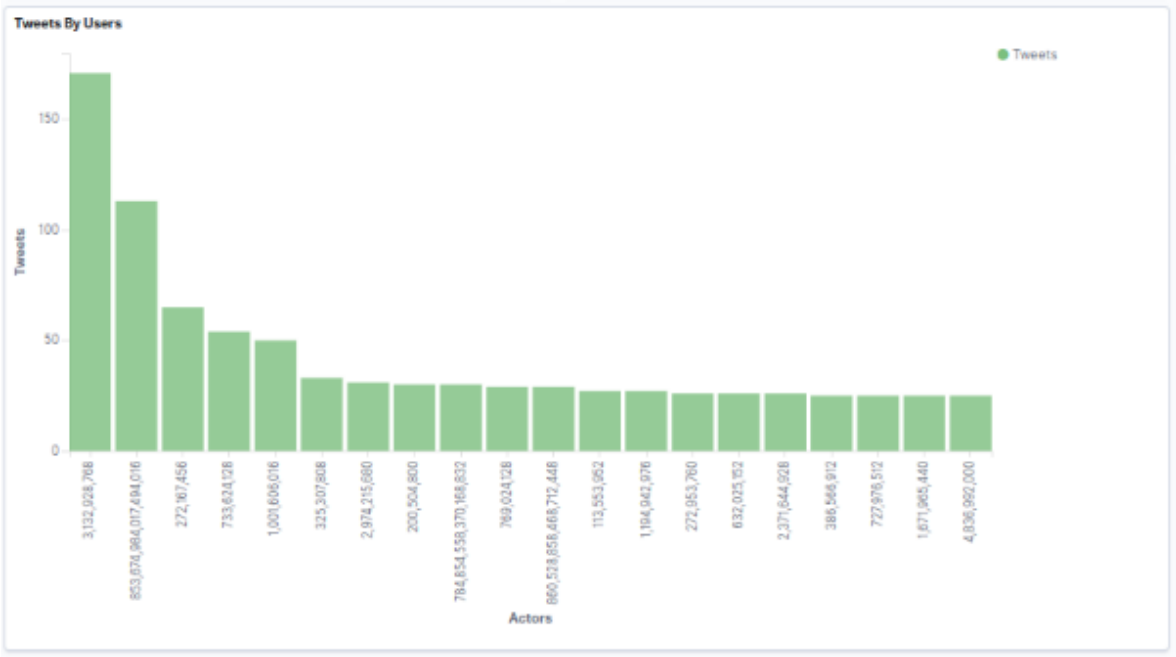

Figura 23. Tweets By User, comunidad con más usuarios

En la gráfica Tweets By User, podemos ver a los veinte usuarios que más tweets han publicado en esta comunidad. Vemos que los cinco primeros usuarios son muy activos con respecto a sus publicaciones, llegando incluso los dos primeros a más de cien tweets en las dos horas y media analizadas. Observando el resto de los usuarios, vemos que se estabiliza sobre los veinticinco tweets la media de publicación.

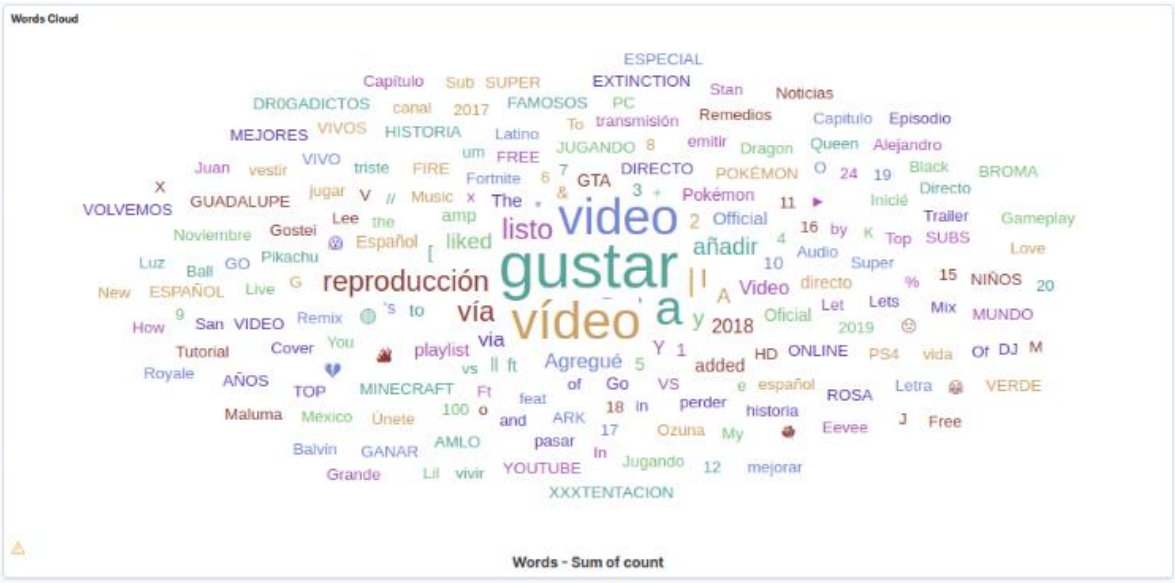

Figura 24. Palabras más mencionadas en la comunidad con más usuarios

En la Figura 24 podemos ver las palabras más mencionadas por los usuarios en la comunidad. Analizándolas podemos ver que destacan sobre el resto algunas como *gustar*, *video y reproducción*, lo que nos indica que la comunidad está hablando sobre videos que les gusta o les ha gustado. Si profundizamos más a fondo, vemos también palabras como *MINECRAFT, GO, GTA, ARK, POKÉMON y Fortnite*, lo que nos indica que los vídeos comentados, en su mayoría, tratan sobre videojuegos.

Figura 25. Gráfica de tiempo de la comunidad con más usuarios

En la Figura 25, vemos la gráfica de tiempo de publicación de los tweets, en intervalos de 1 minuto. Debemos de recordar que se analizado tres periodos concretos y los intervalos intermedios están a cero porque no tenemos datos sobre ellos. Observamos que la mayoría de ellos se han publicado en el primer intervalo, desde la 1:00 hasta la 1:30, más concretamente al principio, llegando a trescientos cincuenta por minuto. También podemos observar que por la tarde sobre las 18:05 ha sufrido un repunte de publicaciones llegando a superar los doscientos por minuto.

Comunidad con más tweets

La comunidad con más usuarios tiene el identificador 366116, por lo que lo ponemos en el filtro y analizamos los resultados, los cuales podemos ver de forma general en la Figura 26.

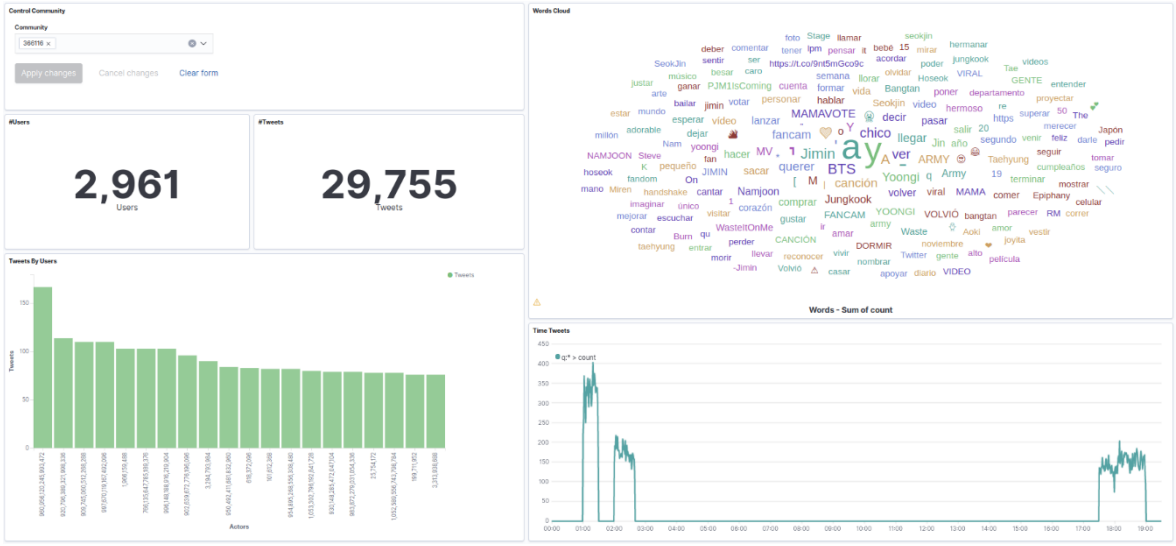

Figura 26. Tablero con la comunidad con más tweets

En un primer lugar vemos que esta comunidad tiene 2.961 usuarios y 29.755 tweets, lo que implica una media de aproximadamente diez tweets por usuario, esto nos indica que los usuarios han participado activamente o hay una serie de usuarios que son programas publicadores y hacen que crezca la media, para poder analizar más a fondo la comunidad, hacemos zoom en el resto de las gráficas.

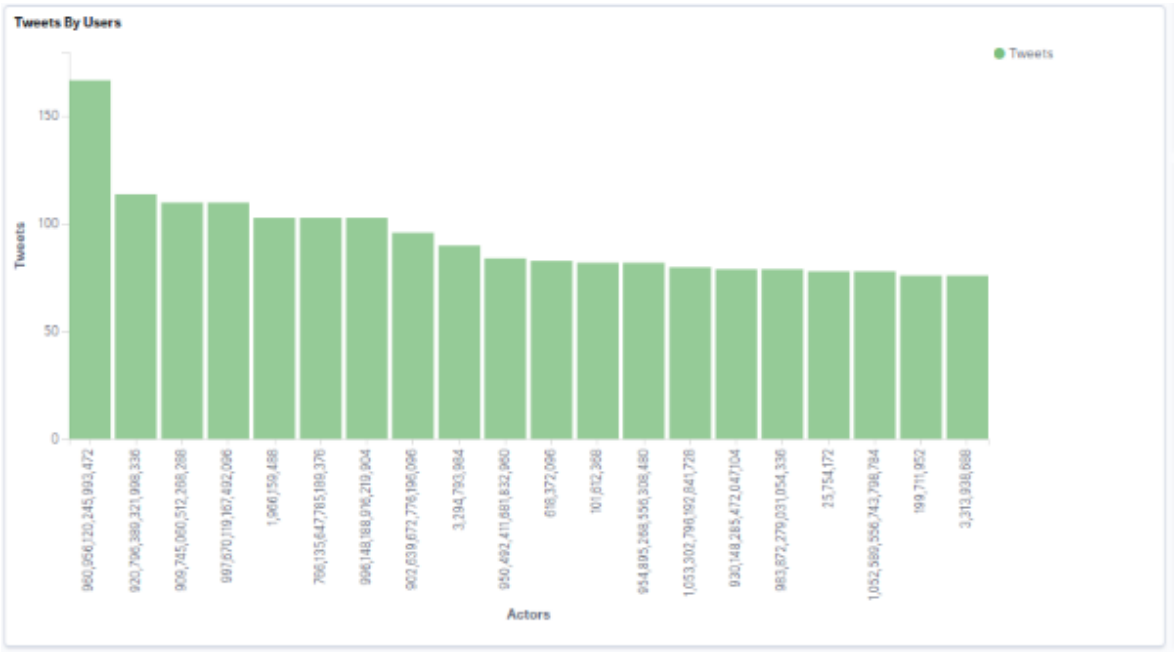

Figura 27. Tweets By User, comunidad con más tweets

En la Figura 27, podemos ver que en esta comunidad los usuarios están más equilibrados con respecto a la cantidad de publicaciones de tweets, a excepción del primero que destaca sobre el resto. También podemos observar, que en esta comunidad los veinte primeros usuarios son más activos, teniendo una media de ochenta tweets publicados en las dos horas y media analizadas.

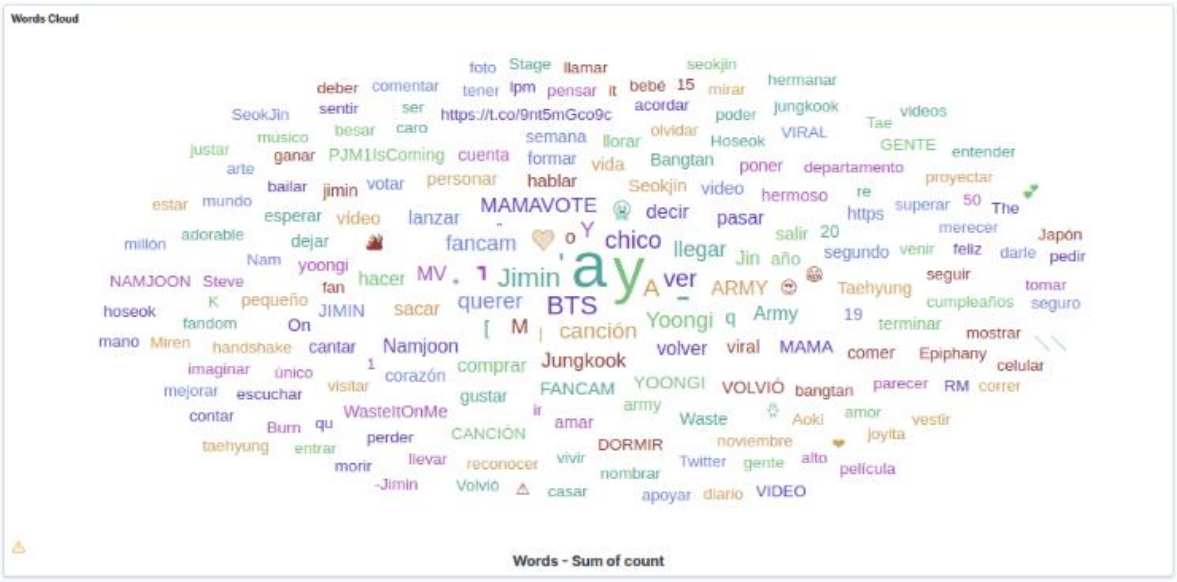

Figura 28. Palabras más mencionadas en la comunidad con más tweets

En la Figura 28 podemos ver las palabras más mencionadas por los usuarios en la comunidad. Analizándolas, observamos las palabras más destacadas como *BTS, Jimin, Namjoon, Jungkook, Yoongi y Seokjin*, lo que nos dice que están comentando sobre el grupo

de música surcoreana BTS y sus miembros. Y si observamos más a fondo podemos ver palabras como *querer, gustar, corazón* y emoticonos como el del corazón, esto nos indica que los comentarios hacia la banda son de aprecio.

Figura 29. Gráfica de tiempo de la comunidad con más tweets

En la Figura 29, vemos la gráfica de tiempo de publicación de los tweets, en intervalos de 1 minuto. Debemos de recordar que se analizado tres periodos concretos y los intervalos intermedios están a cero porque no tenemos datos sobre ellos. Observamos que la mayoría de los tweets se han publicado en el primer intervalo, desde la 1:00 hasta la 1:30, más concretamente al final, llegando a cuatrocientos por minuto.

Observando los tiempos de publicaciones de ambas comunidades podemos ver que durante el periodo de la 01:00 a las 01:30 se han publicado más tweets que en el resto, y esto se puede deber a que el 17 de noviembre de 2018 era sábado.

6 Conclusiones y trabajo futuro

En este capítulo se describen los argumentos y afirmaciones relativos a los resultados obtenidos. También se describe una serie de tareas para mejorar nuestro sistema.

6.1 Conclusiones

El objetivo de este TFM era crear un sistema que analiza un conjunto de tweets publicados, para agrupar a los usuarios en comunidades y obtener la información sobre la que se comenta en cada comunidad.

Para ello se ha dividido el trabajo en cuatro bloques (Ingesta de datos, Procesamiento de datos, Persistencia de datos y Visualización de datos) lo que nos ha ayudado a crear un sistema con módulos independientes y escalable.

Por otra parte, nos hemos apoyado en las herramientas Apache Kafka, Apache Spark, Spacy, Elasticsearch y Kibana para conseguir un sistema desacoplado y escalable, conforme crezca la fuente de datos a procesar.

Una vez analizados los resultados podemos concluir que:

- El sistema cumple con los requisitos iniciales propuestos, hemos podido generar comunidades adecuadas y obtener la información sobre la que se comenta en las de más actividad.
- Los datos de entrada son adecuados para obtener resultados concluyentes.
- La arquitectura diseñada es escalable, lo que nos permite añadir nuevos recursos computacionales al clúster que veamos con la carga al máximo.

Por último, indicar que el TFM me ha ayudado a profundizar más en los temas de Big Data y Data Science vistos en la asignatura de Computación a Gran Escala dentro del Máster.

6.2 Trabajo futuro

Sobre el sistema desarrollado se pueden realizar multitud de mejoras para que nuestro sistema tenga un mejor rendimiento o podamos obtener unos resultados más amplios.

El siguiente paso para realizar es probar el sistema con conjunto de datos en otro idioma o incluso con un conjunto de datos de varios lenguajes, y observar cómo se comporta.

Otra mejora que podemos realizar es automatizar la ampliación de computación de los distintos clústers que disponemos en el sistema, tarea únicamente posible en entornos virtuales. Por ejemplo, si un clúster Kafka está cerca del límite de capacidad, automáticamente añadir un nuevo nodo al clúster. También se puede subir nuestro sistema a alguna cloud pública y aprovechar sus desarrollos para estas ampliaciones automáticas, como puede ser Microsoft Azure y su servicio HDInsight.

Por último, también podemos ampliar la información que obtenemos de cada comunidad, por ejemplo, de donde son la mayoría de los usuarios o de donde son los usuarios quien ha publicado los tweets con mayor frecuencia. Otra mejora posible sería obtener la procedencia de los tweets de dicha comunidad.

● Bibliografía

- [1] Apache Kafka - https://kafka.apache.org/intro - Último acceso 05/08/2019
- [2] Qué es Apache Spark - Abraham Requena Mesa - https://openwebinars.net/blog/que-es-apache-spark/
- [3] Procesamiento del lenguaje natural - https://www.sas.com/es_es/insights/analytics/what-is-natural-language-processing-nlp.html
- [4] spaCy 101: Everything you need to know - https://spacy.io/usage/spacy-101/#features
- [5] Elasticsearch introduction - https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html
- [6] Your window into the Elastic Stack - https://www.elastic.co/es/products/kibana
- [7] https://jupyter.org/

● Anexos

○ *A Ejemplo de registro*

A continuación, vemos un ejemplo de la información que llega por cada tweet desde la fuente de datos y como está estructurada.


```
{
    "id": "tag:search.twitter.com,2005:1063582487501258752",
    "objectType": "activity",
    "actor": {
        "objectType": "person",
        "id": "id:twitter.com:136097751",
        "link": "http://www.twitter.com/ErikR10_",
        "displayName": "Erik Romero",
        "postedTime": "2010-04-23T01:16:48.000Z",
        "image":
"https://pbs.twimg.com/profile_images/1020382743140749317/2w2xJtJ5_normal.jp
g",
        "summary": "Futbolista ⚽ Instagram: ERIKR10_    Mérida,
Venezuela.",
        "links": [
            {
                "href": null,
                "rel": "me"
            }
        ],
        "friendsCount": 679,
        "followersCount": 973,
        "listedCount": 9,
        "statusesCount": 20098,
        "verified": false,
        "preferredUsername": "ErikR10_",
        "languages": [
            "es"
        ],
        "twitterTimeZone": null,
        "utcOffset": null,
        "favoritesCount": 391
    },
    "verb": "share",
    "postedTime": "2018-11-17T00:00:00.000Z",
    "generator": {
        "displayName": "Twitter for Android",
        "link": "http://twitter.com/download/android"
    },
    "provider": {
        "objectType": "service",
        "displayName": "Twitter",
        "link": "http://www.twitter.com"
    },
    "link": "http://twitter.com/ErikR10_/statuses/1063582487501258752",
```

```
    "body": "RT @Mbethaniaml: Jesús “La Pulga” Gómez, uno de los máximos
ídolos de @EstudiantesMEFC, acaba de renovar por dos años con el
académico⍰O□⍰⍰",
    "object": {
        "id": "tag:search.twitter.com,2005:1063582037490184192",
        "objectType": "activity",
        "actor": {
            "objectType": "person",
            "id": "id:twitter.com:86560741",
            "link": "http://www.twitter.com/Mbethaniaml",
            "displayName": "Bethania Martínez Leo",
            "postedTime": "2009-10-31T16:45:20.000Z",
            "image":
"https://pbs.twimg.com/profile_images/1012513410951639043/GAmo1VZy_normal.jp
g",
            "summary": "Periodista deportiva. Mi vida es el fútbol, ni más,
ni menos. Parte del departamento de prensa de @EstudiantesMEFC @FabricaAKD.
Reportera en @LaTeleTuya ⍰□",
            "links": [
                {
                    "href": "http://www.instagram.com/bethaniamartinezleo",
                    "rel": "me"
                }
            ],
            "friendsCount": 651,
            "followersCount": 4348,
            "listedCount": 30,
            "statusesCount": 33499,
            "verified": false,
            "preferredUsername": "Mbethaniaml",
            "languages": [
                "es"
            ],
            "twitterTimeZone": null,
            "utcOffset": null,
            "location": {
                "objectType": "place",
                "displayName": "Venezuela"
            },
            "favoritesCount": 1202
        },
        "verb": "post",
        "postedTime": "2018-11-16T23:58:12.000Z",
        "generator": {
            "displayName": "Twitter for iPhone",
            "link": "http://twitter.com/download/iphone"
        },
        "provider": {
            "objectType": "service",
            "displayName": "Twitter",
            "link": "http://www.twitter.com"
        },
        "link":
"http://twitter.com/Mbethaniaml/statuses/1063582037490184192",
        "body": "Jesús “La Pulga” Gómez, uno de los máximos ídolos de
@EstudiantesMEFC, acaba de renovar por dos años con el académico⍰O□⍰⍰",
```

```
        "object": {
            "objectType": "note",
            "id": "object:search.twitter.com,2005:1063582037490184192",
            "summary": "Jesús “La Pulga” Gómez, uno de los máximos ídolos de
@EstudiantesMEFC, acaba de renovar por dos años con el académico",
            "link":
"http://twitter.com/Mbethaniaml/statuses/1063582037490184192",
            "postedTime": "2018-11-16T23:58:12.000Z"
        },
        "favoritesCount": 1,
        "twitter_entities": {
            "hashtags": [],
            "user_mentions": [
                {
                    "screen_name": "EstudiantesMEFC",
                    "name": "Estudiantes Oficial",
                    "id": 1633936946,
                    "id_str": "1633936946",
                    "indices": [
                        53,
                        69
                    ]
                }
            ],
            "symbols": [],
            "urls": []
        },
        "twitter_filter_level": "low",
        "twitter_lang": "es"
    },
    "favoritesCount": 0,
    "twitter_entities": {
        "hashtags": [],
        "user_mentions": [
            {
                "screen_name": "Mbethaniaml",
                "name": "Bethania Martínez Leo",
                "id": 86560741,
                "id_str": "86560741",
                "indices": [
                    3,
                    15
                ]
            },
            {
                "screen_name": "EstudiantesMEFC",
                "name": "Estudiantes Oficial",
                "id": 1633936946,
                "id_str": "1633936946",
                "indices": [
                    70,
                    86
                ]
            }
        ],
        "symbols": [],
```

```
        "urls": []
    },
    "twitter_filter_level": "low",
    "twitter_lang": "es",
    "retweetCount": 2,
    "gnip": {
        "matching_rules": [
            {
                "tag": "1luisacarcedo",
                "id": 6544094648404346526,
                "id_str": "6544094648404346526"
            }
        ]
    }
}
```