UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA PARA IDENTIFICAR TÓPICOS DE INTERÉS DE USUARIOS CHILENOS EN TWITTER

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

SEBASTIÁN LEONARDO CAMINO ALCALDE

PROFESOR GUÍA:
JUAN DOMINGO VELÁSQUEZ SILVA

MIEMBROS DE LA COMISIÓN:
FELIPE ESTEBAN VILDOSO CASTILLO
ALBERTO RAÚL CABEZAS BULLEMORE
ROCÍO BELÉN RUIZ MORENO

SANTIAGO DE CHILE
2016

RESUMEN DE LA MEMORIA PARA OPTAR AL
TITULO DE: Ingeniero Civil Industrial
POR: Sebastián Leonardo Camino Alcalde
FECHA: 25/11/2016
PROFESOR GUÍA: Juan Domingo Velásquez Silva

DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA PARA IDENTIFICAR TÓPICOS DE INTERÉS DE USUARIOS CHILENOS EN TWITTER

El objetivo general de esta memoria de título es diseñar e implementar un sistema de User Interest Modeling, que sea capaz de identificar tópicos de interés de usuarios chilenos en Twitter. Este trabajo se desarrolla dentro del marco del proyecto OpinionZoom, que es un proyecto de I+D aplicada concursado por InnovaChile de CORFO y dirigido por el Web Intelligence Centre de la Universidad de Chile. El proyecto busca generar un sistema avanzado de análisis de datos extraídos desde redes sociales para obtener información relevante para las instituciones y empresas en relación a sus productos y servicios.

La información obtenida a partir de los usuarios en las redes sociales puede tener muchos usos. Uno de éstos es caracterizar a los usuarios e identificar sus tópicos de interés. Contar con esta información puede ayudar a las organizaciones a conocer mejor a sus clientes, lo que les permitiría tomar mejores decisiones. Los métodos más utilizados para identificar tópicos de interés en Twitter usan el contenido generado por el usuario a caracterizar, sin embargo este enfoque conlleva un problema: la gran mayoría de los usuarios no *tweetea* o lo hace muy poco. Esto significa que los métodos que utilizan este enfoque no podrán identificar los tópicos de muchos usuarios y varios tendrán resultados deficientes, por lo que se necesita un enfoque distinto.

La hipótesis de investigación de este trabajo dice que es posible obtener tópicos de interés de usuarios chilenos de Twitter a partir de sus conexiones en la red social y sin utilizar el contenido generado por ellos, es decir sin utilizar sus *tweets*.

El sistema desarrollado se basa en la metodología propuesta por Bhattacharya *et al.*, pero enfocado en el Español para caracterizar de mejor forma a los usuarios chilenos. Este sistema utiliza la información de las listas de Twitter, para inferir los tópicos de influencia de usuarios populares de la red social, para luego inferir transitivamente los tópicos de interés de los usuarios que los siguen. Está compuesto de 4 módulos principales: el primero se encarga de extraer los datos de Twitter; el segundo procesa el texto de las listas e identifica los tópicos que las caracterizan; el tercero identifica los tópicos de influencia de los usuarios populares de Twitter; finalmente, el último módulo identifica los tópicos de interés agregando la información de los tópicos de influencia.

Se utilizó el sistema para identificar tópicos de interés de algunos usuarios de Twitter y se validó la hipótesis de investigación, ya que el 97 % de los usuarios evaluados se consideró representado por los tópicos identificados. Con esto, se tiene un sistema capaz de identificar tópicos de interés en español de la gran mayoría de usuarios de Twitter, con alta precisión y mejores resultados que la situación actual.

A Fran Urmeneta.

Agradecimientos

A Pipe Vildoso, por todo el apoyo durante el desarrollo de la memoria y la buena onda.

A Juan Velásquez, por todas las oportunidades y la confianza.

A mis compañeros del WIC: Gaspar Pizarro, Rominna Jiménez, Rocío Ruiz, Felipe Vera, Yerko Covacevic, Pangui, Ignacio Díaz, Andrés Córdova, Pía Andrioletti, Felipe Maldonado y Gonzalo Falloux, quienes hicieron que el proceso haya sido más llevadero y me ayudaron cada vez que lo necesité.

A mi familia, por la paciencia y por no dejar de apoyarme durante este largo camino.

A mis amigos más cercanos y a los que el tiempo alejó.

Por último, a quién me acompañó durante toda mi vida universitaria y me dio fuerzas para seguir adelante, Fran Urmeneta.

Tabla de contenido

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

El acceso a internet en el mundo es cada vez mayor. En consecuencia, el número de usuarios en este medio ha crecido muy rápidamente, en promedio un 900,4 % entre el año 2000 y 2016, alcanzando una penetración del 49,2 % o 3,61 mil millones de usuarios [1]. En Chile este número asciende al 77,8 % o 14,1 millones de usuarios en el 2016 [2]. Este rápido crecimiento se debe, en parte, a la Web 2.0, dónde los usuarios ya no son simples receptores de contenido, sino que pueden interactuar con los sitios que visitan y crear contenido.

En América Latina, y particularmente en Chile, los sitios web más visitados son Google, redes sociales y medios informativos [3]. En el apartado de las redes sociales, Chile es el líder en América Latina, con un 99 % de penetración. Los países que lo siguen son Brasil (89 %), México (87 %) y Argentina (77 %) [4].

Algo que caracteriza a los sitios de redes sociales es que los mismos usuarios generan contenido. Este contenido puede tener muchas formas, como expresar una opinión o juicio, compartir una foto o video, comunicarse con alguien conocido, entre otros. Lo importante de esto es que hay muchos datos en la web y de libre acceso, de donde se puede extraer información valiosa. En particular, se pueden obtener los tópicos de interés de los usuarios de las redes sociales, lo que puede ayudar a las empresas e instituciones a acercarse más a las personas. Esto puede ayudar a la gestión, ayudando a tomar mejores decisiones en temas de campañas de marketing, diseño de productos y servicios, entre otros.

1.1 Antecedentes

El trabajo desarrollado en esta memoria es parte del Proyecto OpinionZoom, financiado por INNOVA CORFO, el que es realizado por el Web Intelligence Centre (WIC[1]), perteneciente a la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile.

[1] `http://wic.uchile.cl`

1.1.1 WIC

El WIC es un centro de investigación de la Universidad de Chile que se especializa en inteligencia web y la utilización de herramientas de *Data Science* para resolver problemas.

En el WIC se realiza investigación aplicada en el área de Web Intelligence, y se cuenta con numerosas publicaciones en revistas científicas internacionales. La docencia es otro foco del centro, donde se dictan cursos de orientación práctica acerca de las Tecnologías de Información y su aplicación en los negocios.

El WIC es miembro del Web Intelligence Consortium[2], el que agrupa distintos centros de investigación y desarrollo en Inteligencia Web de todo el mundo.

En el sitio web del WIC se puede ver su misión, su visión y los objetivos que se propone:

Misión

> Desarrollar investigación de frontera en el campo de Tecnologías de Información creando nuevas soluciones para abordar problemas complejos de ingeniería utilizando herramientas basadas en la Web de las Cosas.

Visión

> Ser un líder a nivel internacional en la investigación de tecnologías de información y comunicaciones aplicadas a la resolución de problemas del mundo real.

Objetivos

- Publicar en las principales revistas, conferencias y editoriales relacionadas con Web Intelligence.
- Proveer un servicio profesional, excelente y rápido para todos nuestros clientes.
- Dictar cursos de orientación práctica acerca de las Tecnologías de Información y su aplicación en los negocios.

1.1.2 OpinionZoom

Es un proyecto de investigación aplicada, financiado por INNOVA CORFO y adjudicado por el WIC, con código 13IDL2-23170. El título del proyecto es:

> OpinionZoom: Plataforma de análisis de sentimientos e ironía a partir de la información textual en redes sociales para la caracterización de la demanda de productos y servicios.

Como dice Córdova [5], OpinionZoom surge de la investigación de Marrese-Taylor *et al.*

[2] `http://wi-consortium.org/`, visitado el 14 de Abril del 2016

[6] sobre la aplicación del modelo de minería de opiniones basado en aspectos de Bing Liu [7] en la industria del turismo en Chile [8]. Para dicha investigación se construyó y utilizó una herramienta novedosa de minería de opiniones [9], cuyo impacto dio pie para que se concibiera un proyecto mayor.

El problema que OpinionZoom quiere resolver es el distanciamiento de las organizaciones y empresas en Chile con los ciudadanos y clientes, ayudándolos a acercarse más a ellos utilizando la información disponible en redes sociales, en particular en Twitter. En este sentido, OpinionZoom quiere ser un "Observatorio de Twitter", y entregarle a las organizaciones y empresas información que los ayude a conocer mejor a su mercado objetivo.

El factor diferenciador de OpinionZoom es su enfoque en Chile. Se han desarrollado herramientas específicas para el lenguaje que utilizan los chilenos, que no están presentes en la competencia.

Servicios de OpinionZoom

Los servicios que la plataforma de OpinionZoom se propone a ofrecer se agrupan en tres secciones: Inteligencia de clientes, *Trending Alert* e Impacto de campañas [10].

Inteligencia de clientes

- Identificación: ¿Quiénes y cómo son mis clientes? Algoritmo Líderes de Opinión.
- Conocimiento: ¿Qué les gusta? Algoritmo Interés Complementario.
- Escucha: ¿Qué y cómo están hablando de tu marca y la competencia? Métricas de Medición.

Trending alert

- Alertas de Temas Hot: reclamos puntuales, reclamos generalizados, contingencia.
- Seguimiento de reclamos mediante: análisis de opiniones ex post y generación de encuestas automáticas.
- Análisis de Trend LifeCycle: duración, peaks/valles, velocidades.

Impacto de campañas

- Medición de métricas antes/después de acción de marketing de tu marca y su competencia.
- Conceptos asociados a la campaña y la marca durante la campaña.
- Validación de Productos: Polaridad y conceptos asociados a un producto determinado. Uso de hashtag para capturar información.

– Otros servicios: Automatización de reportes personalizables de redes sociales. Apoyo en investigaciones científicas, etc.

El prototipo resultante de este trabajo de título se enfoca en el servicio de Inteligencia de Clientes, en la sección de Conocimiento y en particular en el Algoritmo de Interés Complementario. El servicio que es entregado a las instituciones es el saber que le interesa a algún segmento de usuarios, lo que puede tener múltiples usos dependiendo de la empresa y el rubro . La información es entregada de forma agregada, es decir sin identificar a un usuario determinado, sino que caracterizando a un grupo en su totalidad. Uno de los valores de OpinionZoom es el respecto por la privacidad e identidad de las personas.

1.2 Justificación

La información obtenida a partir del contenido generado por usuarios en la web puede tener muchos usos. Uno de éstos es caracterizar a usuarios de redes sociales e identificar sus tópicos de interés. Contar con esta información puede ayudar a las organizaciones a conocer mejor a sus clientes, lo que les permitiría tomar mejores decisiones.

Las aplicaciones que se le pueden dar a un conjunto de tópicos de interés de un usuario son variadas. Entre ellas se destacan:

Sitios web adaptativos: Son sitios web que se ajustan a la medida del usuario. Si se tienen los intereses, es posible modificar la estructura, el contenido y la presentación de la información de acuerdo a los gustos del usuario.

Búsquedas personalizadas: El orden o los resultados mostrados en una búsqueda puede ser personalizado considerando los intereses del usuario.

E-commerce personalizado: Los productos ofrecidos al usuario, en el sitio de e-commerce, son determinados por los intereses asociados a su perfil.

Sistemas de recomendación: Los sitios de recomendación en general pueden utilizar los intereses del usuario para realizar mejores recomendaciones.

Simulación de usuarios: Teniendo los intereses se puede simular el comportamiento de un usuario en un sistema.

Predicción de demanda: Si se tienen los intereses o gustos de una cierta cantidad de usuarios, sería posible predecir la demanda que tiene cierto producto, considerando las características que éste tiene.

Dado el potencial que tiene el contar con los intereses de usuarios, ha surgido un área de investigación llamada *User Interest Modeling*, donde se utilizan técnicas de análisis de datos, como Opinion Mining y Sentiment Analysis, y otras como Information Retrieval y Web Mining, para obtener perfiles o modelos de usuarios. También hay empresas que entregan

servicios relacionados, como Klout[3] e IBM[4]. Sin embargo, la mayoría de la investigación y APIs desarrolladas están enfocadas en el idioma inglés, lo que presenta un problema para OpinionZoom, ya que el proyecto está enfocado en usuarios chilenos y uno de sus factores de diferenciación es utilizar el español de Chile. Otro problema para OpinionZoom es que los servicios desarrollados por terceros están enfocados en el contenido generado por los usuarios de Twitter, lo que tiene varias falencias: muchos usuarios utilizan Twitter como un medio informativo [11]; se sabe que el 90 % del contenido de la red social es generado por el 10 % de los usuarios [12]; y que el 44 % de las cuentas de Twitter no tienen *tweets* [13]. Lo que esto genera es que los tópicos de interés que entregan los sistemas sean deficientes o tengan baja cobertura (no se obtienen tópicos de interés de todos los usuarios). Considerando estos problemas, surge la necesidad de un sistema capaz de identificar tópicos de interés en español y de usuarios de Twitter que no generen contenido.

Actualmente en OpinionZoom, para la obtención de los tópicos de interés de los usuarios, se utiliza una API de Klout. Los resultados entregados por esta API tienen varios problemas: los tópicos están en inglés; no existen resultados para todos los usuarios; baja variabilidad de tópicos (muchos tópicos repetidos); bajo poder de segmentación; alta frecuencia de tópicos que no son muy populares; y, por último, los resultados entregados corresponden a tópicos de influencia y no de interés. En resumen, el sistema en uso no entrega información útil.

Este proyecto se enfoca en crear una aplicación que sea capaz de obtener tópicos de interés de usuarios de Twitter de habla hispana, particularmente de Chile. Es importante que se puedan obtener resultados de usuarios que no *twiteen*, o lo hagan muy poco, dado que son la mayoría de la red social. El fin es dotar al proyecto OpinionZoom de un módulo de obtención de intereses más certero, más informativo y con mayor cobertura que el que se utiliza en la actualidad.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar e implementar un sistema de User Interest Modeling, que sea capaz de identificar tópicos de interés de usuarios chilenos en Twitter.

1.3.2 Objetivos específicos

1. Estudiar el estado del arte respecto a metodologías y técnicas de User Interest Modeling, User Profiling y obtención de tópicos de interés de usuarios en redes sociales.

2. Definir, desarrollar e implementar un sistema, a nivel prototipo, que permita la identificación de tópicos de interés de usuarios chilenos en Twitter, utilizando las mejores

[3]https://klout.com
[4]http://www.ibm.com/watson/developercloud/doc/personality-insights/basics.shtml

técnicas encontradas en el estudio del estado del arte.

3. Evaluar y validar los resultados del sistema desarrollado. Se analizará si los tópicos identificados efectivamente representan a los usuarios caracterizados y si estos son útiles para la plataforma de OpinionZoom.

1.4 Hipótesis de investigación

Se pueden obtener tópicos de interés de usuarios chilenos de Twitter a partir de sus conexiones en la red social. Los resultados obtenidos son mejores que los que se obtienen utilizando la API de Klout.

1.5 Resultados esperados

Los resultados esperados de esta memoria son:

- Un marco conceptual que incluye el estado del arte de las metodologías de User Interest Modeling, User Profiling y otras relativas a obtención de tópicos de interés de usuarios en redes sociales.
- Definición del mejor método que permite extraer tópicos de interés de usuarios hispanoparlantes en Twitter.
- Un sistema que entregue los tópicos de interés de un usuario de Twitter.
- Una base de datos que contenga la información y datos necesarios para que el sistema desarrollado pueda entregar los tópicos de interés de un usuario.
- Un reporte de mejoras a ejecutar para entregar más valor.

1.6 Alcances

El sistema desarrollado es un prototipo funcional. Lo más importante es demostrar que la metodología propuesta es factible, sin embargo hay mejoras que se pueden hacer para que el sistema sea más rápido y para que los resultados sean más precisos.

Con respecto a la cobertura, el sistema funciona para cualquier usuario. No obstante, el tiempo de ejecución puede ser alto si el usuario no ha sido procesado con anterioridad por el sistema.

En cuanto a la información entregada por el sistema, los tópicos entregados son específicos para el usuario consultado. Sin embargo, la plataforma de OpinionZoom le entrega la información a sus clientes de forma agregada, por lo que no hay problemas relativos a la

privacidad de las personas, al no entregar información con la que se pueda individualizar a un usuario. El formato utilizado para entregar los tópicos de interés de un usuario permite que la agregación de estos, para caracterizar a múltiples usuarios, se realice fácilmente.

1.7 Metodología

Para llevar a cabo los objetivos propuestos anteriormente, se deben llevar a cabo los siguientes pasos:

1. Estudio del estado del arte de User Interest Modeling, User Profiling y Obtención de intereses de usuarios en redes sociales. En esta etapa se busca documentar las distintas metodologías y técnicas existentes en la literatura para obtener los intereses de usuarios en redes sociales.

2. Definición y diseño del sistema. Utilizando las técnicas y métodos más adecuados se desarrollará un prototipo de la solución.

3. Desarrollo del sistema. Se creará el sistema para obtener los intereses de usuarios chilenos en Twitter, adoptando buenas prácticas y haciéndolo modular. De esta manera será más fácil de extender, mejorar y de usar.

4. Verificar resultados obtenidos. Los resultados que se obtienen de la plataforma deben ser verificados con información conocida.

1.8 Estructura del informe

El informe está compuesto por 6 capítulos. El actual, Capítulo 1, es el capítulo introductorio donde se plantea el problema, los objetivos y alcances del trabajo.

El Capítulo 2 consta del Marco conceptual e introduce los conceptos fundamentales para el desarrollo del trabajo. Además, muestra el estado del arte con respecto a la obtención de intereses de usuarios en redes sociales.

El Capítulo 3 muestra el diseño del sistema de identificación de tópicos de interés de usuarios de Twitter. En esta parte se explica qué es lo que se hizo, sin profundizar en cómo se llevó a cabo.

En el Capítulo 4 se describe en detalle el sistema desarrollado en esta memoria en un aspecto más técnico.

En el Capítulo 5 se realiza una comparación con la situación actual de OpinionZoom, luego se muestran los resultados de la encuesta realizada para validar el sistema, y se finaliza con una discusión sobre los resultados obtenidos.

El Capítulo 6 da a conocer las conclusiones, las que contienen un resumen de los resulta-

dos obtenidos, las implicancias de éstos y sus limitaciones. También se proponen mejoras y trabajo futuro.

Capítulo 2

Marco Conceptual

El objetivo de este capítulo es presentar una base conceptual de los elementos relevantes para esta memoria. Primero se da una descripción de la Web y las redes sociales, con un enfoque en Twitter. Luego se describe el proceso KDD, Minería de datos y Minería de texto, con énfasis en los componentes utilizados para este trabajo. A continuación se define lo relativo a los usuarios, perfiles y modelos de usuario. Posteriormente, se presentan distintos métodos de *User Modeling*, incluyendo el método que se utilizará. Por último, se describe los que son las APIs.

2.1 La Web

La World Wide Web, más conocida como WWW, fue propuesta en 1990 por Tim Berners-Lee [14]. La WWW se define como un sistema de documentos de hipertexto[1] relacionados entre sí y accesibles por medio de internet.

En sus inicios, lo que se conoce por la Web 1.0, era utilizada para compartir información mediante sitios y páginas estáticas. Las actualizaciones eran poco frecuentes. Por esta razón, las empresas la utilizaban para entregar información a sus clientes, pero no existía una interacción. Sumado a esta limitación de la Web, el acceso a internet era bastante restringido. En la actualidad la Web es mucho más amplia y es uno de los principales usos del internet.

2.1.1 Web 2.0

El término Web 2.0 fue creador por Tim O'Reilly, junto a sus compañeros de trabajo, en Octubre de 2004 [15]. Se utiliza para hablar de sitios y aplicaciones web en las que los usuarios son creadores de contenido y donde existe una interacción entre ellos, más

[1]El diccionario en línea de Merriam-Webster define hipertexto como una manera de ordenar información en una base de datos que permite a un usuario obtener información e ir de un documento a otro, mediante el uso de *clicks* en palabras destacadas o imágenes. Visitada el 23 de noviembre de 2016

allá del traspaso de información. En la Web 2.0, existe una interacción bidireccional entre el usuario y el sitio web, ya que el usuario puedo escribir contenido, entre otras cosas, y modificar el sitio. Por esta razón se considera que en la Web 2.0 los sitios web son dinámicos. Algunas de sus principales características son proveer una experiencia de usuario más completa, el énfasis en la participación del usuario, la escalabilidad y generación de contenido dinámico.

Los servicios que se generan a partir de este cambio en la web se pueden categorizar, de forma general, de la siguiente manera [16, 17]:

Blogs: Son diarios en línea, en los que usuarios escriben contenido en forma de bitácora. El contenido es mostrado de forma cronológica, comenzando desde el más reciente. Se suelen combinar con *podcasts*, esto es contenido digital como video o audio que puede ser descargado o transmitido a dispositivos móviles. Por ejemplo: Gizmodo (`http://gizmodo.com`), FayerWayer (`https://www.fayerwayer.com/`).

Redes sociales: Aplicaciones que permiten a usuarios comunicarse y compartir contenido con otros y tener un perfil que otros usuarios pueden visualizar. Por ejemplo: Facebook (`https://www.facebook.com`), Twitter (`https://twitter.com`).

Comunidades de contenido: Sitios web que organizan y comparten tipos particulares de contenido. Pueden estar construidas para compartir videos como Youtube (`https://www.youtube.com`), para compatir imágenes como Instagram (`https://www.instagram.com`), y enciclopedias editadas publicamente, mejor conocidas como wikis, como Wikipedia (`https://www.wikipedia.org`) y Wikia (`http://es.wikia.com/Wikia`).

Foros: Sitios para intercambiar ideas e información, usualmente enmarcados en un tema de interés particular. Por ejemplo Reddit (`https://www.reddit.com`), Yahoo Groups (`https://groups.yahoo.com/`).

Existe una serie de elementos o principios que las aplicaciones de la Web 2.0 deben seguir [16]:

1. Foco en soluciones basadas en servicios, en la simplicidad y de código abierto en la forma de aplicaciones en línea.

2. Desarrollo continuo e incremental de aplicaciones que requiera la participación de usuarios de una manera distinta, no sólo consumiendo sino que también contribuyendo, revisando y editando contenido.

3. Nuevos modelos de negocio basados en servicios y nuevas oportunidades para alcanzar a clientes individuales con productos de bajo volumen.

2.2 Redes sociales

Las redes sociales, o redes sociales en línea, se definen como servicios basados en la web que permiten a usuarios tener un perfil en la plataforma, que puede ser visitado por otros usuarios, e interactuar con otros usuarios con los que se tiene o no una relación [18]. Las relaciones pueden ser bidireccionales o unidireccionales, dependiendo de la plataforma. Un par de ejemplos notables de redes sociales son Facebook[2] y Twitter[3], que es la red social en la que se enmarca esta memoria.

2.2.1 Twitter

Twitter es una red social de *microblogging*[4] que fue lanzada en Julio de 2006. Ha tenido un crecimiento muy acelerado, llegando a ser el noveno sitio más visitado del mundo [19]. Se cuenta con 313 millones de usuarios activos al mes [20] y 100 millones de usuarios activos diarios [13].

Las cuentas de Twitter pueden representar más que sólo a una persona, pasando por empresas, organizaciones (tanto públicas como privadas), personajes ficticios, entre otros. Sin embargo, existe la opción de *verificar* la cuenta, para mostrar que esa cuenta representa realmente a la persona. Esto suele ocuparse para personajes públicos o famosos [21].

Las relaciones entre usuarios de Twitter son unidireccionales. Un usuario puede tener *seguidores*, usuarios que lo siguen, y a su vez puede seguir a otros usuarios, quienes son llamados *amigos*. Cada usuario dispone de un perfil, donde se muestran datos como una descripción del usuario, los últimos tweets, los amigos y el número de tweets emitidos, entre otros [11]. Este perfil es público por defecto.

El contenido generado por los usuarios de Twitter consta principalmente de texto. Ellos publican mensajes cortos (con un límite de 140 caracteres), los que se denominan *tweets*. Un tweet puede tener distintos objetivos, y para ello se utilizan distintas funciones y contenido. Los roles pueden ser de los siguientes:

- **Status:** Este es el tipo de tweet más simple, ya que corresponde a la publicación de contenido, o actualización de estado. El contenido es principalmente texto, pero puede incluir también imágenes, videos e hipervínculos.

- **Retweets:** En este tipo de mensaje, un usuario cita un tweet publicado por otro usuario. El mensaje muestra que se trata de un *retweet*, y el usuario puede hacer un comentario sobre el tweet original. El usuario que publicó el tweet original es notificado, y este tweet almacena un conteo de las veces que ha sido retwitteado.

- **Menciones:** Esta es una de las formas de las que disponen los usuarios para co-

[2] `https://www.facebook.com`

[3] `https://twitter.com`

[4] El diccionario en línea de Merriam-Webster define *microblogging* como blogging realizado con restricciones severas de tamaño o espacio, típicamente publicando frecuentemente mensajes breves sobre actividades personales. A su vez, blogging se define como publicar contenido en un Blog.

municarse entre ellos. Esto se realiza agregando al texto del tweet un arroba (@) y el nombre del usuario. Por ejemplo: @WWF. De esta forma le llega una notificación al usuario mencionado. Hay un tipo particular de menciones que se llama *respuesta*, que es cuando el tweet comienza con la mención al usuario, o utilizando el botón "responder", y donde se genera una conversación pública.

Otra forma que tienen los usuarios para comunicarse es mediante el uso de mensajes directos. Este tipo de mensajes sólo puede ser visto por los usuarios involucrados, es decir que es privado. Un usuario puede enviar este tipo de mensajes a cualquiera de sus seguidores [22].

Existe una función de Twitter que se llama *hashtag*, y que es usado para mencionar explícitamente que se está hablando de un tema en particular. Para ello se utiliza la almohadilla (#) anteponiéndose a una palabra, así la etiqueta se puede identificar de una manera más rápida tanto para usuarios como para el sistema, como por ejemplo #terremoto, #CASOPENTA, entre otros.

Los *hashtags* utilizados con más frecuencia, así como la palabras y frases, se categorizan temporalmente como *Trending Topics*. Estos se muestran a todos los usuarios en la página principal, con la posibilidad de hacer click en ellos y poder ver tweets que hagan referencia a esos tópicos.

Los usuarios pueden organizar a sus amigos utilizando las *listas* de Twitter. Una lista es un grupo de cuentas de Twitter, que posee un título dado por el usuario dueño de la lista y una descripción opcional. Un usuario puede crear sus propias listas o suscribirse a listas creadas por otros usuarios. Cuando un usuario selecciona una lista, se le muestran los tweets emitidos solamente por las cuentas que están en ésta [23]. Es por esta razón que el nombre y descripción de las listas suelen ser bastante descriptivos.

2.3 KDD y Minería de datos

La abundancia de datos y la facilidad del acceso a ellos hoy hacen que el Descubrimiento de Conocimiento y la Minería de Datos tomen particular importancia e incluso se conviertan en necesidad [24]. A continuación se explica en que consisten estos procesos.

2.3.1 Descubrimiento de Conocimiento en bases de datos

Más conocido por su nombre en inglés, *Knowledge Discovery in Databases* o simplemente KDD, es un proceso que permite descubrir conocimiento a partir de datos. En [25] se define KDD como: *"el proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles y comprensibles, en los datos"*. Esta definición implica que se pueden definir medidas cuantitativas para evaluar los patrones extraídos. En muchos casos, es posible determinar medidas de certeza o de utilidad. Es importante que el resultado pueda ser interpretado por el experto de negocio, en caso contrario el conocimiento extraído deja de ser útil.

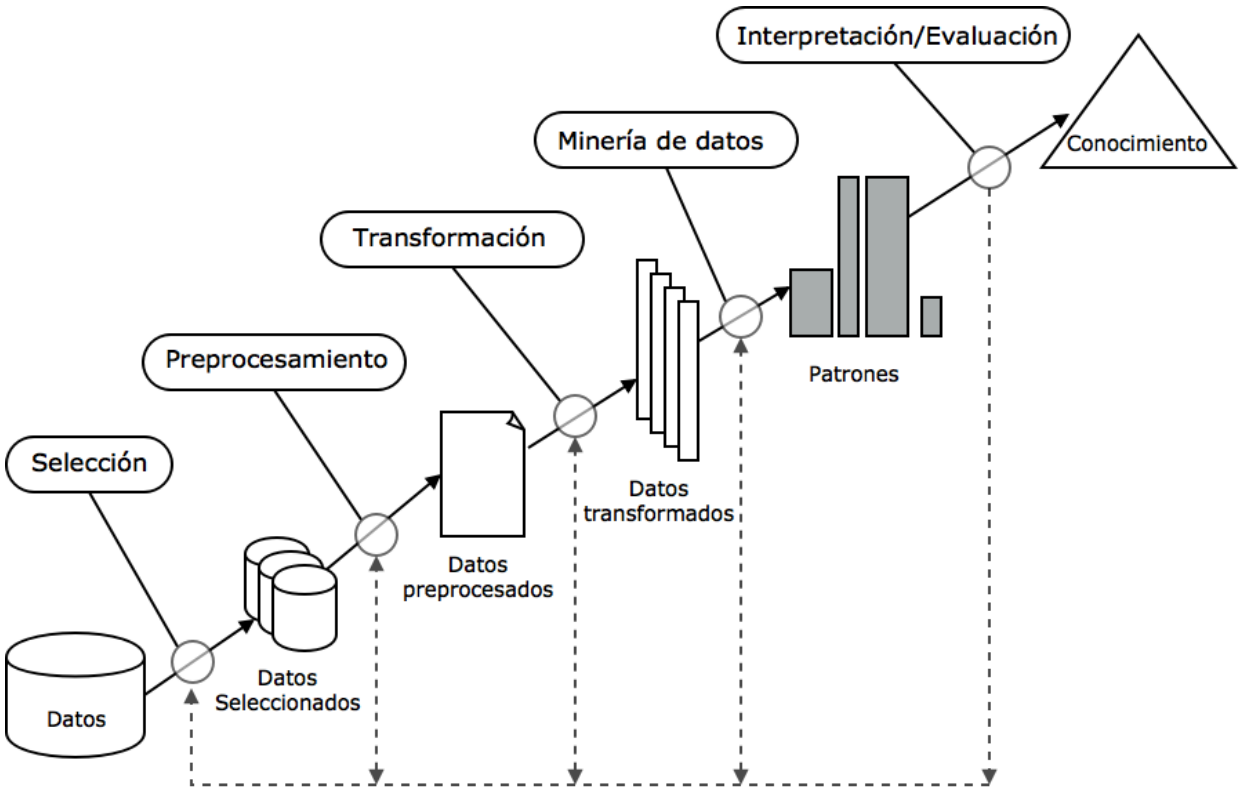

Figura 2.1: Proceso KDD

Fuente: Elaboración propia

La Figura 2.1 muestra las distintas etapas del proceso. Se aprecia que el proceso es iterativo en cada etapa, lo que significa que puede ser necesario volver a realizar ajustes en pasos anteriores. El proceso comienza por determinar los objetivos y termina con la implementación del conocimiento descubierto. A continuación se describe cada uno de los 9 pasos del proceso KDD [24]:

1. **Comprender el dominio de la aplicación:** Las personas a cargo del proyecto KDD deben entender y definir los objetivos del usuario final y el medio en el que el proceso se llevará a cabo. Mientras avance el proceso KDD, pueden haber revisiones y ajustes a este paso.

2. **Seleccionar y crear el set de datos:** Con los objetivos claros, se deben definir los datos que serán utilizados para el proceso KDD. Esto incluye el averiguar que datos están disponibles, obtener datos adicionales y luego integrar todo un un solo conjunto de datos. Este proceso es muy importante porque la etapa de Minería de Datos utilizará estos datos para trabajar. Es la base para construir los modelos y si hay atributos importantes que falten entonces el estudio puede fallar.

3. **Preprocesamiento y limpieza:** En esta etapa se mejora la confiabilidad en los datos. Se incluyen pasos de limpieza de datos, como manejar valores faltantes y remover el ruido y los valores atípicos o fuera de rango. Dependiendo del caso, esta etapa puede ser muy rápida o puede tomar la mayor parte del tiempo del proceso global. Se

pueden utilizar complejos métodos estadísticos e incluso algoritmos de minería de datos en este contexto.

4. **Transformación de datos:** En esta etapa se generan mejores datos para la minería de datos. Algunos métodos pueden incluir reducción de dimensiones y transformación de atributos. Este proceso es crucial para el éxito del proyecto y suele ser específico para cada uno.

5. **Seleccionar la tarea apropiada de Minería de Datos:** En esta etapa se debe decidir que tipo de Minería de Datos se utilizará, como regresión, clasificación o *clustering*. Esto depende de los objetivos del proceso. Hay 2 objetivos principales en Minería de Datos: verificación y descubrimiento, que a su vez se divide en predicción y descripción [25].

6. **Elegir el algoritmo de Minería de Datos:** Con el tipo de Minería de Datos a utilizar claro, se selecciona la técnica que será utilizada. Esta etapa incluye la selección del método específico que será utilizado para descubrir patrones.

7. **Aplicación del algoritmo de Minería de Datos:** Finalmente, se implementa el algoritmo de Minería de Datos seleccionado. En algunos casos puede ser necesario realizar este paso varias veces hasta que se obtenga un resultado satisfactorio, por ejemplo cambiando los parámetros del algoritmo.

8. **Evaluación:** Se evalúan e interpretan los patrones minados con respecto a los objetivos fijados en el primer paso. En este paso se consideran las decisiones tomadas en los pasos de preprocesamiento con respecto a su efecto en los resultados, con la posibilidad de volver y realizar cambios. El foco de este paso es la comprensibilidad y utilidad del modelo. El conocimiento descubierto es documentado para su posterior uso.

9. **Usar el conocimiento descubierto:** Ahora se puede incorporar el conocimiento a otro sistema para realizar otras acciones. El conocimiento se puede utilizar para hacer cambios al sistema y medir sus efectos. El éxito de esta etapa define la efectividad del proceso KDD en su totalidad.

2.3.2 Minería de Datos

La Minería de Datos es una de las etapas del proceso KDD, e involucra ajustar modelos o determinar patrones a partir de datos, lo que cumple el rol del conocimiento descubierto [25]. Que el modelo refleje conocimiento útil o interesante dependerá de el proceso KDD en su totalidad, y se requerirá de la interpretación de un experto.

Los objetivos de la Minería de Datos se definen de acuerdo al objetivo general del proceso de descubrimiento de conocimiento, y se pueden agrupar en 2 tipos: verificación y descubrimiento. En *verificación* el sistema se limita a verificar la hipótesis del usuario. En *descubrimiento* el sistema busca nuevos patrones de forma autónoma. A su vez, el objetivo de descubrimiento se puede dividir en *predicción* donde el sistema busca patrones para

predecir el comportamiento futuro de entidades, y *descripción*, donde el sistema busca patrones que describan la data y sean interpretables por un humano.

Existen 2 enfoques primordiales a la hora de ajustar modelos: estadístico y lógico. El *enfoque estadístico* permite en el modelo utilizar técnicas no deterministas, en cambio el *enfoque lógico* es puramente determinista. El enfoque estadístico es la aproximación que más se utiliza para aplicaciones prácticas de Minería de Datos, dada la presencia de incertidumbre en en los procesos de generación de datos del mundo real.

La frontera entre los métodos predictivos y descriptivos a veces no es clara (algunos modelos predictivos pueden ser descriptivos, hasta el punto que llegan a ser entendibles, y vice versa), sin embargo la distinción es útil para entender el objetivo general de descubrimiento.

2.4 Minería de Texto

La Minería de Texto, más conocido por su nombre en inglés *Text Mining*, se encarga del análisis de texto apoyado por computadores. Usa técnicas de Recuperación de Información, más conocido por su nombre en inglés *Information Retrieval*, y de Procesamiento de Lenguaje Natural (NLP) y los conecta con algoritmos y métodos de KDD, Aprendizaje de Máquinas y estadísticas [26]. En consecuencia, se utiliza un procedimiento similar al KDD, pero no de datos en general, sino que enfocado en el análisis de documentos de texto.

Para minar grandes colecciones de documentos, es necesario preprocesar el texto de éstos y almacenar la información en una estructura de datos que sea más apropiada para un procesamiento y análisis posterior.

2.4.1 Preprocesamiento de texto

Existen varios procesos de preprocesamiento de texto. Estos son necesarios para la posterior aplicación de algoritmos para extraer información del texto. Los siguientes son algunos de los más utilizados y los que serán empleados en este trabajo [26]:

- **Separación de *CamelCase***
  CamelCase es una notación en la que se escriben palabras compuestas donde se omiten los espacios y cada palabra empieza con una letra mayúscula. Es usado comúnmente en los *hashtags* y títulos de listas de Twitter o simplemente para utilizar menos caracteres. Un ejemplo sería escribir “CyberMonday” en vez de “Cyber Monday”. De esa manera se puede incluir todo en un mismo hashtag y ahorrar el caracter espacio. Es necesario separar estas palabras que estén escritas de esta forma, para identificar correctamente los *tokens* que las componen.

- **Tokenización**
  Para obtener las palabras que son usadas en un documento, es necesario realizar un proceso de *tokenización*. Lo que se hace en este paso es dividir el texto original en

palabras, que son posteriormente llamadas *tokens*, generalmente quitando los elementos de puntuación. Se pueden utilizar otros criterios para la obtención de tokens, de acuerdo al análisis que será realizado. Además de palabras, se pueden identificar como tokens algunas abreviaciones, precios, emoticones, entre otros. El arreglo que resulta de juntar los distintos tokens obtenidos de una colección de documentos es llamado *diccionario*.

- ***Case-Folding***
  Se refiere a pasar todas las palabras a minúscula. De esta manera se normalizan los tokens. Es una técnica especialmente útil al momento de comparar tokens y realizar un conteo de éstos.

- **Filtrado**
  Se utiliza para disminuir el tamaño de un diccionario. El método más utilizado es el borrado de *stopwords*. Las stopwords son palabras que no aportan, o aportan muy poca, información como artículos y preposiciones, entre otras. Lo más común para realizar el borrado es comparar las palabras de un diccionario con una *stoplist*, una lista de stopwords, y eliminar las coincidencias.

- **Lematización**
  El objetivo de este proceso es transformar una palabra desde su *forma flexionada* a su forma base, o lema. La forma flexionada se refiere a la conjugación, plural, género, entre otros. Es necesario un análisis morfológico de cada palabra del diccionario, lo que se suele lograr asignando a cada palabra su respectiva etiqueta gramatical (verbo, sustantivo, adjetivo, etc.).

2.4.2 Ponderación de términos

Dado un conjunto de términos de un documento, se puede notar que no todos éstos son igualmente útiles para describir el contenido de éste. Decidir la importancia de un término, para resumir los contenidos de un documento, no es un problema trivial. A pesar de esta dificultad, existen propiedades que son fácilmente medibles y que son útiles para evaluar la importancia de un término. A modo de ejemplo, se considera una colección de cien mil documentos. Si una palabra aparece en cada uno de los documentos, no será útil para caracterizar un documento, puesto que no permite diferenciarlo del resto. En cambio, una palabra que esté sólo en 5 documentos de esta colección es útil porque permite diferenciar fácilmente los documentos. Por lo tanto, la importancia de cada término es variable [27].

TF-IDF

Para reflejar la importancia de cada término en un documento, se utilizan esquemas de ponderación. Lo que hacen es asignar un ponderador a cada término del documento. El más popular es TF-IDF [27].

El ponderador que se le asigna a cada término es el que se muestra en la Ecuación 2.1:

$$w_{t,d} = TF_{t,d} \times IDF_t \tag{2.1}$$

Donde $w_{t,d}$ es el ponderador asignado al término t en el documento d, $TF_{d,t}$ es la frecuencia de t en d y IDF_t es la frecuencia inversa de t en toda la colección de documentos. La forma más común de representar IDF_t se muestra en la Ecuación 2.2:

$$IDF_t = \ln \frac{N}{n_t} \tag{2.2}$$

Donde N es el número de documentos en la colección y n_t es el número de documentos en los que está t. De esta manera, el peso de un término en un documento es creciente mientras más alta es su frecuencia en el documento, y es decreciente en cuanto a la cantidad de documentos de los que es miembro.

2.4.3 Preprocesamiento lingüístico

Muchas veces se pueden aplicar métodos de Minería de Texto sin mayor preprocesamiento. Sin embargo, algunas veces un preprocesamiento lingüístico adicional se puede utilizar para mejorar la información disponible de cada término [26]. Los siguientes enfoques son frecuentemente aplicados:

Etiquetado Gramatical
Se conoce más por su nombre y sigla en inglés, *Part-of-speech tagging* o POS *tagging*, y es la tarea de etiquetar cada palabra en una oración con su etiqueta gramatical correspondiente [28]. En otras palabras, se decide si la palabra es un sustantivo, verbo, adjetivo, etc. A continuación se muestra un ejemplo de una oración etiquetada:

(2.1) *Me gusta la Universidad de Chile*
p v a n s n

Para lograr etiquetar correctamente una palabra, se suelen usar 2 enfoques. Uno utiliza el contexto de la palabra para determinar la etiqueta correcta y el otro simplemente utiliza la etiqueta que es más probable en los documentos de entrenamiento. También se pueden utilizar en conjunto.

Reconocimiento de nombres propios
Se trata de reconocer y clasificar nombres propios en el texto, particularmente nombres que involucren más de una palabra. Por ejemplo, la oración ilustrada en el Ejemplo 2.1 estaría mejor etiquetada si se detectara a "Universidad de Chile" como un nombre propio, como se muestra en el siguiente ejemplo:

(2.2) *Me gusta la Universidad de Chile*
p v a n

2.5 El usuario

Con el objetivo de aclarar conceptos y evitar confusiones al lector en los capítulos posteriores, en esta sección se definen los conceptos: usuario, perfil de usuario y modelo de usuario [18].

Usuario

El World Wide Web Consortium define, en [29], al usuario como la persona, o un grupo de personas actuando como una entidad, que accede a uno o más servicios de un sistema. Para cada usuario existe información personal, lo que lo hace ser identificable. El usuario se puede identificar por sus características y su comportamiento. Las características son información ingresada por el usuario como su nombre, dirección, fecha de nacimiento, etc. El comportamiento se refiere a cómo el usuario interactúa con el sistema.

Perfil de usuario

Los sitios web de redes sociales crean un perfil de usuario en el paso de registro. El perfil puede incluir información demográfica, como el nombre, edad y país [30], así como necesidades, tópicos de interés o deseos [31]. En este tipo de ambientes se tienen 2 tipos de perfil de usuario: uno es el basado en *keywords*, o palabras clave, donde el perfil es representado por una bolsa de palabras, más conocido por su nombre en inglés *Bag of words*; el otro tipo de perfil es el basado en conceptos semánticos[32, 33].

Modelo de usuario

Según Kobsa [34], "los modelos de usuario son colecciones de información y supuestos sobre usuarios individuales (así como usuarios grupales), que se necesitan para el proceso de adaptación de sistemas a acciones individuales de los usuarios". Esto quiere decir, desde el punto de vista de una red social, que un modelo de usuario es el conocimiento dinámico que el sistema recolecta y construye para modificar su interacción con el usuario.

Algunas diferencias que hacen un modelo de usuario distinto de un perfil, son que el modelo permite aprender preferencias del usuario, predecir o inferir preferencias del usuario y mantenerlas dinámicamente [30, 35]. Sitios de redes sociales, aplicaciones web personalizadas y motores de búsqueda personalizados utilizan modelos de usuarios para adaptar los servicios a las necesidades de los usuarios [36, 37].

2.6 User Modeling

User Modeling es la disciplina que se enfoca en crear modelos de usuarios. Se utiliza el término indistintamente con *User Profiling* y *User Profile Modeling*. El objetivo de un mo-

delo de usuario es caracterizar al usuario, y existen muchas formas de hacerlo. La elección del mejor método dependerá de la información que requiera el usuario final, así como del uso que él le dará a ésta. Los métodos se pueden clasificar según la forma de obtener datos, el dinamismo del modelo, la manera de construir el perfil y el tipo de enriquecimiento semántico que se utilice [38].

2.6.1 Recolección de información de usuarios

Los datos recogidos dependen de la naturaleza del sitio web utilizado y de la aplicación que se le dará al modelo. En general se pueden obtener datos explícitos, implícitos, de estereotipos y de conexiones sociales [38, 39].

Datos explícitos

Los datos explícitos son proporcionados directamente por el usuario, ya sea a través de formularios, encuestas, comentarios, búsquedas web y clasificaciones. Generalmente, este tipo de información es opcional, y muchas veces corresponde a información demográfica, como la edad del usuario, género, trabajo, cumpleaños, estado civil y pasatiempos. En algunas ocasiones el usuario puede indicar explícitamente cuales son sus intereses, pero estos también puedes ser inferidos obteniendo palabras clave de sus comentarios o a partir de las calificaciones o *ratings*.

Este tipo de información tiene varios problemas. Primero, los usuarios generalmente no están dispuestos a entregar información llenando largos formularios. Segundo, no siempre dicen la verdad. Tercero, muchas veces los usuarios no saben expresar lo que realmente quieren.

Datos implícitos

En contraste a los datos explícitos, los datos implícitos son los datos inferidos del comportamiento de los usuarios y pueden ser obtenidos estudiando los datos de clicks, transacciones y navegación. Para poder obtener un perfil de usuario a partir de sus acciones, se deben cumplir ciertas condiciones, como que el comportamiento del usuario debe ser repetitivo, ya que si no hay repetición no se podrá descubrir un patrón.

Una característica clave es aprender de la observación y que el perfil sea actualizado según los cambios que tiene el usuario. Las técnicas de User Modeling deben ser capaces de adaptar el contenido del perfil a medida que se registran nuevas observaciones. Otra forma de ingresar información es por retroalimentación, haciendo que el usuario indique, posterior a un servicio, si es que este le gustó o no, y cuánto.

Datos de estereotipos

Estos modelos se basan en la representación de características relevantes y comunes a ciertos subgrupos definidos de usuarios. Los estereotipos fueron de los primeros intentos que se hicieron para diferenciar a un usuario de otros. Generalmente, dependiendo del estereotipo se entregan distintas funcionalidades de un sistema. Esta aproximación es útil cuando no hay más información disponible sobre un usuario, por ejemplo, cuando el usuario no ha usado el sistema con anterioridad. Sin embargo, al ser muy general, no es muy certero y es relativamente estático en el tiempo.

Datos de conexiones sociales

Los datos de conexiones sociales representan las relaciones o interacciones entre usuarios. Las relaciones pueden ser bidireccionales, donde es necesaria la aceptación de ambos usuarios, o unidireccional, como es el caso de los seguidores o amigos en Twitter. Estos datos pueden ser representados como un grafo[5], y el análisis de éstos puede servir para identificar comunidades de usuarios en la red. En general, los grafos sociales se usan para representar una comunidad de confianza del usuario, la que puede ser tratada como usuarios que piensan o tienen opiniones similares. Este método puede reemplazar o complementar el método del vecino más cercano, que depende de las similitudes entre usuarios para identificar usuarios similares. Estos datos también son usados para enriquecer los perfiles de usuarios con más tópicos de interés, asumiendo que un usuario va a estar interesado en tópicos que son comunes con sus amigos o usuarios similares [40].

2.6.2 Dinamismo del perfil de usuario

En Kanoje *et al.* [41] categorizan los modelos según lo dinámicos que son. En otras palabras, si el perfil cambia en el tiempo.

Perfiles Estáticos

La creación estática de perfiles es un proceso en el que se analizan la características estáticas y predecibles de un usuario. En esta aproximación, el comportamiento del usuario se predice analizando la información disponible del usuario. Hay algunos problemas cuando se depende sólo de este método, ya que los usuarios no están interesados en revelar información personal, por temas de privacidad, o simplemente porque llenar un formulario puede ser muy tedioso. En consecuencia, la precisión de este método disminuye a medida que pasa el tiempo.

[5]De manera simplificada, un grafo es una estructura matemática usada para representar relaciones entre objetos. Los objetos son representados por nodos y sus relaciones por arcos. Para mayor información se puede revisar el material del curso de Matemáticas Discretas y Algoritmos, de la Universidad de Stanford en `https://stanford.edu/~rezab/discrete/Notes/2.pdf`. Visitado 5 de oct. de 2016

Perfiles Dinámicos

Los modelos dinámicos se basan más en la información reciente del usuario que en la antigua. Estos modelos tratan de aprender más acerca del usuario. Es por esto que estos sistemas son llamados *Behavioral Profiling*, *Adaptive Profiling* o también *Ontological Profiling*.

2.6.3 Representaciones de perfiles de usuario

Los perfiles de usuario son generalmente representados como grupos de palabras clave ponderadas, redes semánticas o conceptos ponderados. Los perfiles de palabras claves son los más simples de construir, pero como tienen que capturar y representar todas (o la mayoría) de las palabras de las que pueden corresponder a intereses en otros documentos, se requiere mucha retroalimentación de los usuarios para aprender terminologías relativas a cada tópico. Este también es un problema compartido por la mayoría de los perfiles basados en redes semánticas, deben aprender la terminología que concierne a cada concepto. En contraste, los perfiles de conceptos son entrenados con ejemplos para cada concepto, por lo que empiezan con relaciones hechas entre vocabulario y conceptos [30].

Perfiles de palabras clave

La representación más común de perfiles de usuario son grupos de palabras clave. Estas pueden ser extraídas automáticamente desde documentos web, desde sus relaciones con otros usuarios o pueden ser entregadas directamente por el usuario. Los pesos, o ponderaciones, que usualmente están asociados a cada palabra clave, son representaciones numéricas de la importancia de la palabra dentro de los intereses del usuario. Cada palabra clave, mejor conocida por el inglés *keyword*, puede representar un tópico de interés o estas pueden ser agrupadas en categorías, lo que resulta en una representación más estándar de los intereses del usuario. Un ejemplo de un perfil de usuario basado en keywords ponderadas se puede ver en la Figura 2.2.

Perfiles de redes semánticas

Para abordar el problema de la polisemia[6] inherente a los perfiles basados en keywords, los perfiles pueden ser representados por una red semántica ponderada en la que cada nodo representa un concepto. Minio *et al.* [42] trabajaron en un enfoque, basado en este tipo de perfiles, en el que cada nodo contiene una palabra en particular que fue encontrada en el corpus[7] y los arcos son creados basados en la ocurrencia simultánea de 2 palabras.

Otro ejemplo es InfoWeb [43], un sistema de filtrado para documentos de bibliotecas digitales en línea, que también utiliza perfiles basados en redes semánticas y que representan

[6] Según WordReference.com, la definición de polisemia es "Pluralidad de significados de una palabra".

[7] Según WordReference.com, la definición de corpus es "Conjunto de datos, textos u otros materiales sobre determinada materia que pueden servir de base para una investigación o trabajo".

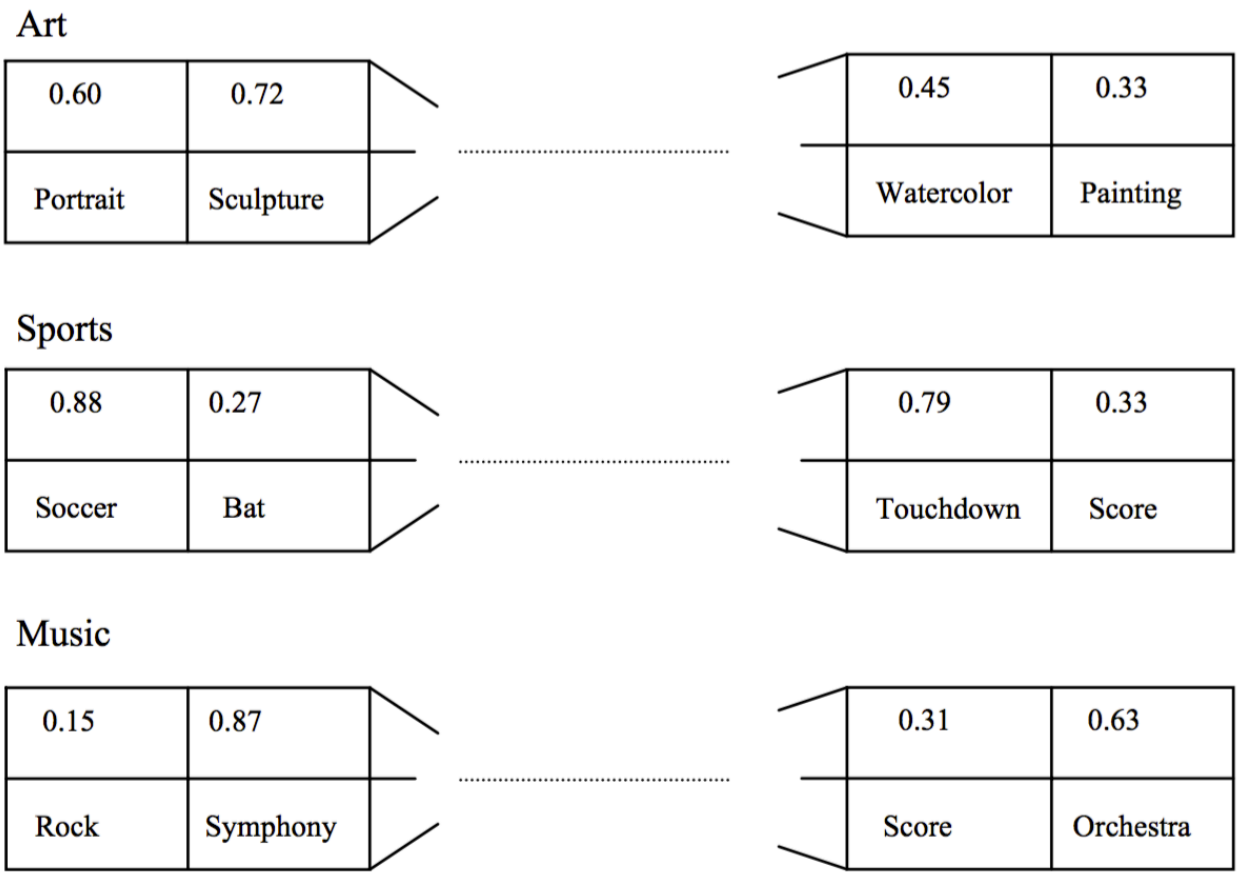

Figura 2.2: Perfil de usuario basado en keywords

Fuente: Gauch *et al.* [30]

intereses de largo plazo. Cada perfil de usuario es representado como una red semántica de conceptos. Inicialmente, cada red semántica contiene una colección de nodos que no están unidos, en la que cada nodo representa un concepto. Los nodos de conceptos, llamados *planetas*, contienen un ponderador que representa la importancia de ese concepto. A medida que se recopila más información del usuario, el perfil es enriquecido agregando palabras (con sus ponderadores respectivos) adicionales asociadas a los conceptos. Estas palabras clave son almacenadas en nodos secundarios, llamados *satélites*, unidas a su planeta asociado. Se agregan también uniones entre planetas representando asociaciones entre conceptos. La Figura 2.3 muestra un ejemplo de un modelo de usuario basado en esta representación.

Perfiles de conceptos

Los perfiles basados en conceptos, también llamados “Perfiles basados en ontologías” [44], son similares a los basados en redes semánticas en el sentido de que ambos son representados por nodos conceptuales y relaciones entre esos nodos. Sin embargo, en los perfiles basados en conceptos, los nodos representan tópicos abstractos que se consideran interesantes por el usuario, a diferencia de usar palabras específicas o grupos de palabras

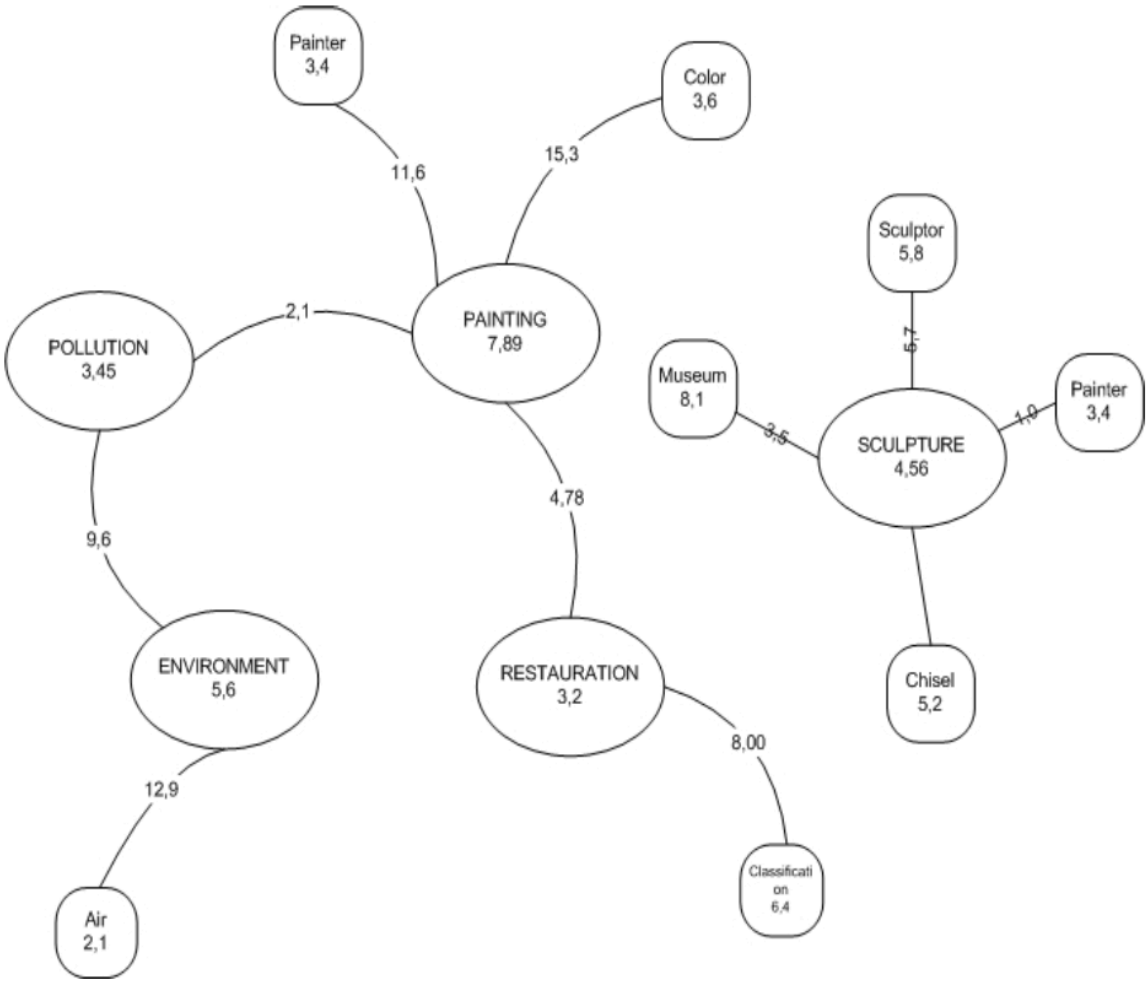

Figura 2.3: Perfil de usuario basado en redes semánticas

Fuente: Gauch *et al.* [30]

relacionadas. Los perfiles de conceptos también son similares a los perfiles de keywords en el sentido de que ambos se representan como vectores de características ponderadas, pero las características representan conceptos y no palabras ni grupos de palabras. Se usan varios mecanismos para expresar qué tan interesado está el usuario en cada tópico. La técnica más simple es un valor numérico, o ponderador, asociado a cada tópico.

Uno de los primeros proyectos en construir perfiles de usuarios basados en conceptos fue el proyecto OBIWAN [45]. El proyecto utilizó varias jerarquías de referencia, pero finalmente seleccionó a *Open Directory Proyect*[8]. La Figura 2.4 muestra un ejemplo de un perfil de usuario creado por el proyecto OBIWAN a partir del historial de búsqueda web del usuario. Otros autores [44] han usado otras jerarquías, como las de Wikipedia[9] y Wordnet[10].

[8] `https://www.dmoz.org`
[9] `https://www.wikipedia.org`
[10] `https://wordnet.princeton.edu`

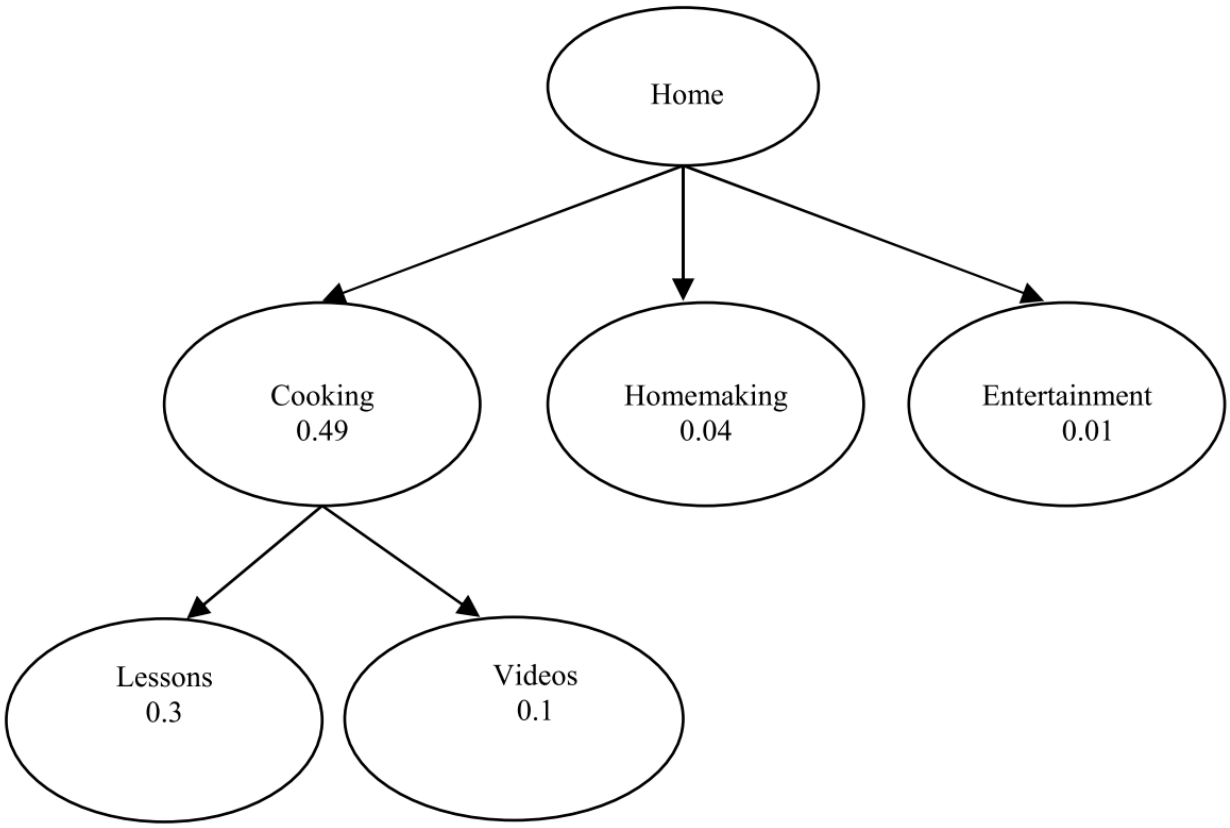

Figura 2.4: Perfil de usuario basado en conceptos

Fuente: Gauch *et al.* [30]

Los perfiles de usuario más simples, basados en jerarquías de conceptos, son construidos a partir de una taxonomía de referencia o tesauros. Para construir perfiles más complejos, se pueden usar ontologías de referencia. En este caso, las relaciones entre conceptos son especificadas explícitamente y el perfil resultante puede incluir información más rica y una amplia gama de tipos de relaciones [30, 44].

2.6.4 Construcción de perfiles de usuario

Los perfiles de usuario son construidos a partir de fuentes de información utilizando una amplia gama de técnicas basadas en aprendizaje de máquinas, conocido también por el inglés *machine learning*, y en recuperación de información, también conocido por el inglés *information retrieval*. Las técnicas a utilizar dependen de la representación de perfil de usuario deseada. Los perfiles pueden ser construidos manualmente por los usuarios o expertos, sin embargo esto es difícil y puede tomar mucho tiempo a muchos usuarios, por lo que sería una barrera a la adopción masiva del servicio. Las técnicas que construyen los perfiles automáticamente, a partir de la retroalimentación de los usuarios, son mucho más populares. A pesar de que algunos enfoques usan algoritmos genéticos o redes neuronales para aprender los perfiles, existen otros enfoques, más simples y eficientes, basados en probabilidades o en modelos de espacios vectoriales que son ampliamente utilizados y que

son efectivos en muchas aplicaciones.

Sin importar cuál método se utilice, el perfil debe mantenerse actualizado para reflejar las preferencias del usuario con exactitud. La actualización del perfil puede ser realizada manual o automáticamente. Se prefieren lo métodos automáticos porque son menos invasivos para el usuario final. Algunos autores [30] recomiendan no hacer uso métodos de actualización absolutamente automáticos, sino utilizar la retroalimentación del usuario, que requiere esfuerzos mínimos. Sin embargo los resultados de los experimentos de actualización automática de perfiles son prometedores [46].

Construyendo perfiles de palabras clave

Los perfiles basado en palabras clave se crean extrayendo palabras clave desde páginas web que se obtienen de alguna fuente de información, como el historial de navegación del usuario o los favoritos del navegador. Se ocupa alguna forma de ponderación de palabras clave para identificar las palabras más importantes de una página web dada, y usualmente el número de palabras extraídas de una sola página es limitado para que sólo los *N* términos con mayor peso de cada página contribuyan al perfil.

La forma más simple de construir perfiles produce sólo un perfil de palabras clave para cada usuario. Un ejemplo es *Amalthaea* [47], uno de varios sistemas que crean perfiles extrayendo palabras clave de páginas web. Para ponderar las palabras ocupan el esquema de TF-IDF de recuperación de información. Se refiere al lector a la sección 2.4.2 para más detalles sobre TF-IDF. La particularidad de este proyecto es que usa un algoritmo de aprendizaje basado en algoritmos genéticos para adaptar y expandir los perfiles de los usuarios. Otro ejemplo que destaca es el sistema *Who Likes What*, desarrollado por Bhattacharya *et al.* [48]. Este sistema es novedoso porque utiliza las conexiones de Twitter para identificar tópicos de interés de usuarios. Esto hace que no se requiera que el usuario ingrese información y también resuelve el problema de no poder caracterizar a un usuario en Twitter si no genera contenido. Para más información se refiere al lector a la sección 2.6.5.

Construyendo perfiles de redes semánticas

Los perfiles basados en redes semánticas son típicamente construidos por recolección de calificaciones, o en inglés *feedback*, del usuario, tanto positiva como negativa. Similarmente a las técnicas de construcción de perfiles de vectores de palabras clave, las palabras claves se extraen de páginas web calificadas por usuarios. Las técnicas difieren de las de la sección anterior en que, en vez de agregar las palabras clave a un vector, las palabras se agregan a una red de nodos. Los nodos pueden representar palabras individuales o, en aplicaciones más complejas, un concepto particular y sus palabras asociadas. Los términos *conceptos* e *intereses* son a menudo usados sin distinción en la literatura. En esta sección, concepto se refiere a una idea específica y a una colección de palabras asociadas, por ejemplo: *perro* y sus sinónimos, mientras que un interés se refiere a tópicos de interés de más alto nivel, como *Derechos de Animales*, que a su vez pueden ser representados por una colección de conceptos asociados.

Los perfiles de usuario semánticos tienen una ventaja sobre los perfiles basados en palabras clave porque puede modelar explícitamente la relación entre palabras particulares y un concepto de alto nivel. En consecuencia, pueden lidiar más efectivamente con la ambigüedad y sinonimia[11] inherente al lenguaje natural. Sin embargo, esto también hace que sea más difícil construir dicho sistema. Los sistemas deben utilizar alguna relación existente entre palabras y conceptos, como la de WordNet. También pueden construir esta relación mediante un algoritmo de aprendizaje de máquinas o manualmente.

Construyendo perfiles de conceptos

Los perfiles de conceptos pueden construirse de muchas maneras y de muchas fuentes de datos distintas, pero tienen en común que utilizan una taxonomía como base para el perfil [30]. Estos perfiles difieren de los perfiles de redes semánticas en que ellos se describen en términos de conceptos preexistentes, en vez de modelar los conceptos como parte del perfil de usuario. En consecuencia, todos ellos requieren una manera de determinar en qué conceptos el usuario está interesado basado en retroalimentación. Aunque algunos sistemas obtienen retroalimentación a partir de documentos previamente clasificados, muchos otros la obtienen a partir de una amplia variedad de documentos y posteriormente realizan clasificación de texto para identificar los conceptos contenidos por cada uno. En la Figura 2.5 se puede ver un esquema que representa la creación y actualización de un perfil en el sistema OBIWAN [45].

2.6.5 Método Who-Likes-What

Vale la pena referirse con mayor detalle a la metodología para encontrar tópicos de interés de usuarios en Twitter llamada *Who-Likes-What*. Esta metodología fue desarrollada por Bhattacharya *et al.* [48], un grupo de investigadores del Max Planck Institute[12] de Alemania y del Indian Institute of Technology Kharagpur[13] de India.

Para explicar esta metodología, se define como u a un usuario dado de Twitter, de quien serán inferidos los tópicos de interés. La metodología propuesta consiste de 2 etapas. En primer lugar, se revisa a qué usuarios el usuario u está siguiendo, quienes son llamados los *amigos* de u. Estos son los usuarios de quienes u está interesado en recibir información. En segundo lugar, se identifican los tópicos de influencia, o sobre los que son considerados expertos, de los *amigos* de u. Estos tópicos se utilizan para inferir los tópicos de interés de u, en otras palabras los tópicos de los que u está interesado en recibir información.

[11] Según WordReference.com, la definición de sinonimia es 'Coincidencia de significados entre dos o más vocablos'.

[12] `http://www.mpi-sws.org`

[13] `http://www.iitkgp.ac.in`

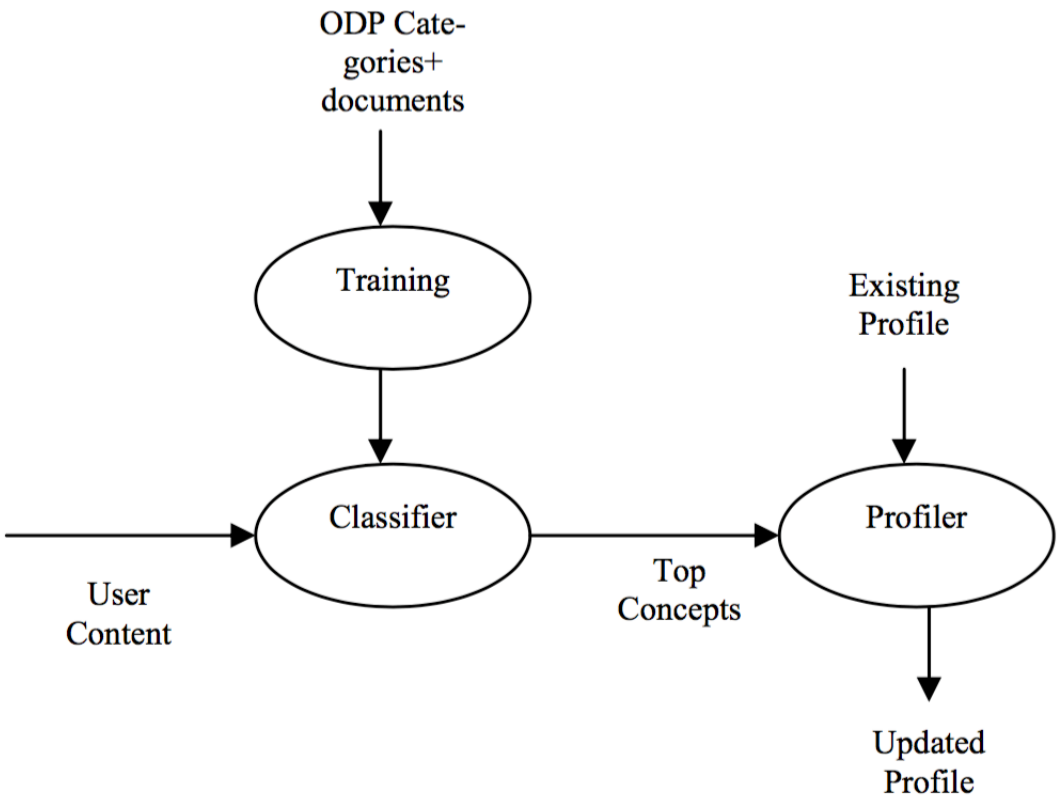

Figura 2.5: Creación de perfil de usuario en OBIWAN

Fuente: Gauch *et al.* [30]

Infiriendo tópicos de expertise usando listas de Twitter

Las listas de Twitter son una forma que tienen los usuarios para agrupar a *amigos* bajo un título y descripción(para más información se puede ver la sección 2.2.1).

En un trabajo previo [49, 50], se propuso una metodología, llamada *who-is-who*, para identificar los tópicos de *expertise* de usuarios populares en Twitter, utilizando los títulos y las descripciones de las Listas que crean los usuarios. Lo que se obtiene son los tópicos que los usuarios utilizan para categorizar a sus *amigos*. Para identificar los tópicos del usuario v, a quién u sigue, se obtienen las Listas que tienen a v como miembro y se extraen los términos más comunes que aparecen en sus títulos y descripciones. Se identifica a v como un experto en el tópico t si v es miembro de al menos 10 Listas que contengan el tópico t. El umbral 10 fue definido basándose en observaciones de sus estudios previos [49, 50]. Similar a lo que se hizo en estos trabajos previos, se consideraron como tópicos solamente unigramas y bigramas, que son identificadas como sustantivo o adjetivos por un etiquetador de parte del discurso (más conocido por el inglés *POS-tagger*). En los trabajos previos se mostró que esta metodología infiere con exactitud los tópicos de *expertise* de millones de usuarios populares de Twitter. Por ejemplo, los tópicos de la cuenta @BarackObama, inferidos por la metodología descrita, se pueden ver en la Figura 2.6. El resultado está en línea y disponible para el público general en `http://twitter-app.mpi-sws.org/who-is-who`.

Figura 2.6: Resultado sistema *who-is-who* para el usuario @BarackObama

Fuente: [50, 51]

Infiriendo tópicos de interés

Para un usuario *u*, se usa la metodología descrita anteriormente para identificar los tópicos sobre los que los *amigos* de *u* son considerados expertos. Intuitivamente, si un usuario se suscribe a recibir tweets de varios usuarios que son considerados expertos un un cierto tópico *t*, entonces es muy probable que al usuario *u* le interese el tópico *t*. Se considera que *u* está interesado en el tópico *t* si *u* sigue por lo menos a 3 expertos en el tópico *t*. Entonces, se obtiene un vector de tópicos de interés para *u*, que es una lista ordenada de acuerdo al numero de usuarios expertos a los que *u* sigue. En la Figura 2.7 se ve un resultado de los tópicos de interés obtenido del sistema *Who Likes What*. Este sistema está disponible para pruebas en línea en `http://twitter-app.mpi-sws.org/who-likes-what/`.

2.7 APIs

API es una sigla para *Application Programming Interface*, que se traduce como "Interfaz de programación de aplicaciones". Una API es un conjunto de definiciones, protocolos y herramientas utilizadas para la construcción de software y aplicaciones.

Como lo dice su nombre, es una *interfaz* que facilita, o permite, la comunicación entre aplicaciones [53]. Es un intermediario que estandariza la comunicación, permitiendo que esta sea más fluida y duradera en el tiempo. En la Figura 2.8 se muestra un esquema de comunicación entre aplicaciones, por medio de una API.

Una API permite interactuar, a un programador, con una aplicación usando una serie de funciones. El objetivo es que se puedan escribir programas que no dejen de funcionar

Figura 2.7: Resultado sistema *Who Likes What*

Fuente: [48, 52]

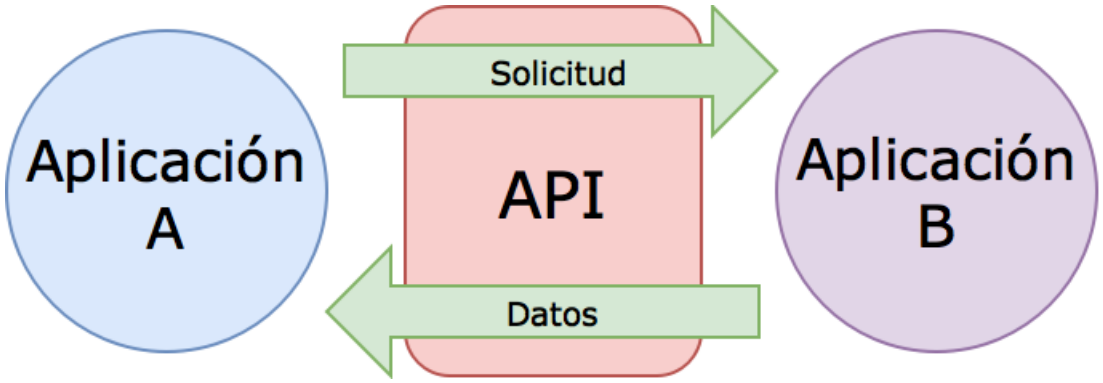

Figura 2.8: 2 aplicaciones comunicadas por medio de una API

Fuente: Elaboración propia

cuando hayan cambios en el sistema subyacente.

Las APIs pueden ser dependientes o independientes del lenguaje de programación. Cuando es dependiente significa que sólo está disponible mediante el uso de la sintaxis y elementos de un lenguaje en particular. Si es independiente, la API puede ser llamada de varios lenguajes de programación distintos.

2.8 Métricas de evaluación

Las dos medidas que se utilizan con mayor frecuencia en sistemas de recuperación de información para medir su efectividad son la precisión y la exhaustividad (más conocidas por sus nombres en inglés, *precision* y *recall*, respectivamente) [54].

La precisión es la fracción de elementos recuperados que son relevantes [27], como se ve en la ecuación 2.3:

$$Precision = \frac{\#\ de\ elementos\ relevantes\ recuperados}{\#\ de\ elementos\ recuperados} \tag{2.3}$$

La exhaustividad se define como la fracción de elementos relevantes que son recuperados, como se ve en la ecuación 2.4

$$Exhaustividad = \frac{\#\ de\ elementos\ recuperados}{\#\ de\ elementos\ relevantes} \tag{2.4}$$

Por ejemplo, se considera una conjunto de 30 elementos que pertenecen a la clase c_i, dentro de un conjunto C. Al hacer una consulta para recuperar documentos de la clase c_i se obtienen 10 documentos que efectivamente pertenecen a esa clase. En este caso los resultados de precisión y exhaustividad están dados por

$$Precision(c_i) = \frac{10}{10} = 100\,\% \tag{2.5}$$

$$Exhaustividad(c_i) = \frac{10}{30} = 33,3\,\% \tag{2.6}$$

Donde se ve que la precisión es del 100 %, sin embargo hay bastantes elementos relevantes que no fueron recuperados, lo que se ve reflejado por su exhaustividad. Suele ser útil el utilizar ambas medidas, cuando es posible. Dependiendo de la aplicación del modelo de clasificación puede ser una más importante que la otra.

Capítulo 3

Diseño del sistema

En este capítulo se describe el diseño del sistema de identificación de tópicos de interés. La idea principal es mostrar qué es lo que se hizo durante el desarrollo, pero no entrar en detalles de cómo se hizo. Los algoritmos utilizados, herramientas de desarrollo y lo relacionado a la programación del sistema son presentados en el Capítulo 4: Desarrollo del sistema.

Dada la revisión bibliográfica realizada, se decidió diseñar el sistema en base a la metodología desarrollada por Bhattacharya *et al.* [48]. Esta metodología se caracteriza por ser capaz de identificar tópicos de interés de cualquier usuario de Twitter, incluso los que no *twiteean*. Esto es posible porque no se utiliza el contenido generado por los usuarios, sino que los amigos de ellos y el sistema de listas de Twitter. Se refiere al lector a la Sección 2.6.5 para una explicación en detalle del método.

A continuación se explica el sistema en general y posteriormente cada uno de sus módulos por separado.

3.1 Arquitectura general

En esta sección se presenta la arquitectura general del sistema. La Figura 3.1 pretende ilustrar la comunicación entre los distintos módulos y el orden del proceso.

El proceso inicia de 2 posibles formas:

1. Un desarrollador hace un pedido a la API desde alguna aplicación.
2. Un usuario utiliza el Módulo de Visualización, el que hace un pedido a la API.

El proceso siguiente es el mismo en cualquiera de los 2 casos mencionados:

1. La API realiza un pedido al Módulo de Identificación de Tópicos de Interés, en adelante **MITI**.

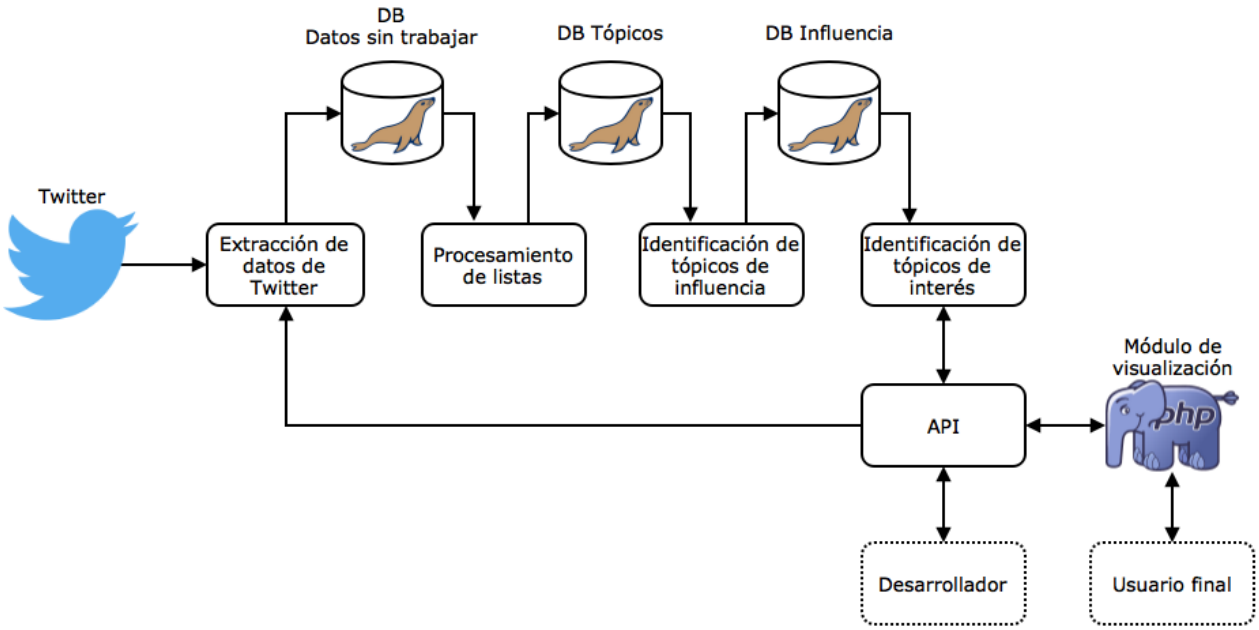

Figura 3.1: Arquitectura General

Fuente: Elaboración propia

2. El MITI tiene 2 opciones, que dependen del caso:

   (a) Si el usuario ha sido caracterizado con anterioridad al pedido actual, se pueden identificar sus tópicos de interés a partir de los tópicos de influencia de cada uno de sus amigos. Se entrega, a la API, el set de tópicos de interés con sus frecuencias.

   (b) Si el usuario no ha sido caracterizado, entonces hay que ejecutar el proceso completo. En este caso el MITI retorna que el usuario no ha sido caracterizado y la API realiza un pedido al Módulo de Extracción de Datos de Twitter, en adelante **MED**, enviándole el ID del usuario a caracterizar.

3. El MED se conecta con la API de Twitter para obtener los datos de los amigos del usuario y de las listas de las que estos son miembros. Estos datos son almacenados en la base de datos.

4. El Módulo de Procesamiento de Listas procesa el texto del título y la descripción de las listas y extrae de ellos los sustantivos. Luego se almacenan estos tópicos en la base de datos de tópicos. En esta base de datos se almacenan también la relación de cada lista con sus respectivos tópicos, lo que será utilizado en el siguiente módulo.

5. El Módulo de Obtención de Tópicos de Influencia utiliza los tópicos que se extraen de las listas de las que cada uno de los amigos del usuario es miembro. Se realiza una agregación tanto por relevancia del tópico en cada lista cómo por su frecuencia. Posteriormente se almacenan los tópicos relevantes a cada amigo en la base de datos, con su respectiva importancia y frecuencia.

6. El MITI obtiene los tópicos de interés del usuario a partir de los tópicos de influencia de sus amigos, igual como se realiza en el punto 2a.

7. La API ahora redirige el set completo con la frecuencia de cada tópico de interés ya sea al programa que lo solicitó o al Módulo de Visualización, dependiendo del origen del requerimiento original.

8. En el caso que el requerimiento provenga del Módulo de Visualización, al usuario final se le muestra una nube de palabras de distintos tamaños que muestra los tópicos de interés y la importancia de cada uno.

3.2 Módulo de Extracción de Datos de Twitter

El Módulo de Extracción de Datos de Twitter se encuentra al principio del sistema y tiene el objetivo de obtener los datos de Twitter relativos al usuario que se quiere caracterizar.

El módulo está compuesto de 2 submódulos:

Módulo de Obtención de Amigos (MOA): Cómo su nombre bien lo dice, es el módulo que obtiene los amigos de Twitter del usuario a caracterizar y los almacena en la base de datos para su posterior análisis.

Módulo de Obtención de Listas (MOL): Este módulo recibe el conjunto de amigos del usuario a caracterizar y obtiene las listas de las que es miembro cada uno de ellos.

El proceso que sigue el Módulo de Extracción de Datos de Twitter, que se puede ver en la Figura 3.2, es el siguiente:

1. La API envía el requerimiento al Módulo de Obtención de Amigos. Este requerimiento incluye el *id* de Twitter del usuario que se quiere caracterizar.

2. El MOA se comunica con la API de Twitter y obtiene de respuesta el conjunto de amigos del usuario.

3. El MOA inserta la información obtenida en la base de datos.

4. Se envía el conjunto de amigos del usuario al siguiente módulo, el Módulo de Obtención de Listas.

5. El MOL se comunica con la API de Twitter y obtiene de respuesta el conjunto de listas de cada amigo del usuario. Dentro de la información obtenida está el título y la descripción de cada lista.

6. Se insertan los datos obtenidos en la base de datos.

7. El MOL envía la lista de amigos al Módulo de Procesamiento de Listas.

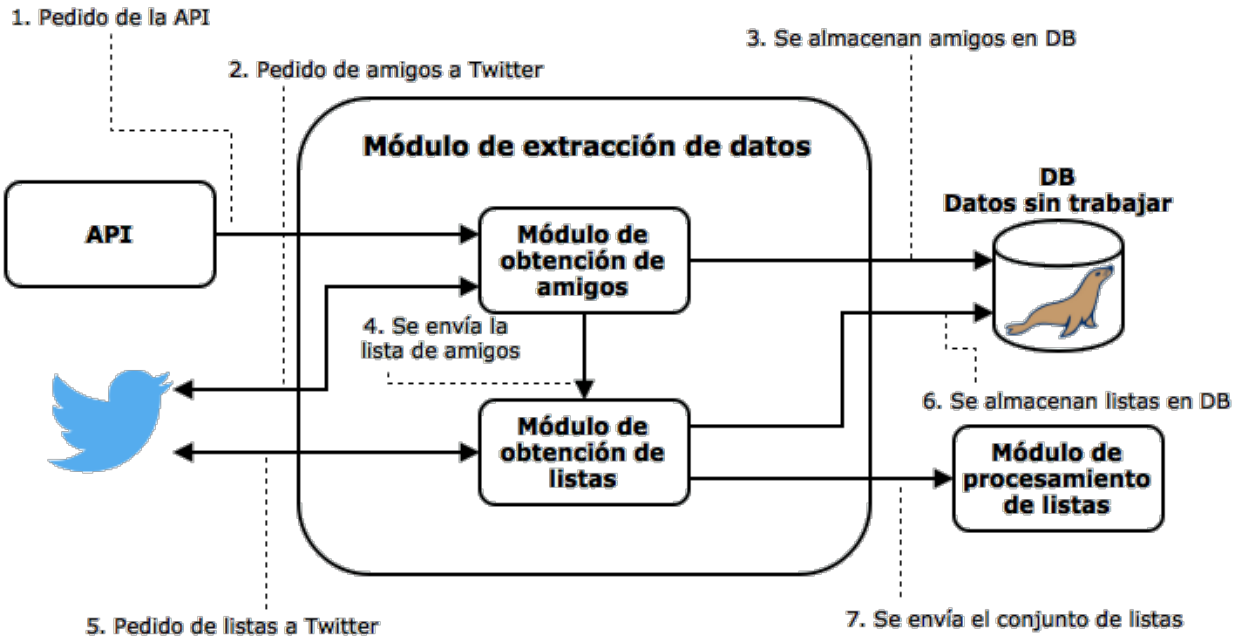

Figura 3.2: Arquitectura de Módulo de Extracción de Datos de Twitter

Fuente: Elaboración propia

3.3 Módulo de Procesamiento de Listas

El Módulo de Procesamiento de Listas es el segundo módulo del proceso. Su tarea es tomar y transformar el texto de las listas de Twitter en una estructura que sea útil para el resto del proceso. La mayoría de las etapas de este módulo corresponden a etapas de Minería de Texto y las otras son tareas específicas necesarias para extraer información de las listas.

En la Figura 3.3 se pueden ver los elementos generales que componen al Módulo de Procesamiento de Listas. Se comienza con la lista de usuarios de Twitter obtenidos en el Módulo de Extracción de Datos de Twitter. Luego el gestor realiza los pasos 2 y 3 para cada usuario. El paso 2 consta de la realización de una consulta a la base de datos, para obtener los datos de las listas de cada usuario. En el paso 3 se envían las listas obtenidas al Módulo de Procesamiento de Texto, el que es explicado en la Sección 3.3.1. El paso 4, donde se insertan los tópicos a la base de datos, es realizado para cada una de las listas procesadas. Por último, cuando no quedan más usuarios de quienes obtener listas ni listas que procesar, se continúa con el Módulo de Identificación de Tópicos de Influencia, el que está descrito en 3.4.

3.3.1 Módulo de Procesamiento de Texto

Este módulo se encarga principalmente, como lo dice su nombre, de procesar el texto del título y la descripción las listas. Sin embargo, tiene etapas que se escapan de lo que se conoce como procesamiento de texto. Estas etapas son la asignación de peso o ponderado-

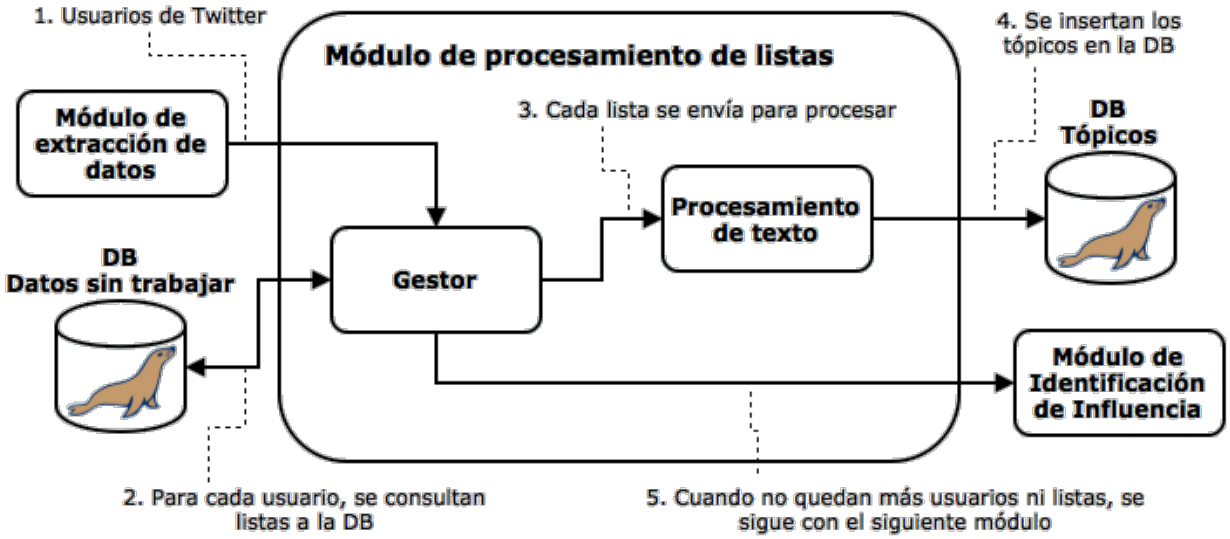

Figura 3.3: Módulo de Procesamiento de Listas

Fuente: Elaboración propia

res, que indican la relevancia del término, y la inserción en la base de datos. El detalle de este módulo se ilustra en la Figura 3.4. A continuación se presenta una explicación sobre cada submódulo.

Tokenizador previo: Separa las palabras que componen al título de las listas. En particular, se separan las palabras que están escritas con *CamelCase* y las que están unidas por otros caracteres como "+", "_", entre otros. Se quitan también algunos caracteres de puntuación para no confundir al segmentador de oraciones.

Unión de título y descripción: El título y la descripción, en caso de que la lista tenga una, están almacenadas por separado. En esta etapa se unen para mejorar el desempeño del detector de idioma.

Limpiador: Se remueven los "#", muy utilizados en los *hashtags* de Twitter.

Detector de idioma: Realiza un análisis del texto del título y descripción de la lista e identifica el idioma en el que está escrito. Este resultado es usado en las otras etapas, que poseen funciones distintas dependiendo del idioma.

Tokenizador: Separa el texto de la lista en sus componentes o *tokens*: palabras, números y puntuación.

Segmentador de oraciones: Agrupa las palabras obtenidas del tokenizador en oraciones.

Análisis morfológico: En esta etapa se realizan varias tareas, que incluyen la detección de puntuación, detección de números, detección de fechas y reconocimiento de *entidades nombradas*. Las entidades nombradas son un sustantivo propio que tiene 2 o más palabras, por ejemplo "Universidad de Chile".

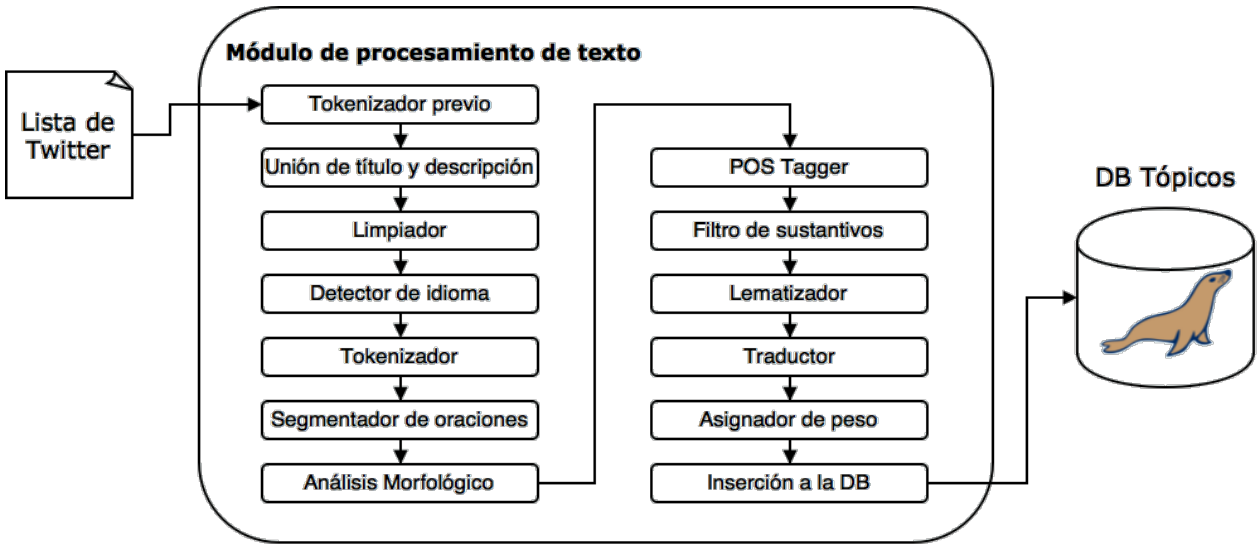

Figura 3.4: Etapas de procesamiento de texto

Fuente: Elaboración propia

POS Tagger: En esta etapa se realiza el etiquetado gramatical de las palabras .

Filtro de sustantivos: En esta etapa se le dan uso a las etiquetas gramaticales y se mantienen solamente los sustantivos.

Lematizador: Se transforma cada uno de los sustantivos obtenidos a su lema, que es la forma no flexionada de la palabra.

Traductor: Se traducen todos los sustantivos que estén en un idioma distinto al castellano.

Asignador de peso: Para cada lista, y dado el numero de palabras en el título y la descripción de ésta, se le asigna un peso, o ponderador, a cada tópico para reflejar su importancia relativa. El ponderador se calcula repartiendo 100 entre el título y la descripción en partes iguales. Si no hay descripción el título se lleva todo. Posteriormente se hace la repartición del peso entre todos los *tokens* de cada sección, equitativamente.

Inserción a la DB: Se inserta cada tópico con su peso a la base de datos, y su relación con la lista.

A continuación se ilustrará cada paso de la fase de procesamiento de listas. La lista de ejemplo es la siguiente:

(3.1)

Título	*TechNews&Stuff*
Descripción	*Anything tech related goes here.* *Mainly #Uber #Tesla, rumors most of the time.*

Después de aplicar el tokenizador previo, donde se separa el título en 3 palabras y se

remueve el “&”, la lista queda así,

(3.2)

Título	*Tech News Stuff*
Descripción	*Anything tech related goes here.* *Mainly #Uber #Tesla, rumors most of the time.*

Posteriormente se une el texto del título y de la descripción, separados por “±”.

(3.3) *Tech News Stuff ± Anything tech related goes here. ± Mainly #Uber #Tesla, rumors most of the time. ±*

Luego, el Limpiador remueve los elementos no deseados.

(3.4) *Tech News Stuff ± Anything tech related goes here. ± Mainly Uber Tesla, rumors most of the time. ±*

Se pasa el texto por el detector de idioma. Se detecta el idioma inglés. El resto de los pasos se realizan utilizando este idioma. Ahora se realiza la tokenización del texto:

(3.5) *(Tech)(News)(Stuff)(±)(Anything)(tech)(related)(goes)(here)(.)(±)(Mainly) (Uber)(Tesla)(,)(rumors)(most)(of)(the)(time)(.)(±)*

A continuación se realiza la identificación de oraciones.

(3.6) *[(Tech)(News)(Stuff)] [(Anything)(tech)(related)(goes)(here)(.)] [(Mainly) (Uber)(Tesla)(,)(rumors)(most)(of)(the)(time)(.)]*

Luego se muestra el resultado del etiquetado grmatical, que fue posible gracias al análisis morfológico realizado previamente.

(3.7) *Tech News Stuff*
n n n
Anything tech related goes here .
p n v v r f
Mainly Uber Tesla , rumors most of the time .
r n n f n r i d n f

Se aplica el filtro de sustantivos y el texto queda:

(3.8) *[(Tech)(News)(Stuff)] [(tech)] [(Uber)(Tesla)(rumors)(time)]*

Luego, el lematizador lleva cada sustantivo a su forma base.

(3.9) *[(tech)(news)(stuff)] [(tech)] [(uber)(tesla)(rumor)(time)]*

Ahora se traducen los términos, en el caso de estar en inglés. El texto resultante es:

(3.10) *[(tecnología)(noticias)(cosas)] [(tecnología)] [(uber)(tesla)(rumor)(tiempo)]*

Posteriormente se le asigna un peso, o ponderador, a cada palabra. Esto es para mostrar su importancia relativa.

(3.11) *tecnología noticias cosas*
$_{16.7}$ $_{16.7}$ $_{16.7}$
tecnología
$_{10}$
uber tesla rumor tiempo
$_{10}$ $_{10}$ $_{10}$ $_{10}$

Finalmente se insertan los tópicos a la base de datos. En el caso de que los tópicos se repitan, se suman los pesos. Para el ejemplo presentado, el resultado es el siguiente:

(3.12) *tecnología noticias cosas uber tesla rumor tiempo*
$_{26.7}$ $_{16.7}$ $_{16.7}$ $_{10}$ $_{10}$ $_{10}$ $_{10}$

3.4 Módulo de Identificación de Tópicos de Influencia

La tarea de este módulo es identificar los tópicos que caracterizan a usuarios de Twitter, en base a cómo otros usuarios los categorizan. Este módulo explota la información de las listas de Twitter, y se basa en la metodología desarrollada por Sharma *et al.* [50].

Las etapas del Módulo de Identificación de Tópicos de Influencia se pueden ver en la Figura 3.5. A continuación una descripción de los pasos en más detalle:

1. Llega una serie de `ids` de Twitter al Módulo de identificación de Influencia.

2. El Orquestador envía uno de los `ids` al Agregador. Este paso se repite para todos los `ids` que llegaron al Orquestador. Se envía un nuevo `id` cada vez que el Agregador termina los pasos 3 y 4.

3. El Agregador realiza una consulta a la base de datos de tópicos. Esta consulta obtiene como resultado los tópicos utilizados para caracterizar al usuario en Twitter.

4. Este paso se realiza para cada uno de los tópicos obtenidos en el paso 3. Lo que se hace es agregar el tópico y su relación con el usuario en la base de datos, para su posterior consulta.

5. Cuando no quedan más usuarios de quienes obtener tópicos, ni tópicos que insertar a la base de datos, se procede al Módulo de Identificación de Tópicos de Interés.

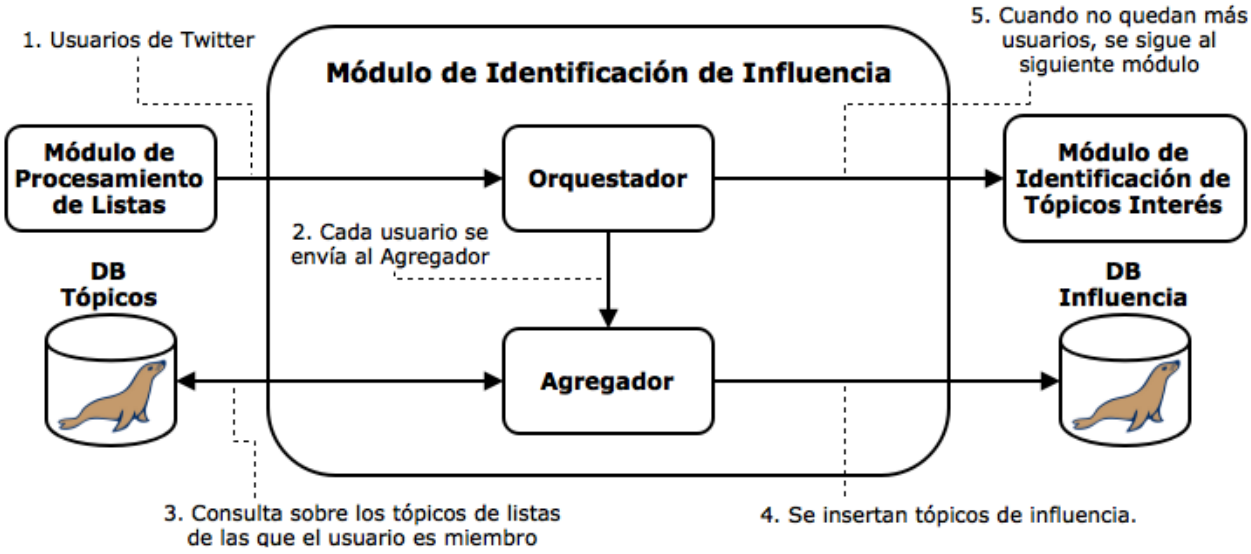

Figura 3.5: Etapas del Módulo de Identificación de Influencia

Fuente: Elaboración propia

3.5 Módulo de Identificación de Tópicos de Interés

El Módulo de Identificación de Tópicos de Interés es el que identifica los tópicos de interés del usuario, a partir de los tópicos de influencia, desde la base de datos y los devuelve a la API. Se compone principalmente de 3 etapas, que se pueden ver en la Figura 3.6. En resumen, el módulo comienza con la solicitud de la API y el envío de un `id` de un usuario de Twitter de quien se quieren obtener los tópicos de interés. En el paso 2 se hace una consulta a la base de datos y se obtienen los tópicos de interés del usuario. Luego, el filtro elimina de la lista los tópicos que no entregan información. En la siguiente etapa se les aplica el esquema de pesos TF-IDF a los tópicos. Se refiere al lector a la sección 2.4.2 para más información sobre TF-IDF. Por último, los tópicos y ponderadores se pasan a formato JSON[1] para ser enviados a la API.

A continuación se entrega más detalle sobre las funciones de cada parte de este módulo.

3.5.1 Agregador

Este submódulo realiza una consulta a la base de datos de influencia. La consulta obtiene los tópicos que caracterizan a todos los usuarios que sigue el usuario de quien se desean obtener tópicos de interés. Estos tópicos son agrupados y los contadores, que indican la importancia de ese tópico para un usuario dado, son sumados. Luego este resultado se envía al submódulo de filtro.

[1]JSON (JavaScript Object Notation) es un formato de intercambio de datos basado en etiquetas. Es ligero y fácil de interpretar y generar tanto por personas como por máquinas. Es un formato ampliamente utilizado en múltiples lenguajes de programación, lo que lo hace ideal para el intercambio de datos. Para más información visitar `http://www.json.org/json-es.html`. Visitada el 23 de noviembre de 2016.

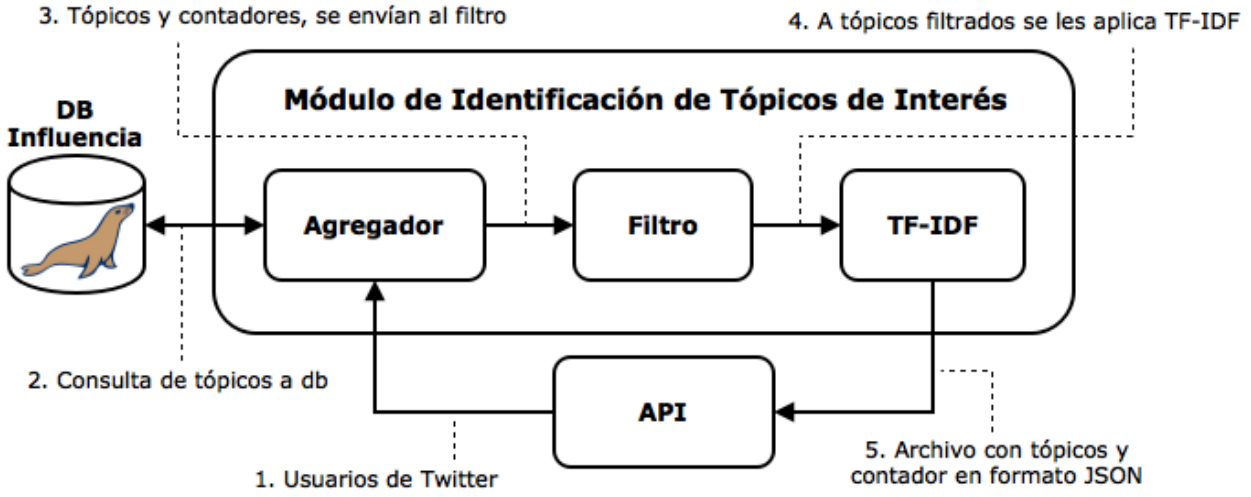

Figura 3.6: Etapas del Módulo de Identificación de Tópicos de Interés

Fuente: Elaboración propia

3.5.2 Filtro

El submódulo de Filtro recibe el conjunto de tópicos y contadores obtenidos por el agregador, y los compara con una *lista negra*, la que consta de tópicos que no aportan información. El submódulo filtra los tópicos que coincidan con los de la lista negra, dejando pasar sólo los que no aparecen en ella.

En la Tabla 3.1 se muestra un extracto de los tópicos de la lista. Se puede ver que varios son referentes a Twitter, y fueron removidos porque no aporta información que a usuarios de Twitter les interese esta plataforma. Otros como “getlistomatic” son nombres de aplicaciones para crear listas, y que ponen su nombre en la información de las listas creadas, por lo que se remueven de los tópicos de interés. Otros tópicos simplemente no aportan información sin mayor contexto como “cuentas”, “personajes”, “personas” y “organizaciones”. La lista negra fue creada a base de la inspección de los resultados obtenidos, tomando en consideración que uno de los objetivos de los tópicos de interés de este sistema es que sean informativos.

3.5.3 TF-IDF

Este módulo es muy importante, porque es el que le asigna el ponderador final al tópico de interés. Para hacerlo se utiliza el esquema de ponderación de términos *TF-IDF*. Para esto se utiliza el `contador`, obtenido en el submódulo Agregador, y la frecuencia en todos los documentos (en este caso `Listas`) de cada tópico.

El objetivo es dotar a cada tópico de una medida de importancia. Lo que se se intenta reflejar con ese esquema de ponderación es que si un tópico es muy común en las listas de Twitter, entonces es poco representativo. Por otro lado, si el tópico es poco común, entonces tiene gran poder de segmentación.

Tópicos	
getlistomatic	tweets
lista	otros
gente	seguidores
seguidores	favoritos
cuentas	instituciones
usuarios	organizaciones
amigos	varios
cosas	twitter
cuentas	tuiteros
compañías	hecho
personajes	mis seguidores
personas	twitteros
⋮	⋮

Tabla 3.1: Lista Negra

Fuente: Elaboración propia

En resumen, mientras más común es el tópico, menor será su importancia. A su vez, mientras mayor sea la frecuencia de este tópico para el usuario, su importancia será mayor. Se refiere al lector a la Sección 2.4.2 para más información sobre TF-IDF.

3.6 API

La API es una interfaz que permite relacionar el sistema de obtención de tópicos de interés con aplicaciones externas. Se refiere al lector a la Sección 2.7, para más información sobre APIs en general. Además de ser una interfaz, funciona como un orquestador, ya que hace llamados a distintos módulos dependiendo de la solicitud. En el caso de este trabajo, la API tiene 2 funciones.

Obtener Tópicos de Interés

Si la solicitud es a la función de Obtener Tópico de Interés, la API se conecta directamente con el Módulo de Obtención de Tópicos de Interés. Este se conecta con la base de datos y, previo procesamiento de los datos, entrega los tópicos de interés a la API.

Agregar Usuario al sistema

Cuando la solicitud va a la función de Agregar usuario, la API orquesta todos los módulos del sistema, como se ve en la Figura 3.1. En este caso se siguen los siguientes pasos:

1. Se envía el `id` del usuario u a caracterizar al Módulo de Extracción de Datos de Twit-

ter, el que retorna una lista con los amigos de *u*.

2. Se ingresa la lista de amigos al Módulo de Procesamiento de Listas.
3. Se ingresa la lista de amigos al Módulo de Obtención de Tópicos de Influencia.
4. Por último, se ingresa la lista de amigos al Módulo de Obtención de Tópicos de Interés. Este módulo retorna la lista de tópicos que son de interés para el usuario *u*.

Para entender a fondo el desarrollo de la API, se refiere al lector a la sección 4.6.

3.7 Módulo de Visualización

El Módulo de Visualización es un anexo al sistema de Identificación de Tópicos de Interés. Está pensado para permitir a un usuario visualizar los resultados y para dar un ejemplo de uso, pero el objetivo del sistema es proveer de información a aplicaciones, por medio de la API. En la Figura 3.7 se muestra un esquema de su funcionamiento.

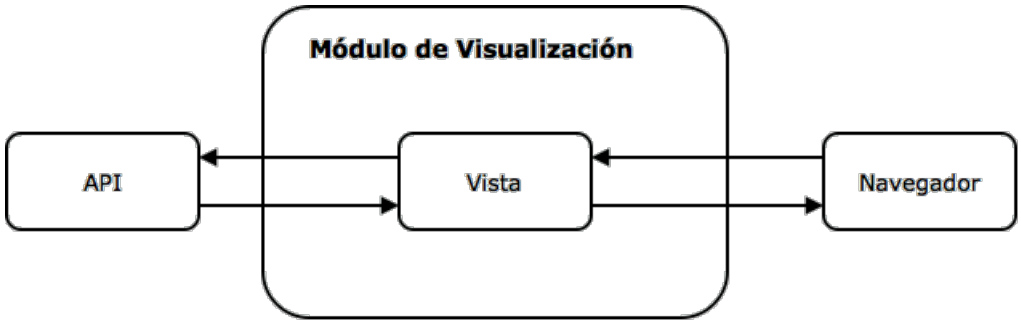

Figura 3.7: Módulo de Visualización

Fuente: Elaboración propia

Este módulo está compuesto de sólo un submódulo, la Vista. El usuario puede interactuar con el sistema únicamente por medio del navegador. El módulo de Vista es el encargado de traducir los requerimientos del Usuario a algo que la API pueda entender, y hacer los requerimientos a esta. A su vez es el encargado de dar formato a las respuestas de la API para que el navegador lo pueda entender y mostrar al usuario.

Como se ve en la Figura 3.8, el usuario tiene 2 opciones. Puede obtener la nube de tópicos de interés (de un usuario insertado con anterioridad), o agregar un usuario al sistema.

El la Figura 3.9 se muestra el resultado de cualquiera de las 2 solicitudes. Las 2 funciones de la API retornan los tópicos de interés del usuario, por lo que la página para desplegar los resultados es la misma. Sin embargo, el proceso de fondo es muy distinto, así como el tiempo que toma cada solicitud. En este módulo, el resultado se muestra como una nube de palabras, en la que el tamaño de la palabra indica su relevancia para el usuario. Otra opción para entregar resultados podría ser una tabla con el valor numérico de cada tópico, pero se eligió esta representación porque se refleja la relevancia de una forma más gráfica.

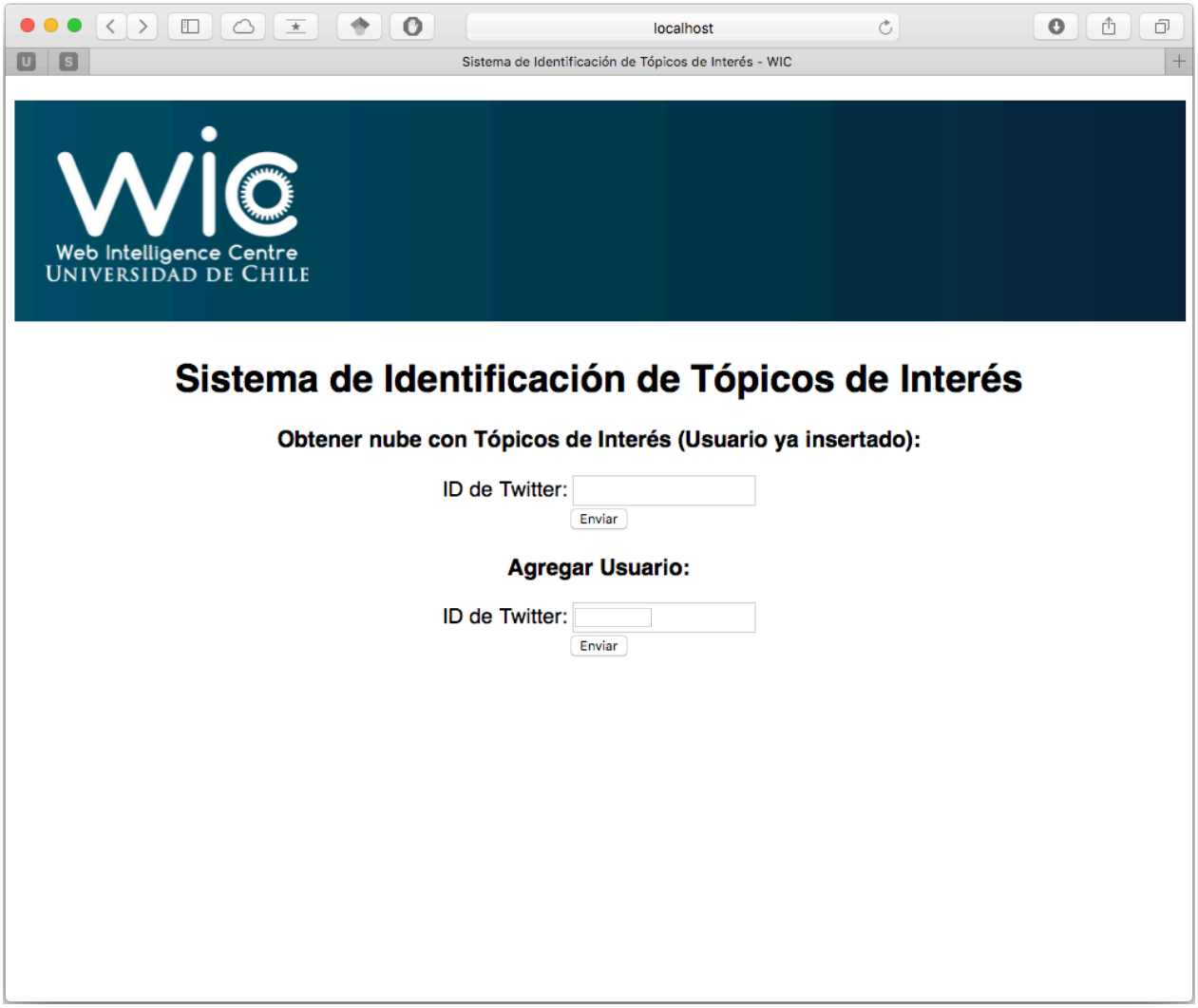

Figura 3.8: Landing page de Módulo de Visualización

Fuente: Elaboración propia

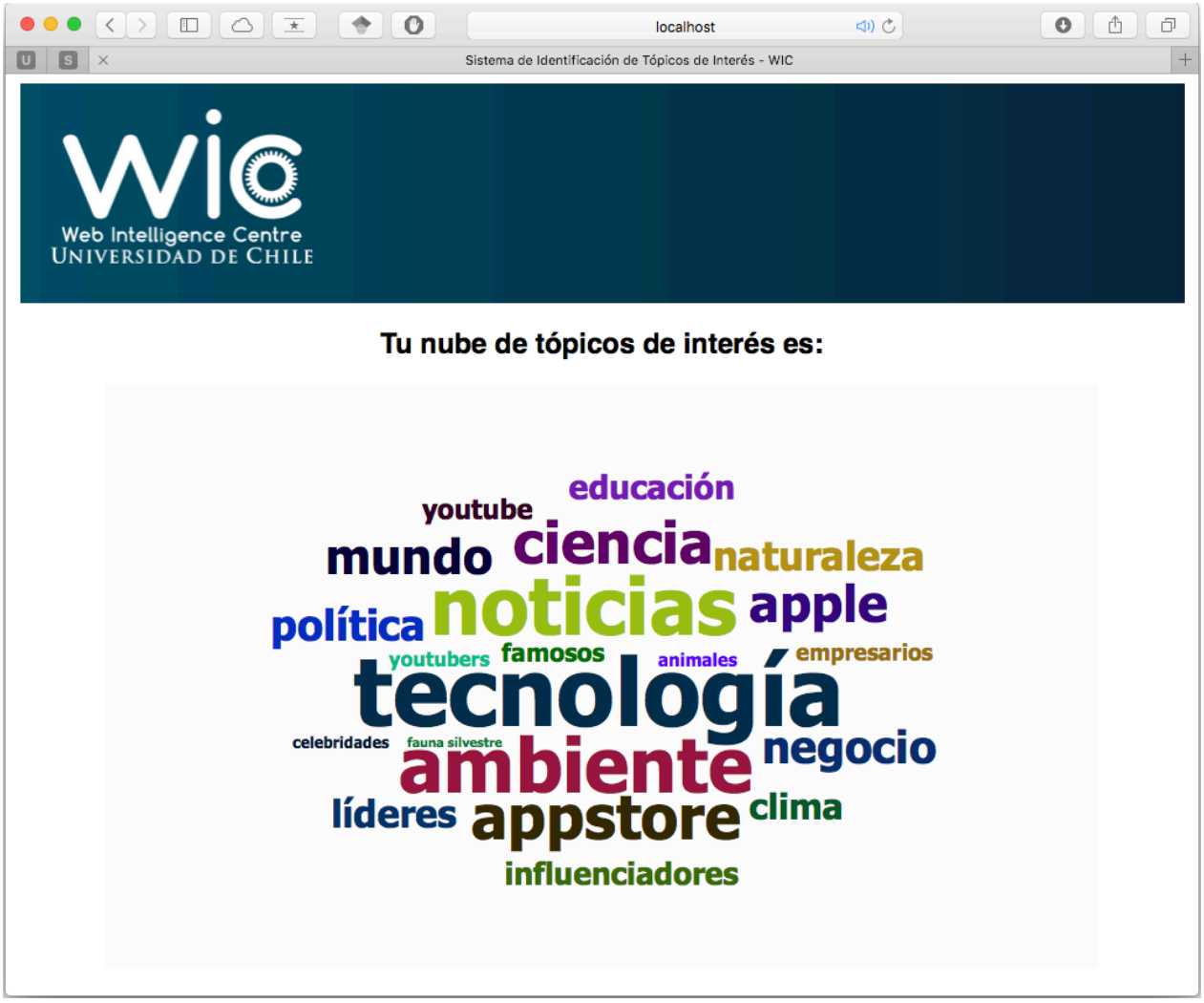

Figura 3.9: Página de resultado de Módulo de Visualización

Fuente: Elaboración propia

Capítulo 4

Desarrollo del sistema

Este capítulo pretende mostrar qué es lo que hace la aplicación y cómo lo hace, por lo que la información mostrada es de un nivel más técnico que el anterior capítulo. Está pensado para entender en detalle los algoritmos utilizados y cómo fue programada la solución.

El capítulo está estructurado de la siguiente manera: En la primera sección de detallan las herramientas tecnológicas utilizadas. Las siguientes 4 secciones se refieren a los módulos que se pueden ver en la Figura 3.1 del capítulo anterior. Lo sigue una sección que detalla el funcionamiento de la API desarrollada y se termina con la implementación de esta en un módulo de visualización.

4.1 Herramientas tecnológicas

4.1.1 Java

El lenguaje de programación utilizado para el desarrollo del sistema es Java[1]. Es un lenguaje orientado a objetos, basado en clases y de uso general.

Fue desarrollado por James Gosling mientras trabajaba en Sun Microsystems, que fue adquirida posteriormente por Oracle. Tiene parentescos con los lenguajes C y C++. Es considerado un lenguaje de relativamente alto nivel (especialmente en comparación a C y C++). Las aplicaciones en java son compiladas y pueden ser ejecutadas en cualquier sistema operativo.

Se eligió este lenguaje por la portabilidad entre distintos sistemas, por su robustez y por el amplio respaldo en la industria.

[1] `http://docs.oracle.com/javase/8/docs/`, visitada el 23 de noviembre de 2016.

4.1.2 PHP

PHP[2] es un lenguaje de programación de código de lado del servidor diseñado principalmente para desarrollo web, pero también es usado como un lenguaje de uso general. El código PHP puede ser incluido directamente en el HTML. Eso uno de los lenguajes más utilizados para el desarrollo web, con sitios como Wikipedia y Facebook que lo utilizan, y plataformas como Wordpress.

Este lenguaje es utilizado sólo en el módulo de visualización de este trabajo, a modo de ejemplo de uso del sistema. Se eligió por su facilidad de uso y por el amplio soporte en la industria.

4.1.3 MariaDB

MariaDB[3] es un servidor de bases de datos que está dentro de los más populares del mundo. Es desarrollado por los desarrolladores originales de MySQL[4] y es garantizado que permanecerá con código abierto. Usuarios notables incluyen a Wikipedia, Facebook y Google.

MariaDB convierte los datos en información estructurada en un amplio rango de aplicaciones, desde bancos hasta sitios web. Es una mejora a MySQL, que está hecho para que sea fácil migrar a el. MariaDB es usado porque es rápido, escalable y robusto, con un rico ecosistema de motores de almacenamiento, plugins y muchas otras herramientas que lo hacen muy versátil para una gran variedad de usos.

MariaDB es desarrollado como un software de código abierto y como una base de datos relacional que provee una interfaz SQL para el acceso a los datos. Las últimas versiones de MariaDB también incluyen acceso a funcionalidades GIS y JSON [55].

Se utiliza este motor por su respaldo en la industria y su facilidad de uso.

4.1.4 REST API de Twitter

La REST API de Twitter provee acceso, a través de código, para escribir o leer datos de Twitter [56]. La API funciona con *credenciales*, que están relacionados a aplicaciones que se pueden crear en el sitio de desarrolladores de Twitter[5]. Estas aplicaciones son identificadas por medio de *OAuth*. Cada credencial tiene un límite de consultas dentro de un intervalo de 15 minutos, que depende de la consulta. La API tiene muchas funcionalidades, pero las utilizadas en este trabajo, junto con sus límites, se muestran en la Tabla 4.1. Todas las respuestas a las solicitudes son entregadas en formato JSON.

[2] `http://php.net`. Visitada el 9 de Agosto, 2016.
[3] `https://mariadb.org/`. Visitada el 9 de Agosto, 2016.
[4] `http://www.mysql.com`. Visitada el 9 de Agosto, 2016.
[5] `https://dev.twitter.com`. Visitada el 9 de Agosto, 2016.

Consulta	Límite de consultas	Resultados por consulta
Amigos del usuario	15	5000
Miembros de una Lista	75	1000
Estado de credencial	180	1

Tabla 4.1: Límites REST API de Twitter

Fuente: Elaborado a partir de [57]

4.1.5 Twitter4J

Twitter4J es una librería no oficial, desarrollada para Java, para integrar la API de Twitter en aplicaciones [58]. Esta librería está compartida bajo la licencia Apache 2.0. Dentro de sus características destaca que es gratis y de código abierto, es compatible con versiones de Java 5 y superior, e incorpora soporte para *OAuth*, el sistema de verificación de credenciales que utiliza la API de Twitter.

Esta librería se utiliza en el Módulo de Extracción de Datos de Twitter, especialmente para obtener los amigos de un usuario, las listas a las que pertenecen ellos, la información de las listas y para la identificación por medio de credenciales con la API.

Se utiliza esta librería porque tiene todas las funciones que se requieren en este trabajo para conectarse a la API de Twitter, y porque es una de las más utilizadas, por lo que el respaldo en la comunidad es amplio.

4.1.6 Wordcloud2.js

Wordcloud2.js[6] es una librería, implementada en Javascript, para crear nubes de palabras en HTML a partir de JSON. Es desarrollada por Timothy Guan-tin Chien y compartida bajo la licencia de MIT.

Esta librería es utilizada en el Módulo de Visualización por ser una forma amigable de representar los resultados.

4.1.7 Freeling

FreeLing [59] es la herramienta utilizada para el análisis de texto. El proyecto FreeLing fue creado y es actualmente liderado por Lluís Padró cómo un medio para hacer disponible a la comunidad los resultados de la investigación llevada a cabo por el grupo de investigación de procesamiento de lenguaje natural de la Universidad Politècnica de Catalunya[7].

FreeLing es una librería en C++ que tiene funcionalidades de análisis del lenguaje para muchos idiomas (entre ellos Inglés y Castellano). Estas son algunas de sus funciones:

[6] `https://github.com/timdream/wordcloud2.js` Visitada el 15 de Septiembre, 2016

[7] `http://www.talp.upc.edu`, visitada el 23 de noviembre de 2016.

- Análisis morfológico.
- Detección de entidades nombradas.
- Etiquetado gramatical (más conocido por el inglés PoS-tagging).
- Tokenización.
- Detección de idioma.
- Lematización.

Se utilizó esta librería por los buenos resultados que entrega en español, por su soporte de múltiples idiomas y por ser de código abierto.

4.1.8 Netbeans

Para el desarrollo se utilizó el IDE (sigla en inglés para Entorno de Desarrollo Integrado) Netbeans 8.1 [60]. Este software es creado por Oracle y entrega herramientas para la edición y análisis de código para trabajar con las últimas tecnología de Java, entre otros. Es compatible con proyectos Maven y permite correr el código en el editor. También es compatible con GIT[8], lo que lo hace muy útil para código utilizado en varios equipos y personas, además de poder tener un respaldo del trabajo y de guardar versiones.

4.1.9 Maven

Es una herramienta que puede ser usada para construir y gestionar cualquier proyecto basado en Java. El objetivo principal de Maven es permitir que un desarrollador comprenda el estado completo de un proyecto de desarrollo en el menor tiempo posible. Para lograr este objetivo hay varias áreas importantes que Maven intenta manejar:

- Hacer el proceso de compilación y chequeo de dependencias más fácil.
- Proveer un sistema uniforme de compilación y chequeo de dependencias.
- Proveer información de calidad del proyecto.
- Proveer guías para desarrollo utilizando mejores prácticas.
- Permitir una migración transparente a nuevas funcionalidades.

Para describir el proyecto a construir y sus dependencias de otros módulos o componentes externos se utiliza el Project Object Model (POM). Otra característica muy importante de Maven es que permite descargar plugins desde la web, lo que hace que sea fácil tener

[8]GIT es un sistema de control de versiones gratuito y de código abierto, diseñado para manejar proyectos de software tanto pequeños como muy grandes, con rapidez y eficiencia. Más información en `https://git-scm.com`, visitada el 23 de noviembre de 2016.

acceso y usar proyectos de código abierto en Java.

En este trabajo se utiliza principalmente para integrar fácilmente librerías al código en Java.

4.1.10 Spring

Spring es un *framework* que provee un modelo de programación y configuración para aplicaciones basadas en Java [61]. Un elemento clave de Spring es su soporte de infraestructura a nivel de la aplicación, lo que da facilidades al desarrollador para enfocarse en la lógica de negocio. Algunas de sus características son:

- Inyección de dependencias
- Programación Orientada a Aspectos incluyendo la gestión de transacciones declarativa de Spring
- Spring MVC: framework de aplicaciones web y servicios web REST
- Soporte para JDBC, JPA, JMS

Spring se utilizará en este trabajo para crear la API. Se utilizará Spring MVC para hacer la interfaz entre la aplicación, programada en Java, y un sitio web. Se optó por utilizar Spring por su respaldo, facilidad de uso e instalación. En particular se utilizó Spring Boot[9], que hace que el *setup* sea más rápido y sencillo, pero permite menos configuración.

4.2 Extracción de Datos de Twitter

En esta sección se describe la extracción de datos de Twitter y su almacenamiento en la base de datos, como fue explicado en la sección 3.2 y en la Figura 3.2. El las siguientes 2 secciones se explicará con mayor detalle el funcionamiento de los 2 submódulos: El de obtención de amigos y el de obtención de listas.

4.2.1 Obtención de amigos

Este módulo es el que obtiene los *amigos* de un usuario *u* de Twitter. Para obtener los amigos de *u* se utiliza la REST API de Twitter.

La lógica de la obtención de amigos se puede ver en el Algoritmo 4.1. Los datos requeridos son el `id` de Twitter del usuario que se quiere caracterizar. El módulo entrega como resultado un arreglo con los `ids` de los amigos *u*. En la línea 1 se inicializa el `cursor`, que es lo que indica el inicio, el punto de referencia, para hacer la consulta a la API. El -1 indica que es el comienzo y el 0 que no quedan más resultados. En la siguiente línea se inicializa un

[9] `http://projects.spring.io/spring-boot/` Visitada el 13 de Septiembre, 2016

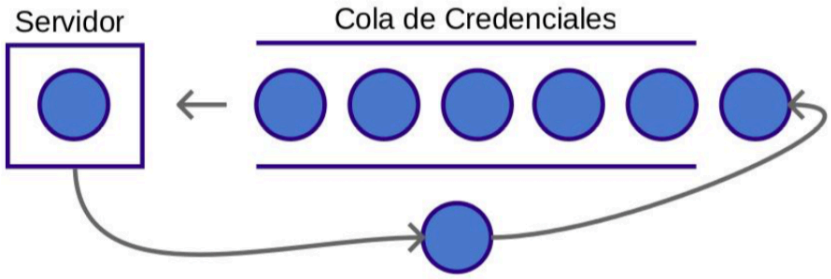

Figura 4.1: Cola de credenciales de Twitter

Fuente: Cortés [62]

Data: id: el identificador de Twitter del usuario u a caracterizar
Result: $amigos$: la lista de amigos de Twitter del usuario u

```
1  cursor = −1;
2  amigos ← se inicializa un arreglo vacío;
3  while cursor ≠ 0 do
4  |  if a credencial no le quedan consultas then
5  |  |  obtener nuevo credencial;
6  |  end
7  |  amigos' ← se obtienen amigos de u con cursor y credencial;
8  |  se agrega amigos' al arreglo amigos;
9  |  cursor ← se obtiene el nuevo cursor;
10 end
11 foreach amigo ∈ amigos do
12 |  se inserta amigo en la base de datos, si es que no estaba ;
13 end
14 return amigos;
```

Algorithm 4.1: Obtención de amigos de Twitter.

arreglo vacío donde se almacenarán los amigos que se obtendrán en los siguientes pasos. La línea 3 es un *while* que hace que las líneas 4-9 se ejecuten mientras queden resultados por obtener. Las líneas 4-6 revisan si el credencial es válido para la consulta, y si no lo es se obtiene uno nuevo y se mueve el anterior al final de la cola. En la línea 7 se hace una consulta para obtener los `amigos` de *u* a la API de Twitter, utilizando el credencial válido y el `cursor`. La línea 8 agrega los `ids` del arreglo `amigos'` al arreglo `amigos`, que es el que será retornado al terminar el módulo. El último paso de la iteración obtiene el `cursor`, o posición de referencia, donde terminó la última consulta. Al terminar la iteración se han obtenido todos los `amigos` del usuario *u* y ahora se deben insertar en la base de datos. En las lineas 11-13 se recorre el arreglo `amigos`, y para cada `amigo` se realizan 2 consultas SQL:

```
(4.1)  INSERT IGNORE INTO Amigos(TwitterId) VALUES (id)
```

Donde `Amigos` es la tabla en la que se almacenan los `id` de todos los amigos de usuarios que pasan por el sistema. El `IGNORE` se utiliza porque no se quiere agregar un usuario repetido a la tabla y es necesario que el programa siga ejecutándose.

```
   Data: amigos : lista de amigos del usuario u
   Result: Ninguno. Se rellena una base de datos en cada loop.
 1 amigosInsertados ←: arreglo ordenado de ids de los amigos con listas ya insertadas;
 2 listas ← inicializar un arreglo vacío;
 3 cursor = −1;
 4 foreach amigo ∈ amigos do
 5     if amigo ∉ amigosInsertados then
 6         while cursor ≠ 0 && tamaño de listas ≤ 2000 do
 7             if a credencial no le quedan consultas then
 8                 obtener nuevo credencial;
 9             end
10             listas′ ← obtener Listas a las que amigo pertenece con cursor y credencial;
11             se agrega listas′ al arreglo listas;
12             cursor ← se obtiene el nuevo cursor;
13         end
14         if amigo es miembro de alguna lista then
15             foreach lista ∈ listas do
16                 insertar lista en tabla de Listas;
17                 insertar amigo y lista en tabla de relación Listas_Amigos;
18             end
19         else
20             insertar amigo y listaDummy en tabla de relación Listas_Amigos;
21         end
22     end
23 end
```

Algorithm 4.2: Obtención de listas de Twitter.

(4.2) `INSERT INTO AmigoUsuario(idUsuario, idAmigo) VALUES (idU,id)`

`AmigoUsuario` es la tabla donde se almacena la relación entre el usuario con `idUsuario = idU` y el amigo con `idAmigo = id`. Cuando se terminan todas estas inserciones, el módulo termina retornando el arreglo `amigos`, el que será utilizado en el módulo de obtención de listas.

4.2.2 Obtención de Listas

Este módulo sigue al Módulo de Obtención de Listas y utiliza su resultado, un arreglo con los `amigos` del usuario *u*. La lógica de la obtención de listas se puede ver en el Algoritmo 4.2. En la línea 1 se obtienen las `ids` de Twitter de los usuarios que ya fueron procesados por el sistema, en caso de que el sistema se haya utilizado para algún usuario antes de *u*. Esto se realiza mediante la siguiente consulta SQL:

(4.3) `SELECT DISTINCT idAmigos FROM Listas_Amigos ORDER BY idAmigos ASC`

Luego, en la línea 2, se inicializa el arreglo donde se irán guardando las listas extraídas desde Twitter. En la línea 3 se crea el `cursor` que, al igual que en el módulo anterior, es lo que indica a la API de Twitter la posición de la consulta. En el caso de valer -1 significa que está en el comienzo.

Posteriormente, en la línea 4, se inicia un *for* que itera sobre todos los `amigos`. En la línea 5 se impone la condición de que el `amigo` que se va a procesar no haya sido procesado con anterioridad. Esta restricción hace que el módulo funcione más rápido y que no se utilicen credenciales de más.

En las líneas 6-13 es dónde se extraen las `listas` de Twitter. En particular, en la línea 6 se inicia un *while* con la condición de que queden `Listas` por obtener de la API y que no se sigan obteniendo si número de `Listas` obtenido sobrepasa los 2000 [50]. En la línea 7 se revisa el estado del credencial, y en caso de que no le queden consultas se cambia por el siguiente y el ya utilizado se posiciona al final de la cola. La obtención de `Listas` se realiza en la línea 10, donde a la API de Twitter se le entrega el `id` del `amigo`, el `cursor` y un `credencial` válido. En este paso se decidió solicitar a lo más 700 listas a la API puesto que la tasa de error en la respuesta del servidor era muy alta cuando el número era mayor. En la línea 11 se agregan las `Listas` obtenidas al arreglo *listas*. El último paso de esta iteración es obtener el `cursor`, y si este es distinto de 0 se volverán a obtener `listas` de la misma manera ya explicada.

Si no quedan más `Listas` que obtener, o el número obtenido es mayor o igual a 2000, se procede a insertar las `Listas` y la relación entre estas y los `amigos` a la base de datos. En la línea 14 se revisa si el usuario es miembro de alguna `Lista`, y en ese caso se itera sobre cada una de las `Listas` y se realizan 2 consultas SQL para cada una:

(4.4) `INSERT IGNORE INTO Listas(idTwitter, Titulo, Descripcion) VALUES (id, titulo, descripcion)`

En esta consulta se inserta en la tabla *Listas* el `id` de Twitter de la `Lista`, junto con su `título` y `descripción`. Dentro de la consulta se incorpora el término `IGNORE` para que no se repita la inserción de una `Lista` e ignorar el error si esto pasa.

(4.5) `INSERT INTO Listas_Amigos(idListas, idAmigos) VALUES (idLista, idAmigo)`

En esta consulta se almacena la relación de tiene el `amigo` de `id = idAmigo` con la `Lista` de `id = idLista`.

Hay muchos usuarios que no son miembros de ninguna lista, por lo que la condición de la línea 14 no se cumplirá siempre. Para hacer más rápido el procesamiento de futuros usuarios que sigan a `amigo`, se realiza la siguiente consulta:

(4.6) `INSERT INTO Listas_Amigos(idListas, idAmigos) VALUES (1, idAmigo)`

Dónde se sabe que si un usuario *pertenece* a la `Lista` con `idListas = 1`, entonces el usuario no tiene tópicos de influencia. Esto será utilizado en la sección 4.4.

```
   Data: Amigos: usuarios de Twitter seguidos por el usuario u.
   Result: Ninguno. Se rellena una base de datos en cada loop.
 1 amigosProcesados ← obtener amigos con listas procesadas;
 2 listasProcesadas ← obtener listas procesadas con anterioridad;
 3 foreach amigo ∈ Amigos do
 4     if amigo ∉ amigosProcesados then
 5         listas ← obtener listas de amigo;
 6         foreach lista ∈ listas do
 7             if lista ∉ listasProcesadas then
 8                 topicos = procesarTexto(lista);
 9                 foreach topico ∈ topicos do
10                     insertar topico en base de datos;
11                 end
12             end
13             insertar relación de amigo y lista en base de datos;
14         end
15     end
16 end
```

Algorithm 4.3: Procesamiento de listas de Twitter.

4.3 Procesamiento de listas

En esta sección se detalla el desarrollo del Módulo de Procesamiento de Listas, presentado en la sección 3.3.1. Se explicará el funcionamiento general del módulo y luego se dará énfasis en el procesamiento del texto de las listas. En el Algoritmo 4.3 se presenta el funcionamiento general del módulo de procesamiento de listas de Twitter.

El módulo recibe de *input* un arreglo con los `ids` de los `amigos` de el usuario *u*. Estos son los usuarios de quienes se obtuvieron las `listas` a las que pertenecen en la sección 4.2.2. Se comienza, en la línea 1, por obtener un arreglo ordenado con los `ids` de todos los usuarios de Twitter que ya tienen procesadas sus `listas` y están insertadas en la base de datos. Posteriormente se obtiene otro arreglo ordenado de `ids` de Twitter, pero esta vez corresponde a las `listas` que ya han sido procesadas y tienen sus tópicos insertados en la base de datos. En la línea 3 se comienza la iteración que recorrerá a todo el arreglo de `amigos` de *u*. Luego, se impone la condición de que el `amigo` no debe haber sido procesado con anterioridad. Esta restricción tiene la finalidad de disminuir el tiempo total que toma el paso por este módulo. En la línea 5 se realiza una consulta a la base de datos, para obtener los datos de las `listas` del `amigo`. La consulta SQL es la siguiente:

(4.7)
```
SELECT idTwitter, Titulo, Descripcion FROM Listas, Listas_Amigos WHERE
Listas.idTwitter = Listas_Amigos.idListas AND idAmigos = var1
```

Dónde `var1` es el `id` de Twitter del `amigo` de quién se están obteniendo las `listas`. A continuación, en la línea 6 se inicia una iteración que recorrerá todas las `listas` obtenidas en la consulta anterior. Sin embargo, si la `lista` es parte del arreglo $listasProcesadas$, como

se ve en la línea 7, entonces simplemente se ignora y se sigue con la `lista` siguiente. Esto se hace para disminuir el tiempo de ejecución del módulo, al no procesar `listas` más de una vez. En la línea 8 se toma el texto de la `lista` y es procesado. El procesamiento de texto se explica en detalle en la sección 4.3.1. El resultado del procesamiento de texto son los `tópicos` de la `lista` con un ponderador, que indica la importancia relativa de cada `tópico`. Se itera sobre cada uno de estos `tópicos`, como se ve en la línea 9-10, y luego se procede a la inserción a la base de datos, la que es realizada con las siguientes 2 consultas SQL:

(4.8) `INSERT INTO Topicos(Topico) VALUE(var1) ON DUPLICATE KEY UPDATE Topico = Topico`

Dónde `var1` es el tópico a insertar. En esta consulta se inserta el tópico a la tabla que almacena todos los tópicos. Esta consulta está pensada para que éstos no se repitanen la tabla, por lo que si se intenta insertar uno que ya está en ella, simplemente se omite.

(4.9) `INSERT INTO Listas_Topicos(Listas_idListas, Topicos_idTopicos, Peso) VALUES (idLista, idTopico, peso) ON DUPLICATE KEY UPDATE Peso=Peso+peso`

Dónde `idLista` es el `id` de Twitter de la `lista` procesada, `idTopico` es el `id` del tópico en la tabla `Topicos`, y `peso` es el ponderador que indica la importancia del tópico en la `lista`. En esta consulta, lo que se hace es guardar tópico y la importancia que tiene éste. La consulta está pensada para que cada dupla (`lista, topico`) sea única, pero que si un `tópico` aparece más de una vez en la `lista`, entonces sus ponderadores se sumen.

4.3.1 Procesamiento de texto

En esta sección se detalla el desarrollo del módulo de Procesamiento de Texto. Esta es la etapa nombrada en la sección 4.3, en particular en el Algoritmo 4.3 línea 8. Se muestra un esquema general en la Figura 3.4. El módulo cuenta con partes desarrolladas en este trabajo y otras en las que se utilizan APIs de terceros, como FreeLing y Google. En el algoritmo 4.4 se observa la lógica del módulo de procesamiento de texto. Se refiere al lector a la sección 3.3.1 para ver ejemplos de cada uno de los pasos del algoritmo.

El primer paso del algoritmo, como se ve en la línea 1, es tokenizar el título de la lista. Esta es una tokenización previa que es muy importante identificar correctamente el idioma en un paso posterior, ya que aplica reglas que no están incluidas en la tokenización de FreeLing. Es realizada utilizando varios *regex*, o expresiones regulares, en *Java*, como se ve en la Tabla 4.2. Cada vez que se cumple la expresión regular, se inserta un espacio en blanco (en el caso de *CamelCase*) o se reemplaza el término por un espacio en blanco (en todos los otros casos).

El tokenizador previo entrega un *string* que se concatena con la descripción, para obtener un sólo *string* que contiene todo el texto, como se ve en la línea 3. Es importante mencionar que el título y descripción se separan con el caracter "±", para poder ser separados en pasos posteriores. La unión del título y descripción es realizada para mejorar

```
   Data: lista: tupla que contiene (id, titulo, descripcion).
   Result: topicos: arreglo de tuplas (topico, peso).
 1 titulo ← tokenizar titulo de lista;
 2 descripcion ← descripcion de lista;
 3 texto ← concatenar titulo con descripcion;
 4 texto ← realizar limpieza de texto;
 5 idioma ← identificar idioma de texto;
 6 if idioma = castellano then
 7 |   l ← tokenizar texto en Castellano;
 8 |   ls ← separar l en oraciones en Castellano;
 9 |   ls ← análisis morfológico en Castellano de ls ;
10 |   ls ← PoS tags en Castellano de ls;
11 |   sustantivos ← obtener sustantivos de ls en Castellano;
12 |   foreach sustantivo ∈ sustantivos do
13 |   |   if sustantivo ∈ titulo then
14 |   |   |   peso ← calcular peso de sustantivo;
15 |   |   else
16 |   |   |   peso ← calcular peso de sustantivo;
17 |   |   end
18 |   end
19 else
20 |   l ← tokenizar texto en Inglés;
21 |   ls ← separar l en oraciones en Inglés;
22 |   ls ← análisis morfológico en Inglés de ls ;
23 |   ls ← PoS tags en Inglés de ls;
24 |   sustantivos ← obtener sustantivos de ls en Inglés;
25 |   sustantivosTraducidos ← traducir sustantivos al castellano;
26 |   foreach sustantivo ∈ sustantivos do
27 |   |   if sustantivo ∈ titulo then
28 |   |   |   peso ← calcular peso de sustantivo;
29 |   |   else
30 |   |   |   peso ← calcular peso de sustantivo;
31 |   |   end
32 |   end
33 end
34 return topicos;
```

Algorithm 4.4: Procesamiento de texto.

la identificación de idioma de FreeLing, pero deben ser separados posteriormente para la asignación de ponderadores.

En la línea 4 se aplica la limpieza de texto, que en realidad es eliminar los "#". Luego, como se ve en la línea 5, se identifica el idioma del texto. Esto se realiza mediante la función `identifyLanguage()` de la API para Java de FreeLing, y retorna un *string*. Para este trabajo se configuró para que retorne "es" o "en", para castellano e inglés, respectivamente.

Tipo de texto	Regex
CamelCase	`(?<!(^|[A-Z]))(?=[A-Z])|(?<!^)(?=[A-Z][a-z])`
Signo más (+)	`\\+\\s*`
Coma (,)	`,\\s*`
Punto (.)	`\\.\\s*`
Slash (/)	`/\\s*`
Guión (-)	`-\\s*`
Guión bajo (_)	`_\\s*`
Arroba (@)	`@\\s*`

Tabla 4.2: Expresiones regulares utilizadas en tokenización previa

Fuente: Elaboración propia

En la línea 6 se aplica una condición que divide el resto del procesamiento de texto en las funciones específicas de cada idioma. Los primeros 4 pasos para cada idioma (líneas 7-10 y 16-19) son funciones de FreeLing, y para efecto de este trabajo se considerarán como *caja negra*, ya que no es realmente importante cómo funcionan, sino el resultado que entregan. Las funciones utilizadas son las siguientes:

- `tokenize()`: esta función se aplica sobre un objeto de clase `Tokenizer` de FreeLing y se le da de *input* el texto a tokenizar. Retorna un objeto de clase `ListWord` con el texto tokenizado. Lo que indica en qué idioma se va a tokenizar es el objeto `Tokenizer`.

- `split()`: esta función se aplica sobre un objeto de clase `Splitter` de FreeLing y se le da de *input* el objeto de clase `ListWord` obtenido de la función `tokenize()`. Retorna un objeto de clase `ListSentence` con el texto tokenizado separado en oraciones. Lo que indica en qué idioma se van a segmentar las oraciones es el objeto `Splitter`.

- `analyze()`: esta función se aplica sobre un objeto de clase `Maco` de FreeLing y se le da de *input* el objeto de clase `ListSentence` obtenido de la función `split()`. El resultado el el mismo objeto `ListSentence` entregado, pero ahora con información adicional (como detección de números, fechas, puntuación, etc.). Lo que indica en qué idioma se va a realizar el análisis morfológico es el objeto `Maco`.

- `analyze()`: esta función se aplica sobre un objeto de clase `HmmTagger` de FreeLing y se le da de *input* el objeto de clase `ListSentence` obtenido del análisis morfológico. El resultado el el mismo objeto `ListSentence` entregado, pero además se incorporan etiquetas gramaticales. Lo que indica en qué idioma se va a realizar el etiquetado gramatical es el objeto `HmmTagger`.

En la línea 11 y 24 se separan los sustantivos del resto de los tokens, utilizando las etiquetas gramaticales. En ambos casos se eligen los *tokens* con una etiqueta que empiece con “N”, lo que indica que es un sustantivo. Como parte del análisis morfológico, FreeLing identifica los lemas de los *tokens*. Aprovechando este análisis previo, se obtienen los lemas y no el *token* original.

Hay un paso que es exclusivo para las listas en Inglés, como se ve en la línea 25, que es

traducir los tópicos al castellano. Para hacer esto se utiliza la API de Java de el traductor de idioma de Google[10].

El último paso de este módulo, realizado en las líneas 12-18 y 26-32, es asignar los ponderadores de importancia, o pesos, a cada uno de los tópicos obtenidos. Cada lista tiene un peso total que es igual a 100. Este peso se divide 50 % al título y 50 % a la descripción, es decir 50 y 50. Dentro de título se dividen los 50 en partes iguales entre todos los sustantivos obtenidos, y lo mismo se hace en la descripción.

4.4 Identificación de Tópicos de Influencia

En esta sección se detalla el desarrollo del módulo de Identificación de Tópicos de Influencia, presentado en la sección 3.4. El objetivo es identificar los tópicos que los usuarios de Twitter utilizan para clasificar a los usuarios famosos, y la importancia relativa entre los tópicos. En el algoritmo 4.5 se puede apreciar la lógica de de este módulo.

```
   Data: amigos: arreglo de ids de Twitter a ser procesados.
   Result: Ninguno. Se rellena una base de datos en cada loop.
1  amigosInsertados ← arreglo ordenado de amigos ya procesados;
2  foreach amigo ∈ amigos do
3  |  if amigos ∉ amigosInsertados then
4  |  |  tuplas ← se obtienen tópicos e importancia de amigo;
5  |  |  foreach tupla ∈ tuplas do
6  |  |  |  se inserta a la DB topico, peso, contador y la relación con amigo;
7  |  |  end
8  |  end
9  end
```

Algorithm 4.5: Módulo de Identificación de Tópicos de Influencia

El módulo recibe de *input* un arreglo de `ids` de Twitter. Estos son lo *amigos* de quienes se obtendrán los tópicos de influencia. En la línea 1 del algoritmo 4.5 se obtiene un arreglo con los usuarios con tópicos de influencia ya identificados en ejecuciones anteriores del sistema. Este arreglo es luego utilizado en la línea 3 para hacer más rápida la ejecución del módulo. Para obtener este arreglo, se hace la siguiente consulta SQL a la base de datos:

```
(4.10)  SELECT DISTINCT idAmigos FROM Amigos_Influencia GROUP BY idAmigos
        ORDER BY idAmigos ASC
```

Luego, en la línea 2, se itera sobre cada uno de los `amigos` y, en la línea 3, se impone la condición de que el `amigo` no debe haber sido procesado con anterioridad. Esto se hace comparando el `id` con los valores del arreglo `amigosInsertados`.

[10] `https://developers.google.com/api-client-library/java/apis/translate/v2` Visitada el 26 de Septiembre, 2016

Para los amigos que cumplen la condición, se obtienen tuplas de los tópicos de influencia y su información. Las tuplas tienen la siguiente estructura:

$$(id_i, topico_i, peso_i, contador_i) \tag{4.1}$$

Dónde id_i corresponde al id del tópico i en la base de datos, $topico_i$ es el tópico en si, $peso_i$ es la suma de los pesos en todas las ocurrencias del tópico para el usuario, y $contador_i$ es la frecuencia del tópico para el usuario. Para obtener estas tuplas se realiza la siguiente consulta SQL:

(4.11)
```
SELECT count(Listas_Topicos.Peso) as count, sum(Listas_Topicos.Peso)
as peso, Topico, idTopicos FROM Topicos, Listas_Topicos, Listas,
Listas_Amigos WHERE Topicos.idTopicos = Listas_Topicos.idTopicos AND
Listas_Amigos.idListas = Listas.idTwitter AND Listas_Topicos.idListas
= Listas_Amigos.idListas AND Listas_Amigos.idAmigos = var1 GROUP BY
Topico ORDER BY peso DESC
```

La consulta 4.11 es bastante compleja y es lo que toma más tiempo en todo el sistema. Como se puede ver, relaciona 4 tablas de la base de datos. Para cada tópico se realizan 2 operaciones: se obtiene la frecuencia y la suma del peso del tópico en todas las listas de las que el usuario es miembro. El resultado obtenido se ejemplifica en la Tabla 4.3, donde se muestran los primeros resultados para la cuenta del jugador de baloncesto Stephen Curry[11].

idTopicos	Topico	sum	count
856	nba	21032.15	259
862	deportes	17638.75	215
857	baloncesto	6083.72	82
932	guerreros	3229.17	42
861	atletas	3093.34	48
⋮	⋮	⋮	⋮

Tabla 4.3: Resultado Consulta 4.11

Fuente: Elaboración propia

Cada fila de esta tabla resultado es una de las tuplas mencionadas. Terminando con la explicación del algoritmo, lo que queda es recorrer toda la tabla de resultado y hacer la inserción a la base de datos, como se ve en las líneas 5-7. La siguiente consulta SQL realiza esta acción:

(4.12)
```
INSERT INTO Amigos_Influencia(idAmigos, idTopicos, Topico, Contador,
Peso) VALUES (var1, var2, var3, var4, var5)
```

Dónde var1 es el id de Twitter del usuario, var2 es el id del tópico en la tabla de tópicos, var3 es el tópico a insertar, var4 es el peso y var5 el la frecuencia del tópico.

[11] http://www.biography.com/people/stephen-curry Visitada el 11 de Septiembre, 2016

4.5 Identificación de Tópicos de Interés

En esta sección de detalla el desarrollo del módulo de Identificación de Tópicos de Interés, presentado en la sección 3.5. Este módulo es llamado por la API, enviando un `id` de un usuario de Twitter. En el algoritmo 4.6 se muestra la lógica de este módulo.

Data: $id \leftarrow$ el identificador de Twitter del usuario u a caracterizar
Result: $json \leftarrow$ archivo en formato JSON con tópicos e información
1 $tuplas \leftarrow$ se obtienen los $topicos$, $frecuencia$ e idf de u desde la DB;
2 $listaNegra \leftarrow$ lista con palabras que no aportan información;
3 $json \leftarrow$ se inicializa un JSON vacío;
4 **foreach** $topico \in topicos$ **do**
5 **if** $topico \notin listaNegra$ **then**
6 se agrega $topico$ a $json$;
7 $ponderador \leftarrow$ aplicar tf-idf a $frecuencia$;
8 se agrega $frecuencia$ a $json$;
9 **end**
10 **end**
11 **return** $json$

Algorithm 4.6: Módulo de obtención de tópicos de interés

Se comienza por realizar una consulta a la base de datos donde se obtienen tuplas de interés, las que tienen la siguiente estructura:

$$(topico_i, frecuencia_i, idf_i) \tag{4.2}$$

Dónde $topico_i$ es el tópico de interés, $frecuencia_i$ es la cantidad de veces que el tópico se repite dentro de las `listas` en las que los *amigos* de *u* son miembros, y idf_i es la frecuencia del tópico en todas las `listas` de la base de datos. La consulta obtiene los tópicos de influencia de los *amigos* de *u* y sus contadores. Además realiza una agregación según tópico, es decir suma los contadores para cada tópico. La consulta SQL realizada es la siguiente:

(4.13)
```
SELECT Topico, Topicos.Contador as idf, sum(Amigos_Influencia.Contador)
as contador FROM Amigos_Influencia, AmigoUsuario, Topicos
WHERE idUsuario = var1 AND idIntereses = Amigos_Influencia.idTopicos
AND AmigoUsuario.idAmigo = Amigos_InfluenciaidAmigos GROUP BY Topico
ORDER BY contadorsum DESC LIMIT 50;
```

Dónde `var1` es el `id` del usuario *u* a caracterizar. En la línea 2 se lee un archivo de texto llamado *listaNegra*, que incluye las palabras que suelen repetirse dentro de los tópicos pero que no aportan información. Por ejemplo: “listas”, “twitter”, “twitteros” y “usuarios”. En la línea 3 se inicializa un archivo JSON en el que se insertarán las duplas de tópicos de interés con su contador, y otra metadata. Luego, en la línea 4, se recorre cada una de las tuplas. Si el tópico no está en la lista negra, se agrega al JSON. Cabe mencionar que se

utiliza el formato JSON porque es un estándar en la industria, se utiliza en muchas APIs (todas las APIs utilizadas en este trabajo utilizan este formato) y es fácil de entender.Luego, en la línea 7, se hacen operaciones relativas al ponderador del tópico, en particular aplicar el esquema *TF-IDF*. Las operaciones se resumen en la ecuación 4.3, dónde $frecuencia_i$ y idf_i son los valores de las tuplas obtenidas en la consulta 4.13.

$$ponderador_i = frecuencia_i \times \ln(idf_i) \tag{4.3}$$

Se termina la iteración de cada tópico por agregar el ponderador al JSON. Por último, el módulo retorna el JSON con todos los tópicos y ponderadores, como el que se muestra en la Figura 4.2. Además de los tópicos de interés con sus ponderadores, se incluye el `id` del usuario caracterizado, el valor máximo (`max`) de los ponderadores y el `tiempo` que demoró la consulta.

4.6 API

En esta sección se detalla el funcionamiento y desarrollo de la API, presentada en la sección 3.6. La API funciona como controlador e interfaz entre el sistema y aplicaciones externas. En este momento cuenta con 2 funciones, *Obtener Tópicos de Interés* y *Agregar Usuario*. La API fue desarrollada en Java utilizando el framework Spring Boot para ser llamada a través de la web.

4.6.1 Agregar Usuario

Esta función se ejecuta al llamar a la URL de la API con el agregado `/agregarusuario`. El parámetro que necesita es el `id` de Twitter del usuario que se quiere agregar al sistema. En el algoritmo 4.7 se puede apreciar la lógica del la función *Agregar Usuario*.

Data: $id \leftarrow$ el identificador de Twitter del usuario u a caracterizar
Result: $json$: archivo en formato JSON con los $topicos$ de interés
1 $amigos \leftarrow$ se obtienen $amigos$ de Twitter de id y se insertan a la DB;
2 se obtienen $listas$ de los $amigos$ de Twitter del usuario u y se agregan a la DB;
3 se obtienen $topicos$ de las $listas$ de $amigos$ de u y se agregan a la DB;
4 se obtienen $topicos$ de influencia de $amigos$ de u y se agregan a la DB;
5 $json \leftarrow$ se obtienen $topicos$ de interés de usuario u y se entregan en JSON;
6 **return** $json$

Algorithm 4.7: Función para agregar usuarios al sistema

Esta función hace un llamado a todos los módulos del sistema. En la línea 1 del algoritmo se obtienen los `amigos` de *u* y se insertan a la base de datos. Se utiliza el Módulo de Extracción de Datos de Twitter, en particular el submódulo de Obtención de Amigos. En la línea 2, se obtienen las `listas` que tienen de miembros a los `amigos` de *u*. Para esto se utili-

```
{
    "intereses": [
        ["música", 84043],
        ["noticias", 72720],
        ["tecnología", 29037],
        ["política", 26263],
        ["medios", 24089],
        ["chile", 22468],
        ["tech", 23035],
        ["músicos", 18994],
        ["periodistas", 16534],
        ["políticos", 18362],
        ["bandas", 16374],
        ["televisión", 13180],
        ["artistas", 12972],
        ["mundo", 10913],
        ["información", 10783],
        ["rock", 12248],
        ["ciencia", 9553],
        ["famosos", 8578],
        ["prensa", 8776],
        ["gaming", 9614],
        ["metal", 9291],
        ["actualidad", 8050],
        ["electrónica", 9501],
        ["medios de comunicación", 6603],
        ["hardware", 8140],
        ["bandas", 7363],
        ["opinión", 6004],
        ["juegos", 6267],
        ["entretenimiento", 5478],
        ["comunicación", 5713],
        ["cultura", 5093],
        ["educación", 5212],
        ["radio", 4958],
        ["deportes", 4787],
        ["celebridades", 4727],
        ["actores", 4689],
        ["gobierno", 4664]
    ],
    "max": 84043,
    "id": 86443290,
    "tiempo": 19580
}
```


Figura 4.2: JSON de resultado

Fuente: Elaboración propia

za el Módulo de Extracción de Datos de Twitter, en particular el submódulo de Obtención de Listas.

Luego, en la línea 3, se muestra el paso de procesamiento de de listas e identificación de tópicos de estas. Para esto se utiliza el Módulo de Procesamiento de Listas. El paso que sigue es utilizar los tópicos de las listas para identificar los tópicos de influencia de los `amigos` de u, como se ve en la línea 4. Para esto se utiliza el Módulo de Identificación de

Tópicos de Influencia.

Por último, se identifican los tópicos de interés de u, utilizando el Módulo de Identificación de Tópicos de Interés. El resultado se entrega en formato JSON, como se mostró anteriormente en la Figura 4.2.

4.6.2 Obtener Tópicos de Interés

Esta función se ejecuta al llamar a la URL de la API con el agregado `/intereses`. El parámetro que necesita es el `id` del usuario de quien se desean identificar los tópicos de interés.

Data: $id \leftarrow$ el identificador de Twitter del usuario u a caracterizar
Result: $json$: archivo en formato JSON con los $topicos$ de interés
1 $json \leftarrow$ se obtienen $topicos$ de interés de usuario u y se entregan en JSON;
2 **return** $json$

Algorithm 4.8: Función para Obtener Tópicos de Interés

Como se ve en el algoritmo 4.8, al estar el usuario ya agregado, esta función solamente llama al Módulo de Identificación de Tópicos de Interés. El resultado se entrega en formato JSON, como se mostró anteriormente en la Figura 4.2.

4.7 Visualización

En esta sección se detalla el funcionamiento y desarrollo del Módulo de Visualización, presentada en la Sección 3.7. Este módulo es un ejemplo simple de uso del sistema y API, y está pensado en mostrar la información obtenida de una forma amigable para el usuario. Está desarrollado en PHP principalmente, pero también se utiliza Javascript. Este módulo es la Vista y la API funciona como controlador de los módulos.

La API tiene 2 funciones y esté módulo ejemplifica el uso de ambas. En cualquiera de los 2 casos, el resultado es mostrar los tópicos de interés del usuario. Para ello se utiliza la librería *wordcloud2.js* desarrollada en Javascript. Esta librería sirve para crear *nubes de palabras*, y se utilizará para mostrar los tópicos de interés y su importancia, dependiendo del tamaño de cada palabra.

El módulo comienza con una pagina desarrollada en HTML en la que se dispone de 2 campos. Uno es para agregar usuarios al sistema y otro es sólo para mostrar la nube de tópicos de interés. Cada uno de estos formularios envía el `id` del usuario, en adelante `idUsuario`, a `nube.php`, pero con un nombre distinto, para poder distinguir cual es la función de la API que se utilizará.

Si la variable `idUsuario` es llamada `id`, entonces se llama a la función de Obtener Tópicos de Interés. Esto se hace mediante un llamado *curl* a la URL de la API en Spring, pero

agregando `/intereses/?id=idUsuario`. En cambio, si la variable `idUsuario` es llamada `id2`, entonces se llama a la función de Agregar Usuario al Sistema. Esto se hace mediante un llamado *curl* a la URL de la API, pero agregando `/agregarUsuario/?id=idUsuario`.

En ambos casos, el resultado que se recibe es un *string* en formato JSON. Este string es entregado a la función WordCloud, la que genera la nube de palabras en un canvas HTML.

Capítulo 5

Resultados

El capítulo de resultados se divide en 2 partes. En la primera parte se muestra el proceso de validación del nuevo sistema mediante una encuesta y se analizan sus resultados. En la segunda parte, la discusión, se realiza una comparación de resultados utilizando la API de Klout y el sistema desarrollado en esta memoria, y luego se muestra una discusión general. Se realiza esta comparación porque el proyecto OpinionZoom utiliza el servicio de Klout para obtener los tópicos de interés que entrega en la plataforma, y el nuevo sistema fue creado para reemplazarlo. En ambas partes se utilizan los resultados y respuestas a la encuesta de 34 usuarios chilenos de Twitter.

5.1 Validación

En esta sección se describe el proceso de validación de resultados del sistema desarrollado en este trabajo. En primer lugar se detalla la metodología realizada para la validación y posteriormente se presentan los resultados obtenidos.

5.1.1 Metodología

La validación de los resultados obtenidos por el sistema consistió en realizar una encuesta a 34 usuarios chilenos de Twitter, para que ellos evaluaran qué tan representativos son los tópicos de interés arrojados por el sistema con respecto a sus reales intereses en Twitter.

El primer paso consistió en conseguir usuarios de Twitter, que utilicen la plataforma ya sea para *twittear* o para recibir información de otros usuarios, y que estén dispuestos a contestar una pequeña encuesta.

En segundo lugar, se procedió a ejecutar los distintos módulos del sistema para obtener los tópicos de interés de cada usuario. Los resultados obtenidos fueron representados en una nube de palabras, igual que en el módulo de visualización. En la Figura 5.1 se muestran

2 ejemplos de nubes de palabras enviadas a los usuarios.

A continuación se envió al e-mail de cada usuario la nube de palabras generada para él y la encuesta que debe responder. La encuesta fue creada utilizando la plataforma Google Forms[1]. Los usuarios debieron responder 3 preguntas, además de la posibilidad de dejar comentarios. Las preguntas de la encuesta fueron las siguientes:

1. **¿La nube de palabras representa tus intereses en Twitter?**

   Las alternativas de esta pregunta fueron Sí y No. El objetivo de esta pregunta es determinar si la percepción de los usuarios, con respecto a los resultados del sistema, fue positiva o negativa.

2. **De los 20 tópicos de la nube, ¿Cuántos consideras que te representan?**

   Esta pregunta tenía de alternativas los números enteros del 0 al 20. A diferencia de la pregunta 1, en esta pregunta se pretende obtener precisión de la predicción..

3. **¿El tamaño de las palabras de tu nube refleja la importancia relativa de tus tópicos de interés?**

   Las alternativas fueron 1, 2, 3 y 4, donde 1 indica que el tamaño refleja muy bien los tópicos de interés y 4 que no los refleja en absoluto. Esta pregunta tiene como objetivo identificar si la importancia relativa de cada tópico en la nube fue bien identificada.

Luego de 2 semanas, a partir del envío de la encuesta a los usuarios, se descargaron las respuestas obtenidas. Posteriormente se analizaron los resultados y se representaron de forma gráfica, para una mejor interpretación.

La precisión se define como la proporción de elementos clasificados correctamente dentro de una clase [27]. En el caso de este trabajo hay solamente una clase, que se puede describir como “el tópico identificado es un tópico de interés del usuario”, lo que puede ser verdadero o falso.

5.1.2 Resultados

En esta sección se muestran los resultados obtenidos de la encuesta realizada a los usuarios y un resumen sobre los tiempos de ejecución. En total se obtuvieron 34 respuestas. A continuación se muestra cada una de las preguntas realizadas con sus resultados y análisis respectivos.

1. **¿La nube de palabras representa tus intereses en Twitter?**

   Las respuestas a esta pregunta son muy favorables, ya que de los 34 usuarios que respondieron la encuesta, 33 respondieron la alternativa “Sí”. Esto quiere decir que el 97,1 % de los usuarios considera que los tópicos identificados sí representan sus tópicos de interés en Twitter, como se observa en la Figura 5.2.

[1] `https://www.google.com/intl/es-419/forms/about/` visitada el 23 de Septiembre, 2016

Figura 5.1: Nubes de palabras de usuarios encuestados

Fuente: Elaboración propia

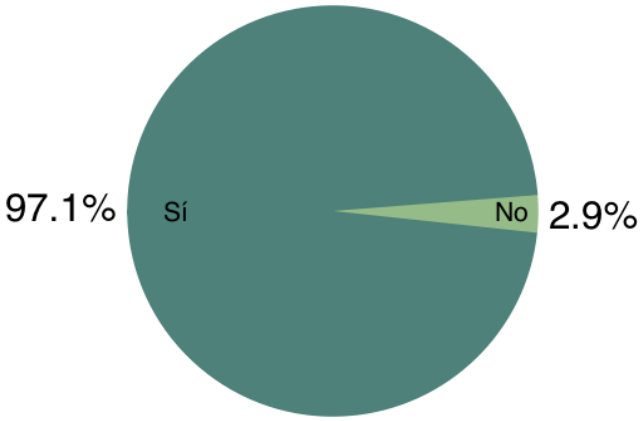

Figura 5.2: Representatividad de los tópicos de interés

Fuente: Elaboración propia

2. **De los 20 tópicos de la nube, ¿Cuántos consideras que te representan?**

   El resultado de esta pregunta se puede ver en la Figura 5.3, que muestra un histograma de las respuestas, es decir la frecuencia de cada una de éstas. Se puede ver que un gran porcentaje, el 67 % de las respuestas, se concentra entre los 15 y 20 tópicos representativos.

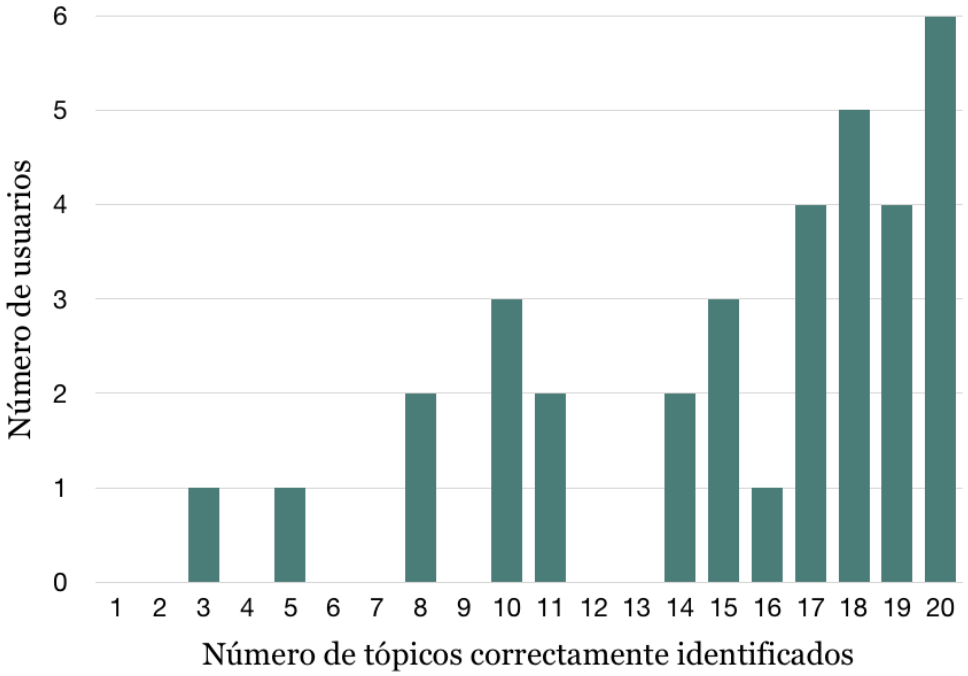

Figura 5.3: Histograma de tópicos correctamente identificados

Fuente: Elaboración propia

Con estos resultados se puede calcular la precisión de la identificación de tópicos de interés, que es el porcentaje de tópicos correctamente identificados. Este resultado

se considera bueno, sin embargo hay espacios para mejora.

$$Precisin = 76,3\,\% \tag{5.1}$$

Otra forma útil de mostrar los resultados es a través de las probabilidades de que el número de tópicos correctamente identificados se encuentre en determinado segmento y la probabilidad acumulada de éstos, como se ve en la Tabla 5.1. Donde la probabilidad acumulada es la probabilidad de pertenecer al segmento dado y a los superiores. Por ejemplo, la probabilidad acumulada del segmento $[16, 13]$ es la suma de la probabilidad de ese segmento más la probabilidad de $[20, 17]$.

Segmento	Probabilidad	Probabilidad acumulada
$[20, 17]$	$55,9\,\%$	$55,9\,\%$
$[16, 13]$	$17,6\,\%$	$73,5\,\%$
$[12, 9]$	$14,7\,\%$	$88,2\,\%$
$[8, 5]$	$8,8\,\%$	$97,0\,\%$
$[4, 0]$	$3,0\,\%$	$100,0\,\%$

Tabla 5.1: Probabilidad de identificación de tópicos de interés

Fuente: Elaboración propia

3. **¿El tamaño de las palabras de tu nube refleja la importancia relativa de tus tópicos de interés?**

   Las respuestas a esta pregunta muestran que el tamaño de las palabras de la nube entregada a cada usuario sí refleja el orden de importancia de éstos para ellos, como se ve en la Figura 5.4.

   En la Tabla 5.2 se muestran las 4 opciones dadas en la pregunta y el porcentaje de usuarios que seleccionó cada alternativa. De aquí vale la pena notar que el 79,4 % de los usuarios consideran que la importancia relativa de los tópicos está “muy bien” o “bien”.

Respuesta	Probabilidad	Probabilidad acumulada
Muy bien	$44,1\,\%$	$44,1\,\%$
Bien	$35,3\,\%$	$79,4\,\%$
Mal	$11,8\,\%$	$91,2\,\%$
Muy mal	$8,8\,\%$	$100\,\%$

Tabla 5.2: Probabilidad de importancia relativa de tópicos de interés

Fuente: Elaboración propia

En la Tabla 5.3 se muestra un resumen de los tiempos de ejecución del sistema desarrollado. Se muestran los promedios y desviaciones estándar de la ejecución del sistema en su totalidad, así como de cada módulo por separado. Este promedio considera a los mismos 34 usuarios encuestados. Se puede ver que la mayoría del tiempo lo utiliza el Módulo de

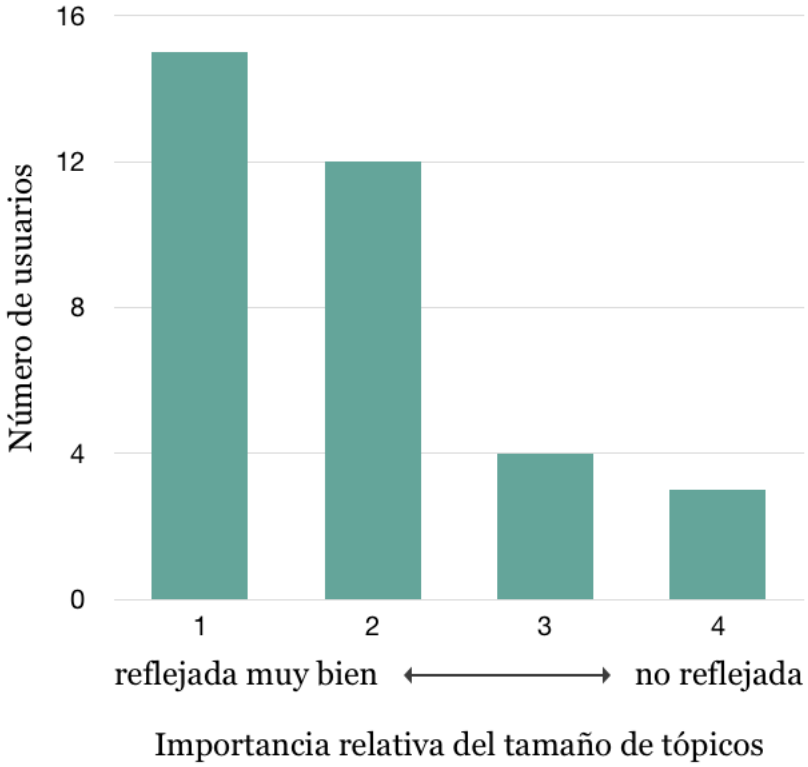

Figura 5.4: Representatividad del tamaño de los tópicos de interés

Fuente: Elaboración propia

Extracción de Datos de Twitter, con un 76,4 % del total. El tiempo de insertar a un nuevo usuario al sistema es, en promedio, de 2046 segundos, sin embargo mientras más usuarios se inserten este tiempo disminuirá, dado que no se tendrán que extraer tantos datos de Twitter. El tiempo de la obtención de los tópicos de interés de un usuario ya procesado es de aproximadamente 2,4 segundos.

	Tiempo promedio	Desv. Est. tiempo	% del tiempo total	Desv. Est. del % del tiempo
Mod. Twitter	1601, 0	1896, 9	76, 4	8, 4
Mod. Listas	409, 6	484, 8	21, 4	8, 0
Mod. Influencia	33, 4	29, 6	2, 0	1, 0
Mod. Intereses	2, 4	2, 1	0, 2	0, 2
Total	2046, 3	2395, 2	100	-

Tabla 5.3: Tiempos de ejecución del sistema

Fuente: Elaboración propia

5.2 Discusión

En esta sección se comparan los resultados del sistema desarrollado con los que se obtienen con la API de Klout, principalmente considerando la variedad de resultados, lo informativos que son y si éstos están alineados con la literatura. Por último se realiza una discusión sobre la validación anterior y la comparación realizada.

5.2.1 Comparación con Klout

Para poder comparar los resultados de ambos sistemas, se utilizaron los mismos 34 usuarios en la obtención de éstos. En el caso de Klout se obtuvieron entre 0 y 5 tópicos para cada usuario, los que no tienen ningún tipo de ponderador, es decir que todos tienen la misma importancia. El 5.9 % de los usuarios no tuvo resultados, y sólo el 82.4 % tuvo 5 tópicos en su resultado.

N°	Tópico 1	Tópico 2	Tópico 3	Tópico 4	Tópico 5
1	-	-	-	-	-
2	Chile	Santiago	Osorno	Sony	Faith No More
3	Chile	Santiago	Osorno	Faith No More	Aurora Colorado
4	Chile	Santiago	Osorno	Faith No More	Lady Gaga
5	Chile	Santiago	Osorno	Faith No More	Entrepreneurship

Tabla 5.4: Ejemplo resultados de Klout

Fuente: Elaboración propia

La Tabla 5.4 muestra los primeros 5 resultados obtenidos con Klout[2]. Se puede ver que uno de los usuarios no tiene tópicos de interés identificados, mientras los otros 4 usuarios comparten 4 de sus 5 tópicos, por lo que no se puede segmentar de buena manera. Además los tópicos identificados están en inglés, lo que no es útil para OpinionZoom. En el Anexo B, en la Tabla B.2, se muestran todos los resultados obtenidos. Por temas de privacidad, los nombres de los usuarios encuestados se omitieron en los resultados. En esta tabla se puede ver que los grupos de tópicos de interés no son únicos, es decir que se repiten entre los usuarios.

El sistema desarrollado en esta memoria entrega 20 tópicos, en castellano, para el 100 % de los usuarios. Además cada uno de los tópicos identificados tiene un ponderador asociado que indica la importancia de ese tópico para el usuario. Esto permite reflejar de mejor manera los tópicos de interés.

Las Figuras 5.5 y 5.6 muestran las frecuencias de todos los tópicos identificados utilizando la API de Klout y el nuevo sistema, respectivamente. Se aprecia que la variedad de tópicos el sistema nuevo es mucho mayor al de Klout (146 contra 39), además la frecuencia de los tópicos decrece más lentamente en el nuevo sistema, por lo que la capacidad de

[2]Los resultados obtenidos de Klout que se utilizan en esta memoria fueron obtenidos el día 30 de septiembre de 2016.

segmentación es mayor.

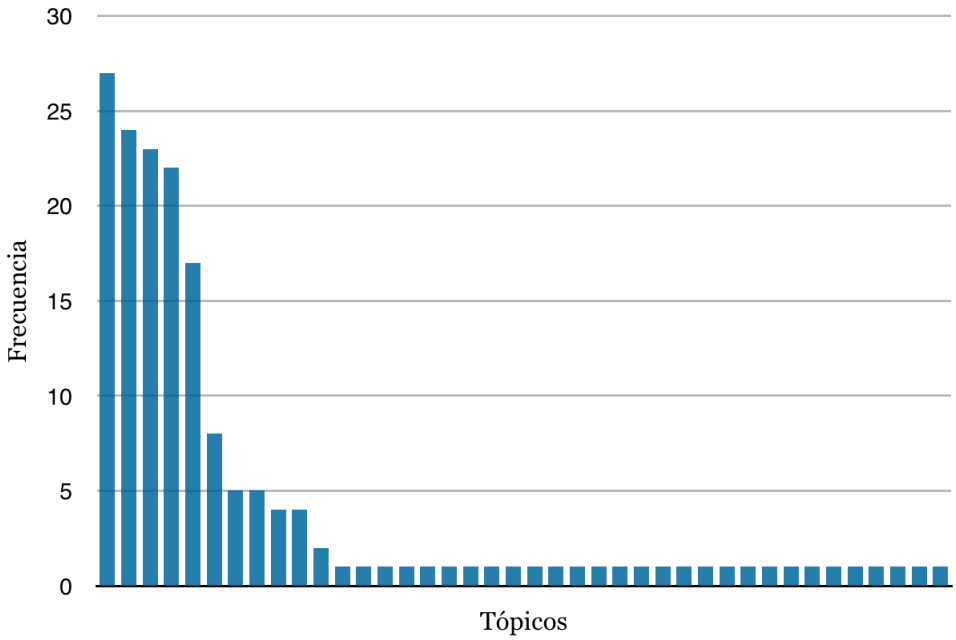

Figura 5.5: Frecuencia de tópicos en 34 usuarios de Klout

Fuente: Elaboración propia

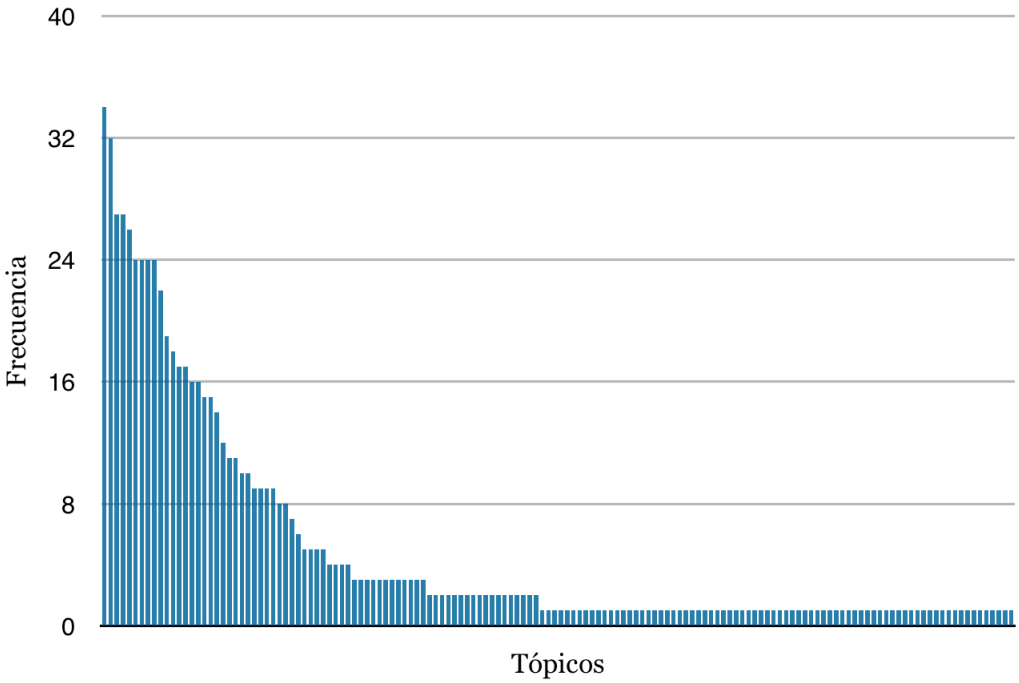

Figura 5.6: Frecuencia de tópicos en 34 usuarios del nuevo sistema

Fuente: Elaboración propia

Las Tablas 5.5 y 5.6 muestran los tópicos con mayor frecuencia de ambos sistemas. En la primera se muestran los obtenidos con Klout, y se muestran todos los tópicos con fre-

cuencia mayor o igual a 2. En la segunda se muestran los resultados del nuevo sistema, donde sólo están los tópicos con resultados mayores o iguales a 14. Las tablas con todos los resultados de las frecuencias se pueden ver en el Anexo B, en particular en la Tablas B.1, B.3 y B.4.

Tópico	Frecuencia	Porcentaje
Santiago	27	79,4 %
Chile	24	70,6 %
Osorno	23	67,6 %
Faith No More	22	64,7 %
Aurora Colorado	8	23,5 %
Television	5	14,7 %
Radio	5	14,7 %
Twitter	4	11,8 %
Richard Nixon	4	11,8 %
Samsung	2	5,9 %

Tabla 5.5: Tópicos con más frecuencia de Klout

Fuente: Elaboración propia

Tópico	Frecuencia	Porcentaje
noticias	34	100 %
mundo	32	94,1 %
chile	27	79,4 %
medios	27	79,4 %
televisión	26	76,5 %
política	24	70,6 %
información	24	70,6 %
periodistas	24	70,6 %
famosos	24	70,6 %
música	22	64,7 %
entretenimiento	19	55,9 %
medios de comunicación	18	52,9 %
deportes	17	50 %
tecnología	17	50 %
políticos	16	47,1 %
ciencia	16	47,1 %
prensa	15	44,1 %
artistas	15	44,1 %
fútbol	14	41,2 %

Tabla 5.6: Tópicos más frecuentes del nuevo sistema

Fuente: Elaboración propia

Se puede ver que los tópicos extraídos de Klout no son muy informativos ni variados. Los 3 de mayor frecuencia son ubicaciones, lo que no es útil para segmentar intereses. Twitter es un tópico que no agrega valor, puesto que los tópicos de interés que se extraen son de

esa plataforma. Hay tópicos que parecieran ser de nicho que aparecen dentro de los más frecuentes de Klout como "*Osorno*[3]", "Faith No More[4]", "Aurora Colorado[5]" y "Richard Nixon[6]", cuando dentro de los tópicos más frecuentes lo que debería primar son tópicos comunes.

En [11] se dice que los usuarios más seguidos de Twitter son cuentas de noticias o celebridades. En consecuencia, dentro de los intereses más comunes debieran estar tópicos relacionados con esos temas. En la Tabla 5.6, relativa al nuevo sistema, se puede ver que dentro de los tópicos más frecuentes hay una amplia variedad de tópicos de interés generales, como es de esperar de un sistema de intereses. Hay tópicos relativos a muchas áreas, incluidas las noticias y el entretenimiento. Con esta variedad de tópicos es posible segmentar usuarios de acuerdo a sus gustos. Sin embargo, se puede ver que el tópico noticias es parte del resultado de todos los usuarios. Ese tópico no permite segmentar, pero tiene sentido que esté en la nube, y muestra que a todos los usuarios encuestados les interesan las noticias.

Se puede ver que los tópicos generados con el nuevo sistema son más informativos y tienen mayor potencial de segmentación que los resultados entregados por Klout. Para validar que los resultados obtenidos realmente reflejan los intereses de los usuarios se realizó una encuesta, la que se muestra en la siguiente sección.

5.2.2 Discusión General

La comparación realizada entre los resultados entregados por Klout y los obtenidos con el nuevo sistema, muestran que la variedad de tópicos es mayor en el último, que su capacidad de segmentación es mayor y que es consistente con la literatura referente a los intereses de los usuarios en Twitter.

Se realizó una encuesta a los mismos 34 usuarios de quienes fueron identificados los tópicos de interés en la sección de comparación. A partir de la encuesta realizada, se obtuvieron métricas y comentarios sobre el sistema, que revelaron una percepción positiva de parte de los usuarios. Los comentarios entregados indican que los tópicos identificados en la nube de palabras reflejan de buena manera los tópicos de interés de la mayoría de los usuarios. Varios de los encuestados expresaron que se sintieron completamente representados por los tópicos entregados, otros mencionaron que no se les ocurrieron tópicos que hayan faltado en la nube, y otros indicaron que los resultados eran buenos.

Sin embargo, algunos usuarios encontraron aspectos que mejorar en los tópicos entregados. Uno de éstos es que hay tópicos que son muy similares entre sí, por lo que no aporta información que aparezcan ambos. Podría ser mejor que un solo tópico que englobe todos

[3]Osorno es una comuna del sur de Chile, en la Región de los Lagos. Para mayor información visitar `http://www.municipalidadosorno.cl/sitios/cp/webimo/`. Última visita el 2 de octubre de 2016

[4]*Faith No More* es una banda musical estadounidense. Su biografía completa se puede visitar en `http://www.fnm.com/bio.shtml`. Última visita el 2 de octubre de 2016

[5]*Aurora* es una ciudad ubicada en Colorado, uno de los estados de Estados Unidos de América. Para mas información visitar `https://www.auroragov.org`. Última visita el 2 de octubre de 2016

[6]*Richard Nixon* fue un presidente de los Estados Unidos de América. Su biografía está en `http://www.biography.com/people/richard-nixon-9424076`. Última visita el 2 de octubre de 2016

los términos, y que su ponderador sea mayor. Por ejemplo: “políticos” y “política”; “periodistas”, “medios”, “medios de comunicación”, “noticias”, “prensa” y “actualidad”; “celebridades” y “famosos”.

Además hay algunos tópicos que no aportan mucha información acerca de un usuario, y aparecen dentro de los tópicos más importantes. Algunos de estos son “periodistas”, “medios”, “noticias”, “televisión”, entre otros. Esto se puede deber a que muchos usuarios utilizan Twitter como una plataforma para informarse, por lo que tiene sentido que sigan a muchos usuarios relacionados con los tópicos nombrados. Sin embargo, los usuarios no consideran que esos tópicos los identifiquen tanto como otros que caracterizan a menos de sus amigos.

En algunos casos, dentro de los resultados se encuentra el mismo tópico en singular y plural. Esto se debe a que la lematización se realiza previo a la traducción, y en algunas ocasiones el traductor entrega la misma palabra en plural. Algunas de estas ocurrencias en las nubes de palabras de los usuarios encuestados fueron: “deporte” y “deportes”; “celebridad” y “celebridades”; “juego” y “juegos”.

En algunos comentarios, algunos usuarios notaron que ciertas palabras no los identificaban. Dentro de las que más se repiten en esta categoría están “famosos” y “farándula”. Esto puede deberse a que los usuarios siguen a usuarios famosos, pero no los siguen por ese hecho, sin embargo son categorizados de esa manera.

Algunos usuarios comentaron que hay tópicos que habrían esperado que aparecieran dentro de sus tópicos de interés y que no aparecieron. La razón de esto puede ser que los usuarios no siguen a otros usuarios por los mismos tópicos que son caracterizados según la mayoría de sus seguidores. Otra razón es que el sistema considera sólo sustantivos y palabras individuales, a excepción de las *entidades nombradas* que detecta FreeLing. Esto provoca que algunos tópicos que tienen más de una palabra no estén incluidos, como “inteligencia artificial”. El sistema solo detectaría “inteligencia” como un tópico, por lo que el concepto original no estaría completamente representado.

Otros dijeron que los tamaños no eran muy representativos. Esto puede deberse a que la importancia de un tópico de interés no es necesariamente proporcional al número de usuarios que se caracterizan con ese tópico. Por otro lado, para calcular el ponderador de la importancia cada tópico de interés de un usuario u, se utilizó la suma del ponderador de influencia de todos los amigos de u, lo que puede no ser representativo.

Capítulo 6

Conclusiones

En este capítulo se dan a conocer las conclusiones de esta memoria, en particular sobre los resultados obtenidos, las implicancias de éstos y sus limitaciones. Posteriormente se proponen mejoras al sistema desarrollado y temas de investigación relacionados.

6.1 Conclusiones generales

En este trabajo se diseñó y desarrolló un sistema de identificación de tópicos de interés basado en la metodología propuesta por Bhattacharya *et al.* [48]. Las principales diferencias con la metodología original es que la nueva está enfocada al Español y que no se cuenta con una gran base de datos de información de Twitter, por lo que se tiene que hacer una extracción exhaustiva de datos utilizando la API de Twitter. Para la elección del método, se llevó a cabo un estudio del estado del arte de metodologías de identificación de tópicos de interés en redes sociales, y en particular en Twitter, considerando los desafíos que presenta esta red social con respecto al contenido generado por sus usuarios. Por esta razón, se eligió un método que utiliza las conexiones entre los usuarios de Twitter, de forma de poder identificar los tópicos de cualquier usuario, incluso si no *twittea*.

El sistema procesó a 34 usuarios de Twitter, y se obtuvieron sus tópicos de interés. Estos resultados se analizaron de dos maneras. Primero se realizó una comparación de resultados, de los mismos 34 usuarios, con la situación actual de OpinionZoom, que es la API de Klout. Se realizó un análisis cualitativo de los tópicos y se concluyó que los entregados por el nuevo sistema son mucho más variados que los de Klout, la cobertura es del 100 % de los usuarios, el número de tópicos por usuario es mayor (20 contra un máximo de 5), los tópicos poseen mayor poder de segmentación y son coherentes con la literatura. En segundo lugar, para otorgar validez a los resultados obtenidos, se realizó una encuesta a los usuarios caracterizados. Los resultados de esta validación fueron favorables, puesto que 33 de los 34 encuestados dijeron estar representados por los tópicos identificados, y también se obtuvieron buenos resultados de precisión y buena recepción de relevancia de los tópicos. Tomando los 2 puntos anteriores en consideración, se puede decir que se valida la hipótesis de investigación, ya que se pudo desarrollar un sistema de identificación de tópicos de

interés de usuarios chilenos de Twitter utilizando a sus amigos de la red social.

La principal ventaja del sistema desarrollado es también una limitación. El identificar los tópicos de interés basándose en los amigos de un usuario permite obtener resultados para la gran mayoría de ellos, mucho más que los sistemas que utilizan sólo los *tweets* generados. Lo único que se necesita es que ellos sigan a usuarios que hayan sido agregados a listas. Esto no es una condición muy restrictiva, ya que la gran mayoría de los usuarios sigue a algún usuario "famoso", que a su vez es miembro de alguna lista. De hecho, en la validación realizada, el 100 % de los usuarios cumplió esta condición.

El desarrollo del sistema de forma modular constituye una ventaja, ya que permite llevar a cabo mejoras sin la necesidad de reescribir completamente el sistema. Esto significa que se puede modificar un módulo, respetando el formato de entrada y salida de éste, y el sistema se mantendrá en funcionamiento.

Cabe destacar que el sistema fue desarrollado para satisfacer las necesidades de OpinionZoom, sin embargo su utilidad no está acotada a este proyecto. Además de ser de utilidad para empresas de análisis de redes sociales, puede ser un insumo para sistemas de recomendación y personalización. El saber lo que le interesa a un usuario puede ser muy útil para determinar qué producto recomendar en un *e-commerce*, el orden de los resultados en un buscador e incluso qué publicidad mostrar. En cualquier caso hay que preocuparse de no individualizar más de la cuenta a los usuarios, para no pasar a llevar su privacidad, y utilizar información agregada cada vez que sea necesario.

Con respecto a los desafíos que existen para llevar el sistema desarrollado al mercado, el mayor de ellos corresponde a lograr que las empresas e instituciones comprendan el real alcance de conocer los intereses de los usuarios, e identifiquen cuáles son las formas de utilizar esta información dentro de su negocio. Si bien los usos del sistema son múltiples, para que determinadas empresas se interesen en adoptarlo, se debe lograr que descubran los beneficios particulares que conllevaría aplicar este sistema.

6.2 Trabajo futuro

A partir de los resultados obtenidos, los comentarios de los usuarios y las conclusiones obtenidas, se pueden proponer mejoras al sistema de identificación de tópicos de interés:

- Para reducir considerablemente el tiempo de ejecución del sistema se puede considerar el paralelizar las consultas a la API de Twitter. Ese es el gran cuello de botella, por lo que al hacer consultas simultáneas se podría disminuir por un gran margen el tiempo que toma el ingresar a un nuevo usuario al sistema.

- Se debe explorar la manera de que los tópicos de interés de un usuario no contengan tópicos con un significado muy similar o igual. El problema de esto es que los tópicos que sean muy similares, tendrán asignado un peso por separado, por lo que no reflejarán su real importancia, y además quita que otro tópico pueda aparecer dentro de los primeros resultados. Para evitar este problema se pueden incorporar técnicas

de *topic modeling*, que consiste en agrupar tópicos según un concepto abstracto, o el uso de jerarquías de conceptos.

- Hay un grupo de tópicos que se repiten mucho entre los usuarios, que aparecen en las primeras posiciones y que parecieran no aportar mucha información. Esto se obtiene a partir de la exploración de resultados y de los comentarios de los usuarios caracterizados. Para evitar este problema se puede utilizar un esquema de peso de término distinto a *TF-IDF* o utilizar una modificación de éste.

- Muchas veces en los resultados ocurren tópicos que aparecen tanto en su forma singular como plural, lo que claramente es redundante. Esto se puede deber a varias razones, entre ellas que el identificador de idioma no identifique correctamente el idioma, por lo que el lematizador no funciona correctamente. Otra posibilidad es que el traductor inglés-castellano traduzca algunas palabras singulares a su forma plural.

- Algunos usuarios comentaron que hay palabras que no los identificaban. Esto se puede deber a que un mismo usuario de Twitter puede ser influyente en varios temas, y ser caracterizado de distinta manera. Para evitar esto se puede evaluar el número de veces que debe aparecer un tópico, dentro de los tópicos de influencia de sus *amigos*, antes de que se muestra cómo un tópico de interés.

- Hay tópicos de interés de usuarios que no están incluidos en los resultados que entrega el sistema. Una posible solución es permitir que existan tópicos que tengan más de una palabra (n-gramas), y que se permita que éstos sean sustantivos y adjetivos. Para hacer esto puede ser necesario incorporar una lista de tópicos de más de una palabra, y que el tópico se muestre sólo si es parte de esa lista.

- Algunos tópicos no deberían traducirse, a pesar de estar en inglés. Existen nombres propios que al ser traducidos pierden sentido, y otros que se utilizan comúnmente por su nombre en inglés. Una posible solución es incorporar una lista de tópicos que no deben ser traducidos.

- La importancia relativa de los tópicos de interés podría ser mejorada. En general los usuarios se mostraron conformes, pero hay bastante espacio para mejora. El sistema actual considera que si un usuario es más famoso que otro, entonces sus tópicos ponderan más que el de un usuario más desconocido. Esta condición podría cambiarse por que el peso de todos los usuarios seguidos sea el mismo y podrían mejorar los resultados. Otra posibilidad es cambiar el esquema de pesos que se usa (el actual es *TF-IDF*).

- Una adición que podría mejorar los resultados, y dar más opciones de visualización de los tópicos, es incorporar jerarquías de tópicos de interés. Para esto se pueden utilizar ontologías o jerarquías de base, como las de WordNet o Wikipedia, o complementar el modelo con *Topic Modeling*. Implementar un sistema con estos elementos ayudaría a poder definir el nivel de detalle que se desea en el resultado de tópicos de interés.

A partir del sistema desarrollado se desprenden futuros temas de investigación que pueden ser de interés:

- La metodología propuesta identifica tópicos de influencia, y a partir de éstos se obtienen los tópicos de interés. Podría ser útil el desarrollar una API que explote los tópicos de influencia de los usuarios y que cuantifique el grado de influencia.

- Sería interesante monitorear el cambio en el tiempo de tópicos de interés de un usuario. Para realizar esto se pueden ir capturando los intereses de cada usuario, o grupo de usuarios, y analizar su dinamismo en el tiempo. Se puede contrastar con sucesos de la vida diaria y determinar qué hace que un tópico de interés se convierta en tal o se pierda.

Bibliografía

[1] Internet World Stats, *World internet users statistics and 2016 world populations stats*, 2016. dirección: http://www.internetworldstats.com/stats.htm (visitado 29 de sep. de 2016).

[2] Internet Live Stats, *Internet users by country (2016)*, 2016. dirección: http://www.internetlivestats.com/internet-users-by-country/ (visitado 29 de sep. de 2016).

[3] Portada, *Ránking de usuarios únicos: los 10 sitios más visitados en latam: país a país*, 2015. dirección: http://mercadotecnia.portada-online.com/2015/11/17/ranking-de-usuarios-unicos-los-10-sitios-mas-visitados-en-latam-pais-a-pais/ (visitado 29 de sep. de 2016).

[4] J. Domenech, *Chile es el país de américa latina con mayor penetracion de las redes sociales*, 2015. dirección: http://www.siliconweek.com/e-marketing/socialmedia/chile-es-el-pais-de-america-latina-con-mayor-penetracion-de-las-redes-sociales-62017 (visitado 29 de sep. de 2016).

[5] A. Córdova, *Diseño y construcción de un sistema web de análisis de opiniones en twitter integrando algoritmos de data mining*, 2015.

[6] E. Marrese-Taylor, J. D. Velásquez y F. Bravo-Marquez, «A novel deterministic approach for aspect-based opinion mining in tourism products reviews», *Expert Systems with Applications*, vol. 41, n.º 17, págs. 7764-7775, 2014. dirección: http://wic.uchile.cl/wp-content/uploads/2015/09/A-novel-deterministic-approach-for-aspect-based-opinion-mining-in-tourism-products-reviews.pdf.

[7] B. Liu, *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Science & Business Media, 2007.

[8] E. Marrese-Taylor, «Diseño e implementación de una aplicación de web opinion mining para identificar preferencias de usuarios sobre productos turísticos de la x región de los lagos», 2013.

[9] E. Marrese-Taylor, J. Velasquez y F. Bravo-Marquez, «Opinion zoom: a modular tool to explore tourism opinions on the web», en *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, IEEE, vol. 3, 2013, págs. 261-264.

[10] F. Ponce de León, «Uso de la ingeniería de negocios en diseño e implementación de negocios con tecnologías de información», Tesis doct., Universidad de Chile, 2015.

[11] H. Kwak, C. Lee, H. Park y S. Moon, «What is twitter, a social network or a news media?», en *Proceedings of the 19th international conference on World wide web*, ACM, 2010, págs. 591-600.

[12] Business Standard, *Content creators leave social networks when messaging is easy*, 2015. dirección: http://www.business-standard.com/article/pti-stories/content-creators-leave-social-networks-when-messaging-is-easy-115040600802.html (visitado 26 de sep. de 2016).

[13] L. S. Lowe, *125 amazing social media statistics you should know in 2016*, 2016. dirección: https://socialpilot.co/blog/125-amazing-social-media-statistics-know-2016/ (visitado 29 de sep. de 2016).

[14] T. Berners-Lee y R. Cailliau, «Worldwideweb: proposal for a hypertext project, 1990», *URL http://www. w3. org/Proposal. html*, 2009.

[15] D. Best, «Web 2.0: next big thing or next big internet bubble», *Technische Universiteit Eindhoven*, 2006.

[16] E. Constantinides y S. J. Fountain, «Web 2.0: conceptual foundations and marketing issues», *Journal of direct, data and digital marketing practice*, vol. 9, n.º 3, págs. 231-244, 2008.

[17] S. Aghaei, M. A. Nematbakhsh y H. K. Farsani, «Evolution of the world wide web: from web 1.0 to web 4.0», *International Journal of Web & Semantic Technology*, vol. 3, n.º 1, pág. 1, 2012.

[18] A. Al-kouz, «Interests discovery in social networks based on a semantically enriched bayesian network model», Tesis doct., 2013.

[19] Alexa, *Twitter.com traffic statistics*, 2016. dirección: http://www.alexa.com/siteinfo/twitter.com (visitado 28 de sep. de 2016).

[20] Twitter, *About twitter*, 2016. dirección: https://about.twitter.com/company (visitado 27 de sep. de 2016).

[21] ——, *About verified accounts*, 2016. dirección: https://support.twitter.com/articles/119135 (visitado 27 de sep. de 2016).

[22] ——, *About direct messages*, 2016. dirección: https://support.twitter.com/articles/14606 (visitado 28 de sep. de 2016).

[23] ——, *Using twitter lists*, 2016. dirección: https://support.twitter.com/articles/76460 (visitado 6 de oct. de 2016).

[24] O. Maimon y L. Rokach, *Data mining and knowledge discovery handbook*. Springer, 2005, vol. 2.

[25] U. Fayyad, G. Piatetsky-Shapiro y P. Smyth, «The kdd process for extracting useful knowledge from volumes of data», *Communications of the ACM*, vol. 39, n.º 11, págs. 27-34, 1996.

[26] A. Hotho, A. Nürnberger y G. Paaß, «A brief survey of text mining.», en *Ldv Forum*, vol. 20, 2005, págs. 19-62.

[27] R. Baeza-Yates y B. Ribeiro-Neto, *Modern Information Retrieval*, 2nd. USA: Addison-Wesley Publishing Company, 2008, ISBN: 9780321416919.

[28] C. D. Manning y H. Schütze, *Foundations of statistical natural language processing*. MIT Press, 1999, vol. 999.

[29] W3C, *The platform for privacy preferences 1.0 (p3p1.0) specification*, 2002. dirección: `http://www.w3.org/TR/P3P/` (visitado 1 de ene. de 2016).

[30] S. Gauch, M. Speretta, A. Chandramouli y A. Micarelli, «User profiles for personalized information access», en *The adaptive web*, Springer, 2007, págs. 54-89.

[31] G. Amato y U. Straccia, «User profile modeling and applications to digital libraries», en *International Conference on Theory and Practice of Digital Libraries*, Springer, 1999, págs. 184-197.

[32] F. Carmagnola, F. Cena, O. Cortassa, C. Gena e I. Torre, «Towards a tag-based user model: how can user model benefit from tags?», en *International Conference on User Modeling*, Springer, 2007, págs. 445-449.

[33] D. Vallet, P. Castells, M. Fernández, P. Mylonas e Y. Avrithis, «Personalized content retrieval in context using ontological knowledge», *IEEE Transactions on circuits and systems for video technology*, vol. 17, n.º 3, págs. 336-346, 2007.

[34] A. Kobsa, «Supporting user interfaces for all through user modeling», *ADVANCES IN HUMAN FACTORS ERGONOMICS*, vol. 20, pág. 155, 1995.

[35] J. Fink y A. Kobsa, «User modeling for personalized city tours», *Artificial intelligence review*, vol. 18, n.º 1, págs. 33-74, 2002.

[36] R. A. Gotardo, C. A. C. Teixeira y S. D. Zorzo, «Ip2 model-content recommendation in web-based educational systems using user's interests and preferences and resources' popularity», en *2008 32nd Annual IEEE International Computer Software and Applications Conference*, IEEE, 2008, págs. 460-463.

[37] X. Shen, B. Tan y C. Zhai, «Implicit user modeling for personalized search», en *Proceedings of the 14th ACM international conference on Information and knowledge management*, ACM, 2005, págs. 824-831.

[38] A. Abdel-Hafez e Y. Xu, «A survey of user modelling in social media websites», *Computer and Information Science*, vol. 6, n.º 4, págs. 59-71, 2013, ISSN: 1913-8997. DOI: `10.5539/cis.v6n4p59`. dirección: `http://www.ccsenet.org/journal/index.php/cis/article/view/28626`.

[39] S. Schiaffino y A. Amandi, «Intelligent user profiling», *Artificial Intelligence: An International Perspective-IFIP Book Series*, págs. 193-216, 2009.

[40] J. Hannon, K. McCarthy, M. P. O'Mahony y B. Smyth, «A multi-faceted user model for twitter», en *International Conference on User Modeling, Adaptation, and Personalization*, Springer, 2012, págs. 303-309.

[41] S. Kanoje, S. Girase y D. Mukhopadhyay, «User profiling trends, techniques and applications», *International Journal of Advance Foundation and Research in Computer*, vol. 1, n.º 11, págs. 2348-4853, 2014.

[42] M Minio y C Tasso, «User modeling for information filtering on internet services: exploiting an extended version of the umt shell», en *UM96 Workshop on User Modeling for Information Filtering on the WWW*, 1996, págs. 2-5.

[43] G. Gentili, A. Micarelli y F. Sciarrone, «Infoweb: an adaptive information filtering system for the cultural heritage domain», *Applied Artificial Intelligence*, vol. 17, n.º 8-9, págs. 715-744, 2003.

[44] M. Wasim, I. Shahzadi, Q. Ahmad y W. Mahmood, «Extracting and modeling user interests based on social media», *Proceedings of the 14th IEEE International Multitopic Conference 2011, INMIC 2011*, págs. 284-289, 2011. DOI: 10.1109/INMIC.2011.6151489.

[45] A. Pretschner y S. Gauch, «Ontology based personalized search», en *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on*, IEEE, 1999, págs. 391-398.

[46] P. K. Chan, «Constructing web user profiles: a non-invasive learning approach», en *Web usage analysis and user profiling*, Springer, 2000, págs. 39-55.

[47] A. Moukas, «Amalthaea information discovery and filtering using a multiagent evolving ecosystem», *Applied Artificial Intelligence*, vol. 11, n.º 5, págs. 437-457, 1997.

[48] P. Bhattacharya, M. B. Zafar, N. Ganguly, S. Ghosh y K. P. Gummadi, «Inferring user interests in the twitter social network», en *Proceedings of the 8th ACM Conference on Recommender systems - RecSys '14*, 2014, págs. 357-360, ISBN: 9781450326681. DOI: 10.1145/2645710.2645765. dirección: http://dl.acm.org/citation.cfm?doid=2645710.2645765.

[49] S. Ghosh, N. Sharma, F. Benevenuto, N. Ganguly y K. Gummadi, «Cognos: crowdsourcing search for topic experts in microblogs», en *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, ACM, 2012, págs. 575-590.

[50] N. K. Sharma, S. Ghosh, F. Benevenuto, N. Ganguly y K. Gummadi, «Inferring who-is-who in the twitter social network», *ACM SIGCOMM Computer Communication Review*, vol. 42, n.º 4, pág. 533, 2012, ISSN: 01464833. DOI: 10.1145/2377677.2377782.

[51] M. B. Zafar, K. P. Gummadi, N. Sharma, S. Ghosh, N. Ganguly y F. Benevenuto, *Who is who? - discover crowdsourced opinion on twitter users!*, 2012. dirección: http://twitter-app.mpi-sws.org/who-is-who/users.php?id=barackobama (visitado 6 de oct. de 2016).

[52] M. B. Zafar, S. Ghosh, K. P. Gummadi, P. Bhattacharya y N. Ganguly, *Who likes what? - discover topical interests of twitter users!*, 2014. dirección: http://twitter-app.mpi-sws.org/who-likes-what/index.php (visitado 29 de sep. de 2016).

[53] D. Berlind, *What is an api, exactly?*, 2015. dirección: http://www.programmableweb.com/news/what-is-an-api/analysis/2015/12/03 (visitado 13 de sep. de 2016).

[54] C. D. Manning, P. Raghavan y H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008, ISBN: 9780521865715.

[55] MariaDB, *About mariadb*, 2016. dirección: https://mariadb.org/about/ (visitado 29 de sep. de 2016).

[56] Twitter, *Rest apis*, 2016. dirección: https://dev.twitter.com/rest/public (visitado 22 de nov. de 2016).

[57] ——, *Rate limits: chart*, 2016. dirección: https://dev.twitter.com/rest/public/rate-limits (visitado 22 de nov. de 2016).

[58] Y. Yamamoto, *Twitter4j*, 2016. dirección: http://twitter4j.org (visitado 10 de oct. de 2016).

[59] L. Padró y E. Stanilovsky, «Freeling 3.0: towards wider multilinguality», en *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, ELRA, Istanbul, Turkey, 2012.

[60] Oracle, *Netbeans ide 8.1*, 2016. dirección: https://netbeans.org/community/releases/81/ (visitado 10 de oct. de 2016).

[61] Pivotal Software, *Spring framework*, 2016. dirección: https://projects.spring.io/spring-framework/ (visitado 29 de sep. de 2016).

[62] V. Cortés, *Diseño e implementación de un sistema para monitorear el consumo y opinión sobre la marihuana en twitter*. 2016.

Anexo A

Encuesta de validación

Encuesta de validación

Fuente: Elaboración propia

Anexo B

Listado de tópicos

Tópico	Frecuencia	Tópico	Frecuencia
Santiago	27	Vogue	1
Chile	24	California	1
Osorno	23	Astronomy	1
Faith No More	22	Reality Television	1
-	17	Apps	1
Aurora Colorado	8	Shoes	1
Television	5	SpaceX	1
Radio	5	Sony	1
Twitter	4	BlizzCon	1
Richard Nixon	4	Soundgarden	1
Samsung	2	Twilight Saga	1
Shinedown	1	Nickelodeon	1
The Voice	1	CBS	1
Software	1	Facebook	1
Victoria's Secret	1	Lady Gaga	1
Antofagasta	1	Entrepreneurship	1
Dropbox	1	League of Legends	1
Music	1	Travel	1
Disney Channel	1	Nine West	1
Apple	1	Careers	1

Tabla B.1: Frecuencia de tópicos de Klout

Fuente: Elaboración propia

Nº	Tópico 1	Tópico 2	Tópico 3	Tópico 4	Tópico 5
1	-	-	-	-	-
2	Chile	Santiago	Osorno	Sony	Faith No More
3	Chile	Santiago	Osorno	Faith No More	Aurora Colorado
4	Chile	Santiago	Osorno	Faith No More	Lady Gaga
5	Chile	Santiago	Osorno	Faith No More	Entrepreneurship
6	Samsung	Santiago	Osorno	BlizzCon	League of Legends
7	Chile	Antofagasta	Santiago	Osorno	Faith No More
8	Chile	Santiago	Osorno	Faith No More	Aurora Colorado
9	-	-	-	-	-
10	Samsung	Dropbox	Santiago	Faith No More	Aurora Colorado
11	Chile	Santiago	Osorno	Faith No More	Richard Nixon
12	Chile	Aurora Colorado	Television	Radio	Santiago
13	Television	Chile	Santiago	Osorno	Faith No More
14	Shinedown	Music	Twitter	-	-
15	Chile	Santiago	Television	Osorno	Faith No More
16	Chile	Santiago	Osorno	Soundgarden	Faith No More
17	Chile	Santiago	Osorno	Faith No More	Radio
18	Chile	Santiago	Osorno	Faith No More	Aurora Colorado
19	Chile	Santiago	Astronomy	Osorno	Faith No More
20	The Voice	Disney Channel	Reality Television	Twilight Saga	-
21	Chile	Osorno	Santiago	Faith No More	Richard Nixon
22	Chile	Santiago	Aurora Colorado	Osorno	Faith No More
23	Chile	Santiago	Osorno	Radio	Faith No More
24	Twitter	Chile	Santiago	Faith No More	Travel
25	Chile	Santiago	Television	Osorno	Faith No More
26	Osorno	Faith No More	Santiago	Richard Nixon	Nine West
27	Software	Apple	Apps	Nickelodeon	Chile
28	Chile	Santiago	Osorno	Faith No More	Radio
29	Television	Chile	Radio	Santiago	Osorno
30	Twitter	Chile	Santiago	Aurora Colorado	Osorno
31	Victoria's Secret	Vogue	Shoes	CBS	-
32	Chile	Santiago	-	-	-
33	Santiago	Osorno	Faith No More	Richard Nixon	Aurora Colorado
34	Twitter	California	SpaceX	Facebook	Careers

Tabla B.2: Resultados Klout 34 usuarios

Fuente: Elaboración propia

Tópico	Frecuencia	Tópico	Frecuencia
noticias	34	opinión	4
mundo	32	juegos	4
chile	27	moda	4
medios	27	películas	4
televisión	26	empresas	3
política	24	negocio	3
información	24	datos	3
periodistas	24	universidades	3
famosos	24	márketing	3
música	22	deportistas	3
entretenimiento	19	farandula	3
medios de comunicación	18	ayuda	3
deportes	17	humor	3
tecnología	17	cantantes	3
políticos	16	libros	3
ciencia	16	película	3
prensa	15	programas	2
artistas	15	gaming	2
fútbol	14	servicios	2
celebridades	12	revistas	2
actualidad	11	youtuberos	2
cultura	11	comida	2
tech	10	matemáticas	2
actores	10	pll	2
gobierno	9	pequeños mentirosos	2
educación	9	citas	2
jugadores	9	rock	2
bandas	9	autores	2
deporte	8	lobo	2
músicos	8	metal	2
notificaciones	7	escritores	2
espacio	6	astronomía	2
comunicación	5	frases	2
emergencias	5	celebridad	2
juego	5	aplicaciones	1
youtubers	5	software	1

Tabla B.3: Frecuencia de tópicos de nuevo sistema - Primera parte

Fuente: Elaboración propia

Tópico	Frecuencia	Tópico	Frecuencia
videojuegos	1	onda	1
web	1	comedia	1
vídeo	1	alegría	1
energía	1	dirección	1
idioma	1	ídolos	1
belleza	1	ajedrez	1
bloggers	1	venezuela	1
maquillaje	1	diputados	1
youtube	1	senadores	1
gurús	1	transporte	1
blogs	1	cyclisme	1
estilo	1	metro	1
tenis	1	tránsito	1
seo	1	velo	1
influenciadores	1	ciclismo	1
cerveza	1	santiago	1
liga	1	cine	1
leyendas	1	tv	1
fábricas de cerveza	1	nba	1
lol	1	equipos	1
sismos	1	sidemen	1
emergencia	1	arsenal	1
canales	1	hombre	1
juegos olímpicos	1	manchester unido	1
innovación	1	atletas	1
startups	1	futbolistas	1
diarios	1	baloncesto	1
río	1	clubs	1
psicología	1	apple	1
física	1	modelos	1
investigación	1	amor	1
salud	1	modelo	1
ingenieria	1	vida	1
tronos	1	estrellas	1
banda	1	linux	1
periodismo	1	emprendimiento	1
internet	1	startup	1

Tabla B.4: Frecuencia de tópicos de nuevo sistema - Segunda parte

Fuente: Elaboración propia