UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

IDENTIFICACIÓN DE CONTENIDO MULTIMEDIA RELEVANTE A PARTIR DE EVENTOS UTILIZANDO SU INFORMACIÓN SOCIAL

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

MAURICIO DANIEL QUEZADA VEAS

PROFESORA GUÍA:
BÁRBARA POBLETE LABRA

MIEMBROS DE LA COMISIÓN:
SERGIO OCHOA DELORENZI
MAURICIO MARÍN CAIHUAN

SANTIAGO DE CHILE
DICIEMBRE 2012

Resumen

Este trabajo consistió en el diseño e implementación de una metodología para la generación automática de resúmenes a partir de documentos de contenido tanto textual como multimedial. La medida de relevancia para la extracción de documentos en el proceso de la generación de resúmenes consideró la inclusión de indicadores *sociales*, es decir, se consideran más importantes los documentos con mayor impacto en medios sociales, como las redes sociales online.

El problema central fue la generación de resúmenes de eventos bien definidos, es decir, no se consideró el problema de identificación de eventos en medios sociales. Se utilizó una estrategia de clustering particional para la identificación de subtópicos de cada evento, y una estrategia simple para ponderar la relevancia de cada documento. Este tipo de trabajos no ha sido profundamente estudiado en las áreas de investigación asociadas.

Se utilizaron los servicios de Google News y Last.fm para la obtención de eventos noticiosos y musicales, respectivamente. Además, se utilizó la red social Twitter para el enriquecimiento y generación de documentos con información social. Se utilizó el algoritmo de clustering K-means para la identificación de subtópicos mediante una representación adecuada de los documentos que no considerara su contenido.

Índice general

Índice de figuras

Capítulo 1

Introducción

Al igual que en el buffet de un restaurante, por mucho que se quisieran comer todos los platos, es imposible comer todo lo que uno quisiera por razones obvias. Una posibilidad es probar un poco de cada comida, para así saber qué es lo más delicioso y comer hasta hartarse.

Pero, ¿qué hacer si hay demasiados platos y no se conocen todos? de alguna manera hay que saber cuáles hay que probar, si el objetivo es comer lo mejor posible. Un amigo, u otras personas que hayan comido en el restaurant pueden recomendar una u otra comida, lo cual puede servir para orientarse. Entonces se pueden escoger muestras de acuerdo a las recomendaciones.

Pasando a un contexto diferente, supóngase que este gran buffet es la Web y los distintos platos corresponden a contenido publicado en ella. Por lo tanto, dada la gran cantidad de información disponible, se hace necesario poder encontrar lo más atractivo de acuerdo a la preferencia de los usuarios. Nótese que se está haciendo otra suposición importante con esta analogía, y es que se está considerando que la información es íntegramente para ser *consumida*, y no, por ejemplo, para generar más contenido, conocimiento, o para ser utilizada por máquinas, etc. Dentro de este contexto se plantea la pregunta de cómo seleccionar el contenido más atractivo dentro de todo lo que hay disponible en un momento dado.

Para hacerse la idea de cuál es la mejor oferta gastronómica de un restaurant, sin tener que probar todos los platos, se puede preguntar a la gente que come habitualmente ahí qué platos fueron los que más les han gustado, y así definir la calidad de éste y/o escoger los mejores platos. O bien, filtrar los documentos que hablan de un mismo evento de acuerdo a las preferencias de los usuarios, y luego presentarlo en la forma de un *resumen* de cada uno.

Este trabajo consistió en el desarrollo de una metodología que permite generar resúmenes automáticos de eventos determinados a partir de los documentos Web que hablan de éstos. No se hace ninguna suposición sobre el *contenido* de estos documentos (pueden ser de texto, imágenes, videos, *multimedios* en general). Los documentos del resumen son seleccionados de acuerdo a *indicadores sociales*: los elementos con mayor impacto en las redes sociales son considerados más importantes.

La metodología diseñada se implementó sobre dos tipos de eventos: noticias y conciertos mu-

sicales. Para obtenerlos, se utilizó el servicio de Google News[1] y Last.fm[2]. Los documentos y sus indicadores sociales se obtuvieron de la red social Twitter[3]; y utilizando una representación adecuada de los documentos como vectores, se utilizó una estrategia de clustering para identificar los subtópicos de cada evento y así generar un resumen, basándose en los indicadores obtenidos.

La estructura de este informe es como sigue: en este capítulo se comenta el contexto dentro del cual se desarrolló este sistema, las contribuciones realizadas, los objetivos y una descripción general de la solución; en Capítulo 2 se discute el estado del arte y el marco teórico del cual se desprende este trabajo; el Capítulo 3 describe más en detalle el problema a resolver, su relevancia y sus dificultades, para luego, en el Capítulo 4, describir la solución implementada, más un par de casos de estudio sobre los resultados obtenidos, para finalizar con las conclusiones de este trabajo en el Capítulo 5.

1.1. Contexto y Motivación

La tasa de crecimiento de la cantidad de datos en la Web, y en particular, en las *redes sociales online* (OSN, *Online Social Networks*), es de tal magnitud que se vuelve necesario encontrar formas de filtrar y buscar sólo la información relevante dentro de todas las fuentes que hablan del mismo evento.

Por otra parte, para poder entender lo que está pasando en el mundo a partir de los puntos de vista de toda las personas que manifiestan su opinión o los hechos recabados en las redes sociales, se hace necesario poder procesar toda esta información. Nuevamente, dado el gran volumen de datos, es necesario poder resumir esta información para ser rápidamente consumida.

En el contexto de las redes sociales online, cada día los usuarios publican millones de mensajes cortos con respecto a distintos tópicos, ya sean conversacionales, sobre eventos noticiosos, etc.[4] Además, el auge de los teléfonos inteligentes o *smartphones* con mayor capacidad de procesamiento e integrados con todo tipo de sensores (cámaras fotográficas, de vídeo, acelerómetro, osciloscopio, etc.), hace posible el generar aun más información y e incluso en tiempo real sobre lo que acontece en el mundo, en Internet, o bien sobre el estado particular de cada usuario.

Este aumento y evolución de la generación de datos no sólo influye en la riqueza de éstos, sino también en el comportamiento de los usuarios a lo largo del tiempo. Actualmente, una gran parte de éstos valora más el contenido de tipo multimedia (imágenes y videos) en las redes sociales online[5]. Este contenido no sólo corresponde a documentos Web en el sentido tradicional, sino a objetos de más alto nivel como *tweets* (mensajes cortos de la red social Twitter), imágenes (de servicios

[1] http://news.google.com/

[2] http://last.fm/

[3] http://twitter.com/

[4] Pear Analytics. Twitter Study http://es.scribd.com/doc/18548460/Pear-Analytics-Twitter-Study-August-2009

[5] The Rise of Visual Social Media http://www.fastcompany.com/3000794/rise-visual-social-media. En el artículo se menciona un estudio sobre comportamiendo y preferencias de los usuarios en las redes sociales llevado a cabo por ROI Research: http://www.slideshare.net/performics_us/performics-life-on-demand-2012-summary-deck

como Instagram[6], Tumblr[7], etc.), vídeos (Youtube[8], Vimeo[9]) e incluso sonidos (Soundcloud[10]). Se hace entonces necesario encontrar formas para satisfacer estas necesidades de los usuarios, las cuales ya han sido abordadas en parte, como la generación de resúmenes automáticos orientado a motores de búsqueda, o la determinación de la relevancia tanto de documentos en la Web como de mensajes en las redes sociales.

Surge como motivación el poder identificar y extraer contenido relevante de lo que está pasando en las redes sociales, clasificados dentro del concepto de eventos, y además avanzar un paso más arriba en el nivel de abstracción: considerar los documentos no por su contenido textual, lo que permite abarcar imágenes, vídeos, sonidos y multimedios en general. Algunas aplicaciones directas de esto son, entre otras:

- Ayudar al trabajo periodístico mediante una colección de contenido multimedia relacionado a un evento noticioso. Por ejemplo, la versión online de Radio Biobío[11] frecuentemente publica breves artículos sobre sucesos que tienen impacto en las redes sociales, mostrando un pequeño conjunto de mensajes con comentarios de la gente[12]. Una aplicación directa involucraría considerar además contenido multimedia, y organizar este contenido de acuerdo a la relevancia que tiene dentro de las redes.

- Enriquecer la búsqueda en la Web a través de contenido multimedia. Una persona buscando información sobre un concierto podría obtener imágenes y vídeos de éste fácilmente una vez identificado el concierto.

- Siguiendo lo anterior, un grupo musical podría obtener toda la información multimedia asociada a su concierto, tanto para sus fans como para ellos mismos, potenciando su popularidad.

- Poder distinguir entre eventos similares rápidamente. Por ejemplo, un usuario que desee obtener información sobre "Gaza", puede referirse tanto a la banda de música como al conflicto en Israel. El poder distinguir rápidamente mediante una imagen o un vídeo acelera mucho el proceso. *Una imagen vale más que mil palabras.*

La metodología implementada es una primera aproximación que puede satisfacer los ejemplos mencionados.

[6] http://instagr.am/
[7] http://tumblr.com/
[8] http://youtube.com/
[9] http://vimeo.com/
[10] http://soundcloud.com/
[11] http://www.biobiochile.cl/
[12] Como muestra: http://www.biobiochile.cl/2012/12/01/aporte-de-lustrabotas-de-santiago-a-la-teleton-provoca-admiracion-en-redes-sociales.shtml, y http://www.biobiochile.cl/2012/12/01/rechazo-provocan-condicionamientos-de-compra-de-ripley-y-unimarc-para-donar-a-la-teleton.shtml

1.2. Objetivos

1.2.1. Objetivo general

El objetivo principal de este trabajo fue el siguiente:

Diseñar e implementar un sistema que permita generar resúmenes automáticos de *eventos*: información temporal publicada en redes sociales online sobre sucesos en particular. Esta información se basa en contenido textual y multimedial generado por los usuarios de estas redes, cuya relevancia se basa en el impacto generado en éstas.

1.2.2. Objetivos específicos

1. Extraer datos relacionados a eventos en la Web, principalmente aquellos generados en redes sociales online. Estos datos pueden componerse tanto de información textual como multimedial.
2. Agrupar la información extraída de un evento en subtópicos.
3. Seleccionar los elementos más relevantes de cada subtópico para producir un resumen del evento.
4. Analizar la efectividad de la metodología propuesta sobre un conjunto de eventos noticiosos y conciertos.

1.3. Descripción general de la solución

Para enfrentar el problema se diseñó una solución que consiste en tres componentes para resolverlo, resultando cada una en diferente grado de complejidad, siendo posible además ser mejoradas en el futuro. Para probar la viabilidad de la metodología, ésta fue aplicada sobre un conjunto de casos de prueba.

En particular:

- Se llevó a cabo una metodología para la obtención de documentos y enriquecerlos con datos obtenidos de fuentes sociales;
- Se diseñó un procedimiento que separar estos documentos en *clusters*, *sin considerar su contenido*. Sólo se utilizó la información social asociada; y
- Se diseñó además un procedimiento para *rankear* u ordenar los resultados de acuerdo a *relevancia*, siendo ésta medida de acuerdo a la información social asociada a los documentos generados.

Las componentes diseñadas fueron las siguientes:

1. La que obtiene descripciones de eventos a partir de fuentes de éstos en la Web, enriqueciéndolos con información social;
2. Otra componente que procesa y separa los documentos a partir de la información social; genera *objetos Web* y los separa en subtópicos de cada evento, respectivamente; y
3. La componente que entrega los k documentos más relevantes por cada evento obtenido, basándose en los subtópicos identificados.

Capítulo 2

Antecedentes

Para poder describir correctamente tanto el problema como la solución implementada, es necesario dar los punteros y conceptos básicos que los involucran. En este capítulo se discutirán los siguientes tópicos:

- La red social Twitter, la cual es utilizada como fuente de datos y documentos para este trabajo.
- Clustering de documentos, y en general, estrategias de clustering, las cuales tienen muchas aplicaciones prácticas. En este trabajo fue utilizada una de estas estrategias para poder determinar los subtópicos de cada evento.
- La identificación automática de eventos, la cual, si bien se utilizó un enfoque más simple para este trabajo, sirve para indicar en qué aspectos es posible extender este trabajo en el futuro.
- Resúmenes automáticos: una sucinta definición, y algunos enfoques que han existido en el tiempo para este procedimiento.
- Ranking de documentos, o cómo generar órdenes de acuerdo a relevancia.

Casi todos estos tópicos, a excepción del primero, involucran técnicas de Minería de Datos, Recuperación de la Información y Aprendizaje de Máquinas, entre otras áreas.

2.1. Twitter

Twitter es una red social online que permite conectar a personas mediante la comunicación de mensajes cortos, rápidos y frecuentes[1]. Estos mensajes son publicados en el perfil del usuario que los emite, pueden ser vistos directamente por los seguidores de este usuario o ser vistos directamente en el perfil o buscándolos mediante una funcionalidad que provee el servicio. Además, un usuario puede *seguir* a otros para poder ver en su *timeline* o perfil privado los mensajes de todos a quienes sigue.

[1]https://support.twitter.com/groups/31-twitter-basics/topics/104-welcome-to-twitter-support/articles/13920-get-to-know-twitter-new-user-faq

Estos mensajes, o *tweets*, sólo son cadenas de caracteres con metadatos que el mismo servicio asigna una vez enviado a la red social. Desde sus inicios (año 2007) se han añadido algunas capacidades adicionales a estos mensajes, como la de poner URLs, imágenes, vídeos, etc. Además, existen varias convenciones que han surgido a lo largo del tiempo. A continuación se describe una lista de tipos de mensajes que existen en Twitter, originados por estas convenciones:

1. Respuestas o *replies*: son mensajes del tipo `@usuario [texto]`, que ocurren usualmente en una conversación entre dos usuarios.
2. Menciones o *mentions*: un poco más general a una respuesta, el nombre del usuario mencionado puede estar en cualquier parte del mensaje. La diferencia semántica es que no se le habla "directamente" al usuario mencionado, como en una respuesta, sino que sólo es mencionado por si el mensaje es de su interés o no.
3. *Retweets*: son mensajes del tipo `RT @usuario: [texto]`. Ocurren cuando se quiere compartir el mensaje de otro usuario, o citarlo para mencionarlo en el mismo mensaje.
4. *Hashtags*: son palabras precedidas por el caracter #, que indican un identificador a cierto evento o suceso dentro o fuera de la red. Suelen usarse para categorizar de cierta forma un tópico, pero son libres de usarse como los usuarios quieran.
5. Mensaje simple: un mensaje sin menciones ni hashtags.

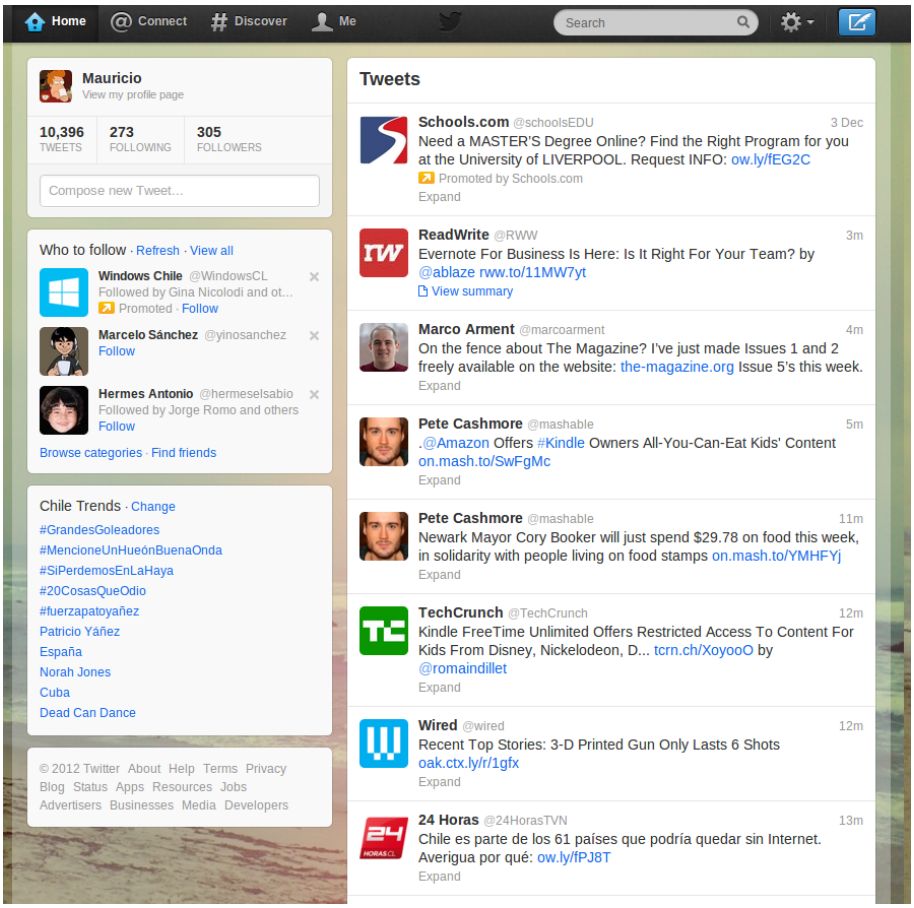

Figura 2.1: Timeline de Twitter. En éste se ve una lista en orden cronológico de los tweets generados por los usuarios que sigue el usuario actual. Además, el sitio incluye tweets promocionados por cuentas que pagan por dicho servicio, como es el caso del primer tweet en la lista.

Ejemplos:

- Mensaje simple: `Jason Funk disipa patitos`;
- Respuesta: `@jason estoy de acuerdo con lo que dices`;
- Mención: `creo que @jason es una cumbre de sabiduría`;
- Retweet: `RT @jason: Jason Funk disipa patitos`; y
- Hashtag: `Estoy escribiendo mi memoria #dcc #summarization`

Estos mensajes están limitados a 140 caracteres de extensión. Sumando esto a la integración de la red con otros servicios y dispositivos, y a la cantidad de mensajes publicados cada minuto, permite utilizar esta red como una gran fuente de datos.

Twitter además provee varios servicios adicionales, como por ejemplo, un servicio de acortamiento de URLs, para permitir incluir una URL larga sin perjudicar la cantidad de caracteres restantes para el mensaje; un servicio de alojamiento de fotos y vídeos, para hacer más sencilla la publicación de mensajes multimedia desde dispositivos móviles; un servicio de búsqueda que permite buscar una cantidad determinada de tweets sobre un término de búsqueda o un hashtag, entre otros servicios.

2.2. Clustering de documentos

El análisis de clusters o clustering es el proceso de encontrar grupos de objetos, tal que los objetos en un grupo sean similares entre sí (o relacionados) y que sean diferentes (o no relacionados) a los objetos de otros grupos. Algunas aplicaciones del análisis de clusters son, entre otras:

- Encontrar clusters naturales y describir sus propiedades (*data understanding*);
- Encontrar agrupamientos útiles (*data class identification*);
- Encontrar representantes de grupos homogéneos (*data reduction*);
- Encontrar perturbaciones aleatorias de los datos (*noise detection*);
- Encontrar objetos inusuales (*outliers detection*);
- etc.

Se denomina cluster a un grupo de objetos, mientras que clustering puede referirse al conjunto de clusters o al proceso de encontrarlos. Existen diversos tipos de procesos de clustering, una de las distinciones más importantes es entre los clusters jerárquicos y los particionales:

- Un clustering jerárquico es un conjunto de clusters anidados, organizados más bien como un árbol. Cortando el árbol en cualquier nivel da como resultado un clustering potencialmente distinto.
- El clustering particional es un conjunto de clusters de forma de partición del conjunto total, es decir, cada objeto está contenido en un sólo subconjunto o cluster.

Para describir el proceso aplicado a documentos, primero se describirán los modelos de representación más importantes para, de forma de definir la noción de documento y luego los algoritmos

de clustering aplicados a éstos.

2.2.1. Modelos de representación de documentos

Standard Boolean Model

El modelo booleano es un modelo de representación de documentos. En él, los documentos son vectores de *términos*:

$$\mathrm{d} = (w_1, w_2, \ldots, w_m)$$

Donde un término es un n-grama del texto del documento.

Definición 2.1 *Un n-grama es una secuencia contigua de n ítems a partir de un texto.*

La definición de ítem dependerá de la aplicación: en lenguaje natural el texto a su vez dependerá del idioma, por ejemplo, si el texto está en inglés o en japonés, la distinción entre palabras es distinta para cada uno.

No existe una medida de "similitud" como tal en este modelo, sino que se considera el calce exacto entre los términos de una query q y un documento d. La query puede ser una consulta hecha por un usuario al conjunto de documentos, o bien un documento del mismo conjunto.

Una consulta es una fórmula de lógica proposicional que pide los documentos que contengan o no ciertos términos.

Bag of words Model

En el modelo Bag of Words un documento d es representado como un conjunto de pares $(w_\mathrm{i}, f_\mathrm{i})$, $\mathrm{i} \in [1..m']$, donde w_i es un término del documento, f_i es la frecuencia de w_i en el mismo, y m' es la cantidad de términos distintos en el documento.

La ventaja principal por sobre el modelo anterior es que permite hacer calces parciales entre consultas y documentos. Este modelo es comúnmente utilizado para hacer clasificación de documentos, por ejemplo, para determinar si un correo electrónico es o no spam.

Vector Space Model

El *Vector Space Model* es un modelo un poco más general que el anterior. Un documento d es representado como un vector de pesos asociados a los términos:

$$\mathrm{d} = (f(w_1), f(w_2), \ldots, f(w_m))$$

Cada dimensión de este vector corresponde al peso asociado a un término del documento.

El peso puede ser directamente la frecuencia del término dentro del documento:

$$\mathrm{freq}(w, \mathrm{d}) = |\{w : w \in \mathrm{d}\}|$$

O bien, normalizar esta frecuencia para evitar que documentos más largos sean más relevantes sólo por su extensión:

$$\mathrm{tf}_0(w, \mathrm{d}) = \begin{cases} 1 & \text{si } w \in \mathrm{d} \\ 0 & \text{si no} \end{cases}$$

tf_0 o *Term Frequency* es una primera aproximación a medir la frecuencia de un término en un documento. Sin embargo, esta nueva aproximación sufre de la desventaja de que ahora un documento con una ocurrencia del término será igual de relevante que algún documento que mencione varias veces el término (por ejemplo, un diccionario que tiene el término una vez contra un artículo sobre el tema). Otra alternativa, considera no castigar demasiado a los documentos con pocas ocurrencias, pero tampoco beneficiar mucho a los que tengan muchas:

$$\mathrm{tf}_1(w, \mathrm{d}) = 1 + \log(\mathrm{freq}(w, \mathrm{d}))$$

La solución más utilizada considera la proporción con respecto al término con más ocurrencias, para esto, se normaliza por el tamaño del documento:

$$\mathrm{tf}(w, \mathrm{d}) = \frac{\mathrm{freq}(w, \mathrm{d})}{max\{\mathrm{freq}(t, \mathrm{d}) : t \in \mathrm{d}\}}$$

Otro problema que tiene utilizar esta medida como los pesos de los términos, es que un término muy repetido entre todos los documentos que hablan de un mismo tema puede significar que no es muy relevante (por ejemplo, las *stopwords* o palabras vacías, son por lo general las preposiciones, artículos, pronombres, etc.). Para esto, se considera además ponderar por el inverso de la frecuencia entre los documentos; es decir, un término frecuente entre todos los documentos ve su peso castigado a diferencia de un término que sólo es mencionado una vez en un documento. Esta medida es llamada *Inverse Document Frequency* o idf:

$$\mathrm{idf}(t, D) = \log \frac{|D|}{1 + |\{\mathrm{d} \in D : t \in \mathrm{d}\}|}$$

Finalmente, el peso de un término es la ponderación de su frecuencia dentro del documento con el inverso de la frecuencia entre los documentos, o tf-idf:

$$\text{tf-idf}(t, \mathrm{d}, D) = \text{tf}(t, \mathrm{d}) \times \text{idf}(t, D)$$

Para el resto de este trabajo se considerará la representación de documentos en este modelo.

2.2.2. Medidas de similitud

A lo largo de los años, dos formas han sido las usuales para determinar similitud entre documentos[19]: calculando el coseno de los dos vectores, y la distancia euclidiana.

El coseno de dos documentos $\mathrm{d_i}$ y d_j se define como

$$\cos(\mathrm{d_i}, \mathrm{d}_j) = \frac{\mathrm{d_i^t d}_j}{\|\mathrm{d_i}\|\|\mathrm{d}_j\|}$$

Si ambos $\mathrm{d_i}$ y d_j son iguales, entonces la fórmula anterior se evalúa a 1, y 0 si no tienen nada en común, es decir, ambos vectores son ortogonales.

La distancia euclidiana entre $\mathrm{d_i}$ y d_j se define como

$$\mathrm{d}(\mathrm{d_i}, \mathrm{d}_j) = \sqrt{(\mathrm{d_i} - \mathrm{d}_j)^t(\mathrm{d_i} - \mathrm{d}_j)} = \|\mathrm{d_i} - \mathrm{d}_j\|$$

En este caso, si la distancia es 0, entonces ambos documentos son idénticos. Y si ambos son ortogonales, la distancia será $\sqrt{2}$.

Si ambos documentos están normalizados (su norma es 1), entonces ambas medidas serán muy parecidas, aun así siendo una de similitud y la otra de distancia.

2.2.3. Algoritmos de clustering

Se considerará la siguiente definición para el problema de clustering:

Dado un entero k y un conjunto de n puntos en $\mathbb{R}^v$, el objetivo es determinar k puntos tal que se minimice ϕ, la suma de las distancias al cuadrado de cada punto y su centro más cercano.

Este problema es NP-hard, sin embargo, existe un algoritmo que permite realizar una búsqueda local y determinar k centros en un tiempo razonable.

K-means[15] es un algoritmo de clustering particional, en el cual cada cluster tiene un *centroide* asociado, esto es, típicamente, un punto el cual es el promedio de todos los puntos del cluster, y no necesariamente corresponde a un dato real.

El algoritmo genera k clusters, donde k es un parámetro del algoritmo, tal que cada punto pertence al cluster cuyo centroide es el más cercano a éste.

Algoritmo 1: K-means

Data: $k > 0$, conjunto D de puntos
Result: Asignación de cada punto a un cluster i $\in [1..k]$
1 Seleccionar k puntos como centroides iniciales;
2 **while** *Los centroides cambien a cada iteración* **do**
3 Formar k clusters asignando todos los puntos al centroide más cercano;
4 Recalcular los centroides de cada cluster;

Los clusters generados dependerán de la elección inicial de los centroides, y usualmente basta con pocas iteraciones para la convergencia. La complejidad de este algoritmo es $O(nkIv)$, donde n es el número de puntos, k el parámetro de la cantidad de clusters, I es la cantidad de iteraciones que hará el algoritmo y v es la cantidad de dimensiones de los vectores.

K-means también puede verse desde el enfoque de optimizar una función criterio. La medida más común es la *suma del error cuadrado* o Sum of Squared Error (SSE). Para cada punto, su *error* es la distancia al cluster más cercano:

$$SSE = \sum_{\mathrm{i}=1}^{k} \sum_{x \in C_{\mathrm{i}}} \mathrm{dist}^2(m_{\mathrm{i}}, x)$$

Donde x es un punto en el cluster C_{i}, y m_{i} es su centroide. La distancia dist suele ser distancia euclidiana entre los dos puntos. Dados dos clusters, en cada paso se escoge el que tenga menor error.

K-means converge rápidamente a un óptimo, sin embargo, éste puede ser un óptimo local, dado que es altamente dependiente de la elección de los centroides iniciales. Dado que la probabilidad de acertar a los centroides "reales" (precisión) es muy baja, es posible que un clustering tenga un mal error cuadrático, de hecho, el radio competitivo con la solución óptima es no acotado incluso para k y n fijos. Existen varias formas de evitar este problema, por ejemplo, ejecutando varias veces K-means, usar clustering jerárquico para determinar los centroides iniciales, determinar más de k centroides y luego elegir los k más separados, etc.

Un enfoque utilizado es el de usar el algoritmo K-means++[2], el cual tiene como objetivo determinar los centroides iniciales para K-Means. Este algoritmo garantiza un radio competitivo de $O(\log k)$ con respecto al error cuadrático esperado del óptimo. El algoritmo K-means++ escoge los centroides con cierta probabilidad que depende de la distancia al centroide más cercano, lo cual garantiza una buena elección inicial, y luego continúa con el algoritmo usual. Sea $D(x)$ la

distancia de x al centroide más cercano. El algoritmo es como sigue:

Algoritmo 2: K-means++

Data: $k > 0$, conjunto D de puntos
Result: Asignación de cada punto a un cluster i $\in [1..k]$
1 Escoger un centroide c_1 al azar uniformemente de D;
2 Escoger un nuevo centroide c_i, escogiendo $x \in D$ con probabilidad $\frac{D(x)^2}{\sum_{x' \in D} D(x')^2}$;
3 Repetir paso 1 hasta haber escogido k centros;
4 Proceder con K-means estándar;

Existen muchas otras variantes para resolver el problema de clustering, principalmente dependientes de la naturaleza de los datos. En la Figura 2.2[2] se puede apreciar una comparación entre distintos algoritmos de clustering y los resultados obtenidos. MiniBatch K-means es una variante de K-means que mejora sustancialmente el tiempo de convergencia escogiendo subconjuntos en vez de centroides al inicio, aunque la calidad de los resultados puede ser levemente menor a la del algoritmo clásico[16].

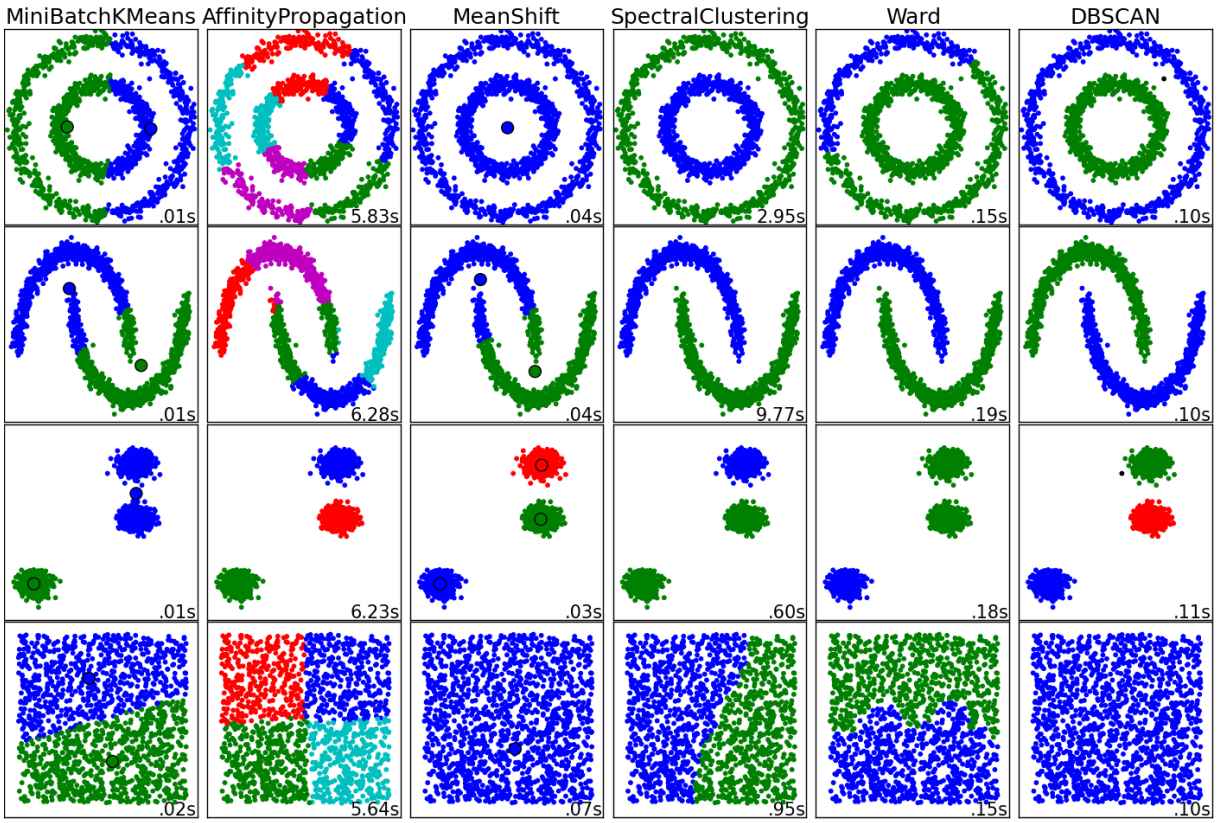

Figura 2.2: Comparación de algoritmos de clustering. En la figura se muestran las características de distintos algoritmos de clustering sobre datos en 2 dimensiones. El último caso muestra datos uniformes en el cual sólo habría 1 cluster.

[2]Imagen obtenida de http://scikit-learn.org/dev/auto_examples/cluster/plot_cluster_comparison.html

2.2.4. Evaluación de clusters

Existen dos enfoques para evaluar clusterings: validación interna y externa[3] La validación interna considera evaluar el clustering con respecto a los datos que han sido agrupados (es decir, sin información adicional), mientras que la evaluación externa considera datos adicionales, como etiquetas a los datos determinadas con anterioridad.

La idea básica detrás de las medidas internas directamente de la definición de clustering. Una buena solución debería agrupar objetos en varios clusters, de forma que los objetos dentro de un cluster sean más similares entre sí, y que los objetos de clusters distintos lo sean lo menos posible. La calidad de la solución se mide en términos del promedio de la similitud interna, y en términos del promedio de la similitud externa, y el radio entre estos dos promedios. Entre más alto es el radio, mejor es la solución entregada.

Por otra parte, dos de las métricas para validación externa más usadas para evaluar clusterings, segun [19] son:

- **Entropía**: mide cómo son distribuidas las clases de documentos entre cada cluster. Se define formalmente como

  $$E(C_r) = -\frac{1}{\log q} \sum_{\mathrm{i}=1}^{q} \frac{n_r^{\mathrm{i}}}{n_r} \log \frac{n_r^{\mathrm{i}}}{n_r}$$

  donde q es el número de clases en el dataset, y n_r^{i} es el número de documentos de la i-ésima clase que fue asignado al r-ésimo cluster C_r. La entropía del clustering se define como la suma ponderada de la entropía de cada cluster:

  $$\text{Entropia} = \sum_{r=1}^{k} \frac{n_r}{n} E(C_r)$$

  Un clustering perfecto tendrá clusters tal que cada cluster contenga documentos de una sola clase, en ese caso la entropía será 0. En general, conviene tener bajos valores de entropía.
- **Pureza**: mide la cantidad de documentos de la clase más grande en un cluster dividida por el tamaño del cluster. La pureza de un cluster C_r se define como

  $$P(C_r) = \frac{1}{n_r} \max_{\mathrm{i}}\{n_r^{\mathrm{i}}\}$$

  A mejor pureza, mejor es la solución.

2.3. Identificación automática de eventos

La identificación automática de eventos consiste en, dado un conjunto de documentos, donde cada documento está asociado a un evento (desconocido), poder particionar el conjunto de docu-

[3] http://en.wikipedia.org/wiki/Cluster_analysis#Evaluation_of_clustering_results

mentos en clusters, de forma que cada cluster corresponda a todos los documentos asociados a un evento.

La definición de "evento" dada por [18], considera lo siguiente:

Definición 2.2 *Un* evento *es un suceso que ocurre en un período de tiempo determinado y en un lugar específico.*

Otra definición de "evento" considera además la información asociada a éste[1]:

Definición 2.3 *Un* evento *es una ocurrencia en el mundo real* e *con (1) un período de tiempo asociado* T_e*, y (2) una secuencia ordenada cronológicamente de mensajes* M_e *de volumen sustancial, que discuten la ocurrencia y que son publicados durante el período* T_e

Si los eventos son conocidos con anticipación, el problema consiste en recuperar documentos que los discutan. Las fuentes de datos son por lo general muy especializadas, por ejemplo, en el caso de las noticias[10] o documentos estructurados, donde es posible utilizar técnicas de procesamiento de lenguaje natural para detectar características de los documentos y utilizar un clasificador para identificar el evento al cual pertenecen. Algunos trabajos utilizan otras fuentes de datos, tal como de fotografías o vídeos, donde estas tienen sus propias características (descripción, tags, coordenadas geográficas, marca de tiempo, etc.), las cuales son aprovechadas para aplicaciones ad-hoc[14]. Otros esfuerzos consideran utilizar distintas fuentes de datos y aprender sus características, combinándolas todas utilizando alguna metodología entrenada con anticipación, tales como [4], o [3], en el cual se generan distintas queries de manera automática para realizar búsquedas de documentos en la Web, basadas en las características de los eventos. Esto permite enriquecer los eventos con información variada sobre ellos.

Cuando los eventos no son conocidos con anticipación, algunos de los enfoques mencionados anteriormente también pueden ser utilizados, por ejemplo, suponer que ocurre un evento si muchos documentos (imágenes, vídeos, tweets, etc.) fueron generados en la misma zona geográfica en un período acotado de tiempo, como en [14], o si se publican muchos mensajes de un tópico en particular en un período acotado (llamado un *burst* o ráfaga de mensajes); en ese caso también se pueden identificar eventos cuando el tópico es conocido, por ejemplo, en [7] utilizan eventos deportivos para identificar las partes más relevantes de cada partido, basándose en los bursts de tweets ocurridos en Twitter.

2.4. Resúmenes automáticos

Debido a la gran cantidad de información en Internet, los sistemas de *Information Retrieval* (IR) se volvieron necesarios para la extracción de contenido relevante a partir de los documentos encontrados. Sin embargo, dada aún la gran cantidad de documentos retornados por estos sistemas, un siguiente nivel de abstracción ha sido necesario para éstos, el cual es la necesidad de construir resúmenes de estos documentos[12]. Un survey más exhaustivo sobre el tema puede ser consultado en [9].

Entre las múltiples aplicaciones de los resúmenes automáticos se pueden mencionar:

- Extracción de información de múltiples fuentes;
- Rápida adquisición de conocimiento;
- Responder preguntas automáticamente de tópicos muy específicos;
- Generación de reseñas biográficas, etc.

Los resúmenes pueden ser clasificados según los siguientes criterios:

- **Detalle**: Indicativo / Informativo;
- **Granularidad**: Específico a un evento / Mirada general;
- **Técnica**: Extractivo / Abstractivo;
- **Contenido**: Generalizado / Basado en consulta; y
- **Aproximación**: Dominio / Específico de un género / Independiente.

Las categorías más importantes a la hora de clasificar un resumen son la técnica y el contenido, mientras que otra forma de clasificar resúmenes se basa en la entrada de la aplicación: resumen de un solo documento o bien multi-documento.

Un resumen extractivo apunta a extraer las oraciones más importantes de un documento, mientras apunta a mantener una baja redundancia en el resumen. Un resumen abstractivo es más complejo: la idea es poder generar un resumen utilizando palabras no necesariamente sacadas de los documentos, o dicho de otra forma, estos sistemas intentan *comprender* el contenido de los documentos de forma de generar un resumen abstracto. La forma clásica de generar resúmenes es utilizando métodos extractivos, dada la complejidad y baja calidad aún de los enfoques abstractivos.

El criterio de contenido de un resumen se basa principalmente en cuál es la base para generarlo: si es basado en consulta, entonces la entrada del sistema es una frase y el resumen intenta encontrar las frases u oraciones más similares a la consulta, como una especie de respuesta a una pregunta. Un resumen de contenido general simplemente genera un resumen de todo el o los documentos de la entrada.

Existen diversos enfoques para generar un resumen en base a uno o múltiples documentos: el más usual es utilizando clustering. Sin embargo, a veces son consideradas algunas características propias del documento: por ejemplo, si son noticias, usualmente las primeras oraciones del texto contienen la información más relevante[5]. Algunas características generales usuales son:

- Ocurrencia de palabras clave: oraciones con palabras clave usualmente son usadas en el resumen.
- Palabras claves de título: a su vez, oraciones que contengan palabras del título también son consideradas importantes.
- Heurísticas de ubicación: como fue mencionado, según la naturaleza del documento, la ubicación de las frases tienen cierta relevancia, tales como en documentos noticiosos, o artículos técnicos.
- Frases indicativas: “en este informe”, o “en conclusión”.
- Frases cortas: usualmente las frases muy cortas no son tomadas en consideración.

- Características de palabras en mayúsculas: acrónimos o nombres propios son considerados importantes.

Entre las distinas estrategias para generar resúmenes, aparte de clustering usando una representación adecuada de los documentos, se cuentan:

- Similitud directa entre documentos, usando el coseno de los vectores;
- Aproximaciones usando teoría de grafos, en el cual, por ejemplo, cada nodo es una frase, y una arista existe entre dos nodos si su similitud está sobre un umbral; luego se determina la centralidad de los nodos mediante un camino aleatorio u otras estrategias;
- Usando aprendizaje de máquinas: mediante un clasificador aprender las características de un conjunto de prueba;
- Utilizando modelos más elaborados, como Modelos Ocultos de Markov[4] para determinar características ocultas entre los datos, como fue usado en [7]; etc.

2.4.1. Evaluación de resúmenes

ROUGE (*Recall Oriented Understudy for Gisting Evaluation*)[13] es una medida intrínseca para la evaluación semi-automática de resúmenes. No es tan efectiva como una evaluación hecha por humanos, dada la naturaleza del problema, pero es más conveniente y escalable.

Dado un documento D y su resumen automático X, y M personas que generan un resumen de referencia sobre D, ROUGE-n se determina de la siguiente forma:

$$\text{ROUGE-}n = \frac{\sum_{S \in \text{Ref}} \sum_{\text{n-grama i} \in S} \min(\text{count}(\text{i}, X), \text{count}(\text{i}, S))}{\sum_{S \in \text{Ref}} \sum_{\text{n-grama i} \in S} \text{count}(\text{i}, S)}$$

Donde Ref es el conjunto de resúmenes de referencia, y count cuenta las ocurrencias del n-grama i en X o S. ROUGE-n mide la cantidad de n-gramas de los resúmenes de referencia que aparecen en el resumen automático, por lo que es necesario que hayan muchos resúmenes de referencia dada la potencial subjetividad a la que son sujetos.

2.5. Ranking de documentos sociales

Siguiendo este proceso, primero identificando eventos, luego generando resúmenes a partir de ellos, el último paso corresponde a determinar, a partir de un conjunto pequeño de documentos (que si bien, no corresponden directamente a un *resumen* de acuerdo a lo dicho en la sección anterior, puesto que estos resúmenes consideran la extracción de frases a partir de los documentos, y no documentos enteros), cómo determinar cuáles son más relevantes que otros. Otra aplicación directa consiste en determinar, más que relevancia, *importancia* de un documento o mensaje, de acuerdo a características propias de éste o bien de su autor. En [6], los autores determinan una

[4] http://en.wikipedia.org/wiki/Hidden_Markov_model

metodología para verificar automáticamente la *credibilidad* de mensajes en Twitter, usando para ello indicadores obtenidos tanto de los mensajes como de sus autores.

En este aspecto hay varias alternativas para generar rankings de documentos:

- Si los documentos son documentos Web, existen varias estrategias, correspondientes principalmente al campo de IR y motores de búsqueda (PageRank, HITS, etc.), como puede ser visto en [17].
- Si se toma en consideración el contenido en texto de los documentos (por ejemplo, noticias, documentos legales, históricos, técnicos, etc.), teniendo como base una consulta, se pueden utilizar métricas de similitud para ordenar los documentos de acuerdo a la similitud a la consulta. Si no existe una consulta, se utiliza esta medida entre los mismos documentos.
- Si los documentos corresponden a mensajes cortos, como usualmente es visto en los medios sociales, existen algunas aproximaciones que utilizan metodologías de Learning to Rank para entrenar un sistema de *ranking* de documentos. Un trabajo empírico puede ser consultado en [11]. Learning to Rank[5] es un problema semi-supervisado de aprendizaje de máquinas para construir un modelo de ranking de datos, es decir, poder generar una permutación de los datos de acuerdo a ciertos criterios que pueden ser entrenados con anticipación por este modelo.

[5] http://en.wikipedia.org/wiki/Learning_to_rank

Capítulo 3

Especificación del Problema

El problema a resolver consiste en poder *resumir* eventos en base al contenido textual y multimedial de la información publicada en medios sociales, tales como las redes sociales online.

Se considerará un evento como una ocurrencia en el mundo real con un período de tiempo asociado y un conjunto de mensajes, de volumen considerable, que discutan la ocurrencia y además publicados dentro del período de tiempo.

El resumir un evento consiste en entregar un subconjunto de mensajes o de documentos mencionados en ellos que tengan relación directa con el evento en cuestión. Por ejemplo, el evento `Anef anuncia movilización nacional` contiene muchos mensajes, tales como:

- `RT @econtingencia: Reajuste: Trabajadores del sector público convocan a movilizaciones: La negociación entra a su recta final: El l... h ...`
- `CUT de La Araucanía se adherirá a marcha nacional en rechazo a reajuste salarial: La Central Unitaria de Trabaja... http://t.co/IcgohlTR`
- `Reajuste: Trabajadores del sector público convocan a movilizaciones http://t.co/pWC7q6NI`
- `RT @biobio: CUT de La Araucanía se adherirá a marcha nacional en rechazo a reajuste salarial ofrecido por el Gobierno http://t.co/IOieDido`
- `http://t.co/ttFp0XgH Trabajadores del sector público anuncian movilizaciones`
- `RT @DanielaLopezLv: http://t.co/ttFp0XgH Trabajadores del sector público anuncian movilizaciones`
- `RT @Barbara_figue: Reajuste: Trabajadores del sector público convocan a movilizaciones http://t.co/DeXKACSR vía @nacioncl`

El resumen consistirá en una colección de objetos, documentos o mensajes, donde en este caso los documentos están representados por las URLs contenidas en los mensajes, de forma que esta colección represente lo mejor posible el evento, siendo de tamaño considerablemente menor al total de mensajes/documentos del evento; garantizando además baja redundancia entre ellos.

El problema de identificar contenido multimedial en medios sociales a partir de eventos de

por sí es un problema desafiante dada la heterogeneidad y la naturaleza ruidosa de los datos: los mensajes son breves, y pueden contener errores gramaticales o de ortografía; además, pueden ser ambiguos respecto al evento que están haciendo referencia (por ejemplo, un mensaje que mencione la palabra "Gaza" puede referirse al conflicto en medio oriente o bien a la banda musical "Gaza").

Este problema no ha sido abordado en profundidad con anterioridad, sino que los esfuerzos se han concentrado principalmente en la generación de resúmenes textuales por una parte[8, 1, 5, 10], y en la recolección de contenido multimedia para eventos por otra[3, 14, 4]. Sin embargo, aún no han habido muchos esfuerzos en cuanto a la generación de resúmenes visuales o resúmenes multimedia en cuanto a eventos.

Las aplicaciones de una solución comprenden muchas en común con las aplicaciones de clustering y resúmenes automáticos, además de la obtención de conocimiento:

- Poder distinguir entre dos eventos rápidamente, por ejemplo, si "Gaza" se refiere al evento musical o al conflicto en medio oriente.
- Comprender la información contenida en grandes fuentes de datos rápidamente mediante un resumen visual.
- Mejorar la calidad del servicio de distintos rubros, al conocer de manera más eficaz el feedback de los usuarios (por ejemplo, un *review* en vídeo del tablet Nexus 7 de Google, o fotografías de baterías de teléfonos móviles que explotan sin razón aparente).
- Tener información más completa sobre eventos musicales, por ejemplo, una persona que quiera asistir a un festival de música puede ver rápidamente fotografías y vídeos del evento realizado en otras partes, para poder tomar una mejor decisión.
- Mejorar el trabajo periodístico, al tener mejor cobertura de eventos masivos como manifestaciones o eventos políticos como elecciones o discursos, entre otros.

Capítulo 4

Descripción de la Solución

La solución propuesta consistió en generar un resumen utilizando documentos representados apropiadamente con la información de medios sociales que los mencionan. La relevancia de estos documentos también se obtuvo con la información social más otros indicadores de la solución parcial.

Para esto, se asume que existe una fuente de eventos, de esta forma, el enfoque de la solución radica principalmente en la generación de los resúmenes y no en la identificación de eventos en medios sociales.

La Figura 4.1 muestra de manera general el modelo propuesto. El modelo considera varias etapas: obtención de eventos, enriquecimiento desde medios sociales para la generación de documentos, identificación de subtópicos y la selección de los k más relevantes.

1. La obtención de eventos consiste en la recolección de metadatos sobre eventos del mundo real, para esto se asume una fuente existente, siendo el problema de identificación de eventos fuera del alcance de este trabajo.
2. Con los metadatos recolectados, se realizan búsquedas en medios sociales para la obtención de mensajes a partir de estos metadatos. Con estos mensajes, se generan documentos utilizando la información que contienen, mediante una representación adecuada.
3. Se identifican los subtópicos de cada evento a partir de los documentos, generando clusters de documentos.
4. Finalmente, se utiliza la información social de los documentos (obtenida a partir de los medios sociales) para generar una medida de relevancia, y poder seleccionar los k documentos más relevantes.

Se utilizaron las API de Google News y de Last.fm para la obtención de eventos: noticias y conciertos, respectivamente. Para el enriquecimiento de los eventos se utilizó la información social que provee la red social Twitter y su API de búsqueda de *tweets*. De la misma forma, se consideraron los metadatos de los mismos mensajes para medir la relevancia de los documentos generados.

Un documento es identificado por la URL que lo ubica en la Web. El contenido no es más que la concatenación de los tweets que mencionan al documento. Se realizó una limpieza y preproce-

samiento de los datos, quitando las *stopwords* y realizando *stemming* sobre el contenido en texto. Luego, se aplicó *tf-idf* sobre los documentos, representándolos como vectores en el *space vector model*. Para identificar los subtópicos de un evento se utilizó el algoritmo de clustering k-means sobre los vectores.

Para el ranking de los documentos se decidió usar una ponderación simple sobre una serie de indicadores que dependen de los tweets y de las URLs de cada evento.

Entre las herramientas utilizadas, se usó lenguaje de programación Python, varias librerías para el manejo de datos (tales como `nltk`, `scipy`, `scikit-learn`, por nombrar las más importantes), el sistema de almacenamiento Redis, entre otras herramientas que son mencionadas en la descripción detallada de la solución.

En la Sección 4.1 se describe el modelo utilizado para la implementación, con una especificación detallada de la solución. En la Sección 4.2 se describe la metodología de desarrollo y la implementación práctica realizada para representar el modelo formal, la cual considera la obtención de datos dentro del proceso. Luego se comentan los problemas técnicos que fueron enfrentados para terminar en la Sección 4.4, donde se discuten un par de casos de estudio sobre los resultados obtenidos.

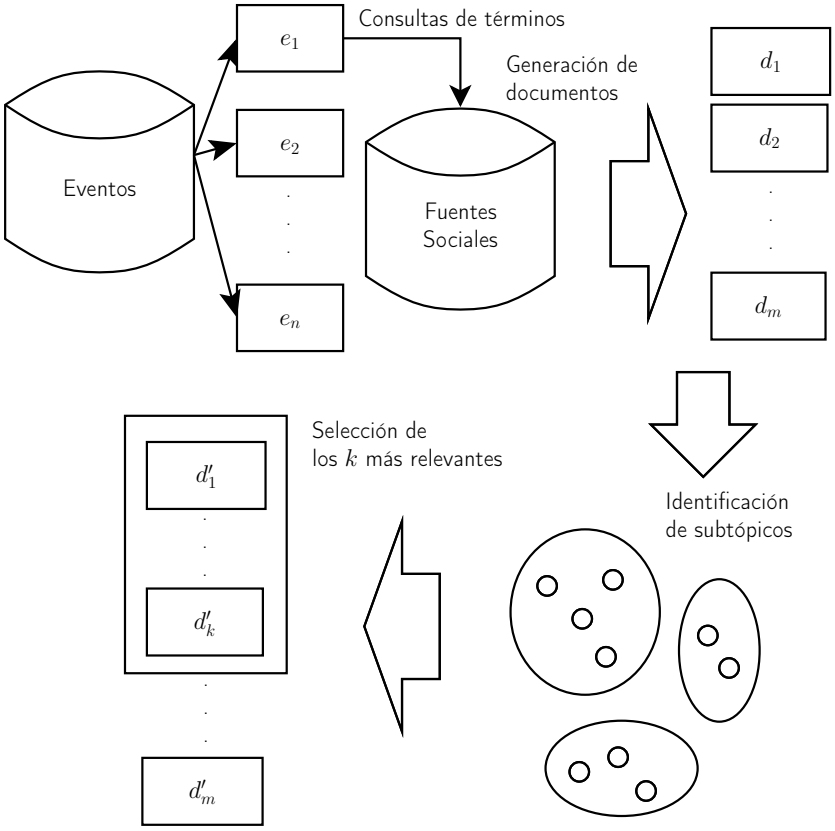

Figura 4.1: Vista general de la solución. En ésta se pueden apreciar las etapas de la metodología: (1) obtención de los eventos, (2) generación de los documentos a partir de mensajes obtenidos de medios sociales, (3) identificación de subtópicos de cada evento, y (4) selección de los k más relevantes.

4.1. Descripción detallada

Para enfrentar el problema descrito se decidió utilizar una representación apropiada de los documentos que permita abstraerse de su contenido, utilizando la información social asociada a éstos. Se decidió que el problema de identificación de eventos está fuera del alcance de este trabajo, por lo cual, se asume que los eventos son dados como input a la solución diseñada. A partir de un evento determinado, se identifican los subtópicos del evento utilizando los documentos, y luego, para cada subtópico se determinan los documentos más relevantes utilizando esta información social.

Para esto, fue necesario contar con dos *fuentes de datos*: una fuente de eventos y otra de *contenido social*, en la forma de mensajes y actualizaciones de estado.

Se asumió que estas fuentes satisfacen los siguientes requerimientos:

- La fuente de eventos debe entregar una lista de eventos rápidamente, la cual debe contener los siguientes datos para cada entrada:
  - Un título del evento y *términos asociados*. Los términos asociados son breves frases o palabras que describan al evento, como por ejemplo, tags o etiquetas.
  - Como datos opcionales: breve descripción del evento, fecha de inicio y término, ubicación y direcciones Web.
- La o las fuentes de contenido social deben entregar una lista de mensajes, con algunos metadatos tales como la fecha de creación, si el mensaje fue compartido, etc. Además, algunos datos sobre el autor del mensaje, como la cantidad de conexiones en la red, y en general, datos que permitan comparar dos autores.

Utilizando estas dos fuentes, el siguiente paso luego de obtener una lista de eventos fue enriquecerlos utilizando las fuentes sociales, generando documentos del tipo $\mathrm{d} = (s_1, s_2, \ldots, s_m)$, donde s_i, $\mathrm{i} \in [1..m]$ es un mensaje de alguna fuente social, con los metadatos asociados. El documento es identificado por la URI de algún documento en la Web, de forma que todos los mensajes que contengan una URI en particular, corresponderán al mismo documento d.

A continuación, utilizando alguna representación adecuada (*vector space model*, *bag of words*, etc.), se generaron clusters de documentos de un mismo evento, identificando los subtópicos. Con ellos fue posible generar un resumen que abarcara todos los aspectos del evento, en contraste con seleccionar directamente los documentos más relevantes del evento en su conjunto, lo cual puede dejar puntos de vista sin ser considerados por su extensión.

Finalmente se seleccionaron los $k > 0$ documentos más representativos de cada cluster, utilizando como criterio los metadatos de los mensajes de la fuente social. De esta forma, se ordenan los documentos dejando como más "relevantes" los que más interés atrae de los usuarios.

4.2. Metodología de desarrollo e implementación

La implementación consistió en las siguientes etapas:

- Obtención del dataset de eventos y documentos.

  Para esto, se utilizaron dos fuentes de eventos: Google News y Last.fm para recolectar noticias y conciertos musicales (incluidos festivales), respectivamente. Como fuente de datos sociales se utilizó la red social Twitter, que dispone de una API para realizar búsquedas por *keywords*. Esta etapa comprendió la recolección de eventos y de documentos con información social asociada a éstos.
- Generación de clusters para la identificación de subtópicos para cada evento.

  Una vez identificados los eventos y los documentos asociados, se generaron clusters usando el algoritmo K-means con K-means++ para la inicialización. Se impuso un valor de $k = 5$ clusters por evento. En la sección de casos de estudio se discuten las alternativas y la evaluación de algunos clusters del dataset.
- Extracción de documentos relevantes para cada evento.

  Una vez identificados los subtópicos de cada cluster, se extrajeron de éstos los documentos más relevantes, utilizando la información social de cada uno de ellos, en conjunto con otros indicadores globales del clustering (como que incluyan una URL dentro de las más mencionadas dentro del cluster, entre otras). Estos documentos corresponden al output o salida del sistema.

4.2.1. Obtención de datos

Se describe a continuación el proceso diseñado para la obtención de datos, tanto de eventos como de documentos con sus indicadores sociales respectivos.

Las etapas de generación del dataset fueron las siguientes:

- Recolección de eventos (noticias y conciertos);
- Enriquecimiento de los eventos existentes mediante tweets; e
- Identificación de documentos a partir de los tweets por cada evento.

Se recolectaron datos (eventos y tweets) desde el 19 de noviembre de 2012 hasta el 30 de noviembre del mismo año, todos los días desde la medianoche hasta que el proceso termina exitosamente.

Recolección de eventos

Se consideraron dos tipos de eventos para el sistema: noticias y conciertos musicales. Los conciertos incluyen festivales de varios artistas.

- **Noticias**

  Para obtener las noticias, se utilizó el servicio de Google News. Existe una API (en proceso de obsolescencia[1], pero funcional a la fecha de este trabajo) que permite obtener no sólo los titulares y breve descripción de cada noticia, sino también un conjunto de entre 4-10

[1] http://googlecode.blogspot.com/2011/05/spring-cleaning-for-some-of-our-apis.html

noticias relacionadas de otras fuentes. Esto sirvió para alimentar los términos de búsqueda para la etapa siguiente. Se guardaron los siguientes datos de una noticia:
  - Título,
  - Descripción,
  - URL de la fuente, y
  - Titulares de las noticias relacionadas.
- **Conciertos**

  Utilizando el servicio de Last.fm para obtener los conciertos y festivales de una ubicación en particular[2], se obtuvieron los conciertos y festivales de las siguientes ubicaciones:
  - Santiago, Chile;
  - Londres, Inglaterra;
  - Glastonbury, Inglaterra;
  - Las Vegas, Nevada, EE.UU.; y
  - Estocolmo, Suecia.
- Título del evento (concierto o festival);
- Artistas que participan; y
- Fechas de inicio y término (esta última no siempre está como dato).

  Además de otros datos descriptivos, como la ubicación, descripción breve, sitio web de la banda o festival, etc.

Cada vez que se obtienen los eventos se vuelven a obtener los conciertos, pero sólo agregando los nuevos. Las noticias siempre son nuevas, aun así por implementación no se consideraron los repetidos.

Enriquecimiento de eventos

Se obtuvieron tweets utilizando el servicio de búsqueda que provee Twitter en su API[3]. El objetivo es enriquecer los eventos con la información social que hay en la Web sobre éstos.

Para cada uno de los eventos obtenidos en la fase anterior, se utilizaron los términos de búsqueda asociados a ellos: los titulares de las noticias relacionadas y los nombres de los artistas para los eventos noticiosos y musicales, respectivamente.

- Para las noticias, se hace una búsqueda en Twitter de los titulares al mismo tiempo en que se obtienen de Google News, y nuevamente al día siguiente, es decir, 2 búsquedas por cada titular de un evento. Se quitan las tildes y caracteres no `ASCII` y las stopwords, para evitar problemas con la implementación y no hacer calce de stopwords en la búsqueda de Twitter.
- Para los conciertos y festivales, se utilizaron los nombres de los artistas y del evento como términos de búsqueda. De acuerdo a la información asociada al evento, se busca por una mayor cantidad de días:
  - Se busca desde un día antes de inicio del evento;

[2] http://www.lastfm.es/api/show/geo.getEvents
[3] https://dev.twitter.com/docs/api/1.1/get/search/tweets

- Si está presente la fecha de término del evento, se busca cada día dentro del intervalo "fecha de inicio" a "fecha de término" hasta tres días terminado el evento.
- Si no está presente la fecha de término (por ejemplo, un concierto o un festival de un día), se busca hasta tres días pasada la fecha de inicio.

Identificación de documentos a partir de tweets

Luego de obtener los tweets asociados a cada evento, el siguiente paso fue generar los documentos que fueron usados para la generación de los resúmenes. Nuevamente, el modelo consistió en que cada documento se modeló como un vector de palabras, donde el identificador del documento es una URL, y sus componentes corresponden al contenido de los tweets que tienen esa URL en el texto del mensaje.

El caso en el que un tweet no tenía ninguna URL en su contenido fue abordado de la siguiente forma: la URL asociada es una tal que representa al mismo tweet (utilizando el servicio de Twitter), y el contenido de ese documento es el mismo tweet, de forma de no dejar el tweet sin ser representado.

Este proceso fue abordado recorriendo todos los eventos del dataset, observando todos los tweets asociados a cada evento, extrayendo la URL si es que hay alguna y guardando el documento con el nuevo tweet. Se marcan los tweets observados para no tener que repetir el proceso, ya que es intensivo en conexión a la red.

Dada la condición breve de los mensajes publicados en la red social, muchos de los usuarios y/o servicios que publican mensajes con una URL en su interior suelen utilizar *acortadores* (o *url shorteners*) para los enlaces, y así no utilizar mucho espacio dentro de un mensaje. Otra ventaja que ofrecen es que algunos servicios como Bit.ly[4] dan estadísticas sobre los visitantes a estos enlaces (y así saber quiénes vienen de cierta red social u otra, por ejemplo). Twitter, a su vez, actualmente también ofrece acortamiento de URLs por defecto. Esto suele producir que un enlace acortado se resuelva a otro enlace también acortado, por lo que es necesario resolver la URL completa para evitar duplicados o *pseudo-duplicados* (en el caso en que dos URLs sintácticamente distintas apunten al mismo recurso). En la Figura 4.2 se puede apreciar un ejemplo de cómo estos enlaces pueden apuntar al mismo recurso, por lo cual hay que resolverlos completamente para evitar pseudo-duplicados.

Por lo anterior, una vez identificada la URL del texto de un tweet, se resuelve su URL completa (que puede ya serlo de antemano), lo que consume recursos de ancho de banda y tiempo.

Una vez identificada la URL, si existía el documento previamente, se añade el tweet a éste, sino se crea un nuevo documento cuyo identificador es la URL encontrada:

```
Event: Kamelot
Event type: Concert
Document URL: http://www.youtube.com/watch?v=pkiZOTc_tTw&feature=related
Document content:
```

[4] http://bit.ly

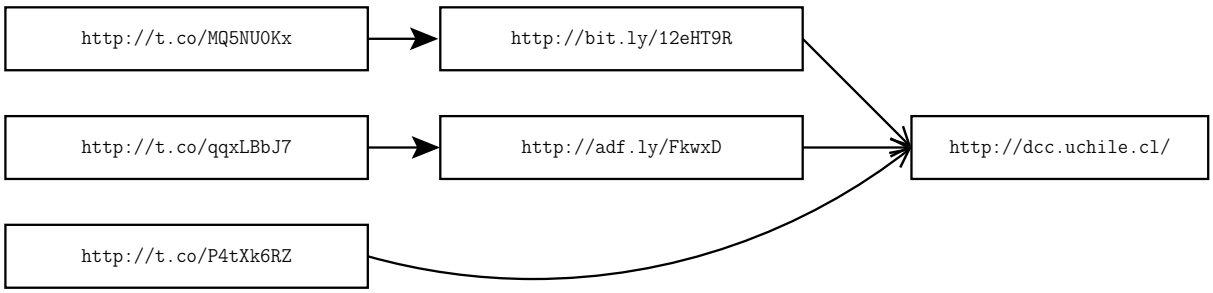

Figura 4.2: Ejemplo de enlaces acortados. Los dos primeros se resuelven primero a otros enlaces cortos que finalmente se resuelven a un nodo terminal (que se resuelve a su misma URL). El tercer link se resuelve directamente al nodo terminal.

```
['#np Kamelot - Can You Remember? :http://t.co/s7MUh7VW',
 'RT @El__Azar: #np Kamelot - Can You Remember? :http://t.co/s7MUh7VW']
```

4.2.2. Identificación de subtópicos

Una vez recolectados tanto los eventos como generados los documentos asociados, se procedió a identificar los subtópicos de cada evento. Para esto, se utilizó el algoritmo K-means para construir clusters a partir de todos los documentos de un solo evento.

Como se mencionó anteriormente, los documentos consisten en vectores del tipo $\mathrm{d}_{\mathrm{URL}} = (t_1, t_2, \ldots, t_m)$, donde t_i es el tweet i-ésimo que contiene a URL dentro del texto del mensaje. Para poder aplicar un algoritmo de clustering, fue necesario procesar nuevamente estos documentos para representarlos como vectores usando el vector space model. El procedimiento consta de dos partes, "normalizar" los documentos, limpiando los términos que puedan afectar al clustering, y luego aplicar tf-idf sobre los documentos normalizados:

Algoritmo 3: Preprocesamiento de documentos

Data: Conjunto de documentos D_e
Result: Conjunto de strings D'_e,

$D'_\mathrm{e} \leftarrow \emptyset$;
for *documento* $\mathrm{d} \in D_\mathrm{e}$ **do**
  $\mathrm{d}' \leftarrow \varepsilon$;
  for *tweet* $t \in \mathrm{d}$ **do**
    $t' \leftarrow \texttt{clean}(t)$;
    $\mathrm{d}' \leftarrow \texttt{concat}(\mathrm{d}', t')$;
  $D'_\mathrm{e} \leftarrow D'_\mathrm{e} \cup \{\mathrm{d}'\}$;

Donde ε es el string vacío, $\texttt{concat}(a, b)$ retorna la concatenación de a y b, y `clean` realiza las siguientes operaciones sobre el tweet:

- Remueve las URLs que contenga el texto;
- Remueve todas las menciones;

- Quita los caracteres "#", dejando los *hashtags* intactos;
- Quita las tildes, acentos, stopwords; y
- Realiza stemming en español o inglés dependiendo del idioma del evento dado por los metadatos de éste. Se utilizó el Snowball Stemmer[5] para esto.

Una vez normalizados los documentos, se convierten a la representación como vectores con pesos por cada término:

Algoritmo 4: Transformación de documentos a vector space model

Data: Conjunto de strings D'_e, vocabulario V de palabras de D'_e
Result: Conjunto de vectores D''_e, representados en vector space model

```
D''_e ← ∅ ;
for documento d ∈ D'_e do
    d' ← map(tf-idf(·, d, D'_e), words(d));
D''_e ← D''_e ∪ {d'};
```

Donde $\texttt{map}(f, l)$ mapea la función o procedimiento f a cada elemento de la lista l, retornando una nueva lista l' en la cual a cada elemento se le aplicó f y `words` retorna una lista con las palabras de d'. Este procedimiento retorna un conjunto de vectores en tf-idf, lo que permite entonces aplicar un algoritmo de clustering para identificar subtópicos.

Se utilizó el algoritmo K-means para generar los clusters. Al aplicarlo sobre el conjunto de vectores, éste retorna una *lista de etiquetas* L, de largo n, la cantidad de documentos utilizados. Cada $L_\mathrm{i} \in [0..k-1], \mathrm{i} \in [1..n]$ indica a cuál cluster corresponde el i-ésimo documento, lo que permite fácilmente filtrar y recuperar los clusters por separado, para pasar a la siguiente etapa.

4.2.3. Ranking de documentos

Para generar un ranking de documentos, se procedió a obtener toda la información relevante de éstos. Se escogió un conjunto de indicadores según [6] y las conclusiones de [11], en las cuales se menciona que la *popularidad* de un autor de un tweet, el largo de un tweet y si éste contiene o no una URL, son los mejores indicadores para medir la relevancia de un tweet, según los experimentos llevados a cabo.

Los indicadores escogidos se pueden apreciar en la Tabla 4.1. En el Código 1 se puede apreciar un ejemplo de la información obtenida para un documento.

[5] http://snowball.tartarus.org/

Indicador	Descripción
TWEETS	Número de tweets del documento
IS_RETWEETED	Número de tweets que son un retweet de otro
RETWEETS	Número de retweets de los tweets del documento
TWEET_LENGTHS	Largos de los tweets del documento, en cantidad de palabras
USER_VERIFIED	Número de tweets cuyo autor está verificado por Twitter
USER_FOLLOWERS	Número de usuarios que siguen a cada autor
USER_LISTS	Número de listas en las cuales están estos autores
USER_STATUSES	Número de tweets que han escrito los autores
USER_FRIENDS	Número de usuarios que siguen y son seguidos por los autores
USER_GEO	Número de autores cuyos tweets tienen ubicación habilitada
USER_CREATED	Fechas de creación de las cuentas (en formato Unix)

Tabla 4.1: Indicadores sociales utilizados para medir relevancia. Todos los indicadores, a excepción de la fecha de creación del usuario, son sumas entre todos los tweets de un documento.

4.3. Desafíos técnicos

4.3.1. Restricciones de la API de Twitter

La API de búsqueda de Twitter permite obtener tweets de acuerdo a un término de búsqueda. Se utilizó este servicio para enriquecer los eventos con información social utilizando como términos de búsqueda tanto los títulos de las noticias como los nombres de los artistas para las noticias y los conciertos, respectivamente.

Funciona de la siguiente forma: cada vez que se hace un request a la URL dada por el servicio, éste retorna a lo más 100 tweets por página, con un máximo de 15 páginas (indicando en el request qué página queremos consultar), dando como total hasta 1500 tweets por búsqueda. Existirán términos de búsqueda que no presenten ningún resultado (ya sea por estar mal escritos o simplemente que no sean un tópico de discusión), o por el contrario, que se generen más tweets que los retornados por la búsqueda por cada ventana de tiempo que demore ésta (por ejemplo, un *trending topic* o tópico que sea muy mencionado en la red social).

Existe una limitación de uso de este servicio: sólo es posible hacer hasta 180 requests por cada 15 minutos, o 1 request cada 5 segundos. Además, sólo retorna tweets de hasta 7 días de antigüedad, y sus resultados no son necesariamente en tiempo real y su estabilidad varía de acuerdo a factores externos.

Los tweets retornados vienen en formato `JSON` (*Javascript Simple Object Notation*), e incluyen varios metadatos sobre el tweet aparte de los principales, como autor, fecha, contenido. Algunos de estos metadatos son:

- Cantidad de *retweets* hechos hasta la fecha;
- Si posee alguna URL o *hashtag* en el texto;
- Si es una *mención* a otro usuario;
- La ubicación de donde se envió el tweet;

```
{
''850a0f7e08e9cf2e080678679857eef9'': {
  ''domain'': ''twitter'',
  ''is_retweet'': [0, 0, 0, 0, 0, 0],
  ''num_tweets'': 6,
  ''retweets'': [''0'', ''0'', ''0'', ''0'', ''0'', ''0''],
  ''tweets_lengths'': [8, 6, 8, 6, 6, 8],
  ''url'': ''https://api.twitter.com/1/statuses/show.json?id=270299718540738560'',
  ''user_created_at'': [1309224895.0,
                        1285052229.0,
                        1309224895.0,
                        1285052229.0,
                        1285052229.0,
                        1309224895.0],
  ''user_followers'': [''45'', ''37'', ''45'', ''37'', ''37'', ''45''],
  ''user_friends'': [''2'', ''25'', ''2'', ''25'', ''25'', ''2''],
  ''user_geo_enabled'': [1, 0, 1, 0, 0, 1],
  ''user_is_verified'': [0, 0, 0, 0, 0, 0],
  ''user_lists'': [''0'', ''1'', ''0'', ''1'', ''1'', ''0''],
  ''user_statuses'': [''12484'', ''4756'', ''12484'', ''4756'', ''4756'', ''12484'']}
}
```


Código 1: Información de un documento, correspondiente al evento "Anef anuncia movilización nacional". Los campos que corresponden a listas indican los valores para cada tweet del documento, en este caso, el documento tiene 6 tweets; por ejemplo, user_followers[23]=45 indica la cantidad de seguidores que tiene el autor del tweet en la tercera posición.

- etc.

Además, incluye datos sobre el autor, como por ejemplo:

- Si la cuenta está *verificada*;
- La cantidad de seguidores del usuario;
- Cantidad de amigos (seguidores que también lo siguen);
- Cantidad de tweets;
- Su descripción, y si incluye alguna URL, etc;
- Ubicación (dada por el mismo usuario);
- Fecha de creación de la cuenta;
- etc.

4.4. Casos de estudio

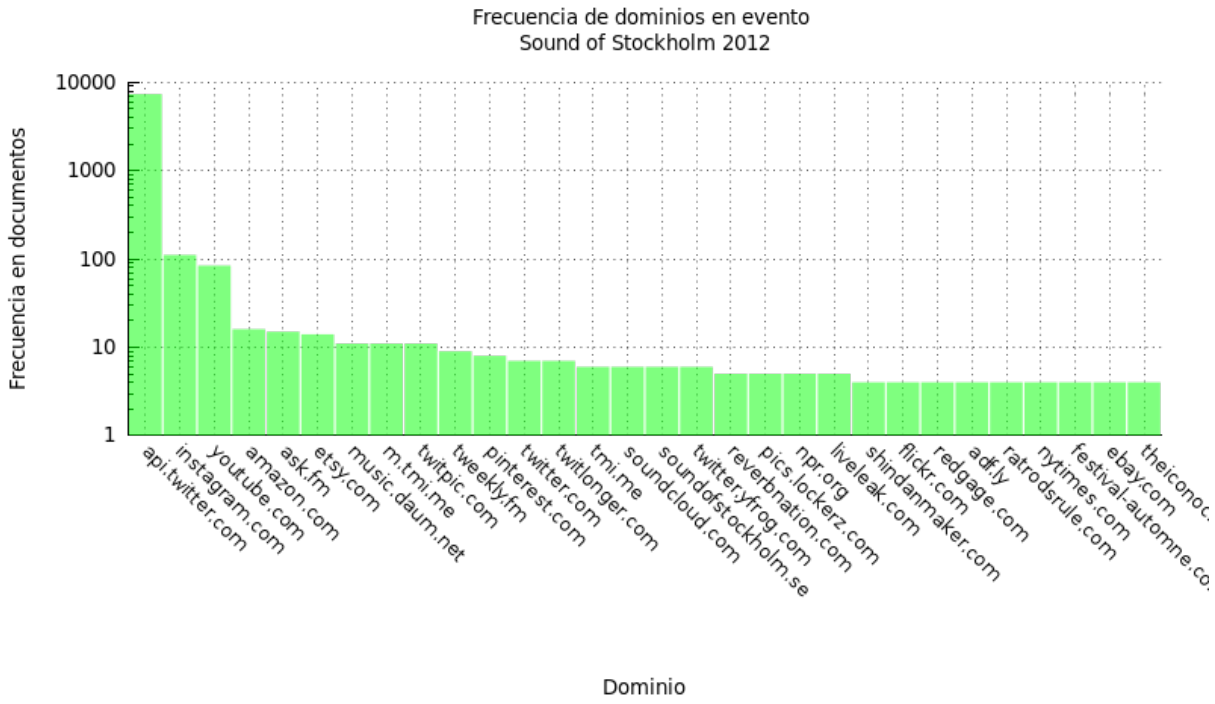

Figura 4.3: Timeline de Twitter. En éste se ve una lista en orden cronológico de los tweets generados por los usuarios que sigue el usuario actual. Además, el sitio incluye tweets promocionados por cuentas que pagan por dicho servicio, como es el caso del primer tweet en la lista.

Capítulo 5

Conclusiones

5.1. Resumen del trabajo realizado

5.2. Objetivos alcanzados

5.3. Relevancia del trabajo realizado

5.4. Trabajo futuro

Bibliografía

[1] J. Allan. *Topic Detection and Tracking: Event-Based Information Organization*. The Kluwer International Series on Information Retrieval. Kluwer Academic, 2002.

[2] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[3] Hila Becker, Dan Iter, Mor Naaman, and Luis Gravano. Identifying content for planned events across social media sites. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 533–542, New York, NY, USA, 2012. ACM.

[4] Hila Becker, Mor Naaman, and Luis Gravano. Learning similarity metrics for event identification in social media. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 291–300, New York, NY, USA, 2010. ACM.

[5] Felipe Bravo-Marquez and Manuel Manriquez. A zipf-like distant supervision approach for multi-document summarization using wikinews articles. In *SPIRE*, pages 143–154, 2012.

[6] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 675–684, New York, NY, USA, 2011. ACM.

[7] D. Chakrabarti and K. Punera. Event summarization using tweets. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pages 66–73, 2011.

[8] Jack G. Conrad, Khalid Al-Kofahi, Ying Zhao, and George Karypis. Effective document clustering for large heterogeneous law firm collections. In *Proceedings of the 10th international conference on Artificial intelligence and law*, ICAIL '05, pages 177–187, New York, NY, USA, 2005. ACM.

[9] Dipanjan Das and André F. T. Martins. A survey on automatic text summarization, 2007.

[10] Nicholas Diakopoulos, Munmun De Choudhury, and Mor Naaman. Finding and assessing social media information sources in the context of journalism. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 2451–2460, New York, NY, USA, 2012. ACM.

[11] Yajuan Duan, Long Jiang, Tao Qin, Ming Zhou, and Heung-Yeung Shum. An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 295–303, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[12] K. Ganapathiraju, Advisors Dr, Jaime Carbonell, and Dr Yiming Yang. Relevance of cluster size in mmr based summarizer: A report 11-742: Self-paced lab in information retrieval.

[13] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 71–78, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[14] Xueliang Liu, Raphaël Troncy, and Benoit Huet. Using social media to identify events. In *Proceedings of the 3rd ACM SIGMM international workshop on Social media*, WSM '11, pages 3–8, New York, NY, USA, 2011. ACM.

[15] S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, September 2006.

[16] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 1177–1178, New York, NY, USA, 2010. ACM.

[17] A. Signorini. A survey of ranking algorithms. 2005.

[18] Yiming Yang, Jaime G. Carbonell, Ralf D. Brown, Thomas Pierce, Brian T. Archibald, and Xin Liu. Learning approaches for detecting and tracking news events. *IEEE Intelligent Systems*, 14(4):32–43, July 1999.

[19] Ying Zhao and George Karypis. Criterion functions for document clustering: Experiments and analysis. Technical report, 2002.

Anexos