UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

MÉTODO PARA LA DETECCIÓN DE TÓPICOS EN LA RED DE MENSAJERÍA INSTANTÁNEA DE LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE

INGENIERO EN CIENCIAS INFORMÁTICAS

AUTOR: ADRIANA PÉREZ NAVARRO
TUTORES: ING. VLADIMIR MILIÁN NUÑEZ
LIC. RAYNEL BATISTA TELLEZ

LA HABANA, SEPTIEMBRE DEL 2020

AGRADECIMIENTOS

DEDICATORIA

DECLARACIÓN JURADA DE AUTORÍA

Declaro por este medio que yo **Adriana Pérez Navarro**, con carnet de identidad **97101104895**, soy el autor principal del trabajo de diploma **Método para la detección de tópicos en la red de mensajería instantánea de la Universidad de las Ciencias Informáticas.**, y que autorizo a la **Universidad de las Ciencias Informáticas** a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Y para que así conste, firmo la presente declaración jurada de autoría en La Habana a los ____ días del mes de _____ del año _____.

Adriana Pérez Navarro

Ing. Vladimir Milián Nuñez

Raynel Batista Tellez

RESUMEN

La detección de tópicos consiste en descubrir las estructuras semánticas subyacentes en documentos y buscar similitudes entre textos. Esta tiene como propósito colocar de forma automática, documentos dentro de un número fijo de categorías (temas o clases) predefinidas, en función de su contenido. El objetivo de esta investigación es desarrollar un método que permita identificar tópicos en los mensajes emitidos en la red de mensajería instantánea de la Universidad de las Ciencias Informáticas (UCI). En la investigación se realizó un estudio sobre conceptos asociados a redes de mensajería instantánea, análisis de la conversación, minería de texto, y detección de tópicos. Además, se describió el procedimiento que sigue el método presentado, se analiza el funcionamiento del Modelo de Tópicos Latent Dirichelt Allocation (LDA) para la detección y se realizaron validaciones en aras de verificar la calidad de la solución. Como resultado final se obtienen visualizaciones de tópicos así como las palabras claves representativas de cada tópico.

Palabras clave: Modelo LDA, detección de tópicos, redes de mensajería instantántanea, minería de texto.

ABSTRACT

Topic detection consists of discovering the underlying semantic structures in documents and looking for similarities between texts. It aims to automatically place documents within a fixed number of predefined categories (topics or classes), depending on their content. The aim of this research is to develop a method that allows identifying topics in messages issued in the instant messaging network of the University of Computer Sciences (UCI). In the research, a study on concepts associated to instant messaging networks, conversation analysis, text mining, and topic detection was carried out. Besides, it was described the procedure that follows the method presented, the functioning of the Latent Dirichelt Allocation (LDA) Topic Model for detection was analyzed and validations were made in order to verify the quality of the solution. The final result is a visualization of topics and the representative keywords of each topic.

Keywords: LDA model, topic detection, instant messaging networks, text mining.

ÍNDICE

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Con el surgimiento y popularización de la Web e Internet se han abierto completamente nuevas vías para permitir la interacción fluida de participantes geográficamente distribuidos Aggarwal (2015), además, posibilitó la creación de redes sociales interactivas donde las personas o instituciones construyen la información Fumero et al. (2010).

Las redes sociales actúan como un punto de encuentro donde millones de usuarios pueden publicar y compartir opiniones, información o simplemente trivialidades sobre todo tipo de temas. Poder analizar y comprender toda esa información pública se está convirtiendo en uno de los principales objetivos por parte de empresas y organizaciones, que ven en estos lugares una fuente de información para analizar qué se dice sobre su área de influencia Vilares Calvo (2014).

Usualmente al escuchar el término red social se piensa en las grandes y conocidas redes sociales Twitter, LinkedIn, Facebook o Instagram, pero estas redes representan solo una pequeña minoría del número de mecanismos de interacción social presentes en la web. Por lo que se hace necesario resaltar que debido a que se basan en la interacción entre diferentes individuos y por tanto, tienen una naturaleza social, los correos electrónicos, blogs, fórums o los sistemas de mensajería instantánea, entre otros, también son considerados redes sociales Aggarwal (2015).

La mensajería instantánea, es una forma de comunicación en tiempo real entre dos o más personas, basada en mensajes de texto, a través de dispositivos conectados a una red, lo cual ha facilitado la comunicación entre las personas. Sin embargo, la mensajería instantánea es un arma de doble filo y podría ser mal utilizada para el intercambio de información ilegítima, influir en el comportamiento y emociones de las personas, incentivar o cometer actos delictivos. Es por ello que identificar, procesar y presentar los temas de conversación dentro de las interacciones de los usuarios en un formato que facilite su comprensión y análisis, ha sido objeto de atención por parte de la comunidad de investigadores y administradores de este tipo de sistemas Brun y Senso (2004).

En los sistemas de mensajería instantánea de redes institucionales, el poder determinar y descubrir las temáticas y/o tendencias de las conversaciones entre los usuarios de dicho sistema, cobra mayor importancia aún, sobre todo en redes institucionales de centros educativos

Ng (2016). En este caso se encuentra la Universidad de las Ciencias Informáticas (UCI), en la cual existe una red de mensajería instantánea (comúnmente conocida como Jabber), de la cual son miembro (o pueden interactuar en ella) todos los estudiantes y profesionales (trabajadores y especialistas) de la universidad.

Es muy interesante tener en cuenta también que los usuarios de las redes de mensajería instantánea suelen escribir a sus contactos cuando algo no anda bien, es decir, para quejarse al respecto de un servicio, producto, evento o persona como forma de desahogo, y por lo general, tiende a compartirlo con más de un contacto, y estos a su vez con sus contactos. Esto significa, que estas conversaciones son un gran termómetro que permiten medir el estado de opinión de los usuarios de la red de mensajería, así como identificar los temas que más preocupan a la comunidad e identificar posibles rumores o comportamientos no permitidos por los reglamentos internos de la universidad.

Para determinar/descubrir cuáles son los temas o tópicos principales sobre los que se habla en la red de mensajería de la universidad, con el objetivo de adoptar estrategias, es necesario aplicar técnicas de minería de texto y procesamiento del lenguaje natural.

El procesado y clasificación automático de textos es un área de investigación formada por diversas disciplinas. Estas incluyen Recuperación de la Información (IR), que se ocupa de encontrar documentos que satisfagan una determinada necesidad de información o consulta dentro de una gran base de datos de documentos; Procesamiento Natural del Lenguaje (NLP), que es una disciplina que abarca todos las técnicas de procesamiento automático tanto de lenguaje escrito como hablado; la Extracción de la Información (IE), que puede ser considerada un campo de NLP y está centrada en encontrar entidades explícitas y hechos dentro de un texto no estructurado. Finalmente, la Minería de Texto es el proceso de analizar el lenguaje natural escrito para descubrir información o conocimientos que son comúnmente difíciles de recuperar Andrade Alvarado et al. (2015).

La importancia de las aplicaciones de minería de texto ha aumentado en los últimos años debido a la gran cantidad de datos de texto, que se crean en una variedad de redes sociales, web y otras aplicaciones centradas en la información. Como resultado, ha incrementado la necesidad de diseñar métodos y algoritmos que puedan procesar una amplia variedad de aplicaciones de texto. Con el objetivo principal de análizar la información para descubrir patrones Aggarwal y Zhai (2012).

Diariamente la red de mensajería instantánea de la UCI genera un gran volumen de datos. Estos datos son analizados con herramientas que no distinguen entre la información relevante y no relevante, por lo que se invierte demasiado tiempo en el análisis de dicha información, lo que trae consigo la pérdida de la oportunidad de prevenir actividades ilegítimas en función de los reglamentos internos de la institución. Además, dichas herramientas limitan la obtención e identificación de los temas tratados en las conversaciones entre los usuarios. Por otra parte,

el carácter subjetivo del análisis realizado, y la carencia del personal necesario para realizar dicha tarea en su totalidad, hace que los resultados conseguidos sean inexactos e incompletos, y se ve afectada la fidelidad y el alcance de los mismos. A partir del análisis anterior es posible afirmar que los resultados obtenidos son ineficaces a pesar de los altos costos en tiempo y recursos destinados.

A partir de la situación anteriormente descrita se define el siguiente **problema a resolver**: el modo en que se analizan los datos generados por el sistema de mensajería instantánea de la Universidad de las Ciencias Informáticas limita la identificación proactiva de ideas, opiniones o expresiones que se repiten con frecuencia en determinadas circunstancias, usualmente conocidas como tópicos.

El problema definido se enmarca en el siguiente **objeto de estudio**: análisis de datos en redes de mensajería instantánea, en el **campo de acción**: métodos de detección de tópicos en la red de mensajería instantánea de la UCI.

El **objetivo general**: desarrollar un método que permita detectar tópicos en los mensajes emitidos en la red de mensajería instantánea de la UCI.

Idea a defender: si se desarrolla un método que contribuya a la detección de tópicos en la red de mensajería instantánea de la UCI, entonces, se contribuirá a la detección proactiva de patrones en el sistema.

Las **tareas de investigación** que se deben cumplir son:

- Análisis de los principales conceptos, trabajos y herramientas relacionados con la detección de tópicos.
- Análisis de los algoritmos existentes para el modelado automático de tópicos.
- Diseño de un método para la detección de tópicos en la red de mensajería instantánea de la UCI.
- Implementación de un método basado para la detección de tópicos en la red de mensajería instantánea de la UCI.
- Validación del comportamiento de la propuesta diseñada.

Para el desarrollo del presente trabajo se emplearon varios **métodos científicos**. Los **métodos teóricos** utilizados son:

Analítico-sintético: para descomponer el problema de investigación en elementos por separado, de esta forma se puede profundizar en el estudio de cada elemento, para luego sintetizarlos en la solución de la propuesta.

Análisis-síntesis: facilitó el procesamiento de la información obtenida en la investigación realizada sobre la detección de tópicos. Se realizó un estudio de los conceptos asociados al análisis de datos en redes de mensajería instantánea basándose en la revisión de sitios web, trabajos de diploma y artículos científicos.

Histórico-lógico: se empleó para investigar las herramientas existentes que permiten la detección de tópicos en textos cortos.

El **Método empírico** empleado fue:

Juicio de expertos: se empleó para determinar que columnas del set de datos eran irrelevantes para la propuesta de solución.

El presente documento cuenta con una estructura de tres capítulos, los cuales abordan los siguientes temas:

- Capítulo 1 Fundamentación teórica sobre el análisis de datos en redes de mensajería instantánea: Se definen los conceptos más relevantes relacionados con las redes sociales, minería de texto, y algoritmos de detección de tópicos, así como el Proceso de Descubrimiento del Conocimiento con las fases que lo componen.
- Capítulo 2 Método para la detección de tópicos en la red de mensajería instantánea de UCI: se definen las tecnologías y herramientas a utilizar en la construcción de la solución. Se presenta un método para determinar tópicos en la red de mensajería instantánea de la UCI a partir de los datos generados por el sistema.
- Capítulo 3 Validación del método implementado: se realiza la validación del método desarrollado mediante varios criterios para determinar que la solución propuesta responde satisfactoriamente al objetivo de la investigación.

CAPÍTULO 1

FUNDAMENTACIÓN TEÓRICA SOBRE EL ANÁLISIS DE DATOS EN REDES DE MENSAJERÍA INSTANTÁNEA

En el presente capítulo se definen los conceptos más relevantes relacionados con las redes sociales, minería de datos, y algoritmos de detección de tópicos, así como el Proceso de Descubrimiento del Conocimiento con las fases que lo componen.

1.1. Análisis de datos en redes de mensajería instantánea

La mensajería instantánea es un tipo de servicio que permite el intercambio de mensajes de texto que llegan en tiempo casi real al destinatario Alsina (2020). La información que se puede obtener de estos mensajes es muy variada: estados de ánimo, intereses personales, emociones, posiciones políticas, entre otros. Por esta razón el análisis de las conversaciones de los usuarios es de gran interés en los procesos de toma de decisiones.

El análisis de la conversación es una línea discursiva de corte etnometodológico interesada en los intercambios verbales y las conversaciones corrientes Atkinson et al. (1984); Drew; Sacks (1972); Schegloff et al. (1977). En el transcurso de las conversaciones se demuestra un grado de competencia social para hacer comprensible ante los demás nuestro comportamiento e intenciones e interpretar el de los otros Godoi et al. (2014). Para esto se hace uso de la minería de datos.

La minería de datos es el estudio de la recogida, limpieza, procesamiento, análisis y obtención de conocimientos útiles a partir de los datos. Existe una amplia variación en términos de los dominios de los problemas, aplicaciones, formulaciones y representaciones de datos que se encuentran en las aplicaciones reales. Por lo tanto, la minería de datos es un término amplio que se utiliza para describir estos diferentes aspectos del procesamiento de datos Aggarwal

(2015) .

La minería de texto es una variación de la minería de datos, que tiene como objetivo analizar y descubrir cualquier patrón interesante, incluidas tendencias y valores atípicos, en datos de texto Aggarwal y Zhai (2012), para ayudar aún más a los usuarios a analizar, digerir información y facilitar la toma de decisiones a partir de diferentes tipos de datos de texto.

Debido a la gran cantidad y variedad de datos que se dispone, el problema de minería de texto ha ido cobrando importancia en los últimos años. Hay una gran necesidad de diseñar métodos (técnicas o algoritmos) que procesen estos datos y extraigan información de forma eficiente Aggarwal y Zhai (2012).

Estos métodos se engloban en lo que se conoce como *machine learning* (ML) o aprendizaje automático Silvestre Gómez (2018). En particular, se define el ML Murphy (2012) como un conjunto de técnicas o métodos que permiten extraer patrones en los datos y usar otros patrones para realizar predicciones u otras decisiones que poseen cierta incertidumbre.

Se pueden diferenciar tres tipos de problemas ML según Silvestre Gómez (2018):

1. Clasificación: es un tipo de problema de aprendizaje supervisado que se caracteriza por tener datos de salida categóricos, es decir, datos que toman valores reales de un conjunto previamente definido. Su objetivo es obtener un dato de salida, y , en función de las características de entrada, x. Se pueden diferenciar dos tipos de clasificadores: binarios, en los que la salida puede tomar dos valores, y multiclase, en los que la salida toma diversos valores. Con el clasificador se busca una función que relacione las entradas con una salida:

$$y = f(x) \tag{1.1}$$

   Esta función es desconocida y se desea encontrar aquella con la que se obtenga menor error de clasificación. Para ello, se emplea un conjunto de datos de entrenamiento cuyas etiquetas son conocidas y, posteriormente, se empleará esta función al resto de datos en los que la salida es desconocida. A esto se le llama generalización. Se debe obtener un clasificador que generalice correctamente a todos los datos de entrada. (Fig 1.1).

2. Regresión: este tipo de problemas se diferencian de los de clasificación en que la salida es una variable continua. Algunos ejemplos de regresión son: la predicción de la bolsa, predicción de localización o predicción de la temperatura. Se pueden encontrar diversas clases:

   a) Regresión lineal, que realiza la predicción empleando una función lineal.

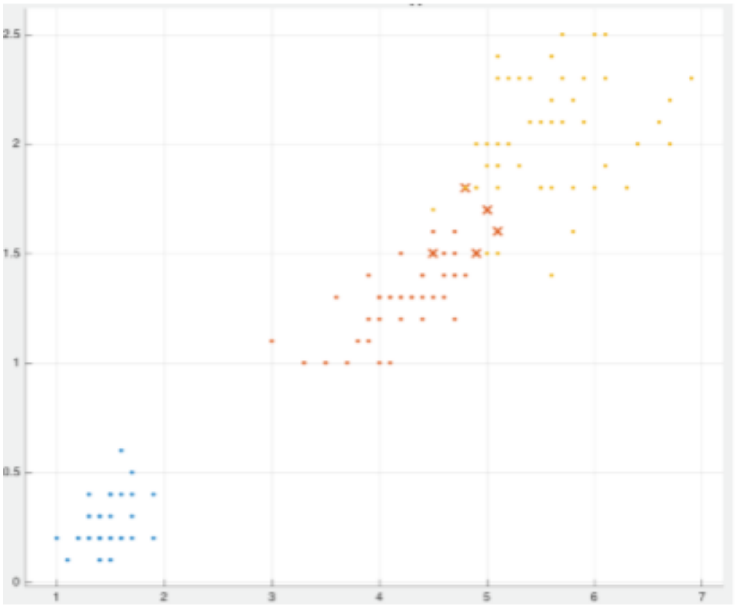

Figura 1.1: Ejemplo de clasificación usando *support vector machine* Athuraliya et al. (2015).

b) Regresiones no lineales, que pueden usar funciones cuadráticas, polinómicas, radiales, ..

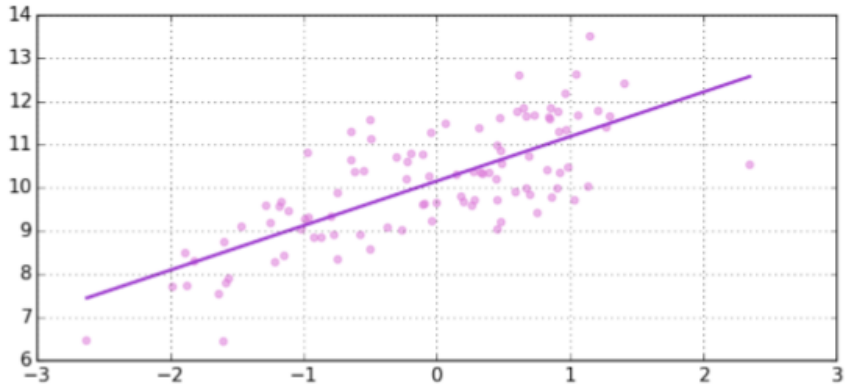

Figura 1.2: Ejemplo de Regresión Athuraliya et al. (2015).

3. Agrupamiento. Es un problema no supervisado en la que los datos no están etiquetados y que busca poner los datos de entrada en diversos conjuntos. El número de conjuntos o *clusters*, K, se encuentra definido por el usuario. En la actualidad, hay diversas técnicas que buscan obtener el número de *clusters* de forma automática, haciendo que la intervención humana sea mínima. (Ver figura 1.3).

Hay diferentes modelos en función de cómo se deseen agrupar los datos: *k-means*, basado en centroides y la distancia de los datos a éstos; algoritmo *expectation-maximization* (EM) que emplea distribuciones estadísticas para definir los *clusters*. La agrupación a menudo se usa como un paso intermedio en otras aplicaciones como análisis y clasificación atípicos.

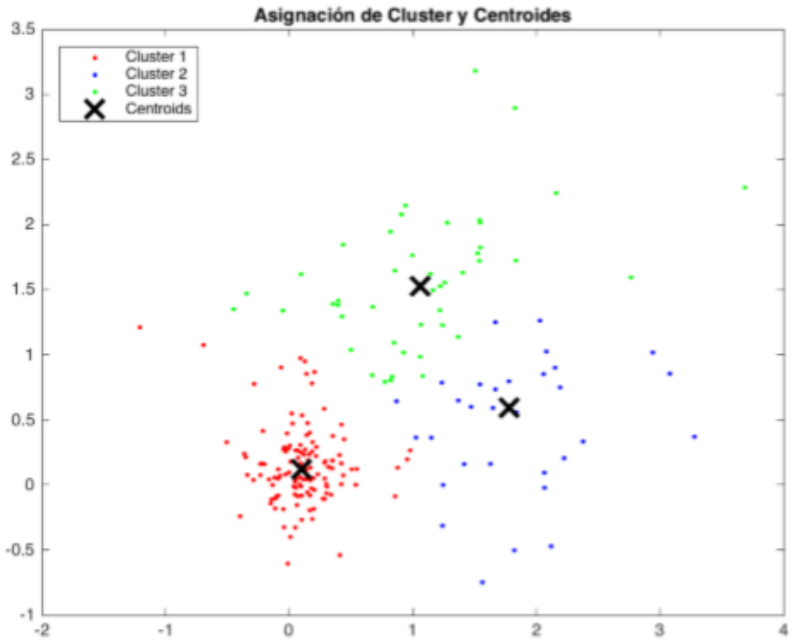

Figura 1.3: Ejemplo de *clustering* usando el algoritmo de *K-means* Athuraliya et al. (2015).

Para darle cumplimiento al objetivo de esta investigación es necesario agrupar mensajes en función de su contenido. Se trata de un problema en el que no se conoce previamente el tema del texto, sino que solo se tiene el contenido del mismo como entrada. Por tanto, se busca emplear técnicas de agrupamiento para agrupar los mensajes por tópicos.

1.2. Detección de tópicos

La detección de tópicos consiste en descubrir las estructuras semánticas subyacentes en documentos y buscar similitudes entre textos Hammoe (2018). El objetivo es colocar, de forma automática, documentos dentro de un número fijo de categorías (temas o clases) predefinidas, en función de su contenido. El procedimiento del modelado es dependiente del tipo de aprendizaje Andrade Alvarado et al. (2015).

Existen multitud de modelos (algoritmos) que se centran en los problemas de modelado de tópicos Jelisavčić et al. (2012). Sin embargo, la mayoría comparten una estructura funcional. Crean un corpus de documentos, de los cuales extraen todas las palabras que lo componen, vocabulario. A este último se le aplican técnicas probabilísticas, dependientes del modelo empleado, para generar cada uno de los tópicos.

El autor Blei y Lafferty (2009), define un tópico como el conjunto de elementos que pueden representar una temática presente en una colección de documentos sin pérdida de información estadística. Entre los modelos de tópicos existentes, se encuentran el modelo *Latent Dirichlet Allocation* y el CTM (*Correlated Topic Model*).

Estos modelos de tópicos se cimientan sobre las siguientes definiciones:

Una palabra w es la unidad elemental de un documento textual y se define como un elemento de un vocabulario indexado V. Para efectos de estos modelos, para representar una palabra se hace uso de vectores unitarios en donde la n-ésima palabra de V se representa con un vector de largo $|V|$ en el cual sólo su componente n-ésima es igual a uno.

Un documento es un arreglo de palabras descrito como $w = (w1, w2, \ldots, wN)$, donde wn es la n-ésima palabra de este.

Un corpus es una colección de documentos descrita como $D = (w1, w2, \ldots, wM)$,

Un tópico es una distribución de probabilidad sobre un vocabulario V fijo. Por ejemplo, el tópico político está descrito por palabras como partido, diputado, senado, ley de manera frecuente y palabras como guerra, marcador, gol con probabilidad casi nula.

1.2.1. Algoritmos de modelado de tópicos

A continuación se muestran algunos algoritmos que buscan resolver el problema de modelado de tópicos según Silvestre Gómez (2018). Estos modelos hacen referencia a los métodos de aprendizaje automático. Entre las características generales de estos algoritmos se hallan la consideración de que cada documento se compone de varios temas o tópicos y que cada tema se supone un conjunto de palabras que lo representa.

1.2.1.1. Modelo Probabilistic Latent Semantic Analysis

Este modelo fue introducido en 1999 y es una técnica del área de procesado del lenguaje natural NLP, *Natural Language Processing*) que representa el contenido de los textos en forma de vector y calcula la similitud de estos textos en función de las palabras que contenga. La finalidad de este algoritmo, como su nombre indica, es extraer información de las relaciones semánticas de los documentos.

Es un modelo generativo que transforma estos vectores en una representación de menor dimensión en la que todo el corpus queda reflejado. Ese espacio se denomina, *latent semantic space*. Se supone un conjunto de documentos $D = \{d_1, \ldots, d_N\}$ que contiene palabras del vocabulario $W = \{w_1, \ldots, w_M\}$. La representación de los datos se puede hacer mediante una matriz $\mathcal{N}$ de tamaño NxM en la que cada elemento $n = (d_i, w_j)$ es el número de ocurrencias de la palabra w_j en el documento d_i. Para ello hace uso de la Descomposición en Valores Singulares (SVD, *Singular Value Decomposition*) que es una técnica que emplea una matriz para obtener $\mathcal{N}$. Esta representación se emplea en los algoritmos de la clase *bag-of-words* en los que la relación entre palabras no se tiene en consideración.

El algoritmo *Probabilistic Latent Semantic Analysis* (pLSA) parte de un modelo de variable latente de la concurrencia de los datos. Asocia una variable no observada, $z \in \mathcal{Z} = \{z_1, \ldots, z_k\}$

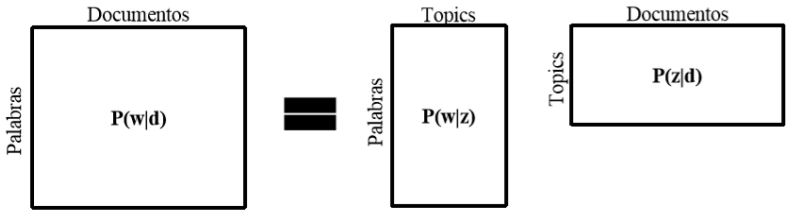

Figura 1.4: Representación de los datos en pLSA (Silvestre Gómez, 2018).

una observación. Se define un modelo de probabilidad conjunta:

$$P(d, w) = P(d)P(w|d) \tag{1.2}$$

en el que la probabilidad condicional viene dada por:

$$P(w|d) = \sum_{z \in \mathcal{Z}} P(w|z)P(z|d) \tag{1.3}$$

Este modelo considera que las variables d y w son estadísticamente independientes condicionadas al estado de la variable latente z. Esto se conoce como modelo de parametrización asimétrica y su representación se muestra en la siguiente figura.

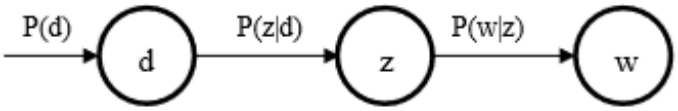

Figura 1.5: Modelo asimétrico Silvestre Gómez (2018).

En esta parametrización cada clase se elige condicionado al documento y cada palabra se genera dependiendo de la clase. Debido a que los valores de z son mucho menores que el número de palabras/documentos, esta variable puede actuar como cuello de botella y conviene emplear otro modelo equivalente de parametrización simétrica:

$$P(d, w) = \sum_{z \in \mathcal{Z}} P(z)P(d|z)P(w|z) \tag{1.4}$$

Este segundo modelo se caracteriza por la obtención tanto del documento como de la palabra a partir de la clase. La representación de este modelo es la siguiente:

Para la estimar la máxima verosimilitud del modelo pLSA se emplea el algoritmo *Expectation Maximization* (EM), un método iterativo se caracteriza por emplear la divergencia de Kullback-Leibler como función objetivo y por alternar dos pasos:

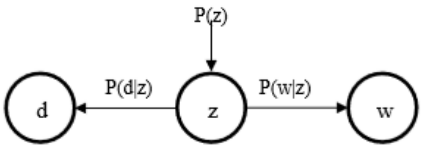

Figura 1.6: Modelo simétrico Silvestre Gómez (2018).

- Expectativa, donde se calcula las probabilidades a posteriori de las variables latentes. Para ello se emplea la siguiente expresión:

$$P(z|d,w) = \frac{P(z)P(d|z)P(w|z)}{\sum_{z' \in \mathcal{Z}} P(z')P(d|z')P(w|z')} \tag{1.5}$$

- Maximización, donde se actualizan los parámetros atendiendo a las siguientes expresiones:

$$P(w,z) \propto \sum_{d \in \mathcal{D}} n(d,w)P(z|d,w) \tag{1.6}$$

Finalmente, con el fin de conseguir una mayor generalización, se emplea una variación del algoritmo EM, *tempered* EM, que añade un parámetro de control a la estimación para evitar el *overfitting* y emplea otra función objetivo, energía libre de Helmholtz:

$$\mathcal{F}_\beta = -\beta \sum_{d,w} n(d,w) \sum_z \widetilde{P}(z;d,w) log P(d,w|z)P(z) + \sum_{d,w} n(d,w) \sum_z \widetilde{P}(z;d,w) log \widetilde{P}(z;d,w) \tag{1.7}$$

El modelo pLSA se caracteriza además por tener en cuenta las palabras polisémicas, consiguiendo en esas situaciones mejores resultados que las técnicas LSA. Sin embargo, presenta algunos inconvenientes, por ejemplo:

- El número de parámetros a estimar depende del tamaño del corpus. Esto puede causar problemas de sobreestimación al incrementar el número de documentos.
- Puede no considerarse un modelo generativo debido a que únicamente tiene en cuenta el vocabulario presente en el set de entrenamiento para generar los documentos. Estos problemas se resuelven con el modelo *Latent Dirichlet Allocation*.

1.2.1.2. Modelo Latent Dirichlet Allocation

Es un modelo probabilístico generativo, no supervisado y no parametrizado. Fue publicado en 2003 por Blei et al. (2003) y se presenta como el primer algoritmo de Modelado de tópicos.

Resuelve el problema del pLSA del incremento de parámetros a estimar, empleando la distribución de Dirichlet que considera cada parámetro como una variable aleatoria. Con esta distribución también se consigue que el modelo sea plenamente considerado generativo. Sin embargo, estas mejoras hacen que el algoritmo sea más complejo, haciéndolo incluso intratable a menos que se empleen aproximaciones con algoritmos, por ejemplo, Gibbs Sampling.

Al igual que pLSI, considera que cada documento es una distribución de tópicos y que cada uno de estos tópicos es una distribución de probabilidad de las palabras de un documento. Considera a los documentos como vectores en los que cada posición se corresponde con el número de veces que está presenten una palabra en el documento. Se considera un modelo generativo debido a que los tópicos se especifican antes de generar los datos.

El modelo LDA se caracteriza por realizar las siguientes suposiciones:

- Los tópicos son distribuciones de un vocabulario fijo. Se asumen K tópicos en la colección.
- Cada documento contiene estos tópicos en diferente proporción. El algoritmo quiere encontrar cada tópico de los documentos del corpus.

LDA se basa en un modelo de variables ocultas que se basan en la interacción de los datos observados con variables aleatorias ocultas. En este caso, los datos observados son cada uno de los textos y las variables ocultas son los tópicos de cada documento. Dada la distribución del posterior de las variables ocultas, se determina la descomposición de tópicos del corpus.

Se suponen K tópicos; V palabras en el vocabulario; $\overrightarrow{\alpha}$ un vector de tamaño K que se emplea como parámetro de la distribución de Dirichlet que define la proporción de los tópicos de cada documento y η, un escalar empleado en la distribución de Dirichlet que define la distribución de cada tópicos en función de las palabras.

El algoritmo LDA se puede resumir en los siguientes pasos:

1. Para cada tópico:

   a) Definir una distribución de las palabras. $\overrightarrow{\beta_k} \sim Dir_v(\eta)$

2. Para cada documento:

   a) Definir un vector de proporciones de tópico. $\overrightarrow{\theta_d} \sim Dir(\overrightarrow{\alpha})$

   b) Para cada palabra:

      1) Asignar un tópico. Para ello se sigue una distribución multinominal.
      $Z_d, n \sim Mult(\overrightarrow{\theta_d}), Z_d, n \in \{1, \ldots, K\}$

      2) Muestrear una palabra del vocabulario. Se escoge siguiendo una distribución multinominal.
      $W_d, n \sim Mult(\overrightarrow{\beta}_{Z_{d,n}}), W_d, n \in \{1, \ldots, V\}$

La siguiente figura, muestra un gráfico del modelo LDA, donde cada nodo representa una variable aleatoria y las flechas, las dependencias entre variables.

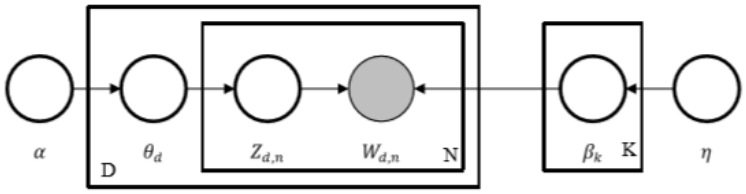

Figura 1.7: Representación gráfica del modelo LDA Silvestre Gómez (2018).

$\overrightarrow{\beta}_{1:K}$ representa la estructura de tópicos; $\overrightarrow{\theta}_{1:D}$, las proporciones de cada tópico por documento y $Z_{1:D,1:N}$, la asignación de un tópico a cada palabra.

Como se ha mencionado anteriormente, el modelo LDA hace uso de la distribución de Dirichlet que pertenece a la familia de las distribuciones exponenciales y se caracteriza porque sus valores suman uno. El algoritmo LDA tiene dos variables Dirichlet, la proporción de los tópicos, $\overrightarrow{\theta}_d$, y la distribución de estos tópicos por el vocabulario, $\overrightarrow{\beta}_k$.

$$p(\overrightarrow{\theta}_{1:D}, z_{1:D,1:N}, \overrightarrow{\beta}_{1:K}|W_{1:D,1:N}, \alpha, \eta) = \frac{p(\overrightarrow{\theta}_{1:D}, \overrightarrow{Z}_{1:D}, \overrightarrow{\beta}_{1:K}|\overrightarrow{W}_{1:D}, \alpha, \eta)}{\int_{\overrightarrow{\beta}_{1:K}} \int_{\overrightarrow{\theta}_{1:D}} p(\overrightarrow{\theta}_{1:D}, \overrightarrow{Z}_{1:D}, \overrightarrow{\beta}_{1:K}|\overrightarrow{W}_{1:D}, \alpha, \eta)} \tag{1.8}$$

Esta expresión es intratable debido a la integral del denominador. Por tanto, es necesario aplicar métodos de aproximación. Cada una de ellas posee ventajas e inconvenientes, pero es muy importante elegir aquella en la que exista un compromiso entre la velocidad, la complejidad, la precisión y la simplicidad conceptual. Existen diversas técnicas:

- Inferencia variacional de campo medio. Este método busca simplificar la expresión del posterior empleando una parametrización que se acerque al verdadero valor. Parte de la idea de que el motivo de que la expresión sea intratable es la dependencia existente entre las variables. Por tanto, se harán uso de unos parámetros con los que se conseguirá la independencia de dichas variables.

  Se asignará un parámetro a cada una de las variables. Finalmente, el cálculo del posterior se realiza con la siguiente expresión:

$$q(\overrightarrow{\theta}_{1:D}, z_{1:D,1:N}, \overrightarrow{\beta}_{1:K}) = \prod_{k=1}^{K} q(\overrightarrow{\beta}_k|\overrightarrow{\lambda}_k) \prod_{d=1}^{D} (q(\overrightarrow{\theta}_d|\overrightarrow{\gamma}_d) \prod_{n=1}^{N} q(Z_{d,n}|\overrightarrow{\phi}_{d,n})) \tag{1.9}$$

  donde cada variable oculta se describe por una distribución: $\overrightarrow{\beta}_{1:K}$ son los tópicos cuyos valores vienen dados por la distribución de Dirichlet de parámetro $\overrightarrow{\lambda}_k$; la proporción de los

tópicos $\overrightarrow{\theta}_{1:D}$ viene dada por la distribución de Dirichlet de parámetro $\overrightarrow{\gamma}_d$; y la asignación del tópico viene descrita por una distribución multinomial de parámetro $\overrightarrow{\phi}_{d,n}$.

Una vez calculado el posterior, la función objetivo a minimizar viene dada por la distancia KullbackLeibler (KL):

$$arg_{\overrightarrow{\gamma}_{1:D},\overrightarrow{\lambda}_{1:K},\overrightarrow{\phi}_{1:D,1:N}} min\, KL(q(\overrightarrow{\theta}_{1:D}, z_{1:D,1:N}, \overrightarrow{\beta}_{1:K})||p(\overrightarrow{\theta}_{1:D}, z_{1:D,1:N}, \overrightarrow{\beta}_{1:K}|W_{1:D,1:N})) \quad (1.10)$$

Esta función objetivo no es tratable y debe reemplazarse por otra que hace uso de los parámetros previamente introducidos:

$$\mathcal{L} = \sum_{k=1}^{K} E[log\, p(\overrightarrow{\beta}_k|\eta)] + \sum_{d=1}^{D} E[log\, p(\overrightarrow{\theta}_d|\overrightarrow{\alpha})] + \sum_{d=1}^{D}\sum_{n=1}^{N} E[log\, p(Z_{d,n}|\overrightarrow{\theta}_d)] + \sum_{d=1}^{D}\sum_{n=1}^{N} E[log\, p(W_{d,n}|Z_{d,n}, \overrightarrow{\beta}_{1:K})] + H(q) \quad (1.11)$$

Donde H hace referencia a la entropía.

- Muestreo de Gibbs. Método propuesto por Griffiths y Steyvers que toma muestras del posterior para aproximarlas con una distribución. Se considera uno de los métodos de Monte Carlo de cadenas de Markov (MCMC, Markov Chain Monte Carlo). Estos métodos construyen cadenas de markov con la distribución deseada a partir de una serie de muestras. El proceso de muestreo comienza tras la ejecución de un gran número de pasos para asegurar la estabilidad. Este método obtiene la distribución condicional de la variable latente B a partir de la expresión:

$$P(Z_j = j|Z_{-i}, w) \propto \frac{n_{-i,j}^{(wi)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta} \frac{n_{-i,j}^{(di)} + \alpha}{n_{-i}^{d_i} + K\alpha} \quad (1.12)$$

donde:
$n_{-i,j}^{(wi)}$ es el número de veces que aparece la palabra w_i en el tópico j
$n_{-i,j}^{(\cdot)}$ es el número total de palabras asignadas al tópico j
$n_{-i,j}^{(di)}$es el número de palabras del documento d_i asignadas al tópico j
n^{d_i} es el número total de otras palabras en el documento d_j

Para realizar el muestreo se inicializa la variable Z_i a un valor comprendido entre uno y el número total de tópicos K y se va actualizando en cada iteración con la expresión. Tras numerosas iteraciones, cuando se alcanza una distribución estacionaria, se tomarán las muestras de Z_i.

1.2.1.3. Modelo Hierarchical Latent Dirichlet Allocation

Este modelo es una extensión del método LDA que organiza los tópicos en una estructura en árbol en lugar de la estructura plana usada en LDA. Este modelo se caracteriza por combinar el cálculo del prior con la verosimilitud basándose en la variación jerárquica del LDA. Dado un corpus en el que cada documento contiene unas palabras, se busca descubrir los tópicos presentes en cada documento, organizándolos jerárquicamente. Emplea una aproximación bayesiana no paramétrica que permite construir el árbol de tópicos conforme van llegando los datos, este modelo se denomina *nested Chinese Restaurant Process* (nCRP) y es una adaptación del algoritmo CRP para emplearlo con jerarquías.

Cada nodo de la estructura representa una variable aleatoria que tiene asociada la distribución palabra-tópico. Este algoritmo tiene en cuenta dos consideraciones: se debe definir la profundidad del árbol y cada documento queda asociado a una rama de dicho árbol, aunque idealmente pueda pertenecer a varias.

Considerando el conjunto de datos de entrada, corpus, formado por documentos que contienen palabras de cierto vocabulario, el modelo asume que estas palabras se han generado siguiendo un método de mezcla de palabras donde cada proporción, θ, es aleatoria. Cada una de estas palabras está asociada a uno o varios tópicos, z. El modelo parte de un árbol previamente definido de L niveles y define un proceso de tres pasos para generar los documentos:

- Primero, se elige un camino desde un nodo raíz a una hoja.
- Segundo, se define un vector de proporciones de tópicos, θ, siguiendo la distribución de Dirichlet.
- Tercero, se generan las palabras del documento mediante una mezcla de tópicos desde el nodo raíz al nodo hoja empleando el vector previamente calculado, θ.

El algoritmo CRP se emplea para generar la estructura del árbol de forma aleatoria, define el prior de las proporciones de cada tópico. Finalmente, el algoritmo *Hierarchical Latent Dirichlet Allocation* (hLDA) puede estructurarse en:

1. Elegir el nodo raíz, C_1
2. Para cada nivel $l \in \{2, \ldots, L\}$:
   a) Elegir el nodo,$C_{\mathcal{I}-1}$, atendiendo a las expresiones del algoritmo de nCRP:

$$p(nodo\ ocupado\ i \mid palabras\ previas\ ) = \frac{m_i}{\gamma + m - 1} \tag{1.13}$$

$$p(siguiente\ nodo\ desocupado\ i \mid palabras\ previas\ ) = \frac{\gamma}{\gamma + m - 1} \tag{1.14}$$

b) Elegir el nodo C_l a partir del anterior

3. Calcular el vector de proporciones de tópicos, θ, empleando la distribución de Dirichlet.
4. Para cada palabra $n \in \{2, \ldots, N\}$:
   a) Seleccionar un tópico $z \in \{2, \ldots, Z\}$ siguiendo la distribución multinomial.
   b) Seleccionar las palabras asociadas al tópico C_z

Este modelo se ilustra en la siguiente figura, donde el nodo T hace referencia al conjunto infinito de ramas de L niveles creados con el algoritmo nCRP. Dado un T, la variable $C_{m,l}$ hace referencia al tópico asociado al nivel l y al camino m. En el caso de que llegue un nuevo documento que no está reflejado en el árbol, se empleará el algoritmo nCRP para asociarlo a su camino. Se debe tener en cuenta que este nuevo documento puede compartir camino con otro documento previamente asociado, o partir de uno completamente nuevo. Esto se traduce en que el posterior de un nuevo documento puede depender del posterior de los documentos previamente procesados.

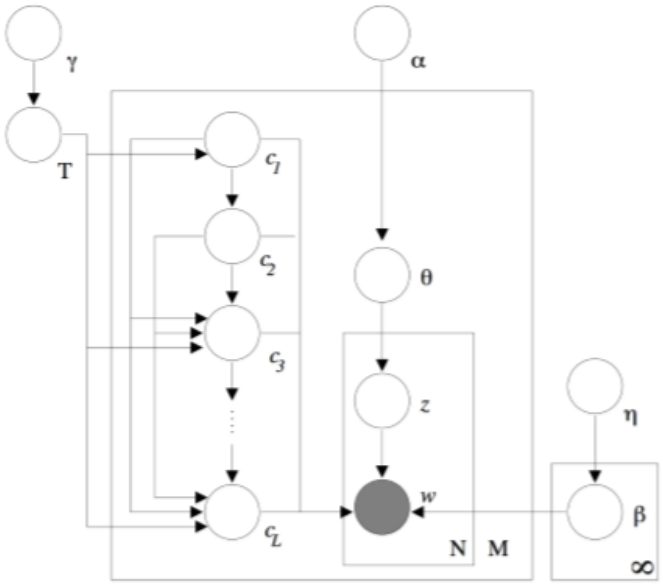

Figura 1.8: Modelo hLDA Silvestre Gómez (2018).

Al igual que en el método LDA, el posterior requiere de una aproximación para poder ser calculable. En este caso, el método es el muestreo de Gibbs, detallado en el apartado anterior. Las variables necesarias son: $W_{m,n}$, corresponde a la n-ésima palabra del m-ésimo documento; $C_{m,l}$, el nodo correspondiente al l-ésimo tópico en el m-ésimo documento; y $Z_{m,n}$, que corresponde a la asignación de la n-ésima palabra del m-ésimo documento a uno de los L tópicos disponibles. Con el método de Gibbs se quiere muestrear las variables $C_{m,l}$ y $Z_{m,n}$.

El proceso de muestreo se divide en dos partes:

1. Muestrear $Z_{m,n}$ con el método descrito en LDA.

2. Dados los valores de las variables ocultas en LDA, muestrear $C_{m,l}$, empleando el prior de CRP. La distribución de los L tópicos asociados al documento m viene dado por:

$$p(C_m|W, C_{-m}, Z) \propto p(W_m|C, W_{-m}, Z)p(C_m|C_{-m}) \tag{1.15}$$

Para calcular la verosimilitud es necesario hacer la aproximación:

$$p(W_m|C, W_{-m}, Z) = \prod_{l=1}^{L} \frac{\Gamma(n^{(\cdot)}_{C_{m,l}}) + W_\eta}{\prod_w \Gamma(n^{(w)}_{C_{m,l}-m} + \eta)} \frac{\prod_w \Gamma(n^{(w)}_{C_{m,l}-m} + n^{(w)}_{C_{m,l}m} + \eta)}{\Gamma(n^{(\cdot)}_{C_{m,l}-m} + n^{(\cdot)}_{C_{m,l}m} + \eta)} \tag{1.16}$$

donde:
$n^{(w)}_{C_{m,l}-m}$ es el número de veces que aparece la palabra w en el tópico $C_{m,l}$ en los documentos que no sean el actual.
W es el tamaño del vocabulario.
$\Gamma(\cdot)$ es la función gamma estándar.

1.2.1.4. Modelo Correlated Topic

Método que mejora el modelo LDA introduciendo correlaciones entre tópicos. En los modelos anteriores, cada uno de los tópicos se suponen independientes del resto. Con este modelo se quiere encontrar una estructura de documentos más compleja teniendo en cuenta las relaciones entre tópicos. Generalmente, este modelo se adapta mejor a los datos debido a la mayor información que se tiene en cuenta.

Correlated Topic Mode (CTM) hace uso de distribuciones normales para el cálculo de las proporciones de los tópicos, permitiendo ver la correlación entre éstos.
La notación que sigue este modelo es la siguiente:

- $W_{d.n}$ es la n-ésima palabra del documento d-ésimo.
- B_k es la distribución del tópico K
- $Z_{d,n}$ se refiere a la asignación de un tópico a la n-ésima palabra del d-ésimo documento.
- θ_d es la proporción de cada tópico del d-ésimo documento

Este modelo sigue el proceso generativo:

1. Estimar la distribución de los tópicos donde η sigue una distribución multinomial.

2. Para cada palabra del documento

   a) Asignar un tópico $Z_{d,n}|\eta \sim Mult(\theta_d)$

b) Asignar una palabra $W_{d,n}|\{Z_{d,n}, \beta_{1:K}\} \sim Mult(\beta_{Z_{d,n}})$

Este proceso se muestra en la siguiente imagen:

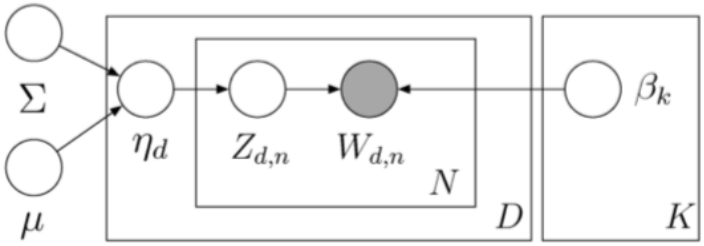

Figura 1.9: Representación gráfica CTM Silvestre Gómez (2018).

Como puede observarse, este método es muy parecido al LDA salvo en la obtención de las proporciones de los tópicos. En este caso se emplea una distribución multinomial en lugar de la de Dirichlet.
Al igual que en LDA, se emplea el método de aproximación de inferencia variacional de campo medio para calcular los parámetros de forma que la divergencia KL entra la aproximación y el verdadero posterior sea mínima. Sin embargo, al emplear distribuciones normales, este proceso se complica.

1.2.1.5. Modelo Author Topic

Se publica en 2004 como una extensión del modelo LDA y, posteriormente, es revisado en 2010.

Se fundamenta en los mismos principios que los modelos anteriores, pero añade el uso de metadatos para extraer la información del autor del texto. Cada palabra de un documento se asociará a un tópico y a un autor. Este modelo ve a cada autor como una distribución de tópicos y, al igual que en LDA, cada tópico es una distribución de palabras.

Con estos modelos se puede extraer más información de los documentos, por ejemplo, los temas sobre los que escribe determinado autor o qué autores escriben sobre un determinado tema.

Parte de la idea de que un grupo de autores, a_d, deciden escribir sobre determinado tema. Cada palabra de esos documentos se elige por el autor siguiendo una distribución y, de la misma manera que en LDA, se elige el tópico siguiendo una distribución tanto de palabras como de autores.

En la siguiente figura 1.10 se representa el *Author Topic Model* (ATM). Donde x es el autor correspondiente a una palabra, w, elegido del conjunto a_d. Los pesos de cada autor se emplean para seleccionar el tópico z y cada palabra se genera siguiendo la distribución ϕ.

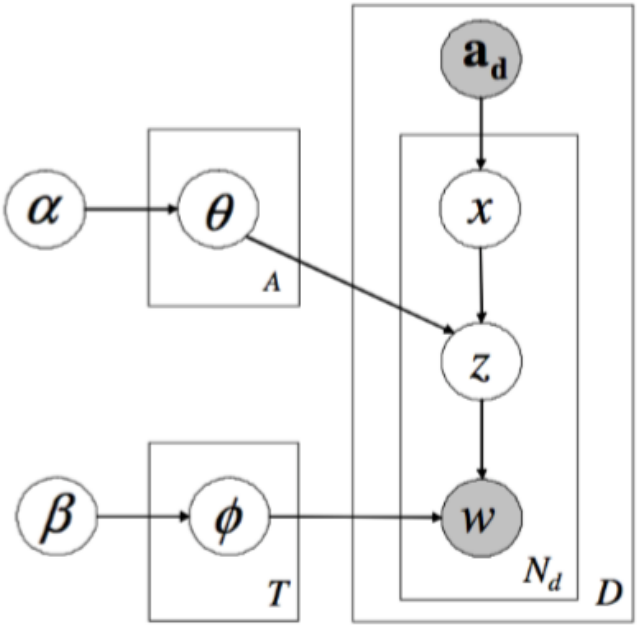

Figura 1.10: Modelo ATM Silvestre Gómez (2018).

Como método de aproximación del posterior se emplea el muestreo de Gibbs debido a su simplicidad. En este caso, el número de variables latentes se incrementa. No sólo se considera la distribución de los tópicos,z, sino que también se ha de tener en cuenta la distribución de los autores, x.

El modelo que se utiliza para desarrollar el método es el ya mencionado LDA. La elección de LDA se debe a que es el algoritmo más utilizado y probado para este tipo de problemas Hammoe (2018). En el paquete gensim de Python existe una implementación de este modelo.

1.3. Proceso de descubrimiento del conocimiento

En todo proceso de análisis de datos a gran escala es necesario seguir una serie de pasos que usualmente están dados por una metodología. Para el desarrollo de esta propuesta de solución se decidió utilizar el Proceso de Descubrimiento del Conocimiento o KDD (*Knowledge Discovery in Databases* por sus siglas en inglés) que, si bien no está considerado como una metodología, constituye y se ha establecido como una guía para proyectos de este tipo.

El KDD es definido por Fayyad et al. (1996b) como el proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles y, en última instancia, comprensibles en los datos.

El proceso KDD es interectivo e iterativo, involucra numerosos pasos son muchas decisiones que deben ser tomadas por el usuario Fayyad et al. (1996b):

1. **Integración y recopilación de datos:** se determinan las fuentes de información que

pueden ser útiles, seguidamente se transforman los datos a un formato común, unificando toda la información recogida, detectando y resolviendo inconsistencias.

2. **Selección, limpieza y transformación:** en esta fase se eliminan o corrigen los datos incorrectos y se decide la estrategia a seguir con los datos incompletos. También se proyectan los datos para considerar solo las variables y atributos que van a ser relevantes para hacer más fácil la tarea propia de la minería y así sean más útiles los resultados de la misma.

3. **Minería de Datos:** esta es la fase donde se define cual es la tarea a realizar y se elige el método que se va a utilizar.

4. **Evaluación e interpretación de los datos:** se evalúan los patrones y se analizan los expertos.

5. **Difusión y uso:** en esta última fase se hace uso del nuevo conocimiento y se hace partícipe de él a todos los posibles usuarios.

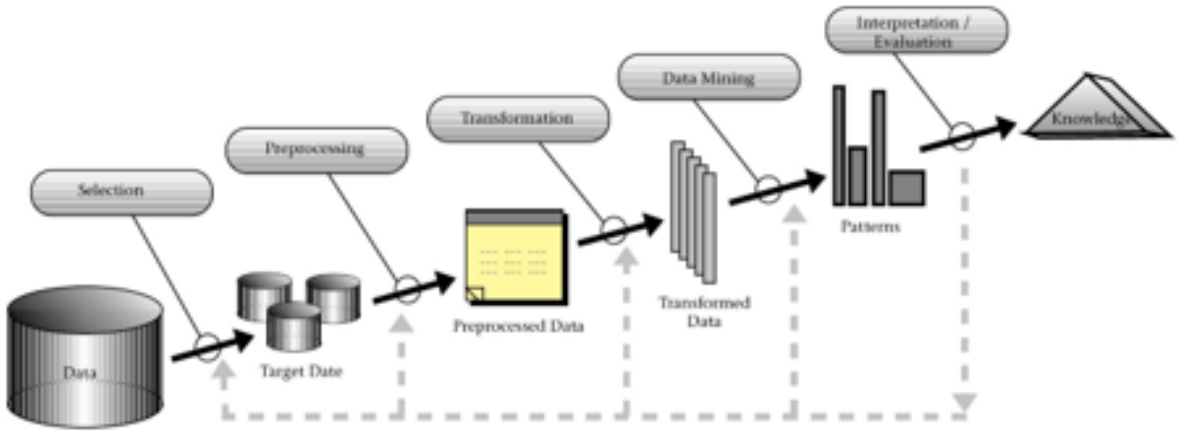

Figura 1.11: Etapas del proceso KDD. Tomado de Fayyad et al. (1996a).

El proceso KDD debe ser precedido por el desarrollo de un entendimiento del dominio de aplicación (área específica donde se aplicará) y la correcta identificación de las metas desde el punto de vista del usuario final Azevedo y Santos (2008), Fayyad et al. (1996a). Una vez finalizado el proceso KDD, el conocimiento descubierto puede ser usado directamente, incorporado a otros sistemas para futuras acciones o simplemente documentado y reportado a las partes interesadas (Fayyad et al., 1996b).

KDD puede involucrar importantes iteraciones y contener bucles entre pares de etapas (Ver fig 1.11). La mayor parte del trabajo en el proceso KDD está enfocado en las etapas de pre-

procesamiento y minería, aunque el resto de las etapas son igualmente importantes para la exitosa aplicación de KDD en la práctica Fayyad et al. (1996a).

1.4. Herramientas existentes para la detección de tópicos

Diferentes soluciones han sido desarrolladas para la detección de tópicos. Entre las soluciones que más se destacan se encuentran las siguientes herramientas:

Google PLAY[1] (tienda oficial de Google para terminales Android, distribuye películas, música, libros y, sobre todo, aplicaciones) sirve de apoyo a los desarrolladores, pues posibilita la clasificación de las revisiones por calificación para detectar fácilmente los más críticos. Sin embargo, algunos usuarios asignan puntuaciones muy bajas sólo para aumentar la visibilidad de su revisión y en muchos casos es más positiva que negativa su contenido Fonseca Torres (2017).

PosNeg Opinion es una herramienta desarrollada por el Departamento de Ciencia de la Computación en la Universidad Central de las Villas, para que el usuario analice un gran cúmulo de opiniones de manera sencilla, ya que se convierten los ficheros XML a texto plano. Fue desarrollada completamente en JAVA, por lo que es multiplataforma, necesita como entrada un fichero XML con todas las opiniones a analizar y como salida muestra cuántas son positivas y cuántas negativas. A petición del usuario también retorna el porcentaje, además de una lista con las opiniones negativas y positivas Amores et al. (2015).

TrendSpottr es una plataforma de inteligencia de tendencias predictivas utilizada por marcas globales, comercializadores y agencias digitales y de relaciones públicas. TrendSpottr predice las tendencias emergentes, el contenido viral y los principales influenciadores de cualquier tema en tiempo real. Detectan tendencias aceleradas con un alto potencial viral y un impacto en el mercado desde Twitter y otras fuentes de datos horas o días antes de que se hayan hecho ”populares” hayan alcanzado la conciencia de la corriente principal Bib (2020a).

Brandwatch es una que herramienta te ayuda a recopilar información clave al analizar miles de millones de conversaciones que se dan cada día en los entornos *online*. *Brandwatch Analytics*, te muestra los temas, tendencias y personas que están teniendo un impacto en tu sector Bib (2018). Además es la empresa líder a nivel mundial en *social intelligence* y tiene a las mejores personas trabajando en ella Bib (2020b).

Sin embargo, actualmente las herramientas que ofrecen estas funcionalidades son todavía escasas o han sido desarrolladas para uso interno exclusivamente. Si bien existen algunas herramienta que podríamos usar de manera gratuita, muchas simplemente ofrecen una versión de prueba con un número limitado de días de uso, o bien ofrecen funcionalidades muy limitadas. PosNeg Opinion, solo es capaz de clasificar las opiniones expresadas en una oración o la

[1] Tienda de aplicaciones de Android, disponible en: https://play.google.com/store

opinión que expresa el texto en su totalidad. Para TrendSpottr y Brandwatch resulta necesario pagar la licencia después de los días de prueba para continuar con su utilización, debido a que fueron concebidas con fines comerciales.

Las herramientas vistas realizan análisis de datos según sus necesidades, pero no satisfacen la solución del problema de la investigación, por lo que desarrollar un método es necesario.

1.5. Conclusiones del capítulo

A partir del análisis de la información presentada en este capítulo puede arribarse a las siguientes conclusiones:

- El estudio de enfoques y tendencias para el análisis de redes sociales permitió identificar diversos campos de aplicación emergentes, denotando el amplio interés de la comunidad científica internacional en el objeto de estudio de la investigación.
- El análisis de la conversación de usuarios en redes sociales permite identificar estados de ánimo, intereses personales, emociones, posiciones políticas, etc.
- Las herramientas actuales dedicadas a la detección de tópicos no satisfacen al problema de investigación.
- El modelo seleccionado para desarrollar el método es LDA por ser algoritmo más utilizado y probado para este tipo de problemas.

CAPÍTULO 2

DESARROLLO DE LA PROPUESTA DE SOLUCIÓN

En el presente capítulo se describe brevemente cuáles son los pasos a seguir para desarrollar el proceso de solución propuesto.

2.1. Presentación de la solución

Para el cumplimento del objetivo general del presente trabajo se deben detectar tópicos a partir de los mensajes de la red de mensajería instantánea de la UCI. El método propuesto sigue el procedimiento que se describe a continuación:

1. Carga y procesamiento de los datos presentes en el conjunto de datos de estudio.
2. Aplicación del modelo LDA a partir de los datos procesados.
3. Detección e interpretación de tópicos.

En la Fig 2.1 se muetra la estructura del metodo antes mencionado.

2.2. Carga y procesamiento de datos

2.2.1. Carga de datos

Para el desarrollo del presente trabajo se tomó como corpus un registro de mensajes de la red de mensajería instantánea de la Universidad de las Ciencias Informáticas almacenado en un archivo de extensión .csv, donde las columnas, que contienen los atributos del mensaje están separadas por comas, y las filas, que contienen un mensaje, por saltos de línea.

Cada mensaje tiene la siguiente estructura:

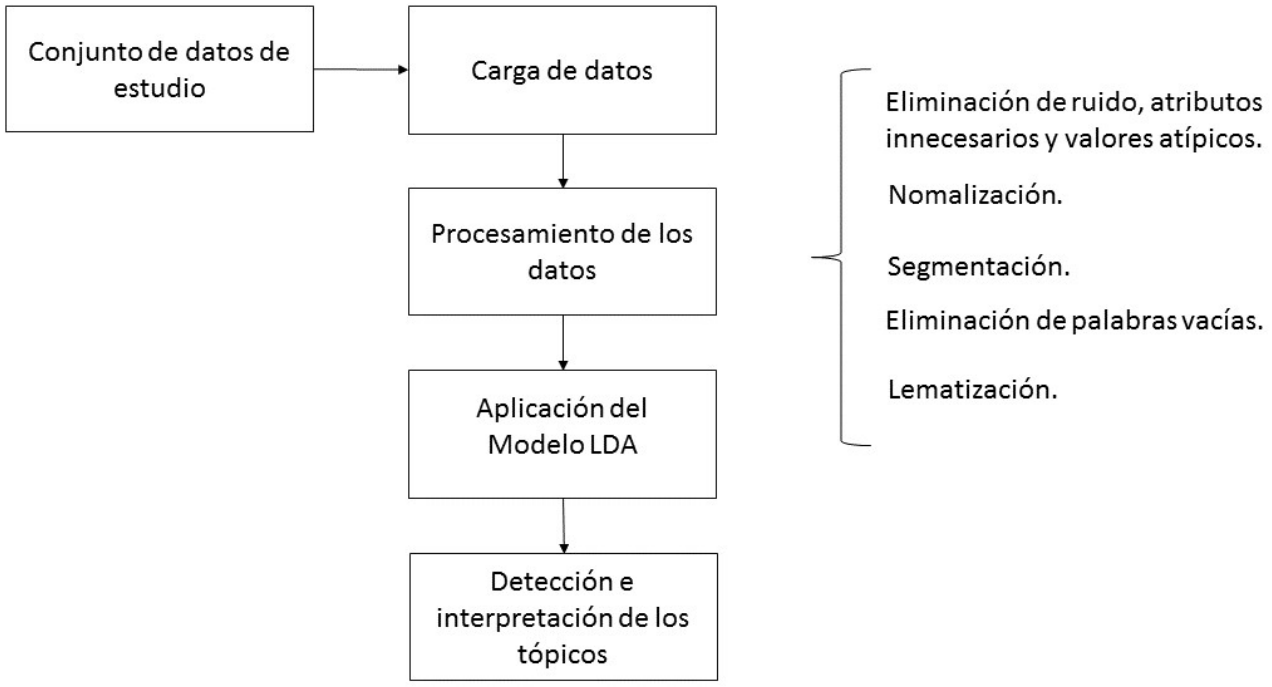

Figura 2.1: Estructura del metodo propuesto. Fuente: elaboración propia.

Conversación, Emisor, Cemisor, Receptor , Creceptor, Tiempo, Contenido

```
107923782,æ¬ð¥á¬,jabber.uci.cu/1a9efae7,xã€¡¬€ç0,jabber.uci.cu/f96e751e,1576026914488,"tengo metodologia  objetivos especificos"
```

Figura 2.2: Estructura de un mensaje.

- Conversación: es un número que identifica una conversación en el dominio.
- Emisor y Receptor: son cadenas de texto únicas que idetifican al usuario que emite y recibe el mensaje respectivamente.
- Cemisor y Creceptor: son cadenas de texto que identifican el cliente de mensajería instantánea del que emite y recibe el mensaje respectivamente.
- Tiempo: es el instante de tiempo en milisegundos en que se envía el mensaje.
- Contenido: es una cadena de texto con el contenido del mensaje.

Para desarrollar correctamente la solución que se propone se hace necesario cargar este registro de mensajes en una estructura que sea fácil y rápida de manipular ya que, una vez que los datos sean correctamente cargados va a ser necesario procesarlos. El manejo de este tipo de estructuras se lleva a cabo, usualmente, con lenguajes de programación de alto nivel. En la presente investigación se decidió hacer uso del lenguaje de programación Python en su versión 3.7.3 debido al gran número de librerías que posee para la manipulación eficiente de grandes volúmenes datos.

La librería de Python, Pandas, facilita el proceso de análisis de grandes volúmenes de datos de manera eficiente, por tal motivo, se decidió hacer uso de la misma en la presente investigación cargando el conjunto de datos de entrada en un objeto DataFrame de Pandas. Un objeto DataFrame de Pandas es una estructura de datos de alto rendimiento con la que es posible manejar fácilmente datos tabulares para el análisis de datos. En estos objetos cada una de las filas representa un registro. Mientras que en cada una de las columnas representan variables y, en este caso se define un nombre para cada una de estas columnas puesto que no poseen en la fuente original.

```
1 import pandas as pd
2 def load_data(path):
3     data = pd.read_csv(path, names=['id_conversation', 'id_sender', '
          id_sender_client', 'id_receiver', 'id_receiver_client', 'time', 'message
          '], nrows=100000)
4     return data
```

Figura 2.3: Función utilizada para cargar los datos de estudio.

Una vez cargado el registro de mensajes se inicia la fase de preprocesamiento de los datos, la etapa más importante en cualquier actividad de minería de textos ya que la calidad de la minería depende directamente de la eficiencia de esta fase Andrade Alvarado et al. (2015).

	id_conversation	id_sender	id_sender_client	id_receiver	id_receiver_client	time	message
20	107915949	€äxçßçþ¬	jabber.uci.cu/4db432fe	»¥þç€¬€¥é»	jabber.uci.cu/1a543b23	1.574226e+12	tu si tienes la apk de instagram
21	107915947	€äxçßçþ¬	jabber.uci.cu/4db432fe	¬é¬¶¬€ç€	jabber.uci.cu/df36ffa9	1.574226e+12	a ese me pincha
22	107915948	¬&x¬þçé	jabber.uci.cu/2cb78369	¬&x¬þçé	jabber.uci.cu	1.574226e+12	9204 1299 7382 2267
23	107915947	¬é¬¶¬€ç€	jabber.uci.cu/df36ffa9	€äxçßçþ¬	jabber.uci.cu/4db432fe	1.574226e+12	t lo envio x dukto
24	107915948	¬&x¬þçé	jabber.uci.cu/2cb78369	¬&x¬þçé	jabber.uci.cu/2cb78369	1.574226e+12	bpa
25	107915947	€äxçßçþ¬	jabber.uci.cu/4db432fe	¬é¬¶¬€ç€	jabber.uci.cu/df36ffa9	1.574226e+12	pera deja pedirle el ip al choco que yo stoy w...
26	107915947	¬é¬¶¬€ç€	jabber.uci.cu/df36ffa9	€äxçßçþ¬	jabber.uci.cu/4db432fe	1.574226e+12	ok
27	107915948	¬&x¬þçé	jabber.uci.cu/2cb78369	¬&x¬þçé	jabber.uci.cu/2cb78369	1.574226e+12	9204 0699 9327 0321 bandec
28	107915946	ü<x¬þé¥ááæ	jabber.uci.cu/8c4e42e5	€äxçßçþ¬	jabber.uci.cu/4db432fe	1.574227e+12	me debes un cerbero
29	107915946	€äxçßçþ¬	jabber.uci.cu/4db432fe	ü<x¬þé¥ááæ	jabber.uci.cu/8c4e42e5	1.574227e+12	jaajaj ok te sirve uno negro?

Figura 2.4: Registro de mensajes cargados en objeto DataFrame de Pandas.

2.2.2. Preprocesamiento de datos

El preprocesamiento abarca todas aquellas técnicas de análisis de datos que permiten mejorar la calidad de un conjunto de datos de modo que las técnicas de extracción de conocimiento puedan obtener mayor y mejor información. Estos tratamientos conllevan a analizar las palabras del lenguaje y conseguir obtener la información relevante y filtrar la que no tiene significado semántico Andrade Alvarado et al. (2015). Dentro del preprocesado de texto, se procede a la eliminación de palabras vacías, lematización, normalización y segmentación del texto. Con estos pasos cada documento de la clase seleccionada se convierte a una representación compacta adecuada para el algoritmo de aprendizaje no supervisado, LDA.

2.2.2.1. Eliminación de ruido, atributos innecesarios y valores atípicos

Inicialmente se descartan las columnas (atributos) innecesarias o irrelevantes para la solución propuesta, se manejan los valores nulos y se elimina el ruido o valores atípicos. Para ello se crea un nuevo DataFrame de Pandas en donde se insertarán los datos luego de ser tratados debidamente.

Utilizando el juicio de expertos se llegó a la conclusión de que columnas como Cemisor y Creceptor eran irrelevantes para la propuesta de solución. También se determinó que la presencia de mensajes con un cambio de línea en su atributo Contenido generaba un nuevo registro con valores nulos y ruido en el conjunto de datos de estudio por lo que debieron de ser tratados.

Luego de descartar atributos innecesarios y tratar correctamente los valores atípicos y el ruido en el set de datos, se procede a la limpieza del texto que servirá de entrada al algoritmo (corpus).

2.2.2.2. Normalización

El proceso de normalización es muy importante en el preprocesamiento de un texto, ya que impide que el programa tome como palabras distintas las palabras que son sintácticamente iguales Andrade Alvarado et al. (2015). Para el caso de este trabajo se pasó todo el texto a minúsculas, se eliminaron todos los acentos y diéresis presentes en los mensajes (documentos). Además, se tuvo en cuenta que por lo general las conversaciones que se llevan a cabo en este sistema de mensajería son, en su mayoría, informales por lo que se decidió eliminar caracteres repetidos consecutivamente y patrones repetidos dentro de una misma cadena.

```
1  import unicodedata
2  import itertools
3  def accent_fold(s):
4      return ''.join(c for c in unicodedata.normalize('NFD', s)
5                     if unicodedata.category(c) != 'Mn')
6  def deleteConsecutiveChars(text):
7      words=[]
8      for word in text:
9          word = ''.join(ch for ch, _ in itertools.groupby(word))
10         word = re.sub(r'(.+?)\1+', r'\1', word)
11         words.append(word)
12     return words
```

Figura 2.5: Funciones utilizadas para normalizar el texto.

2.2.2.3. Segmentación

La segmentación es el proceso de separar el texto en unidades, denominadas *tokens*. En este caso se tokenizaron las palabras que componen cada uno de los contenidos de los mensajes del registro, haciendo uso de la librería de Python spaCy, una librería de código abierto para el procesamiento avanzado de lenguaje natural.

Utilizando también el juicio de expertos se determinó que los tokens que poseen una longitud menor a cuatro y los tokens cuya longitud excede a 20 caractéres no aportan información relevante al descubrimiento de tópicos por lo que fueron descartados.

```
1 from spacy.lang.es import Spanish
2 parser = Spanish()
3 def tokenize(text):
4     lda_tokens = []
5     tokens = parser(text)
6     for token in tokens:
7         if token.orth_.isspace():
8             continue
9         else:
10            lda_tokens.append(token.lower_)
11    return lda_tokens
```

Figura 2.6: Función utilizada para segmentar el texto.

2.2.2.4. Eliminación de palabras vacías

Las palabras vacías (*stopwords*) son palabras que son muy frecuentes y que por lo general no contienen información importante para la extracción de información, por ejemplo: pronombres, preposiciones, conjunciones, artículos, etc.

Estas palabras deben ser excluidas del texto porque se mejora la eficiencia y efectividad de los sistemas de minería de texto. Además de agregar ruido y aumentar el tamaño de la representación.

Empleando el Kit de Herramientas de Lenguaje Natural (NLTK por sus siglas en inglés *Natural Languaje Tool Kit*) de Python, se le dió tratamiento a estas palabras.

En adición al diccionario de palabras vacías que provee NLTK se elaboró un diccionario propio con ciertos términos que no son de interés para el negocio, por tanto, todo los términos que se encontraran en ese diccionario fueron descartados.

2.2.2.5. Lematización

Por lematización nos referimos al proceso de remover los sufijos para reducir una palabra a su lema o raíz.

```
import nltk
stopwords = set(nltk.corpus.stopwords.words('spanish'))
def isStop(token):
    if token not in stopwords:
        return token
```

Figura 2.7: Función utilizada para eliminar las palabras vacías.

Frecuentemente, una palabra no aparece exactamente en un documento, pero sí alguna variante gramatical de la misma como plural, gerundios, sufijos de tiempo verbal, etc. Este problema puede resolverse con la sustitución de las palabras por su raíz.

Haciendo uso de la librería Wordnet que provee NLTK fue posible lematizar todos los tokens.

```
from nltk.corpus import wordnet as wn
from nltk.stem.wordnet import WordNetLemmatizer
def get_lemma(word):
    lemma = wn.morphy(word)
    if lemma is None:
        return word
    else:
        return lemma

def get_lemma2(word):
    return WordNetLemmatizer().lemmatize(word)
```

Figura 2.8: Función utilizada para lematizar los tokens.

Terminado el procesamiento de los datos, el texto está limpio y listo para ser empleado como entrada al algoritmo.

2.3. Aplicación del modelo LDA

Se decidió aplicar la implementación del algoritmo LDA que posee la biblioteca Gensim de Python, diseñada para el modelado de temas no supervisado, el procesamiento de lenguaje natural y el manejo de grandes colecciones de texto.

Las dos entradas principales de dicho algoritmo son el diccionario y el corpus.

```
import gensim.corpora as corpora
dictionary = corpora.Dictionary(preprocessed_data)
corpus = [dictionary.doc2bow(text) for text in preprocessed_data]
```

Figura 2.9: Creación del corpus y diccionario.

El diccionario está compuesto por todas las palabras que se tendrán en cuenta para tratar nuevos textos en la ejecución del modelo. No es más que un arreglo asociativo de palabras del texto, sin repeticiones, y una clave única o Id. El propio Gensim crea dicho identificador para cada palabra en el documento.

<class 'dict'>: 'manana': 0, 'muere': 1, 'copiame': 2, 'pesao': 3, 'copiar': 4, ínstagram': 5, 'pincha': 6, '7382': 7, '9204': 8, énvio': 9, 'choco': 10, 'pedirle': 11, 'stoy': 12, 'wifi': 13, '0321': 14, '9327': 15, 'bandec': 16, 'cerbero': 17, ñegro': 18, 'sirve': 19, 'color': 20, éncontre': 21, éliminar': 22, 'perfiles': 23, 'repent': 24, 'solita': 25, éntro': 26, 'sigo': 27, 'sigueme': 28, 'camila': 29, 'casa': 30, 'piro': 31, 'pasa': 32, 'pena': 33, 'problema': 34, 'repito': 35, 'sientas': 36, 'vida': 37, 'play': 38, 'chama': 39, 'maja': 40, 'bola': 41, 'coreo': 42, 'funciona': 43, 'mandarte': 44, ñovia': 45, 'quererme': 46, 'locuras': 47, 'feliz': 48, 'leagste': 49, 'vivir': 50, 'mala': 51, énsenar': 52, íbas': 53, 'pense': 54, 'responder': 55, 'conclusion': 56, 'simple': 57, 'pensabas': 58, 'rechaza': 59, 'malo': 60, 'pensamos': 61, 'trist': 62, 'corecta': 63, 'rechazan': 64, 'tomar': 65, 'ref': 66, 'preocupes': 67, 'hice': 68, 'papa': 69, 'prueban': 70, 'relajado': 71, 'face': 72, 'salir': 73, 'tube': 74, 'psiphon': 75, 'cuckold': 76, '192.168.173.50': 77, 'mandar': 78, 'lega': 79, 'claudia': 80, 'yesik': 81, énviast': 82, 'mija': 83, 'cuidt': 84, 'monte': 85, 'mete': 86, úiero': 87, éganas': 88, 'hora': 89, 'legar': 90, 'psue': 91, 'ropa': 92, 'tenia': 93, 'poquito': 94, 'tenerte': 95, 'tufaste': 96, 'wuenos': 97, 'quiero': 98, 'conjunto': 99...

Figura 2.10: Visualización del diccionario creado.

El corpus en Gensim es representado como una lista de relaciones $(Id, Frec)$, donde Id es el identificador de cada palabra y $Frec$ es la frecuencia de aparición de dicha palabra en su documento.

<class 'list'>: [[(0, 1)], [(1, 1)], [(2, 1)], [(3, 1)], [(4, 1)], [(5, 1)], [(6, 1)], [(7, 1), (8, 1)], [(9, 1)], [(10, 1), (11, 1), (12, 1), (13, 1)], [(8, 1), (14, 1), (15, 1), (16, 1)], [(17, 1)], [(18, 1), (19, 1)], [(20, 1)], [(21, 1)], [(22, 1), (23, 1), (24, 1)], [(25, 1)], [(26, 1), (27, 1), (28, 1)], [(29, 1), (30, 1)], [(31, 1)], [(32, 1), (33, 1), (34, 1), (35, 1), (36, 1), (37, 1)], [(38, 1)], [(39, 1)], [(40, 1)], [(41, 1)], [(42, 1), (43, 1), (44, 1)], [(30, 1), (45, 1)], [(46, 1)], [(47, 1)], [(48, 1)], [(49, 1), (50, 1)], [(51, 1)], [(52, 1), (53, 1)], [(54, 1), (55, 1)], [(56, 1)], [(57, 1)], [(58, 1), (59, 1)], [(32, 1), (60, 1), (61, 1), (62, 1)], [(63, 1), (64, 1), (65, 1)], [(66, 1)], [(67, 1)], [(68, 1)], [(69, 2)], [(70, 1)], [(71, 1)], [(72, 1)], [(73, 1), (74, 1)], [(75, 1)], [(76, 1)], [(40, 1)], [(77, 1)], [(78, 1)], [(13, 1)], [(79, 1)], [(80, 1), (81, 1)], [(82, 1), (83, 1)], [(84, 1)], [(85, 1)], [(86, 1)], [(87, 1)], [(88, 1), (89, 1), (90, 1), (91, 1), (92, 1), (93, 1)], [(94, 1), (95, 1)], [(96, 1)], [(38, 1), (97, 1)], [(98, 1)], [(99, 1), (100, 1), (101, 1), (102, 1), (103, 1), (104, 1), (105, 1)], [(40, 1), (106, 1)], [(107, 1)], [(6, 1), (108, 1), (109, 1)], [(110, 1)], [(111, 1)], [(112, 1)], [(113, 1)], [(78, 1), (114, 1), (115, 1), (116, 1)], [(117, 1), (118, 1)], [(119, 1)], [(75, 1), (120, 1), (121, 1), (122, 1)], [(123, 1)], [(124, 1)], [(123, 1), (125, 1)], [(121, 1)], [(126, 1)], [(44, 1), (127, 1)], [(0, 1), (69, 1), (128, 1)], [(129, 1), (130, 1)], [(131, 1)], [(132, 1)], [(133, 1)], [(134, 1)], [(135, 1)], [(0, 1), (136, 1)], [(98, 1)], [(137, 1), (138, 1)], [(139, 1)], [(140, 1), (141, 1)], [(142, 1)], [(143, 1), (144, 1), (145, 1)], [(146, 1), (147, 1)], [(148, 1)], [(149, 1...

Figura 2.11: Visualización del corpus.

Por ejemplo, en la figura 2.11, (0, 1) implica que la palabra cuyo identificador es cero aparece una vez en el primer documento. Del mismo modo, la palabra de identificador uno aparece también una única vez y así sucesivamente.

El método que implementa el modelo LDA de la librería Gensim necesita definir, además del

diccionario y el corpus, ciertos parámetros que determinan cómo entrenar y generar el modelo de tópicos, algunos de estos parámetros son:

- iterations: número máximo de iteraciones a través del corpus al inferir la distribución temática de un corpus.
- numTopics: número de temas latentes solicitados que se extraerán del corpus de entrenamiento.
- alpha y beta: hiperparámetros que afectan la densidad de tópicos. El valor por defecto de ambos es 1/numTopics.
- chuncksize: controla cuántos documentos se procesan a la vez en el algoritmo de entrenamiento. Aumentar el tamaño del fragmento acelerará el entrenamiento.
- updateEvery: indica cada cuanto se actualizan los parámetros del modelo.
- passes: controla la frecuencia con la que entrenamos el modelo en todo el corpus. El valor por defecto es 10.

LDA es un algoritmo iterativo, un mayor número de iteraciones mejora la convergencia del modelo. Para la elección de este parámetro se atiende a una relación de compromiso entre el coste computacional, que aumenta con el número de iteraciones, y la convergencia del modelo. El valor que se elije son 300 iteraciones.

Para definir los hiperparámetros alpha y beta se fijó el número de tópicos para analizar el efecto de cada uno de ellos en el modelo generado para el corpus evaluado. Se comienza con los valores por defecto para ir aumentando progresivamente los valores de alpha y beta.

El modelo LDA requiere que se le defina también la cantidad de tópicos (k) a buscar por lo cual es importante hacer un análisis de este parámetro, previo a ejecutar el modelo. Cuando se analizan grandes volúmenes de datos para detectar los tópicos a los que hacen referencia dichos datos es común que salga a flote la siguiente interrogante: ¿cuántos tópicos estamos intentado encontrar?

Para darle solución a esta interrogante es necesario modificar la composición del modelo en un esfuerzo por encontrar la mejor combinación de parámetros, en este caso el parámetro k, para manejar el problema.

Para optimizar el valor de k se decidió generar diferentes modelos LDA con valores de k distintos y luego de esos modelos escoger el modelo óptimo teniendo en cuenta el valor de coherencia de cada uno.

El caso de medir la coherencia de los tópicos se ha estudiado recientemente para remediar el problema de que los modelos de tópicos no garantizan la interpretabilidad de sus resultados Röder et al. (2015).

Medir la coherencia de tópicos es una buena forma de comparar diferentes modelos de tópicos en función de su interpretabilidad humana KUMAR (2017). El valor de coherencia c_v

captura el número óptimo de tópicos al otorgar a la interpretabilidad de estos una puntuación de coherencia. Para calcular dichos valores se utlizó el paquete CoherenceModel de la librería Gensim.
El procedimiento a seguir en el método propuesto puede ser observado en la figura 2.12.

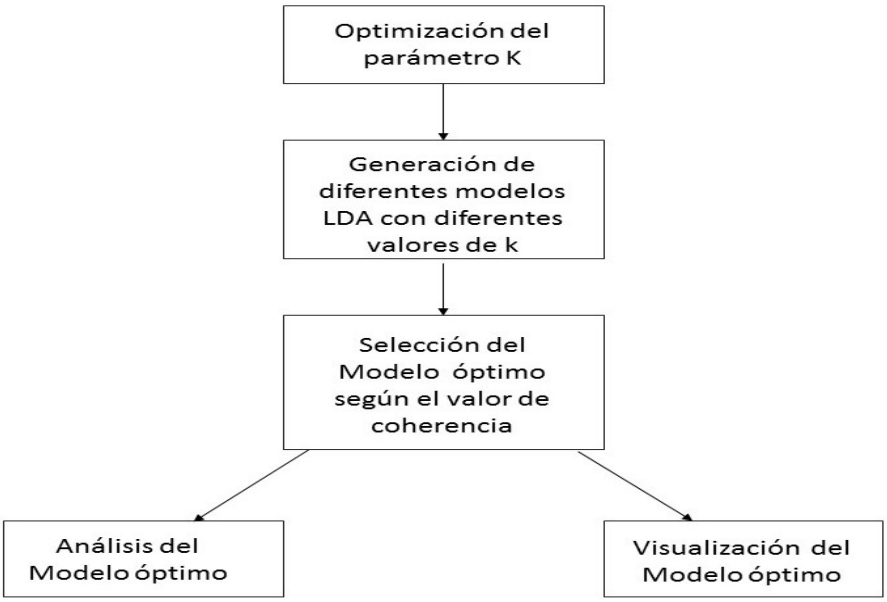

Figura 2.12: Optimización del parámetro k. Fuente: elaboración propia.

A continuación se muestra el código utilizado para calcular los valores de coherencia para cada uno de los modelos LDA generados.

```
import gensim
from gensim.models.coherencemodel import CoherenceModel
def compute_coherence_values(dictionary, corpus, texts, limit, start=2, step=1):
    coherence_values = []
    model_list = []
    for num_topics in range(start, limit, step):
        if __name__ == '__main__':
            freeze_support()
            model = gensim.models.ldamodel.LdaModel(corpus,
                                                    num_topics=num_topics,
                                                    id2word=dictionary,
                                                    passes=value,
                                                    random_state=value)
            model_list.append(model)
            coherencemodel = CoherenceModel(model=model,
                                            texts=texts,
                                            dictionary=dictionary,
                                            coherence='c_v')
            coherence_values.append(coherencemodel.get_coherence())
    return model_list, coherence_values
```

Figura 2.13: Función utilizada para calcular los valores de coherencia.

Los objetos de esta clase CoherenceModel permiten construir y mantener un modelo de coherencia de tópicos. Los variables que recibe como parámetro son: *model, texts, dictionary* del modelo lda creado anteriormente y se especifica en *coherence* la medida de coherencia a utilizar c_v a la cual se le debe proporcionar el texto Kapadia (2019).

Una vez determinado el modelo óptimo dado el valor de coherencia obtenido se procede a generar el modelo LDA.

```
1 import gensim
2 def generateOptimalModel(optimal_numTopics):
3     lda_model = gensim.models.LdaMulticore(corpus=corpus,
4                                            id2word=id2word,
5                                            num_topics=optimal_numTopics,
6                                            random_state=value,
7                                            chunksize=value,
8                                            passes=value,
9                                            iterations=value,
10                                           alpha=value)
11     return lda_model
```

Figura 2.14: Generación del modelo óptimo LDA.

El modelo LDA utilizado en esta herramienta fue optimizado para los datos de prueba procesados, cuando el *dataset* cambie será importante optimizar los parámetros nuevamente para mantener la calidad de los resultados.

2.4. Detección e interpretación de tópicos

2.4.1. Visualización

El objetivo de presentar una visualización de tópicos, es representar los datos a la salida del modelo de forma interpretable. Como establecen Chaney y Blei (2012) los objetivos de la visualización deben ser:

- Resumir el corpus visualizando la composición de los tópicos.
- Revelar las relaciones entre los documentos evaluados y los tópicos descubiertos
- Revelar relaciones entre los documentos evaluados.

Estos objetivos establecen las bases de la extracción de información de una colección de documentos empleando un modelo de tópicos.

En la literatura que versa sobre visualización de tópicos se suele hacer especial hincapié en el primero de los objetivos presentados. Es decir, las visualizaciones suelen centrarse en la de la composición de los tópicos descubiertos. Este tipo de visualizaciones permiten comprender

el corpus de manera global, pero no prestan atención a los documentos a nivel individual. Por tanto, se desatienden las relaciones que se establecen entre documentos Vázquez Marcos (2017).

Sievert y Shirley (2014) proponen un método –LDAvis- para la visualización de la composición de los tópicos. En este trabajo se emplea este método de visualización, se introduce a continuación sus bases de funcionamiento de acuerdo a lo expuesto por Sievert y Shirley (2014).

La herramienta LDAvis permite una visualización interactiva que ofrece las siguientes posibilidades:

- Tópicos representados mediante círculos en el plano de dos dimensiones, donde el área de cada circulo representa la predominancia de cada tópico en el corpus. Los tópicos son ordenado en orden decreciente respecto a su predominancia.
- La posición de cada tópico se computa mediante la distancia entre tópicos, usando MDS para proyectar estas distancias al plano de dos dimensiones.
- Se representa mediante diagramas de barras las palabras más relevantes para el tópico seleccionado. Se superponen una barra roja y otra azul para cada término que representan la frecuencia del término de la palabra evaluado en todo el corpus, es decir, el número de veces que se repite esa palabra en el conjunto de documentos evaluados y la frecuencia del término estimada de la palabra en el tópico seleccionado respectivamente.

Para poder graficar fácilmente los tópicos obtenidos del modelo aplicado se utilizó la librería de Python pyLDAvis

```
1 import pyLDAvis.gensim
2 def show(optimal_model, corpus, dictionary):
3     lda_display = pyLDAvis.gensim.prepare(optimal_model, corpus, dictionary,
          sort_topics=True)
4     pyLDAvis.display(lda_display)
5     pyLDAvis.save_html(lda_display, 'lda.html')
6     pyLDAvis.show(lda_display)
```

Figura 2.15: Función utilizada para visualizar los tópicos obtenidos.

2.5. Conclusiones del capítulo

En este capítulo se propuso un método para la detección de tópicos en los mensajes emitidos en la red de mensajería instantánea de la UCI. Luego de presentada la propuesta, se concluye que:

- El uso del modelo conceptual permitió describir el contexto a partir de la identificación de los principales conceptos asociados a la investigación.
- El KDD se utiliza en la propuesta de solución para identificar los patrones válidos dentro de la fuente de datos y eliminar o corregir los datos incorrectos.
- Se logró detectar tópicos en la red de mensajería instantánea de la UCI utilizando el modelo LDA optimizado para el set de datos procesados.

CAPÍTULO 3

VALIDACIÓN Y ANÁLISIS DE RESULTADOS

En el presente capítulo se realiza la validación del método desarrollado mediante varios criterios para determinar que la solución propuesta responde satisfactoriamente a los objetivos que se trazaron. Se realizan varias pruebas al código para asegurar la calidad del método propuesto verificando que no existan fallas.

3.1. Optimización de hiperparámetros α y β

En la primera fase de la generación del modelo LDA, se estudia el efecto de los parámetros α y β sobre los resultados del modelo. Tras este proceso se decide establecer los siguientes valores: α=0.6 β=0.01

El proceso que condujo a la selección de estos valores se explica a continuación.

En primer lugar, con un número de 5 tópicos preestablecido, se estudió la salida del modelo LDA para el conjunto de evaluación con los valores por defecto de $\alpha(\alpha = 0{,}1)$ y $\beta(\beta = 0{,}01)$. Se observó que a la salida del modelo la mayoría de los documento eran asignados a los distintos tópicos con probabilidades superiores a 0.9. Esto quiere decir, que cada documento expresaba información de pertenencia a un único tópico.

Esta representación no era válida para las colecciones bajo estudio. Dada la heterogeneidad en cuanto a su contenido, una categorización de los documentos tan rígida no expresaba relaciones entre documentos. Como se explica en el segundo capítulo, el valor de α, influye en como se asigna la pertenencia de un documento a los distintos tópicos. A valores más grandes, se considera que un documento está representando por la combinación de un mayor número de tópicos. De esta manera, si una colección de documentos no está claramente categorizada se recomienda el uso de valores de α cercanos a 1.

Ante pruebas con valores de α del orden de 0.8 y 0.9 la salida del modelo para los dos conjuntos de evaluación presentaba documentos en los que casi ningún documento pertenecía a un tópico concreto con probabilidad mayor que 0.5. Esto implica, que los documentos quedaban representados como una combinación de tópicos con parecido valor de pertenencia para cada uno.

Lo que se pretende es tener una representación en un punto medio de los dos extremos antes expuestos. Es decir, tener documentos representados como una combinación de tópicos, pero en los que uno de los tópicos tenga una importancia significativa.

Sobre el parámetro β, se deja su valor por defecto. Cabe mencionar que se probó el efecto de este parámetro sobre la representación. Como se explica en el segundo capítulo, valores más grande de β, aumenta el número de palabras que definen un tópico. A priori se podría pensar que sería necesario aumentar el valor de este parámetro, pero aumentar este parámetro se traduce en que los tópicos estuvieran formados por conjuntos de palabras relevantes muy similares. Esto se observó en la composición de los tópicos mediante PyLDAvis. Por este motivo se selecciono su valor por defecto

3.2. Óptimización del número de tópicos (k)

Como se mostró en le capítulo anterior la solución es un método que utiliza el modelo LDA. Éste modelo impone definir de antemano la cantidad de tópicos a encontrar sobre el set de datos a analizar. Es por esto que el conocimiento del valor de dicho parámetro es de suma importancia conocer antes de entrenar el modelo y por lo tanto comenzaremos explicando cómo optimizar dicho parámetro. Posteriormente, utilizando el parámetro óptimo, se genera el modelo en cuestión observando los tópicos obtenidos y finalmente analizaremos dichos tópicos mediante visualizaciones.

La forma utilizada para obtener el número óptimo de tópicos es construir varios modelos LDA con diferente número de tópicos (k) y tomar aquel que devuelva el valor de coherencia (c_v)[1] más grande considerando el contexto. Dicho valor de coherencia es una forma simple de ver qué tan bueno es el modelo. Para la elección del valor k, el c_v no tiene que ser simplemente el más grande y es por eso que también es importante la cantidad de datos que uno está procesando. Si observamos las mismas palabras clave repetidas en muchos de los tópicos, es probablemente una señal de que el parámetro k es muy grande. El siguiente gráfico (fig 3.1) representa los diferentes valores de coherencia obtenidos para distintos k con los que se entrenó el modelo LDA. Además, hay líneas que conectan el valor medio de cada valor de coherencia para los valores de k Ponweiser (2012).

El gráfico se muestra a partir de los siguientes valores:

[1] https://rare-technologies.com/what-is-topic-coherence/

Num Topics = 3 has Coherence Value of 0.6655

Num Topics = 4 has Coherence Value of 0.6788

Num Topics = 5 has Coherence Value of 0.6790

Num Topics = 6 has Coherence Value of 0.6873

Num Topics = 7 has Coherence Value of 0.695

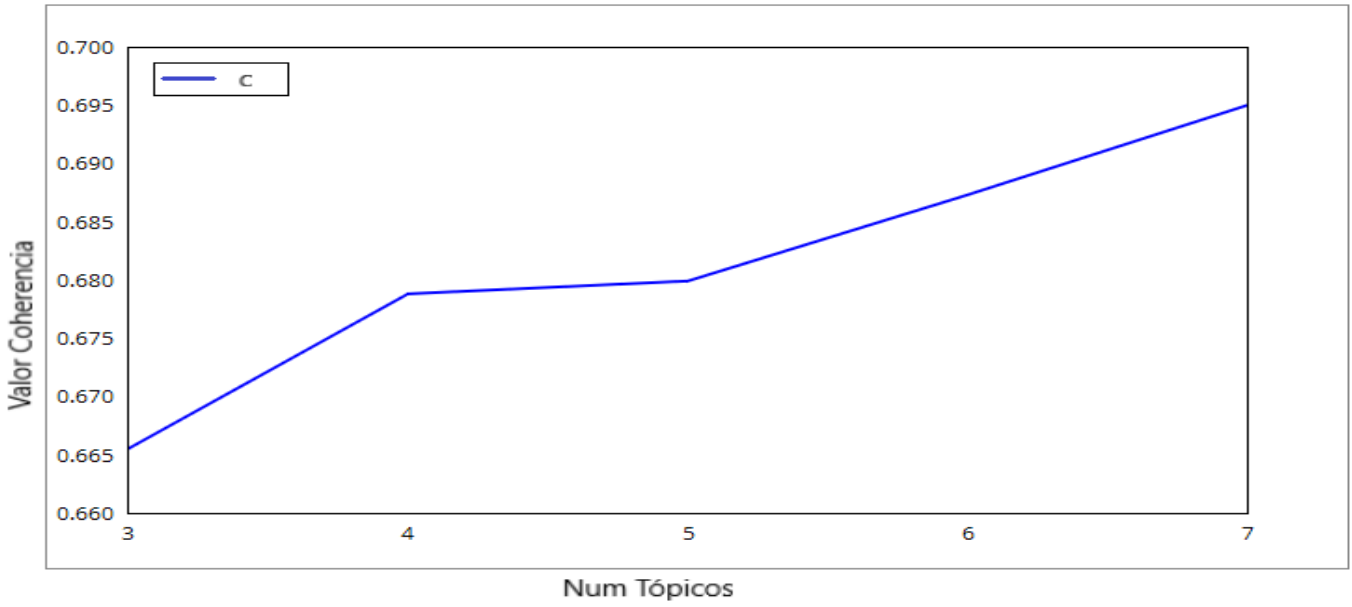

Figura 3.1: Valores de coherencia para diferentes números de tópicos.

Dados los hiperparámetros fijos y una variedad de temas K que van desde de 3 a 7 temas, vemos que un número de 4 tópicos representa lo mejor para el corpus.

Cuando el valor de coherencia sigue aumentando, lo mejor es elegir el que dé el máximo valor de coherencia antes de que la curva se empiece a hacer más llana. Para ilustrar a continuación mostramos algunos valores de tiempo que tomó generar cada uno de los modelos y el proceso total de cálculo del valor de coherencia:

— 76.1792426109314 seconds for model with k = 3 —

— 72.97278571128845 seconds for model with k = 4 —

— 68.66336512565613 seconds for model with k = 5 —

— 67.05760836601257 seconds for model with k = 6 —

— 66.08022141456604 seconds for model with k = 7 —

3.3. Observaciones

Como hemos mencionado previamente, el modelo LDA fue construido, donde cada tópico es una combinación de palabras claves y cada palabra clave contribuye con un cierto peso al tópico.

Podemos ver cada palabra clave de cada tópico y el peso o importancia de cada palabra clave como se muestra a continuación

```
[(0, '0.027*"mira" + 0.015*"paso" + 0.015*"coreo" + 0.012*"hora" + 0.010*"mando"'),
(1, '0.016*"casa" + 0.011*"tenia" + 0.011*"madre" + 0.010*"mande" + 0.006*"necesito"'),
(2, '0.039*"studio" + 0.015*"vale" + 0.011*"pasa" + 0.009*"tesis" + 0.009*"vida"'),
(3, '0.023*"cosa" + 0.015*"falta" + 0.013*"puedes" + 0.013*"quieres" + 0.012*"quiero"')]
```

Figura 3.2: Palabras claves por tópicos.

Los pesos reflejan qué importancia otorga cada palabra clave a ese tema.

Observando las palabras claves se debe poder resumir cuál es el tópico en cuestión al que pertenecen. En el caso del tópico cero podríamos decir que está asociado al tópico “Envío de correo” . De la misma forma, podemos ir determinando las palabras claves de los otros tópicos para definir a qué hace referencia cada uno.

3.4. Visualización

A continuación se presentan tres figuras que ilustran la apariencia que presenta la herramienta LDAvis.

Cada burbuja en la parte izquierda de la visualización representa un tópico. Cuanto más grande la burbuja, más predominante es ese tópico. Un buen modelo de tópicos es aquel que tiene burbujas bastante grandes y que no se solapan, dispersas en todo gráfico en lugar de estar todas juntas y clusterizadas en un único cuadrante.

Un modelo con muchos tópicos seguramente tendrá burbujas pequeñas, ubicadas en una misma región del gráfico y con muchos casos de solapamiento. Si movemos el cursor sobre cada una de las burbujas, las palabras y las barras del gráfico de la derecha se actualizarán. Estas palabras son las palabras clave más importantes del tópico seleccionado.

El diagrama de barras de la figura 3.3, en la que no se ha seleccionado ningún tópico, representa las 30 palabras con mayor prominencia. Por otra parte la figura 3.5 ilustra que si seleccionamos una palabra concreta, podemos ver gráficamente a que tópicos pertenece. El área de los círculos, pasa a indicar el grado de pertenencia de la palabra al tópico.

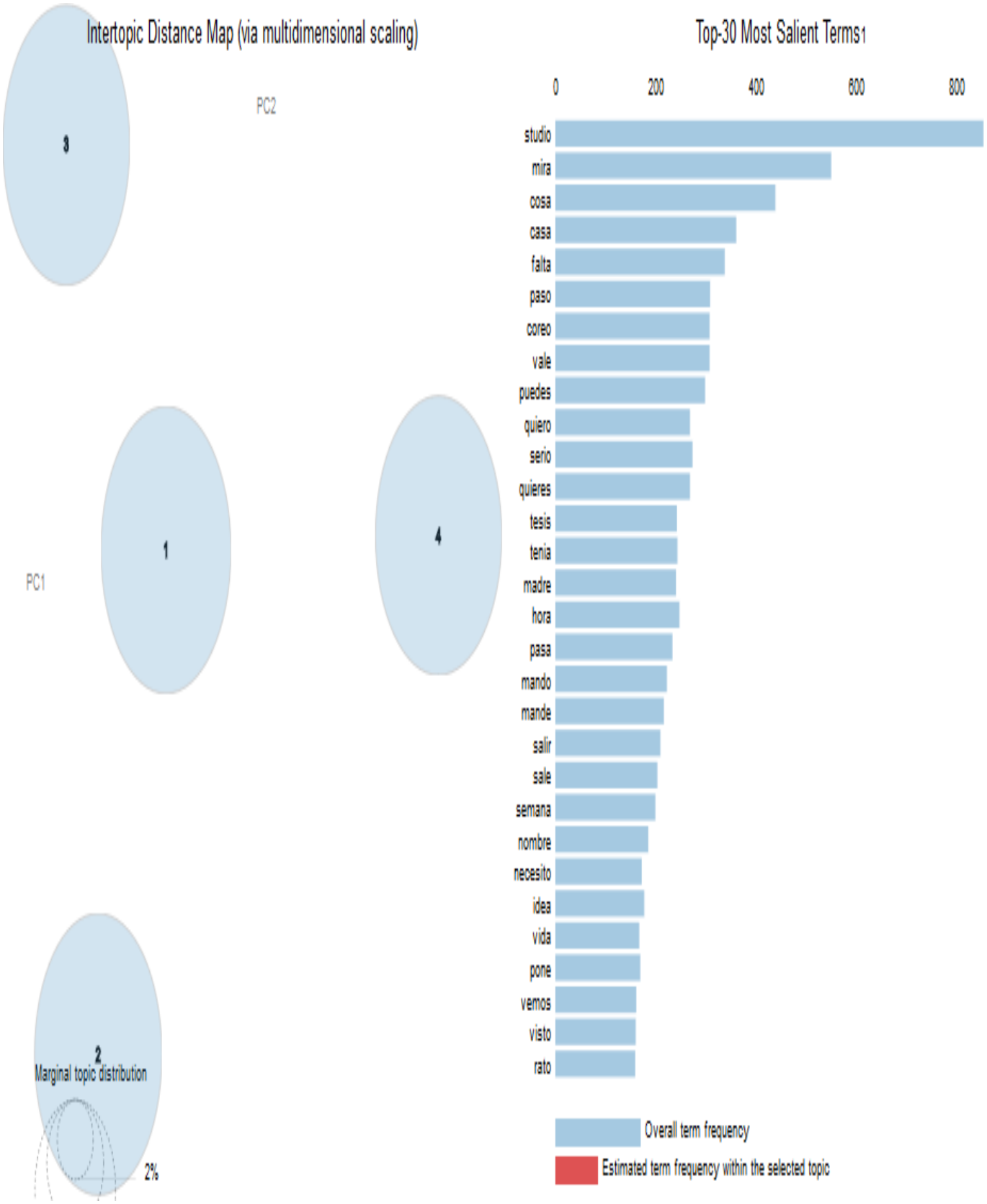

Figura 3.3: Visualización de los tópicos del modelo generado.

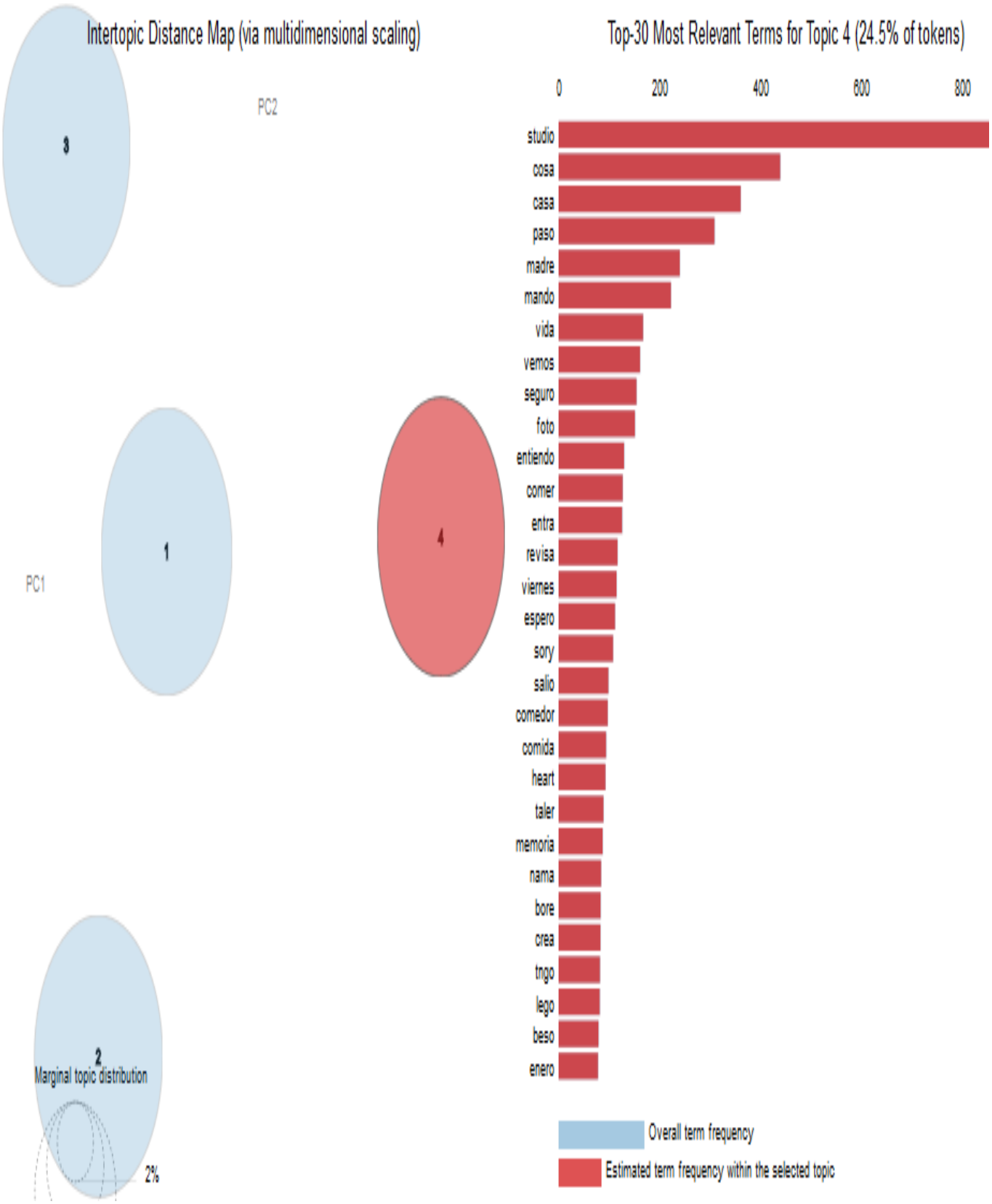

Figura 3.4: Visualización de la composición de un tópico.

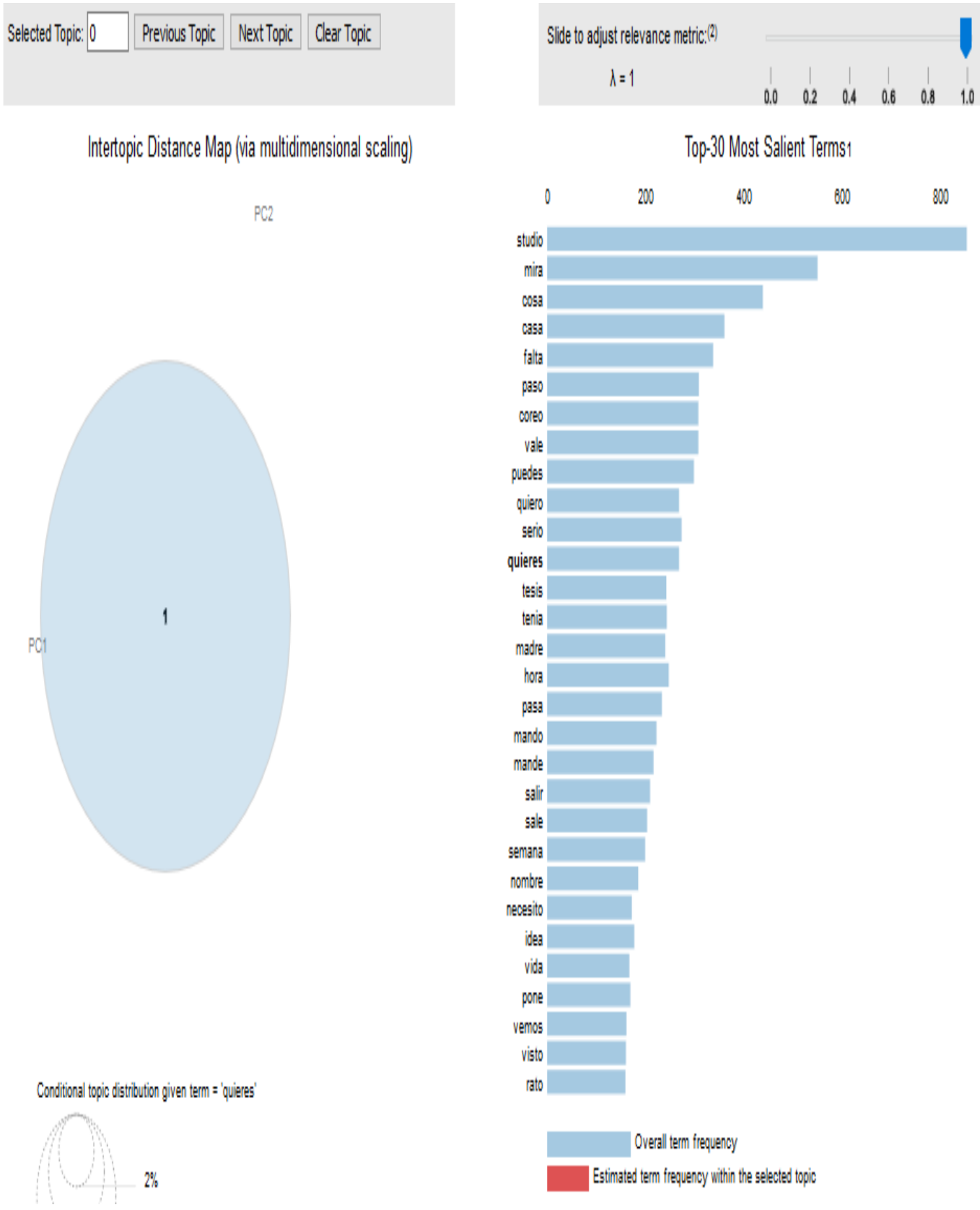

Figura 3.5: Pertenencia de una palabra al tópico.

Mientras que las barras azules representan al atributo *Silency*, una medida de cuánto un término dice acerca de un tópico; y las barras en rojas al atributo *Relevance*, un peso promedio de la probabilidad de una palabra dado un tópico y de esa palabra dado el tópico normalizado por la probabilidad del tópico.

Por último cabe mencionar, que como se observa en la parte superior izquierda, en la herramienta de visualización LDAvis hay un parámetro libre $\lambda \in (0, 1)$. Este parámetro influye a la hora de ordenar los términos más relevantes de cada tópico.

3.5. Conclusiones del capítulo

El proceso de validación a la solución propuesta se desarrolló sin inconvenientes. Luego de realizadas las pruebas de software y de corregirse las no conformidades que fueron detectadas, puede concluirse que el método se encuentra listo para su utilización. Las validaciones realizadas prueban la calidad de la solución desarrollada.

CONCLUSIONES

Al finalizar la presente investigación, arribamos a las siguientes conclusiones que dan solución al objetivo general de la misma:

- La detección de tópicos permitió determinar los temas de conversación, así como palabras claves relacionadas a cada tema dentro de las interacciones de los usuarios de la red de mensajería instantánea de la UCI.
- Se obtuvo un método implementado en Python que facilitó: el procesamiento de un gran volumen de documentos de manera rápida y simple, y el poder detectar tópicos o temas a los que hacen referencia los usuarios de una red social con visualizaciones que permiten analizar de manera sencilla los tópicos.
- Al utilizar el modelo LDA se puede observar que la determinación de palabras claves dentro de un texto es fundamental para el buen desempeño del modelo LDA, donde el pre procesamiento del texto juega un rol fundamental en la segmentación, normalización y eliminación del ruido.
- La validación realizada a los parámetros α, β, y k del modelo LDA, permitieron obtener un modelo óptimo para el set de datos procesados.

RECOMENDACIONES

Una vez cumplido el objetivo de la presente investigación se recomienda:

Optimizar los parámetros nuevamente del modelo LDA cuando el dataset cambie, ya que fueron optimizados para el set de datos de prueba procesados.

Implementar una variante por autores, que posibilite tener una coherencia en función de quienes escriben sobre los temas identificados.

Implementar generación automáticas de etiquetas o nombres para los tópicos. Los que ahora se llaman topic1, topic2, pero realmente son por ejemplo el topic1 es sobre pruebas y el topic2 es sobre la recreación. La generación automática permite tener una idea ya de que va el tema.

Por otro lado el modelo LDA, que si bien es el modelo más utilizado para la detección de tópicos, no es el único existente para resolver el problema. Sería interesante, agregar al método la posibilidad de entrenar un mismo set de datos con diversos modelos (como "*correlated topic model*" o "*biterm topic model*") y poder comparar los resultados obtenidos.

Bibliografía

10 interesantes herramientas para monitorizar temas actuales. 2018. URL https://coobis.com/es/cooblog/10-herramientas-para-monitorizar-temas-actuales. [Online; accessed 3. Mar. 2020].

About Us – TrendSpottr. 2020a. URL http://trendspottr.com/about. [Online; accessed 3. Mar. 2020].

Acerca de nosotros: Brandwatch. 2020b. URL https://www.brandwatch.com/es/about. [Online; accessed 3. Mar. 2020].

Charu C Aggarwal. *Data mining: the textbook*. Springer, 2015.

Charu C Aggarwal y ChengXiang Zhai. *Mining Text Data*. Springer Science & Business Media, 2012. ISBN 978-1-4614-3222-7.

G Alsina. Mensajería instantánea (ip). 2020.

Mario Amores, Leticia Arco, y Michel Artiles. PosNeg opinion: Una herramienta para gestionar comentarios de la Web. *Revista Cubana de Ciencias Informáticas*, 9(1):20–12, 2015. ISSN 2227-1899. URL http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992015000100003.

Vanessa del Rosario Andrade Alvarado et al. *Clasificación de artículos científicos*. Tesis Doctoral, Universidad de Concepción. Facultad de Ingeniería. Departamento de . . . , 2015.

C.D. Athuraliya, M.K.H. Gunasekara, Srinath Perera, y Sriskandarajah Suhothayan. Real-time natural language processing for crowdsourced road traffic alerts. En *2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer)*, págs. 58–62. 2015. ISBN 978-1-4673-9440-6. doi:10.1109/ICTER.2015.7377667. URL https://ieeexplore.ieee.org/document/7377667/.

J Maxwell Atkinson, John Heritage, y Keith Oatley. *Structures of social action*. Cambridge University Press, 1984.

Ana Isabel Rojão Lourenço Azevedo y Manuel Filipe Santos. Kdd, semma and crisp-dm: a parallel overview. *IADS-DM*, 2008.

David M Blei y John D Lafferty. Topic models. En *Text mining*, págs. 101–124. Chapman and Hall/CRC, 2009.

David M Blei, Andrew Y Ng, y Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Ricardo Eíto Brun y Jose A Senso. Artículo minería textual. *El profesional de la información*, 13(1):11, 2004.

Allison June-Barlow Chaney y David M Blei. Visualizing topic models. En *Sixth international AAAI conference on weblogs and social media*. 2012.

Paul Drew. John heritage, eds. 1992. talk at work: Interaction in institutional settings. *Cambridge, UK: Cam-bridge University Press. DrewTalk at Work: Interaction in Institutional Settings1992*.

Usama Fayyad, Gregory Piatetsky-Shapiro, y Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37–37, 1996a.

Usama Fayyad, Gregory Piatetsky-Shapiro, y Padhraic Smyth. The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34, 1996b.

Gabriel Fonseca Torres, Felix; Apolinaire Aguila. *Método para la detección de información relevante de los comentarios de usuarios de aplicaciones de software en la UCI.* Tesis Doctoral, Universidad de las Ciencias Informáticas, 2017.

Antonio Fumero, Genís Roca i Verard, Fernando Sáez Vacas, y José M Cerezo. *Web 2.0*. Pozuelo de Alarcón (Madrid): Fundación Orange España, 2010.

Christiane Kleinübing Godoi, Ana Lúcia de Araújo Lima Coelho, y Araceli Serrano. Elementos epistemológicos e metodológicos da análise sociológica do discurso: abrindo possibilidades para os estudos organizacionais. *Organizações & Sociedade*, 21(70):509–535, 2014.

Luciano Hammoe. *Detección de tópicos: utilizando el modelo LDA*. B.S. thesis, INSTITUTO TECNOLÓGICO DE BUENOS AIRES – ITBA, 2018.

V Jelisavčić, Bojan Furlan, Jelica Protić, y Veljko Milutinović. Topic models and advanced algorithms for profiling of knowledge in scientific papers. En *2012 Proceedings of the 35th International Convention MIPRO*, págs. 1030–1035. IEEE, 2012.

Shashank Kapadia. Evaluate Topic Models: Latent Dirichlet Allocation (LDA). *Medium*, 2019. ISSN 7574-8450. URL https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5

Rohit KUMAR. Natural language processing. En *Machine Learning and Cognition in Enterprises: Business Intelligence Transformed*, págs. 65–73. Apress, Berkeley, CA, 2017. ISBN 978-1-4842-3069-5. doi:10.1007/978-1-4842-3069-5_5. URL https://doi.org/10.1007/978-1-4842-3069-5_5.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Wan Ng. *NEW DIGITAL TECHNOLOGY IN EDUCATION.* Springer, 2016.

Martin Ponweiser. *Latent Dirichlet allocation in R*. Tesis Doctoral, WU Vienna University of Economics and Business, 2012.

Michael Röder, Andreas Both, y Alexander Hinneburg. Exploring the space of topic coherence measures. En *Proceedings of the eighth ACM international conference on Web search and data mining*, págs. 399–408. 2015.

Harvey Sacks. On the analyzability of stories by children. *Directions in sociolinguistics*, págs. 325–345, 1972.

Emanuel A Schegloff, Gail Jefferson, y Harvey Sacks. The preference for self-correction in the organization of repair in conversation. *Language*, 53(2):361–382, 1977.

Carson Sievert y Kenneth Shirley. Ldavis: A method for visualizing and interpreting topics. En *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, págs. 63–70. 2014.

María Silvestre Gómez. *Implementación de Asignación Jerárquica Latente de Dirichlet para Modelado de Temas*. B.S. thesis, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, 2018.

Jorge Vázquez Marcos. *Modelado de Tópicos para perfilado de Blogs*. B.S. thesis, 2017.

David Vilares Calvo. *Análisis de contenidos en Twitter: clasificación de mensajes e identificación dela tendencia política de los usuarios*. B.S. thesis, Universidad de Coruña, 2014.