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

MODELAMIENTO Y SEGUIMIENTO DE TÓPICOS PARA DETECCIÓN DE MODUS OPERANDI EN ROBO DE VEHÍCULOS

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN GESTIÓN DE OPERACIONES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

DIEGO GARRIDO

PROFESOR GUÍA:
RICHARD WEBER

MIEMBROS DE LA COMISIÓN:
PROFESOR 2
PROFESOR 3

Este trabajo ha sido parcialmente financiado por:
NOMBRE INSTITUCIÓN

SANTIAGO, CHILE
2020

Una frase de dedicatoria,
pueden ser dos líneas.

Saludos

Agradecimientos

TODO

Tabla de Contenidos

Índice de Tablas

Índice de Ilustraciones

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS
DE LA INGENIERÍA
POR: **DIEGO GARRIDO**
FECHA: 2020
PROF. GUÍA: RICHARD WEBER

MODELAMIENTO Y SEGUIMIENTO DE TÓPICOS PARA DETECCIÓN DE MODUS OPERANDI EN ROBO DE VEHÍCULOS

En este trabajo se describe una metodología para el descubrimiento de tópicos en el tiempo. La metodología propuesta está basada en (i) discretización del corpus en épocas, (ii) descubrimiento de tópicos en cada época con Hierarchical Dirichlet Process (HDP), (iii) la construcción de un grafo de similitud entre tópicos de épocas adyacentes el cual permite modelar cambios entre los tópicos como: nacimiento, muerte, evolución, división y fusión. En contraste a trabajos anteriores, la metodología propuesta utiliza Word Mover's Distance (WMD) como medida de similitud entre tópicos, permitiendo así una comparación más justa entre tópicos que no poseen un vocabulario común mediante sus word embeddings. Se reportan resultados experimentales tanto cuantitativos como cualitativos en el fenómeno de robo de vehículos en Chile, usando como corpus los relatos de víctimas de robo de vehículo entre los años 2011-2016 provistos por la Asociación de Aseguradores de Chile (AACH). El algoritmo propuesto logra capturar bien los tópicos latentes del corpus, descubriendo delitos tales como "portonazo", apropiación indebida y robo con violencia.

Todo list

Capítulo 1

Motivación

Grandes volumenes de datos digitales son alamacenados día a día, en forma de noticias, blogs, páginas web, artículos científicos, libros, imágenes, sonido, video, redes sociales, etc. Volviéndose cada vez más difícil encontrar y descubrir lo que estamos buscando. Necesitamos herramientas computacionales que ayuden a organizar, buscar y entender grandes colecciones de datos.

Si pudieramos buscar y explorar documentos en base a sus temas, podríamos enfocarnos en temas específicos o más amplios, podríamos observar como estos temas cambian en el tiempo o como se relacionan unos a otros. En vez de buscar documentos a través de únicamente palabras claves, podríamos primero hallar temas que son de nuestro interés, y luego examinar los documentos relacionados a ese tema. Podríamos por ejemplo, descubrir nuevas tendencias de investigación, analizar la evolución de la contigencia social, estudiar la efectividad de campañas publicitarias en base a la opinión de los consumidores, organización y recomendación de contenido, etc.

1.1. Metodología Propuesta

Los modelos de tópicos probabilísticos nos ayudan a descubrir los temas latentes (*clusters*) en una colección de documentos, como estos temas están conectados unos a otros y cómo cambian en el tiempo. Permiten resumir un gran colección de documentos a través de sus temas y organizarlos entorno a estos.

Los modelos probabilísticos tratan un tópico como una distribución de probabilidad discreta sobre el vocabulario del corpus, siendo un práctica habitual interpretar un tópico a partir de sus N palabras más probables. Por ejemplo, para $N = 5$ las palabras más probables de un tópico son: “llaves”, “domicilio”, “individuos”, “casa” y “porton”, por lo que una etiqueta valida para este tópico podría ser “portonazo”.

En este trabajo se propone una metodología para el descubrimiento de tópicos en el tiempo. Esta metodología consisten en la discretización del corpus en épocas, el descubrimiento de tópicos en cada época mediante Hierarchical Dirichlet Process (HDP), la construcción de un grafo de similitud entre tópicos de épocas adyacentes el cual permite modelar cambios entre

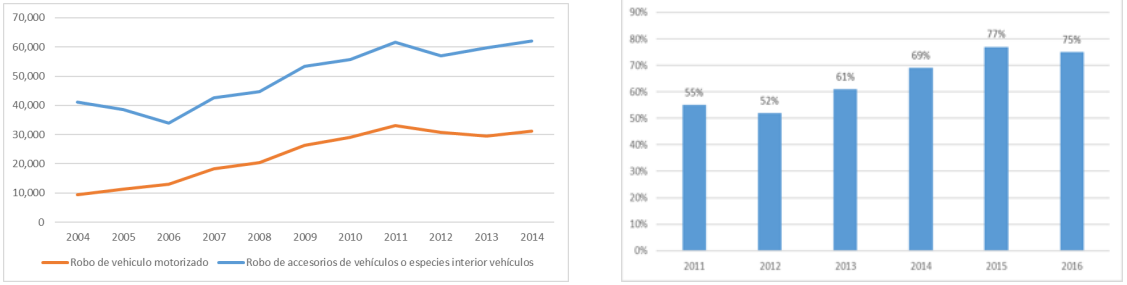
los tópicos como: nacimiento, muerte, evolución, división y fusión. En contraste a trabajos anteriores, la metodología propuesta utiliza Word Mover's Distance (WMD) como medida de similitud entre tópicos, permitiendo así una comparación más justa entre tópicos que no poseen un vocabulario común mediante sus word embeddings.

1.2. Caso de estudio

En este trabajo se aplica la metodología propuesta al fenómeno de robo de vehículos en Chile, usando como corpus una colección de 49.015 relatos de víctimas de robo de vehículo, entre los años 2011-2016, provistos por la Asociación de Aseguradores de Chile (AACH). Cabe destacar que se estima que un tercio del parque automotriz se encuentra asegurado, por lo que se trabaja con una muestra del parque automotriz.

En el contexto de robo de vehículos, los tópicos vendrían siendo los "modu operandi" que utilizan los delicuentes para robar un vehículo. Así, la metodología propuesta permitiría descubrir los *modus operandi* ocultos en los relatos de las víctimas y caracterizarlos a partir de las palabras, como también ver su evolución a través del tiempo, siendo capaz de detectar cuando nacen y mueren, y como cambian en el tiempo.

El robo de vehículos o accesorios de vehículos es un problema que afecta a toda la sociedad en Chile y en el mundo. Este problema se ha vuelto más relevante el último tiempo debido al crecimiento en el robo de vehículo motorizado y de los robos con violencia (ver Figura 1.1).



(a) (b)

Figura 1.1: (a) Cantidad de robos de vehículos y robos de accesorios de vehículos anuales en Chile (2004-2014). Fuente: Informe anual Carabineros, 2004-2014, INE. (b) Tasa de robos con violencia del total de robo de autos de lujo 2011-2016.

Este fenómeno trae consigo un montón de costos para la sociedad, como incremento en la percepción de la seguridad, aumentos en la prima de los seguros de los asegurados, aumento

en los costos de las aseguradoras [1] y el incremento de otros tipos de delitos [2]

1.3. Revisión del estado del arte

El problema planteado consiste en un problema de *clustering*, puesto que no se cuenta con una etiqueta del tema al que corresponde cada documento, siendo el propósito del trabajo descubrirla. Dentro de los métodos de *clustering* que involucran texto el modelamiento de tópicos es el enfoque más prometedor.

El modelamiento de tópicos es una poderosa herramienta que nos permite analizar grandes colciones de documentos. En términos más específicos, los modelos de tópicos corresponden a modelos de *clustering* probabilísticos sobre grupos de datos, donde cada grupo es una colección de datos. En procesamiento de texto, cada grupo es un documento, cada documento es una colección de palabras y el conjunto de documentos lleva por nombre corpus. Estos modelos encuentran los temas (*clusters*) ocultos presentes en el corpus, permitiendo resumir, organizar y explirar grandes colecciones de datos.

Algunas de las técnicas de modelamiento de tópicos están basadas en factorización matricial como LSI (Latent Semantic Indexing) (Dumais, 2004) o NMF (Non-negative Matrix Factorization)(Xu et al., 2003), pero en este trabajo está basado en modelos probabilísticos generativos, como LDA (Latent Dirichlet Allocation)(Blei et al., 2003) o HDP (Hierarchical Dirichlet Process)(Teh et al., 2005). Ambos enfoques tienen sus pros y contras, en este trabajo se prefiere el enfoque probabilístico ya que es capaz de expresar incertidumbre en la asignación de un tópico a un documento y en la asignación de palabras a los tópicos, además, este enfoque suele aprender tópicos más descriptivos (Stevens et al., 2012).

El modelamiento de tópicos dinámico considera los siguientes dinamismos:

1. **Evolución de los tópicos**: la evolución de los tópicos se refleja en el cambio en la distribución sobre las palabras. Esto permite detectar cambios en cómo se comete un mismo tipo de delito, por ejemplo, el "portonazo" en un determinado momento se comete en grupos de 2-3 personas con arma blanca, luego evoluciona de arma blanca a arma de fuego y lo perpetran jóvenes menores de edad.

2. **Dinámismo en la mezcla de tópicos**: esto permite capturar la popularidad de los tópicos en el tiempo.

3. **Nacimiento, muerte, fusión y división de tópicos**: En el contexto de robos es natural que en el tiempo aparezcan nuevos *modus operandi* como también que desaparezcan aquellos que ya no parecen tan atractivos.

[1] Considerando que el costo promedio incurrido en un auto asegurado robado y no recuperado es de $ 5.000.000 de pesos, la pérdida total considerando solo los vehículos no recuperados para el año 2015 es de unos $15.720 millones de pesos.

[2] El destino de los vehículos robados es variado, se usan los autos para perpetrar otros delitos y huir, venderlos por piezas en talleres clandestinos o blanquear sus documentos para pasarlos por la frontera y venderlos o cambiarlos por droga en el extranjero.

En el modelamiento de tópicos estático destaca LDA y HDP. La diferencia principal en estos dos modelos es que el primero necesita de antemano fijar el número de tópicos a descubrir y el segundo lo infiere a partir del corpus.

Dynamic Topic Modelling (DTM) junto Topic Over Time (TOC)(Wang and McCallum, 2006) fueron los primeros modelos de tópicos desarrollados. Estos modelos mantienen el número de tópicos fijo en el tiempo, por lo que si aparece un nuevo tópico este quedará clasificado dentro de un tópico preexistente desde el comienzo, por lo que solo es capaz de capturar el punto 2 y 3.

En (Ahmed and Xing, 2012) se propone Dynamic Hierarchical Dirichlet Process (DHDP), el cual a diferencia de DTM y TOC no mantiene el número de tópicos fijo en el tiempo, sino que lo infiere a partir del corpus. Este modelo no es capaz de capturar fusión y división de tópicos. A diferencia de los otros modelos de tópicos mencionados anteriormente, DHDP no es una tecnología madura, de hecho hasta el idea de hoy no cuenta con una implementación pública disponible, por lo que se desconoce su desempeño en otras fuentes de información.

En (Wilson and Robinson, 2011) y (Beykikhoshk et al., 2018) se propone una metodología que permite capturar los dinámismos mencionados utilizando LDA y HDP respectivamente. Estas consisten en dividir el corpus en épocas, entrenar de forma independiente un modelo de tópico en cada época, para finalmente unir los resultados obtenidos. En este trabajo se utilizan técnicas de modelado dinámico de tópicos bajo este enfoque, usando como HDP para el descubirmiento de tópicos en cada época.

Describir organización del documento

Capítulo 2

Marco teórico

Añadir motivación

Describir organización del capítulo

2.1. Mixture Models

El supuesto básico en *clustering* es asumir que cada observación x_i pertenece a un solo *cluster* k. Podemos expresar la asignación a un *cluster* como una variable aleatoria z_i, donde $z_i = k$ significa que x pertenece al *cluster* k, esta variable no es observada en los datos y se considera una variable oculta. Podemos obtener la distribución que caracteriza a un solo *cluster* k condicionando en z_i

$$p(x_i|z_i = k, \phi) = p(x_i|\phi_k) \tag{2.1}$$
$$\tag{2.2}$$

Además, podemos definir la probabilidad de que una nueva observación pertenezca al *cluster* k

$$p(z_i = k|\pi) = \pi_k \tag{2.3}$$

$\sum_k \pi_k = 1$, ya que π_k son probabilidades de eventos mutuamente excluyentes. La distribución de x_i es entonces de la forma

$$p(x_i) = \sum_k \pi_k p(x_i|\phi_k) \tag{2.4}$$

Podemos escribir $p(x_i|\phi_k)$ como $x_i \sim F(\phi_{z_i})$, donde F es la distribución asociada a las observaciones.

Una representación equivalente para este modelo viende de considerar que el parámetro $\bar{\phi}_i$ usado para generar la observación x_i proviene de una distribucion discreta G, la cual tiene la forma

$$G(\phi) = \sum_k \pi_k \delta_{\phi_k}(\phi) \tag{2.5}$$

Así, G es una mezcla de funciones delta, donde la probabilidad que $\bar{\phi}_i$ es igual a ϕ_k es π_k. Luego, un *mixture model* podría representarse como sigue

$$\phi_{z_i} \sim G \tag{2.6}$$
$$x_i \sim F(\phi_{z_i}) \tag{2.7}$$

Un **Bayesian mixture model** es un *mixture model* con una medida aleatoria para las mezclas. En la sección 2.1.1. y 2.1.2 nos referimos a dos priors ampliamente usados para construir *bayesian mixture model*: la distribución Dirichlet que noes permite construir un **finite mixture model**, donde el número de átomos o *clusters* a descubrir es finito, denotado por K y un prior no paramétrico denominado Dirichlet Process (DP), el cual permite construir un **infinite mixture model**, donde el número de *clusters* no está acotado.

2.1.1. Distribución Dirichlet

La distribución Dirichlet (Minka, 2000) es una generalización multivariada de la distribución beta, la cual tiene soporte sobre un **simplex**, definido por:

$$S_K = \{x : 0 \leq x_k \leq 1, \sum_{k=1}^{K} x_k = 1\} \tag{2.8}$$

Luego, su función de densidad de probabilidad (pdf):

$$Dir(x|\alpha) = \frac{1}{B(\alpha)} \prod_{k=1}^{K} x_k^{\alpha_k - 1} \mathbb{I}(x \in S_K) \tag{2.9}$$

donde $B(\alpha) = B(\alpha_1, \ldots, \alpha_K)$ es la generalización de la función beta a K variables:

$$B(\alpha) \triangleq \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\alpha_0)} \tag{2.10}$$

donde $\alpha_0 \triangleq \sum_{k=1}^{K} \alpha_k$.

En la Figura 2.1 se observa el efecto de los parámetros en la distribución Dirichlet para $K = 3$. Cuando $\alpha_k = 1$ se tiene una distribución uniforme en el dominio S_K. El parámetro α_k controla la *sparsity*, mientras más se acerca a 0 los vectores generados tienen más átomos nulos y se concentra la masa en unas pocas coordenadas, mientras más grande α_k la masa más se concentra en el centro (1/3, 1/3, 1/3), por último, cuando α no es simétrico la masa se concentra proporcionalmente en las coordenadas con α_k mayor.

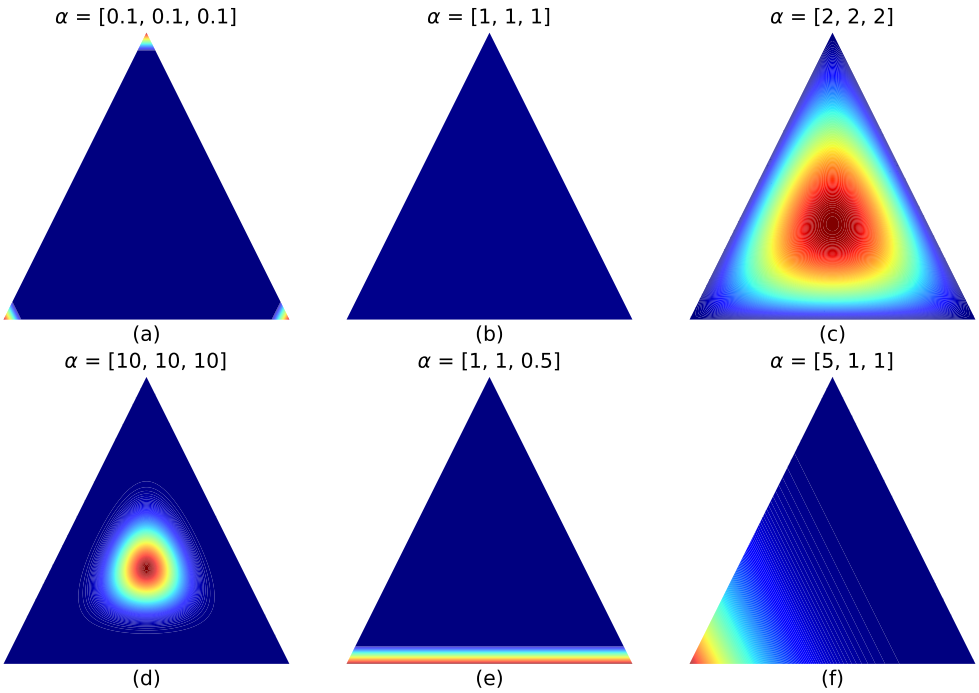


Figura 2.1: Densidad de la distribución Dirichlet para $K = 3$ define una distribución sobre el *simplex*, el cual puede ser representado por una superficie trinagular.

En general se asume simetría en los parámetros de la distribución Dirichlet de la forma $\alpha_k = \frac{\alpha}{K}$, de esta manera se tiene un α funciona como parámetro de concentración. En la Figura 2.2 se oberva una realización de una distribución Dirichlet para $\alpha \in \{0.1, 1, 10\}$ y $K \in \{2, 10, 100\}$, donde podemos observar que a mayor *alpha* las componenetes del vector x más similares se vuelven y esto es más notorio a mayor dimencionalidad debido a que hay más dimensiones donde distribuir la masa.

La distribución Dirichlet es comúnmente usada en estadística Bayesiana, ya que es el prior conjugado de la distribución categórica (multinoulli) y de la distribución multinomial. Así, la distribución Dirichlet puede ser utilizado como prior en un *finite mixture model* asumiendo que $\pi \sim Dir(\frac{\alpha}{K} 1_K)$ y $\phi_k \sim H$.

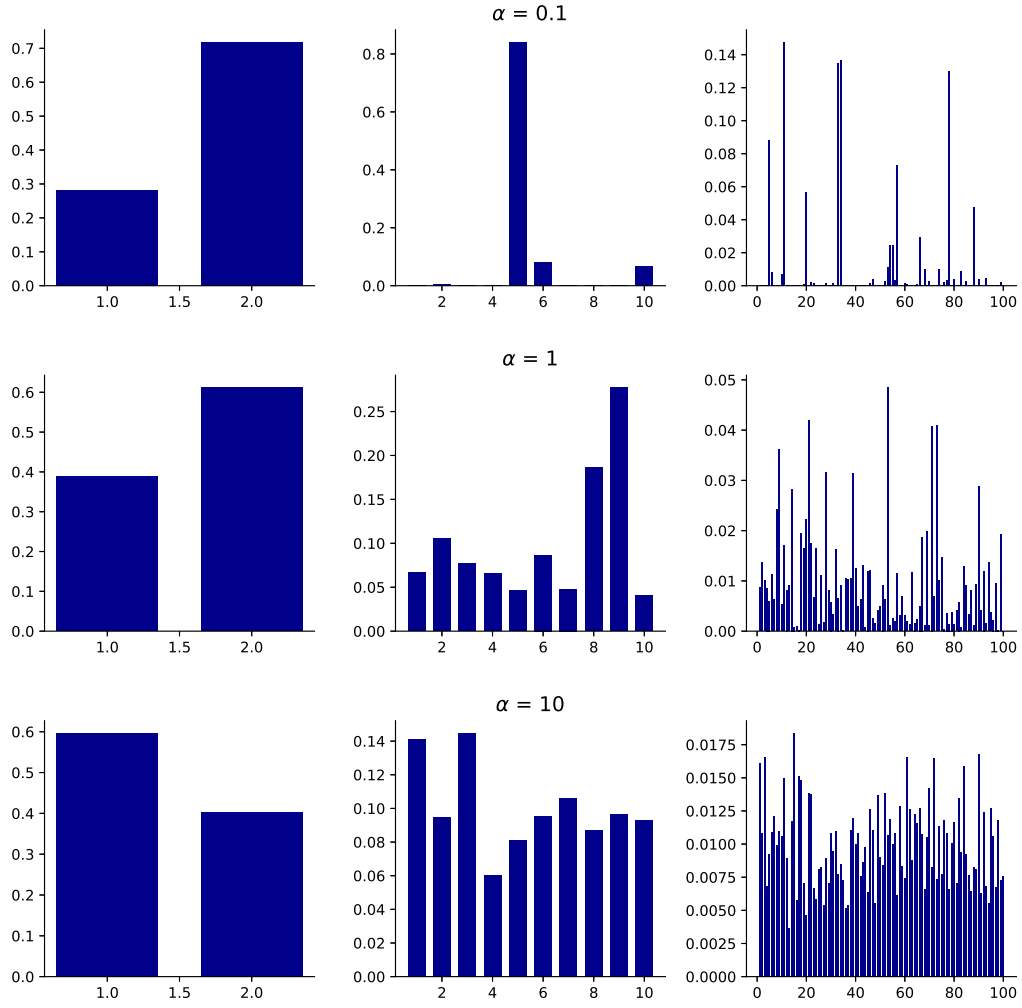


Figura 2.2: Muestra de una distribución Dirichlet simétrica para $\alpha \in \{0.1, 1, 10\}$ y $K \in \{2, 10, 100\}$.

2.1.2. Dirichlet Process

En un *finite mixture model* tenemos $G(\phi) = \sum_{k=1}^{K} \pi_k \delta_{\phi_k}(\phi)$, luego si muestreamos a partir de G, siempre (con probabilidad uno) obtendrémos exactamente K *clusters*. Nos gustaría tener un modelo más flexible, que pueda generar un número variable de *clusters*. La forma de hacer esto es remplazar la distribución discreta G por una medida aleatoria de probabilidad. El Dirichlet Process (Ferguson, 1973), denotado $G \sim DP(\alpha, H)$, es una manera de hacer esto.

Un **Dirichlet Process** (DP) es una distribución sobre medidas de probabilidad $G : \Phi \to \mathbb{R}^+$, donde $G(\phi) \geq 0$ y $\int_\Phi G(\phi) d\phi = 1$. Un DP se define implícitamente por cumplir

$$(G(A_1), \ldots, G(A_K)) \sim Dir(\alpha H(A_1), \ldots, \alpha H(A_K)) \tag{2.11}$$

para cualquier partición finita $(A_1, \ldots, A_k)$ de Φ. En este caso, decimos que $G \sim DP(\alpha, H)$, donde α es llamado el **parámetro de concentración** y $H : \Phi \to \mathbb{R}^+$ es llamado la **medida base**.

Como $p(G(A_1, \ldots, G(A_K)))$ es Dirichlet, la distribución marginal en cada partición distribuye beta $Beta(\alpha H(A_i), \alpha \sum_{j \neq i} H(A_j))$. El DP es considerado consistentemente definido, en el sentido de que si particionamos $\bar{A}_1$ en A_1 y A_2, entonces $G(\bar{A}_1)$ y $G(A_1) + G(A_2)$ siguen la misma distribución beta.

Sea $\phi \sim Dir(\alpha)$, y $z|\phi \sim Cat(\pi)$, si integramos π afuera obtenemos la distribución predictiva del modelo Dirichlet-multinoulli:

$$z \sim Cat(\alpha_1/\alpha_0, \ldots, \alpha_K/\alpha_0) \tag{2.12}$$

donde $\alpha_0 = \sum_k \alpha_k$. Es decir, $p(z = k|\alpha) = \alpha_k/\alpha_0$. Ademas, la posterior de π dada una observación viene dada por

$$\pi|z \sim Dir(\alpha_1 + \mathbb{I}(z = 1), \ldots, \alpha_K + \mathbb{I}(z = K)) \tag{2.13}$$

El DP generaliza el resultado anterior a particiones arbitrarias. Si $G \sim DP(\alpha, H)$, luego $p(\phi \in A_i) = H(A_i)$ y la posterior es

$$p(G(A_1), \ldots, G(A_K)|\phi, \alpha, H) = Dir(\alpha H(A_1) + \mathbb{I}(\phi \in A_1), \ldots, \alpha H(A_K) + \mathbb{I}(\phi \in A_K)) \tag{2.14}$$

Esto se mantiene para cualquier conjunto de particiones. Por lo tanto, si observamos multiples muestras $\bar{\phi}_{1:N} \sim G$, la nueva posterior está dada por

$$G|\bar{\phi}_{1:N}, \alpha, H \sim DP\left(\alpha + N, \frac{1}{\alpha + N}\left(\alpha H + \sum_{i=1}^{N} \delta_{\phi_i}\right)\right) \tag{2.15}$$

Por ende el DP define un prior conjugado para cualquier espacio medible, donde el parámetro de concentración α es como el tamaño de muestro efectivo de la medida base H.

Existen diferentes perspectivas que ayudan a entender la propiedad de *clustering* de un Dirichlet Process. En la sección 2.1.3. y 2.1.4. nos referimos a dos: el Stick Breaking Process y Chinese Restaurant Process (CRP).

2.1.3. Stick Breaking Process

Ahora daremos una definición constructiva para el DP, conocida como *stick breaking process* (Sethuraman, 1994). Sea $\pi = \{\pi_k\}_{k=1}^{\infty}$ una secuencia infinita de mezcla de pesos

derivadas a partir del siguiente proceso:

$$\beta_k \sim Beta(1, \alpha) \tag{2.16}$$

$$\pi_k = \beta_k \prod_{l=1}^{k-1}(1 - \beta_l) = \beta_k(1 - \sum_{l=1}^{k-1} \pi_l) \tag{2.17}$$

Esto se suele denotar como $\pi \sim GEM(\alpha)$, donde GEM representa Griffiths, Engen y McCloskey, ver Figura 2.3 para una ilustración.

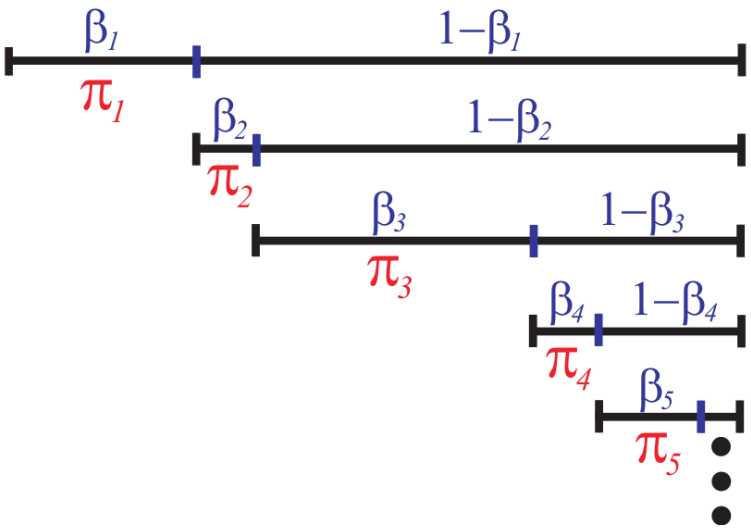


Figura 2.3: Ilustración de *stick breaking process*. Tenemos una barra de largo 1, la cual se rompe en un punto aleatorio β_1, el largo de la pieza que conservamos es llamada π_1, luego recursivamente rompemos la barra restante, así generando $\pi_2, \pi_3, \ldots$. Fuente: Figura 2.22 de (Sudderth, 2006).

Algunos ejemplos de este proceso son mostrados en la Figura 2.4 (a). A mayor α, menos varianza y mayor número de átomos, por el contrario, pequeños valores de α muestran una alta varianza y menor número de átomos, adicionalmente exhiben mayor varianza en el número de átomos.

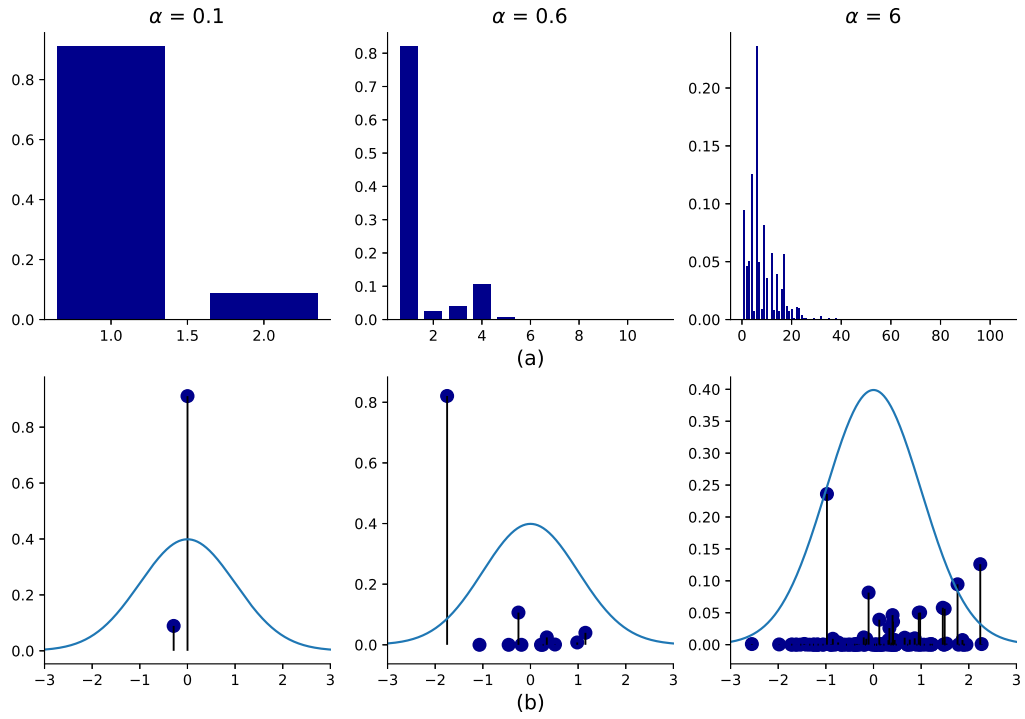


Figura 2.4: (a) Muestra de una distribución GEM para diferentes parámetros de concentración $\alpha \in \{0.1, 0.6, 6\}$. (b) Medidas aleatorias generadas a partir de un Dirichlet Process con medida base normal $\mathcal{N}(0, 1)$ para diferentes parámetros de concentración $\alpha \in \{0.1, 0.6, 6\}$

Se puede demostrar que este proceso terminará con probabilidad uno, a pesar que el número de elementos que este genera incrementa con α. Además, el tamaño del componente π_k decrece en promedio. Ahora definamos

$$G(\phi) = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}(\phi) \tag{2.18}$$

donde $\pi \sim GEM(\alpha)$ y $\phi_k \sim H$. Entonces uno puede demostrar que $G \sim DP(\alpha, H)$. Como consecuencia de esta construcción, las muestras de un DP son **discretas con probabilidad uno**. En otras palabras, al ir muestreando $\bar{\phi}_i \sim G$ veremos valores repetidos, por lo que la mayoría de los datos vendrán de los ϕ_k con π_k más largos. En la Figura 2.4 (b) muestra un par de medidas aleatorias generadas a partir de un DP con una medida base normal.

2.1.4. Chinese Restaurant Process

Trabajar con infinitos átomos puede ser bastante problemático. Para sortear esta dificultad podemos explotar la propiedad de *clustering* de un DP. Sea $\bar{\phi}_{1:N} \sim G$ observaciones generadas a partir de $G \sim DP(\alpha, H)$, sea K los distintos valores de $\bar{\phi}_{1:N}$, luego la distribución predictiva condicionada en las N observaciones está dada por

$$p(\bar{\phi}_{N+1} = \phi | \bar{\phi}_{1:N}, \alpha, H) = \frac{1}{\alpha + N} \left(\alpha H(\phi) + \sum_{k=1}^{K} N_k \delta_{\bar{\phi}_k}(\phi) \right) \tag{2.19}$$

donde N_k es el número de observaciones previas iguales a ϕ_k. Esto esquema de muestreo es llamado *Polya urn* o *Blackwell-MacQueen*. Esta construcción provee una forma constructiva para muestrear de un DP. Es más conveniente trabajar con variables discretas z_i que especifican cual valor de ϕ_k usar, así, definimos $\bar{\phi}_i = \phi_{z_i}$. Basado en esta expresión Tenemos

$$p(z_{N+1} = z | z_{1:N}, \alpha) = \frac{1}{\alpha + N} \left(\alpha \mathbb{I}(z = k^*) + \sum_{k=1}^{K} N_k \mathbb{I}(z = k) \right) \tag{2.20}$$

donde k^* representa un nuevo *cluster* que no ha sido usado aún. Este proceso es llamado Chinese Restaurant Process (CRP) (Aldous, 1985), basado en la oferta aparentemente infinito de mesas en ciertos restaurantes Chinos. La analogía es la siguiente: Las tablas del restaurante son los *clusters* y los clientes son las observaciones. Cuando una persona entra al restaurante, esta puede escoger sentarse en una tabla existente con probabilidad proporcional al número de personas ya sentadas en esa tabla (N_k), en otro caso, con una probabilidad decreciente a medida que más personas entran al restaurante (debido a $1/(\alpha + N)$) escogerá sentarse en una nueva tabla k^*. El resultado de este proceso es una distribución sobre particiones de los naturales, la cual es como una distribución de clientes a tablas.

Al hecho de que las tablas actualmente ocupadas son más probables de obtener nuevos clientes se le suele llamar el fenómeno del *rich get richer*. En efecto, se puede demostrar que la dsitribución del número de *clusters* que oinduce este prior es básicamente una ley de potencia, donde el número de tablas K con probabilidad 1 se aproxima a $\alpha log(N)$ cuando $N \to \infty$, mostrnado que la complejidad del modelo crece logaritmicamente con el tamaño de loos datos.

2.2. Modelos de tópicos

Los modelos de tópicos probabilísticos nos ayudan a descubrir los temas latentes (*clusters*) en una colección de documentos, como estos temas están conectados unos a otros y cómo cambian en el tiempo. Permiten resumir un gran colección de documentos a través de sus temas y organizarlos entorno a estos.

Los modelos probabilísticos tratan un tópico como una distribución de probabilidad discreta sobre el vocabulario del corpus, siendo un práctica habitual interpretar un tópico a partir de sus N palabras más probables. Por ejemplo, para $N = 5$ las palabras más probables de un tópico son: “llaves”, “domicilio”, “individuos”, “casa” y “porton”, por lo que una etiqueta valida para este tópico podría ser “portonazo”.

En *text mining* se suele trabajar bajo la asumpción de **bag of words** (bolsa de palabras), es decir, tanto los documentos como las palabras son tratadas como intercambiables. Es importante hacer notar que itercambiabilidad no es equivalente a que las variables aleatorias

son independientes e identicamente distribuidas. Más bien, intercambiabilidad esencialmente puede ser interpretado como condicionalmente independientes e identicamente distribuidas, donde el condicionamiento es con respecto a los parámetros de una distribución de probabilidad. Por lo tanto, el supuesto de intercambiabilidad es claramente un supuesto de simplificación cuya principal justificación es la construcción de algoritmos computacionalmente más eficientes.

Un *mixture model* bien conocido que trabaja bajo la asupción de *bag of words* es *mixture of unigrams* (Nigam et al., 2000), el cual asume que todos los documentos provienen de un solo *cluster* dentro de un conjunto finito K *clusters*. Los documentos de un *cluster* discuten solo un tópico particular z, y cada tópico z está asociado a una distribución categórica. Así, la verosimilitud de observar un documento d es

$$w|z \sim Cat(\theta_z) \tag{2.21}$$

$$p(w_1, \ldots, w_{N_d}) = \sum_{z=1}^{K} p(z) \prod_{i=1}^{N_d} p(w_i|z) \tag{2.22}$$

En comparación a *mixture of unigrams*, LDA y HDP suponen que las palabras de un documento provienen de un mismo *mixture model*. A nivel corpus los *mixture models* comparten parámetros, que vienen siendo los tópicos, pero las *mixtures of topics* son específicas de cada documento. Esto permite relajar la asumpción de que cada documento es generado por un solo tópico, permitiendo que un documento pueda tener presencia de más de un tema debido a que cada palabra proviene de algún tópico.

En las secciones 2.2.1-2.2.2 se describe en detalle dos modelos de tópicos probabilísticos, Latent Dirichlet Allocation (LDA) y Hierarchical Dirichlet Process (HDP), siendo este último la generalización no parámetrica de LDA, donde el número de tópicos a descubrir no está acotado y se infiere a partir del corpus.

2.2.1. Latent Dirichlet Allocation

En este saección se describe el proceso generativo de Latent Dirichlet Allocation (LDA) (Blei et al., 2003).

En LDA, cada palabra es asignada a su propio tópico, $z_{d,n} \in \{1, \ldots, K\}$ es el tópico del que proviene la palabra n del documento d. Esta asignación a su vez es dibujada de una distribución específica del documento θ_d. Cada tópico es una distribución de probabilidad sobre un vocabulario fijo V. El modelo completo es como sigue

$$\phi_k|\eta \sim Dir(\frac{\eta}{|V|}1_{|V|}) \tag{2.23}$$

$$\pi_d|\alpha \sim Dir(\frac{\alpha}{K}1_K) \tag{2.24}$$

$$z_{d,n}|\pi_d \sim Cat(\pi_d) \tag{2.25}$$

$$w_{d,n}|z_{d,n}, \phi_{1:K} \sim Cat(\phi_{z_{d,n}}) \tag{2.26}$$

Esto es ilustrado en la Figura 2.5.

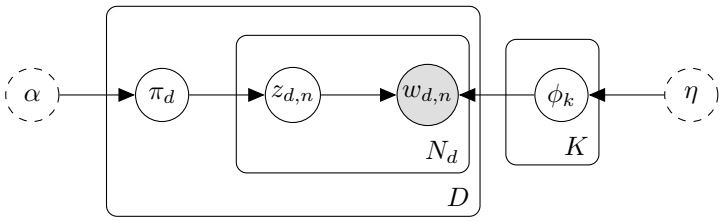


Figura 2.5: Representación gráfica de LDA: círculos denotan variables aleatorias, círculos abiertos denotan parámetros, círculos sombreados denotan variables observadas y los platos indican replicación.

La probabilidad conjunta del modelo:

$$p(\phi, \pi, z, w|\alpha, \eta) = \prod_{k=1}^{K} p(\phi_k|\eta) \prod_{d=1}^{D} p(\pi_d|\alpha) \prod_{n=1}^{N_d} p(z_{n,d}|\pi_d) p(w_{d,n}|\phi_{1:K}, z_{d,n}) \tag{2.27}$$

La distribución a posterior:

$$p(\phi, \pi, z|w, \alpha, \eta) = \frac{p(\phi, \pi, z, w|\alpha, \eta)}{p(w|\alpha, \eta)} \tag{2.28}$$

La distribución posterior es computacionalmente intratable para inferencia exacta, debido a que para normalizar la distribución debemos marginalizar sobre todas las variables ocultas y escribir la constante de normalización en términos de los parámetros del modelo. Para poder computar la posterior es necesario utilizar algoritmos de inferencia aproximada, donde el enfoque habitual es Markov Chain Monte Carlo (MCMC) e Inferencia Variacional (VI).

Una representación equivalente en LDA sería generar cada palabra de un documento d a partir de un tópico dibujado por una distribución G_d,

$$\phi_k|\eta \sim Dir(\frac{\eta}{|V|}1_{|V|}) \tag{2.29}$$

$$\pi_d|\alpha \sim Dir(\frac{\alpha}{K}1_K) \tag{2.30}$$

$$G_d(\phi) = \sum_{k=1}^{K} \pi_{d,k}\delta_{\phi_k}(\phi) \tag{2.31}$$

$$\phi_{d,n}|\pi_d, \phi_{1:K} \sim G_d \tag{2.32}$$

$$w_{d,n}|\phi_{d,n} \sim Cat(\phi_{d,n}) \tag{2.33}$$

2.2.2. Hierarchical Dirichlet Process

Hierarchical Dirichlet Process (HDP)(Teh et al., 2005) es un prior no paramétrico, donde la medida base G_0 de un conjunto de DPs es dibujada a partir de un DP. En el caso de modelamiento de tópicos, temenos un medida global a nivel corpus que es dibujada a partir de un DP con medida base Dirichlet y una medida para cada documento que es dibujada a partir de un DP cuya medida base es G_0. El modelo completo es como sigue

$$H = Dir(\frac{\eta}{|V|}1_{|V|}) \tag{2.34}$$

$$G_0|\gamma, H \sim DP(\gamma, H) \tag{2.35}$$

$$G_d|\alpha, G_0 \sim DP(\alpha_0, G_0) \tag{2.36}$$

$$\phi_{d,n}|G_d \sim G_d \tag{2.37}$$

$$w_{d,n}|\phi_{d,n} \sim Cat(\phi_{d,n}) \tag{2.38}$$

Esto es ilustrado en la Figura 2.6.

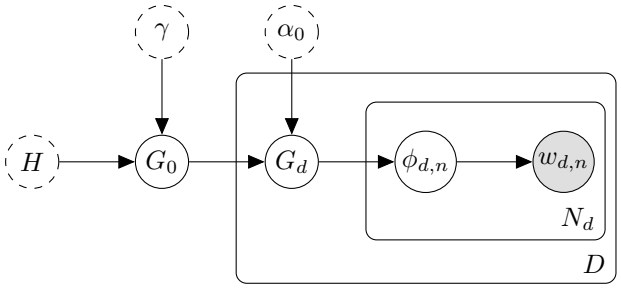


Figura 2.6: Representación gráfica de HDP: círculos denotan variables aleatorias, círculos abiertos denotan parámetros, círculos sombreados denotan variables observadas y los platos indican replicación.

La discretitud a nivel corpus de G_0 asegura que todos los documentos comparten el mismo conjunto de tópicos (*mixture components*). A nivel documento G_d hereda los tópicos de G_0, pero los pesos de cada tópico (*mixture proportions*) es específica del documento.

Aplicando *stick breaking construction* se tiene que para el DP dibujado a nivel corpus la siguiente representación:

$$\beta_k^{'} \sim Beta(1, \gamma) \tag{2.39}$$

$$\beta_k = \beta_k^{'} \prod_{l=1}^{k-1}(1 - \beta_l^{'}) \tag{2.40}$$

$$\phi_k \sim H \tag{2.41}$$

$$G_0(\phi) = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k}(\phi) \tag{2.42}$$

Así, G_0 es discreto y tiene soporte en los átomos $\phi = \{\phi\}_{k=1}^{\infty}$ con pesos $\beta = \{\beta_k\}_{k=1}^{\infty}$, siendo la distribución de β escrita como $\beta \sim GEM(\gamma)$. La construcción a nivel documento de G_d es:

$$\pi_{d,k}^{'} \sim Beta\Big(\alpha_0\beta_k, \alpha_0\Big(1 - \sum_{l=1}^{k}\beta_l\Big)\Big) \tag{2.43}$$

$$\pi_{d,k} = \pi_{d,k}^{'} \prod_{l=1}^{k-1}(1 - \pi_{d,l}^{'}) \tag{2.44}$$

$$G_d(\phi) = \sum_{k=1}^{\infty} \pi_{d,k} \delta_{\phi_k}(\phi) \tag{2.45}$$

$$\phi_{d,n}|\pi_d, \phi_{1:\infty} \sim G_d \tag{2.46}$$

Donde $\phi = \{\phi_k\}_{k=1}^{\infty}$ son los mismos átomos de G_0. Esto es ilustrado en la Figura 2.7.

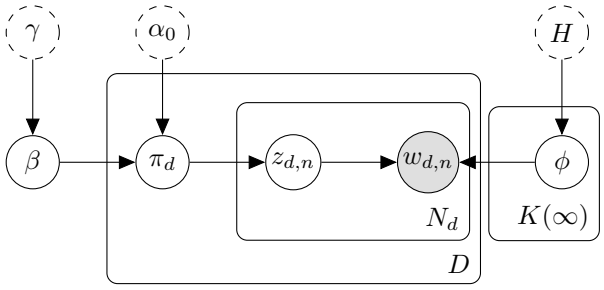


Figura 2.7: Representación gráfica de la construcción stick-breaking de HDP: circulos denotan variables aleatorias, circulos abiertos denotan parámetros, círculos sombreados denotan variables observadas y los platos indican replicación.

Al igual que LDA la distribución posterior es intratable, por lo que en (Teh et al., 2005) se marginaliza G_0 y G_d's, obteniéndose así un nuevo proceso generativo denominado *Chinese restaurant franchise process* (CRF), esta representación permite construir algoritmos eficientes basados en Gibbs Sampling, como en la implementación en C++ utilizada (Wang and Blei, 2010).

2.2.3. LDA versus HDP

HDP es un modelo no paramétrico similar en estructura a LDA, la principal desventaja de LDA frente a HDP es que LDA requiere escoger el número de tópicos K por adelantado, por otro lado, en HDP el número de tópicos no está acotado y es inferido a partir de los datos. En un enfoque tradicional, se requiere de entrenar múltiples veces LDA para diferentes valores de K y se escoge el que tiene mejor la configuración con mejor desempeño en un conjunto de validación, por lo que LDA termina siendo computacionalmente más costoso que HDP, además este enfoque se vuelve impracticable cuando el conjunto de datos es grande. En el aspecto cualitativo ambos modelos entregan tópicos igual de consistentes, en cuanto a métricas de desempeño como *perplexity* HDP suele tener mejor desempeño (Teh et al., 2005).

2.2.4. Interpretación de tópicos

Los modelos de tópicos se caracterizan por tener un alto poder interpretativo, esto se debe a que la distribución de probabilidad de cada tópico sobre el vocabulario nos da una idea del tema al que pertenece, por otro lado la mezcla de tópicos de cada documento muestra que tan importante es cada tópico en la generación de estos, como también dentro del corpus. En este sentido, las visualizaciones nos ayudan a interpretar mejor los resultados de los modelos de tópicos, respondiendo a las siguientes preguntas, ¿Cuál es el significado de cada tópico?¿Cuán predominante es cada tópico?¿Cómo se relacionan los tópicos entre sí?

En (Sievert and Shirley, 2014) desarrollaron una herramienta de visualización web para responder a estas preguntas. Para responder la pregunta 1 se incorpora un gráfico de barras que muestra las palabras más relevantes del tópico seleccionado dado un parámetro $\lambda \in [0, 1]$. A través de una visualización espacial responde la pregunta 2 y 3. La visualización espacial consiste en aplicar técnicas de reducción de dimensionalidad como TSNE (Maaten and Hinton, 2008) o PCA (Wold et al., 1987) (en este caso se utilizó TSNE) a la matriz de distancia entre tópicos, usando Jensen-Shannon divergence (Endres and Schindelin, 2003) como médida de distancia. Una vez cada tópico es mapeado a un punto en un espacio de dos dimensiones se dibuja un círculo con centro en este punto y con radio proporcional a la cantidad de tokens generados por el tópico.

Para interpretar un tópico, lo usual es examinar una lista ordenada de las palabras más probables del tópico, usando ya sea desde cinco a treinta términos. Un problema frecuente que se presenta en este caso es que los términos que son comunes al corpus frecuentemente aparecen en el top de las palabras más probables de un tópico, haciendo difícil discernir el significado de estos. Para esto en (Sievert and Shirley, 2014) se define una métrica denominada *relevance*, la cual define la relevancia de una palabra no solo por su probabilidad dentro del tópico sino también por su exclusivad dentro del corpus. La *relevance* de una palabra w en el tópico k dado λ está dada a través de la siguiente expresión:

$$r(w, k|\lambda) = \lambda log(\phi_{kw}) + (1 - \lambda)\lambda log\left(\frac{\phi_{kw}}{p_w}\right) \tag{2.47}$$

, donde λ determina el peso que se le da a la probabilidad de la palabra w dentro del tópico k (ϕ_{kw}) relativo a su *lift*, el cual se define por el ratio entre la probabilidad de la palabra

dentro del tópico y su probabilidad marginal a lo largo del corpus (p_w). Fijando $\lambda = 1$ se obtiene el ranking de términos decrecientes en orden de su probabilidad dentro del tópico, y fijando $\lambda = 0$ el ranking se basa solo en el *lift*.

2.3. Modelamiento de la evolución de los tópicos en el tiempo

Nuestro objetivo no es solo descubrir los tópicos en el tiempo sino también modelar sus interacciones en el tiempo, como nacimiento, muerte, evolución, divisón y fusión.

En (Wilson and Robinson, 2011) y (Beykikhoshk et al., 2018) se propone una metodología que permite capturar los dinámismos mencionados usando LDA y HDP respectivamente. En estas se propone dividir el corpus en T épocas, en cada época se entrena un modelo de tópicos estático, obteniéndose así T conjuntos de tópicos $\phi = \{\phi_1, \ldots, \phi_T\}$, donde $\phi_t = \{\phi_{t,1}, \ldots, \phi_{t,K_t}\}$ es el conjunto de tópicos que describen la época t, y K_t el número de tópicos inferido en esa época. Una vez descubiertos los tópicos se hace uso de medidas de distancia o similitud para relacionar tópicos de épocas adyacentes

En la sección se describe la metodología propuesta en (Beykikhoshk et al., 2018) para relacionar los tópicos descubiertos de épocas adyacentes.

2.3.1. Gráfo de similitud temporal

Para relacionar los tópicos de una época necesitamos una medida de similitud $\rho \in [0, 1]$, con esta médida de similitud se puede construir un gráfo, donde los nodos son los tópicos de una época y los arcos relacionan tópicos de una época con la siguiente, siendo el peso del arco la similitud entre los tópicos. Una vez construido el grafo se eliminan las conexiones débiles en base a un umbral $\zeta \in [0, 1]$ a definir, reteniendo solo aquellas conexiones entre tópicos suficientemente similares entre épocas adyacentes, matemáticamente podamos el arco entre los tópicos $\phi_{t,i}$ y $\phi_{t+1,j}$ si $\rho(\phi_{t,i}, \phi_{t+1,j}) \leq \zeta$.

Está metodología permite fácilmente detectar desaparición de un tópico, nacimiento de un nuevo tópico, como también dividir o fusionar diferentes tópicos, a continuación se define en detalle cada uno de estos dinamismos:

- **Nacimiento de un tópico:** Si un tópico no tiene ningún arco entrante, por ejemplo, en la Figura 2.8 el tópico ϕ_{j+2} en t.
- **Muerte de un tópico:** Si un tópico no tiene ningún arco saliente, por ejemplo, en la Figura 2.8 el tópico ϕ_j en t.
- **Evolución de un tópico:** Cuando un tópico tiene exactamente un arco de entrada y salida, por ejemplo, en la Figura 2.8 entre las épocas t y $t + 1$ se tiene que el tópico ϕ_{j+2} evoluciona del tópico ϕ_{k+1}.
- **División de un tópico:** Si un tópico tiene más de un arco saliente, por ejemplo, en la Figura 2.8 el tópico ϕ_i de $t - 1$ se divide en $t + 1$ en los tópicos ϕ_j y ϕ_{j+1}.

- **Fusión de un tópico:** Cuando un tópico tiene más de un arco entrante, este tipo de tópicos también pueden ser entendidos como un nuevo tópico, por ejemplo, en la Figura 2.8 los tópicos ϕ_i y ϕ_{i+1} de $t-1$ forman al tópico ϕ_{j+1} en t.

Una ilustración conceptual del grafo de similitud es mostrado en la Figura 2.8, este muestra tres épocas consecutivas.

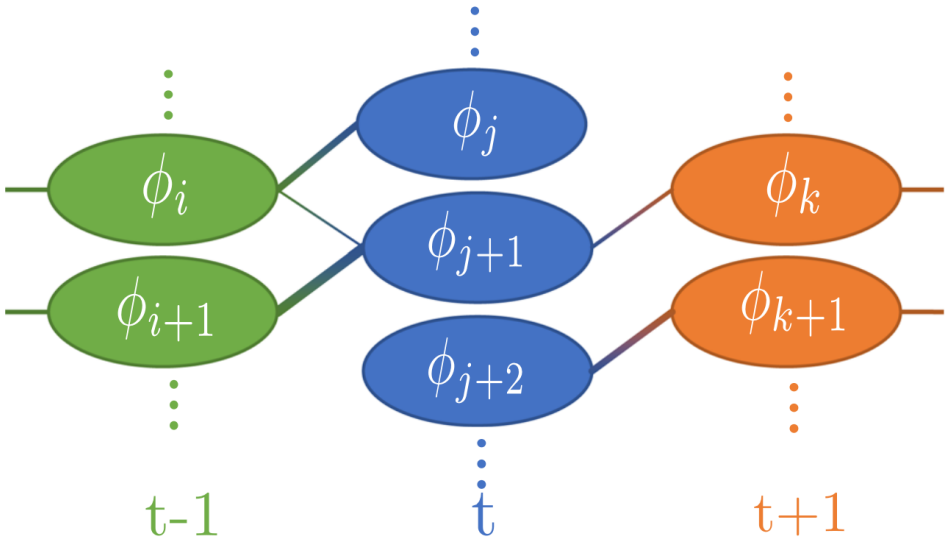


Figura 2.8: Ilustración conceptual del grafo de similitud que modela la dinámica de los tópicos en el tiempo. Un nodo corresponde a un tópico en una época específica; el ancho de los arcos es proporcional a la similitud entre los tópicos, arcos ausentes fueron eliminados por presentar una similitud menor a un umbral. Fuente: Figura 3 de (Beykikhoshk et al., 2018)

Un aspecto relevante de esta metodología es definir el úmbral de corte, el cual no es fácilmente interpretable, además el úmbral depende de la médida de similitud escogida, dificultando así la comparación entre médidas de similitud. En (Beykikhoshk et al., 2018) proponen una alternativa más interpretable para definir el úmbral, para esto estiman la función de distribución acumulada (cdf) del grafo inicial, donde todos los nodos de una época están conectados con todos los nodos de la época adyacente. Sea F_p la cdf sobre las similitudes del grafo inicial, luego sea $\zeta \in [0,1]$ el punto operante de la cdf, luego eliminamos el arco entre los tópicos $\phi_{t,i}$ y $\phi_{t+1,j}$ si $\rho(\phi_{t,i}, \phi_{t+1,j}) \leq F_p^{-1}(\zeta)$, donde $F_p^{-1}(\zeta)$ es el cuantil ζ de F_p.

Ejemplo

2.3.2. Medidas de similitud

Mencionar otras métricas de similitud

Los tópicos son distribuciones de probabilidad sobre un vocabulario fijos de términos. La gran general las medidas de similitud o distancia comparan vectores con el mismo dominio y dimensión, esto significa que los tópicos de épocas adyacentes deben compartir el mismo

vocabulario. Matemáticamente, sea $\phi_{t,i}$ un tópico de la época t y V_t su vocabulario, sea $\phi_{t+1,j}$ un tópico de la época $t+1$ y V_{t+1} su vocabulario, lo más probables es que existan palabras en V_t que no existan en V_{t+1} y viceversa, para poder comparar tópicos en estas épocas adyacentes se debe construir el vocabulario $V'_{t+1} = V_t \cup V_{t+1}$, luego aplicar *padding* a los vectores $\phi_{t,i}$ y $\phi_{t+1,j}$, es decir, se rellenan con ceros las posiciones de palabras que no están en el vocabulario de su dominio.

La gran desventaja del enfoque anterior es que no captura similitud entre palabras, es decir, dos palabras diferentes que pueden llegar a ser sinónimos ocuparan una posición diferente dentro del vector, siendo no robusta a palabras que estan presentes en la época t y no en $t-1$ por lo que no hay forma de compararla. El peor caso sería considerar los vocabularios V_t y V_{t+1}, donde $V_t \cap V_{t+1} = \emptyset$, a pesar de que cada palabra en V_t tiene un sinónimo en V_{t+1} la similitud entre tópicos entre las épocas t y $t+1$ sería cero.

Para lidiar con el problema anterior en (Kusner et al., 2015) se propone una medida de distancia llamada Word Mover's Distance (WMD) para comparar dos documento bajo una representación *bag of words* a través de sus *word embeddings* (Mikolov et al. (2013)).

WMD calcula el costo mínimo de transformar un documento en otro, en esto caso particular sería el costo mínimo de llevar un tópico a otro. Para esto se resuelve el problema de transporte, donde los flujos son los pesos $\phi_{t,i}$ y $\phi_{t+1,j}$ y la matriz de costos es una matriz de distancia euclidiana entre los *word embedding* de todas las palabras deS V_t con V_{t+1}. En la Figura 2.9 se ilustra el espacio en el que viven las palabras de dos tópicos.

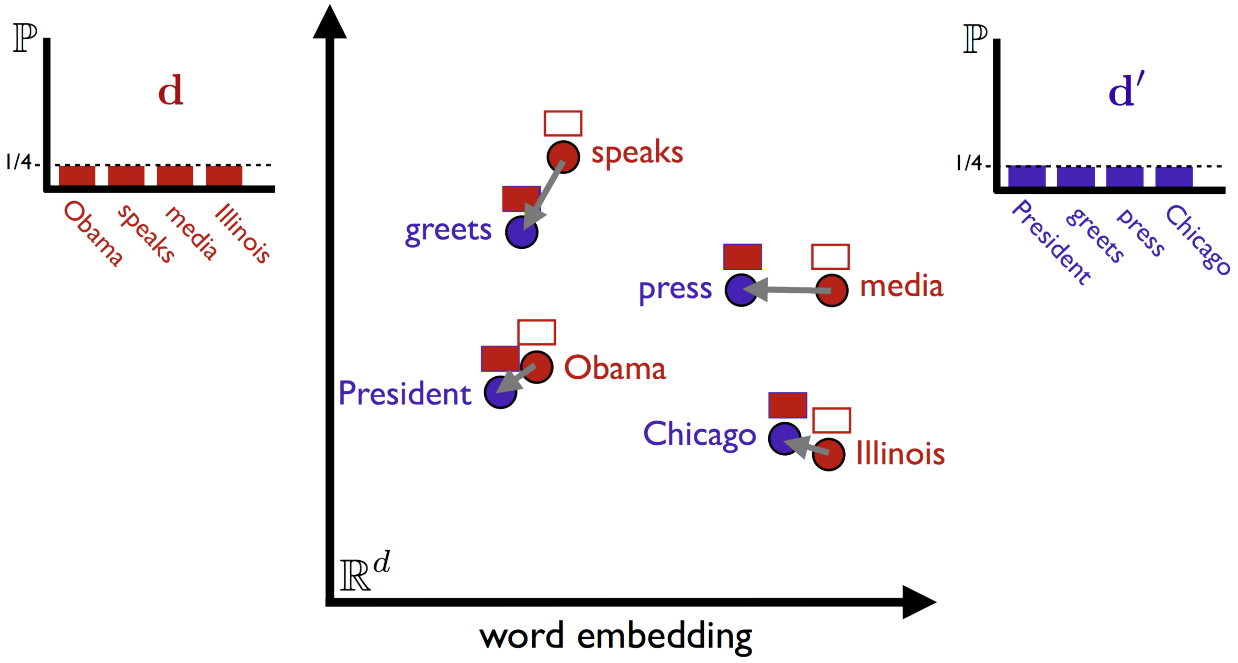


Figura 2.9: Espacio vectorial de los *word embeddings* de las palabras de dos tópicos con un vocabulario de tamaño 4. Fuente: Figura de (Niculae, 2015).

Sea V_i y V_j los vocabularios del tópico i y j respectivamente, luego su WMD viene dado por $WMD(\phi_i, \phi_j)$:

$$\underset{x}{\text{minimize}} \sum_{u \in V_i} \sum_{v \in V_j} c_{u,v} x_{u,v} \tag{2.48}$$

$$\text{s.t.} \quad \sum_{v \in V_j} x_{u,v} = \phi_{i,u}, \ u \in V_i \tag{2.49}$$

$$\sum_{u \in V_i} x_{u,v} = \phi_{j,v}, \ v \in V_j \tag{2.50}$$

$$x_{u,v} \geq 0, \ u \in V_i \ , v \in V_j \tag{2.51}$$

Donde $x_{u,v}$ es el flujo que va de la palabra u del tópico i a la palabra v del tópico j, $\phi_{i,u}$ es la probabilidad de la palabra u en el tópico i, $c_{u,v}$ es el costo de mover una unidad de flujo por el arco (u, v), el costo entre palabras se mide como la distancia euclidiana entre los *word embedding* de dichas palabras.

La primera restricción indica que el flujo que se mueve de una palabra u del tópico i a todas las palabras del tópico j debe sumar su peso ($\phi_{i,u}$), la segunda restricción significa que el flujo que se mueve de una palabra v del tópico j a todas las palabras del tópico i debe sumar su peso ($\phi_{j,v}$). Lo anterior implica que esta medida de distancia es simétrica, es decir, $WMD(\phi_i, \phi_j) = WMD(\phi_j, \phi_i)$.

WMD se puede fácilmente transformar en una médida de similitud, $\rho(\phi_i, \phi_j) = \frac{1}{1+WMD(\phi_i,\phi_j)}$, notar que si la WMD es 0 la similitud es 1 y si es ∞ la similitud es 0.

WMD es una medida de distancia intensiva en recursos computacionales. Para entender mejor esto utilizaremos la representación poliedral del problema, sea N el tamaño del vocabulario entre dos épocas adyacentes, luego la región factible del problema anterior se puede representar como $\{x|Ax = b, x \geq 0\}$, con $A \in \mathbb{R}^{2N \times N^2}$ la matriz de costos, $b \in \mathbb{R}^{2N}$ el flujo disponible y $x \in \mathbb{R}^N$ el flujo a enviar por cada uno de los arcos. Para resolver este problema se utilizó (Doran, 2014), la cual está basada en el algoritmo (Pele and Werman, 2009), cuya complejidad del mejor tiempo promedio escala $\mathcal{O}(N^2 logN)$, por lo si se reduce el vocabulario a un décimo esto traera en el peor caso promedio un *speed up*de de 200.

Los tópicos siguen una distribución de ley de potencia sobre el vocabulario, donde una pequeña fracción de las palabras concentran la mayor parte de la masa de la distribución. Además, en la práctica la interpretación de los tópicos se basa en los top N palabras más probables (o relevantes) con $N \in [5, 30]$, entonces, podemos aprovechar esta estructura para efectos de computar la WMD de un forma más eficiente, por ejemplo, utilizando solo las palabras que capturan un X % de la distribución acumulada del tópico.

Computar WMD requiere contar con *words embeddings*. Para estó se utilizó una de las más grandes colección de *words embeddings* en español (Cañete, 2019a), que cuenta con 1.313.423 *embeddings*, colección obtenida utilizando el algoritmo FasText (Bojanowski et al., 2017) sobre el corpus Spanish Unannotated Corpora (SUC) (Cañete, 2019b), uno de los más

grandes corpus de texto en español. FasText en comparación a otros enfoques para extraer *embeddings* representa los *tokens* a través de n-gramas de caracteres, de esta manera se pueden obtener *embeddings* de *tokens* no vistos durante el entrenamiento a partir de los *embeddings* de los caracteres que lo componen.

2.3.3. Métricas

En la metodología propuesta existen podemos evaluar el desempeño del descubrimiento de tópicos y las conexiones entre tópicos de épocas adyacentes. En ambos casos no se cuenta con el *ground truth* para medir correctamente el desempeño de la metodología propuesta. Si conocieramos el *ground truth*, podríamos utilizar *purity* (Manning et al., 2008) para comparar la asignación de los documentos en torno a los tópicos con la etiqueta. En el caso del grafo temporal, si conocieramos las conexiones presentes y ausentes podríamos utilizar métricas de clasificación.

Los modelos de tópicos suelen tener múltiples hiperparámetros y que no requieren de una etiqueta, en (Blei et al., 2003; Griffiths and Steyvers, 2004; Cao et al., 2009; Deveaud et al., 2014,?; Arun et al., 2010; Zhang et al., 2017) se describen algunas métricas que pueden ser útil para calibrarlos. Cabe destacar que estas métricas carecen de significado y sirven para comparar si un modelo de tópicos es superior a otro.

El trabajo propuesto no tiene por objetivo calibrar los hiperparámetros del HDP y se utiliza la configuración especificada en la sección 2.2.2. Por otro lado, si medimos el desempeño de la evolución de los tópicos y se asume que los tópicos descubiertos están correctos. El desempeño es medido sobre un grafo etiquetado a través de métricas de clasificación. Esto es posible debido a que el fenómeno de robo de vehículos no presenta muchos tópicos a diferencia de los tópicos latentes que podríamos encontrar en Wikipedia.

La métrica escogida en este caso es el *macro average recall* (Forman, 2003), esta corresponde al promedio simple entre los aciertos de la clase presencia y ausencia de conexión. En general, debería haber más ausencia que presencia de conexión, así la clase presencia de conexión no será menospreciada, ya que el macro recall considera que el porcentaje de acertividad de ambas clases son igual de importantes.

Al contar con un grafo etiquetado podemos observar el efecto de los hiperparámetros ζ y q en la métrica de desempeño escogida. El hiperparámetro $\zeta \in [0, 1]$ define el úmbral de corte, representa el punto operante de la cdf del grafo *fully connected*, permite definir el cuantil que se usará como úmbral para eliminar arcos con similitud menor a este. El parámetro $q \in [0, 1]$ define el soporte de los tópicos, utilizando aquellas palabras más probables que explican 100q % de la distribución acumulada del tópico.

2.4. Procesamiento

El proposito del procesamiento en *text mining* es simplificar los datos lo más posible tal que se mantiene el *core* de palabras del corpus. En el caso de modelamiento de tópicos, esta etapa puede reducir significativamente el vocabulario. Como consecuencia, esto puee traer

una mejora en la significancia estadística de los modelos, puesto que se puede obtener un mejor balance entre cantidad de parámetros y observaciones. Adicionalmente, puede facilitar la interpretación de los tópicos, removiendo palabras que aportan poca información.

En este experimento se aplicaron cinco etapas:

1. **Tokenización**: La tokenización es una operación sobre una cadena de caracteres (*string*) que consiste en dividir el *string* (ej: por el caracter espacio) en un conjunto de términos, obteniéndose así una lista elementos llamados *tokens*, que en términos simples pueden considerarse como una palabra.

2. **Procesamiento de caracteres**: En esta etapa se suelen aplicar algunas operaciones básicas de procesamiento. En este proceso se llevan los tokens a unicode y minúsculas. Luego, se eliminan patrones de caracteres que dificilmente pueden tener algún significado, como correos electrónicos, símbolos de puntuación, tokens con números y letras o solo números.

3. **Eliminación de stopwords**: Las *stopwords* son palabras que aportan poca información (ej: artículos, preposiciones y conectores), usualmente tienen un alta frecuencia dentro del corpus. Para esto se utilizó una lista de palabras de *stopwords* disponible en el paquete NLTK de Python de 313 palabras (Bird et al., 2009). Además, esta lista se alimentó con 951 *stopwords* contextuales, palabras especificas del corpus que aportan poca información, en el caso de robo de vehículos palabras relacionas a "robo" o "vehículo" no aportan ninguna información, puesto a que todos los relatos habalan del robo de un vehículo.

4. **Filtro por vocabulario**: Con el proposito de mantener las palabras que son "humanamente legibles" se utiliza un vocabulario. Para esto se utilizó el vocabulario del corpus SUC descrito en la sección anterior. Al ser este gran vocabulario el dominio de los embeddings, se garatiza que al computar WMD se tengan los *embeddings* de las palabras.

5. **Filtro por frecuencia**: El último nivel de procesamiendo corresponde a eliminar *tokens* con baja frecuencia. Esta etapa viene motivada del hecho de que un modelo difícilmente aprenderá algún patrón de un evento que tiene muy pocas realizaciones, menos si tiene una realización única. Esta etapa se aplicó a nivel epoca, eliminando aquellos tokens que aparecen en menos del 0.1 % de los documentos de su época.

Capítulo 3

Experimento

Motivación

Describir organización del contenido

Diseño del experimento

3.1. Datos

Para este experimento se cuenta con las fuentes de datos de la Asociación de Aseguradores de Chile (AACH), corresponde a los relatos que las víctimas del robo de sus vehículos dan a las aseguradoras, lo cual corresponde a 49.015 relatos entre el 2011 y 2016.

Para el uso de WMD es necesario contar con *words embeddings*, para esto se utilizaron los *embeddings* de (**?**), estos *embeddings* fueron obtenidos utilizando el algoritmo FastText (Bojanowski et al., 2017) sobre el corpus Spanish Billion Word Corpus (SBWC) (**?**). FasText en comparación a otros enfoques para extraer *embeddings* representa los *tokens* a través de n-gramas de caracteres, de esta manera se pueden obtener *embeddings* de *tokens* no vistos durante el entrenamiento a partir de los *embeddings* de los caracteres que lo componen.

Descri del dataset: mejorar estilo y añadir ejemplos

3.2. Procesamiento

En minería de texto con el objetivo de extraer el core de palabras del corpus se recurre métodos para reducir el vocabulario, la reducción del vocabulario mejora la significancia estadística de los modelos, puesto que se obtiene un mejor balance entre cantidad de parámetros y cantidad de observaciones, por otro lado puede verse fácilitada la interpretación de los tópicos al remover palabras que aportan poca información.

Rehacer

El paso cero en el procesamiento de textos es tokenizar, la tokenización es una operación sobre una cadena de carácteres (*string*) que consiste en dividir el *string* en un conjunto de términos, en este caso la división se hizo por el carácter espacio, como resultado de esto se obtiene una lista de elementos, a cada elemento de esta lista se le denomina *token* que en términos simples puede considerarse como una palabra para el ejemplo mencionado.

Mover a marco teórico

Mover a marco teórico

Luego, en el primer nivel de procesamiento no interesa hacer distinción entre mayúsculas o minúsculas[1], por ende, los carácteres de cada token son llevados a minúscula, también se eliminaron carácteres y tokens que no aportan información, como símbolos de puntuación, correos electrónicos y tokens que contienen números. En la figura 3.1 se observa la distribución acumulada de los tokens del corpus a este nivel de procesamiento, adicionalmente se tiene que el 50 % de los *tokens* del vocabulario ocurren una sola vez, el 80 % tiene una ocurrencia menor o igual a 5 y el 95 % de la distribución acumulada puede ser explicada con 4199 *tokens* (9 %) del vocabulario, se concluye que la distribución es sumamente pesada y es necesario recurrir a métodos adicionales para su reducción.

Mover a marco teórico

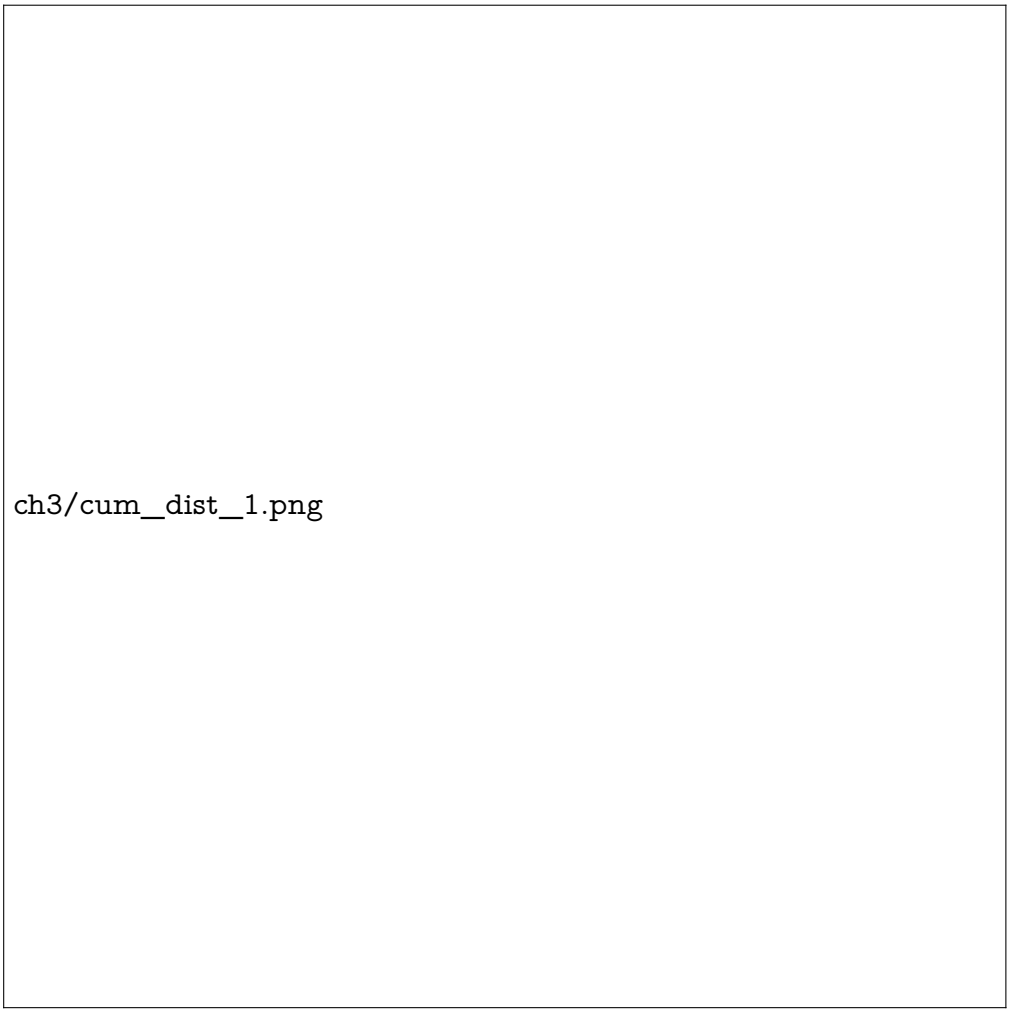


Figura 3.1: Frecuencia acumulada de los tokens únicos aplicando hasta el primer nivel de procesamiento. El eje horizontal es el acumulado de tokens únicos en orden decreciente de ocurrencia. Los puntos corresponden a los cuantiles 60 %, 80 %, 90 %, 95 % y 99 %.

Rehacer

En el segundo nivel de procesamiento se eliminaron las stopwords, palabras que aportan poca información, como artículos, preposiciones y conectores, para esto se utilizó la lista de

[1] En análisis de sentimiento puede ser interesante ya que las personas suelen expresar mensajes de enfado con letras capitales, por lo que las letras capitales añaden información al análisis.

stopwords disponible en el paquete NLTK de Python (Bird et al., 2009) la cual cuenta con 313 palabras. Además, esta lista de *stopwords* se alimentó con *stopwords* contextuales, palabras específicas del corpus que aportan poca información, para esto se hizó un etiquetado de las 1000 palabras más frecuentes del corpus incorporando 417 nuevas palabras, algunos ejemplos son palabras que hacen referencia a vehículo y robo, puesto que todos los documentos corresponden a robos de vehículos.

Mover a marco teórico

El tercer nivel de procesamiento consiste en normalizar los tokens para reducir aún más el vocabulario, como métodos de normalización los más utilizados son *stemming* y lematización. *Stemming* es el proceso de llevar una palabra a su raíz (*stem*), en la práctica *stemming* consiste en aplicar un algoritmo basado en ciertas reglas gramáticales para extraer sufijos (Porter et al., 1980), como desventaja es que stemming no tiene en cuenta el contexto de la palabra por lo que la raíz obtenida puede no corresponder a la raíz verdadera de la palabra, además, para el caso de modelamiento de tópicos los tópicos se vuelve más difícil de interpretar, ya que palabras con significado completamente distinto terminan con la misma raíz o bien la raíz encontrada no tiene un significado claro. Por otro lado, lematización es el proceso de agrupar juntas las formas flexionadas de una palabra para que puedan analizarse como un elemento, identificado como lema, su diferencia principal con *stemming* es que opera con conocimiento del contexto de la palabra para discriminar entre palabras que tienen significado diferente dependiendo del *part of speech tagging* (POST) y de una tabla de búsqueda (*lookup table*). Como método de normalización se decidió utilizar lematización en vez de stemming debido a que tiene menos impacto en la interpretación de los tópicos, sin embargo es una operación más intensiva debido a que stemming es un algoritmo basado en reglas simples mientras que en lematización se suele usar redes neuronales recurrentes (RNNs) para el POST y una vez determinadas las etiquetas gramaticales de las palabras en un documento se utiliza una *lookup table* para encontrar el lema correspondiente. La implementación de lematización utilizada es la implementación de lematización en español del paquete spaCy de Python (Honnibal and Montani, 2017).

Mover a marco teórico

El cuarto y último nivel de procesamiento corresponde a eliminar tokens con baja frecuencia, puesto que el modelo no será capaz de levantar patrones en tokens que aparecen una única vez o con una ocurrencia poco significativa, luego, como el corpus está particionado en épocas, se eliminaron aquellos tokens que aparecen en menos de 5 documentos dentro de una época.

Mover a marco teórico

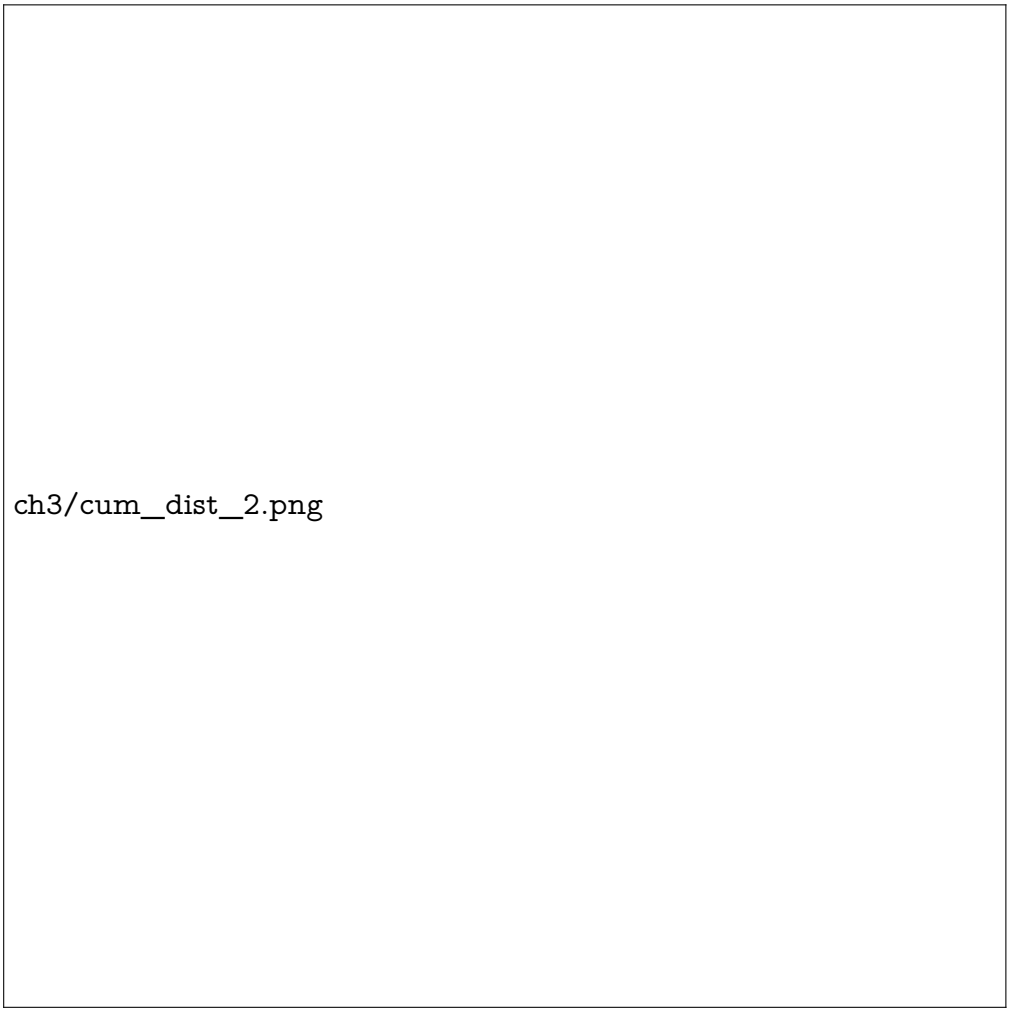


Figura 3.2: Frecuencia acumulada de los tokens únicos aplicando hasta el cuarto nivel de procesamiento. El eje horizontal es el acumulado de tokens únicos en orden decreciente de ocurrencia. Los puntos corresponden a los cuantiles 60 %, 80 %, 90 %, 95 % y 99 %.

Rehacer

En la figura 3.2 se presenta la distribución acumulada del vocabulario hasta el cuarto nivel de procesamiento, en donde se observa que la cola de distribución es bastante menos pesada que bajo el primer nivel de procesamiento, además, como se observa en la tabla 3.1 el tamaño del vocabulario se redujo a menos de un décimo del vocabulario obtenido bajo el primer nivel de procesamiento y es menos de un décimo del tamaño del corpus, por lo que bajo este nivel de procesamiento es posible desarrollar modelos con mayor fuerza estadística.

Rehacer

procesamiento	documentos	vocabulario	tokens
raw	49.015	79.327	2.030.980
ch	49.011	46.708	1.947.235
ch+s+l+f	47.993	4.106	508.987

Tabla 3.1: Estadísticas del corpus bajo distintos niveles de procesamientos, **raw**: sin procesamiento, **ch**: eliminación de símbolos de puntuación, correos electrónicos y tokens con números, **ch+s+l+f**: además incluye eliminación de stopwords (s), lemmatización (l) y eliminación de tokens con baja ocurrencia (f).

Rehacer

En la tabla 3.2 se muestra el detalle del vocabulario para cada una de las épocas tras procesar el corpus, de aquí se extrae que en promedio un 21.28 % del vocabulario se olvida de una época a otra y un 28.19 % es nuevo, es otras palabras, en promedio alrededor de un 50 % del vocabulario no es común entre tópicos de épocas adyacentes, esto justifica la necesidad de utilizar médidas de similitud que capturen la similitud entre palabras de épocas adyacentes ante la renovación que sufre el vocabulario en el tiempo.

Rehacer

época	**old_vocab**	**new_vocab**	**%old_vocab**	**%new_vocab**
2	1.919	1.986	23,35	26.84
3	1.986	2.092	22,61	27.95
4	2.092	2.414	18,21	33.60
5	2.414	2.629	19,80	28.71
6	2.629	2.666	22,44	23.85

Tabla 3.2: Evolución del vocabulario en el tiempo, **old_vocab**: corresponde al vocabulario del período $t-1$, **new_vocab**: corresponde al vocabulario del período t, **%old_vocab**: porcentaje de tokens del período $t-1$ que ya no están en el período t y **%new_vocab**: porcentaje de tokens del período t que no están en el período $t-1$.

Rehacer

3.3. Análisis cuantitativo de resultados

Rehacer

Al aplicar HDP de forma independiente en cada una de los épocas se obtuvo el siguiente número de tópicos [8, 10, 9, 8, 8, 9].

3.3.0.1. Distribución acumulada de los tópicos

En la figura 3.3 se muestra la distribución acumulada promedio de los tópicos, se tiene que en promedio un 8.54 % y 21.42 % del vocabulario se puede capturar un 80 % y 95 % respectivamente de la distribución acumulada de los tópicos, además, para un 99 % de los tópicos basta con un 37 % del vocabulario para capturar el 95 % de su distribución acumulada, por tanto, una representación incompleta de los tópicos usando las palabras más probables que capturan el 80 % de la distribución acumulada trae consigo una disminución importante en el tamaño del vocabulario.

Rehacer

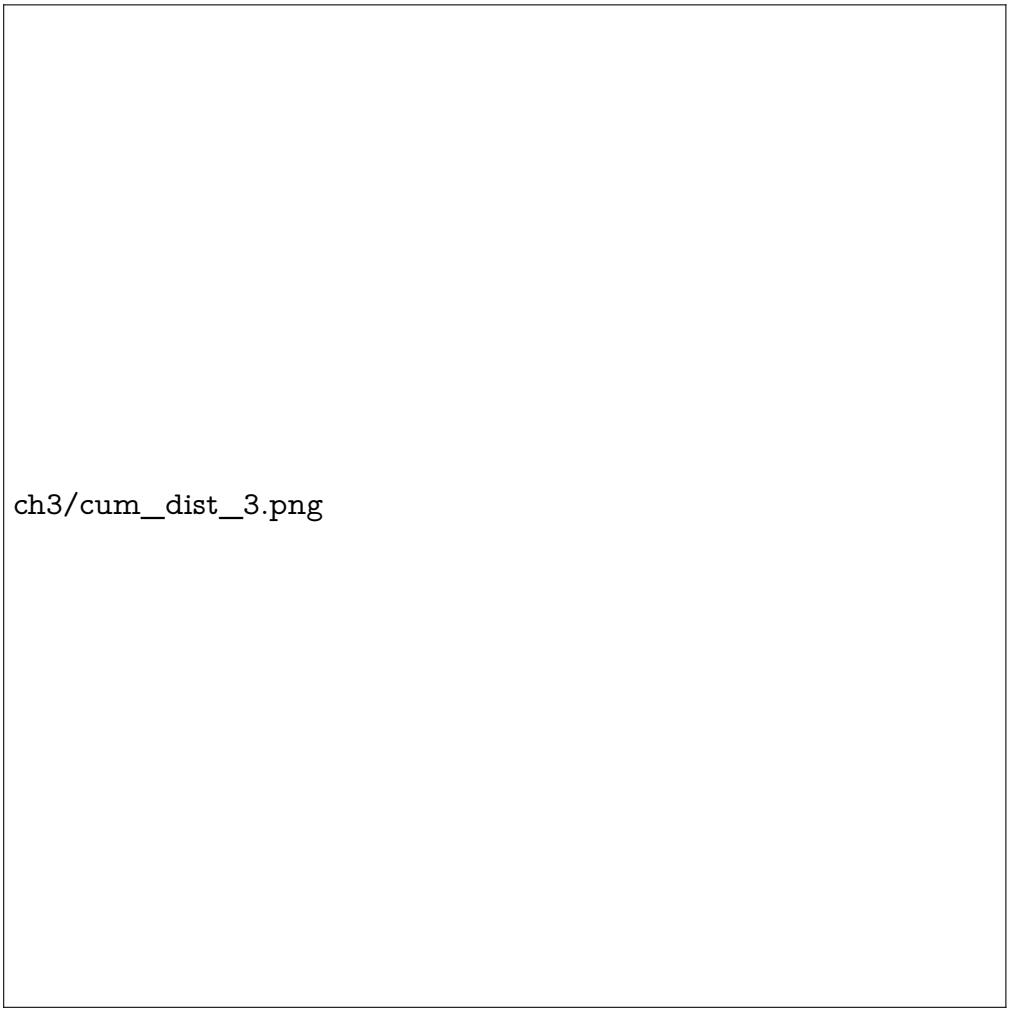


Figura 3.3: Distribución acumulada promedio de los tópicos en función del vocabulario. El punto (x,y) en el gráfico corresponde a la fracción x del vocabulario que explica la fracción y de la distribución acumulada del tópico. Los puntos corresponden a los cuantiles 60 %, 80 %, 90 %, 95 % y 99 %.

Rehacer

3.3.0.2. Construcción del grafo temporal

El modelo propuesto considera tres hiperparámetros:

- $q \in [0, 1]$: para el cálculo de WMD se utilizan las palabras más probables del tópico que explican un 100q % de la distribución acumulada del tópico. Este parámetro genera un nuevo tópico (se normaliza para sumar 1) con un vocabulario más reducido.
- $\lambda \in [0, 1]$: este parámetro pondera la probabilidad de la palabra dentro del tópico con su exclusividad. El nuevo tópico generado es normalizado para sumar 1.
- $\zeta \in [0, 1]$: punto operante de la cdf del grafo inicial, permite definir el cuantil que se usará como úmbral para eliminar arcos con similitud menor a este.

Marco teórico

Para entender de mejor manera la influencia de cada uno de estos parámetros se hizó un etiquetado de los arcos del grafo temporal, asignando un 1 a los arcos que deberían estar presente y 0 a los que no. Luego, se hizó una búsqueda a través de la siguiente grilla de parámetros, $\lambda \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$, $q \in \{0.2, 0.4, 0.6, 0.8, 0.9, 0.95\}$ y $\zeta \in \{0.05, 0.10, ..., 0.90, 0.95\}$.

Marco teórico y diseño de experimento

Como métrica de evaluación se propone *F-score*, definida por:

$$F - score = 2 \times \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.1}$$

, donde *recall* es la tasas de acierto sobre la clase positiva (presencia de un arco) y *precision* es la tasa de acierto de las predicciones sobre la clase positiva. Esta métrica permite balancear la acertividad con la precisión, así una configuración que no pode ningún arco tendra un *recall*=1, pero un bajo *precision* (notar que el númerador decrece más rápido que el denominador).

Usar métrica acorde al marco teórico

De la figura 3.4 se observa que *F-score* tiende a ser creciente en función de ζ, esto se debe a que menor ζ más falsos positivos (pues son más arcos los que sobreviven) empeorando así el *precision* y por consecuencia el *F-score*. Las configuraciones óptimas ocurren en su mayoría en $\zeta = 0.95$ con excepción de tres configuraciones de las treinta posibles de $q \times \lambda$, las cuales se dan en $\zeta = 0.9$ para los parámetros $q = 0.2$ con $\lambda \in \{0.8, 1\}$ y $q = 0.4$ con $\lambda = 0.2$, sin embargo, el valor óptimo alcanzado es bastante cercano al obtenido con $\zeta = 0.95$. En cuanto a λ se observa que no existen muchas diferencias entre $\lambda \in \{0.6, 0.8, 1.0\}$ a diferencia de $\lambda \in \{0.2, 0.4\}$ que suele estar significativamente por debajo de las otras curvas, además se observa una dominancia débil en λ, es decir, en el ζ óptimo dado un (q, λ) un λ mayor no es peor. En el caso del parámetro q se observa que para $q \geq 0.6$ el óptimo obtenido para $\lambda \geq 0.4$ es el mismo, en cambio para $q = 0.4$ esto se cumple para todo $\lambda \geq 0.6$ y con $q = 0.2$ para $\lambda \geq 0.8$.

Rehacer

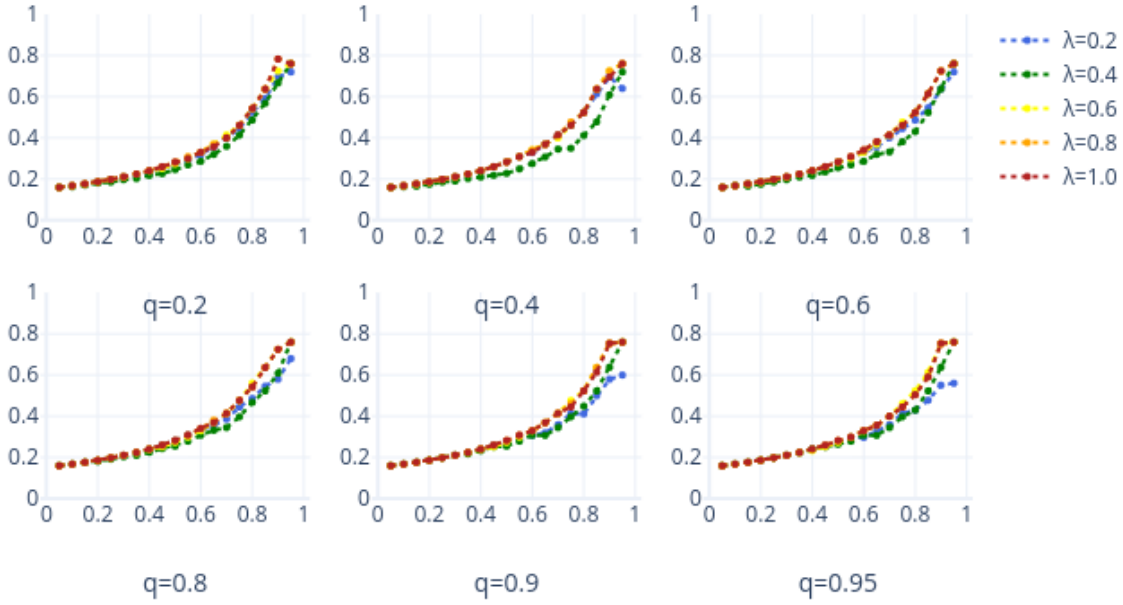


Figura 3.4: F-score (eje vertical) para diferentes configuraciones de los hiperparámetros q, ζ (eje horizontal) y λ.

Rehacer

De la tabla 3.3 se observa que la configuración óptima se alcanza con $q = 0.2$ con $\zeta = 0.9$, además esto ocurre tanto para $\lambda = 0.8$ como $\lambda = 1.0$, por lo que es escoje la configuración $(q, \zeta, \lambda) = (0.2, 0.9, 1.0)$ para construir el grafo temporal. En la figura 3.5 se observa la distribución acumulada de la similitud para el grafo completamente conectado, por lo que el para $zeta = 0.9$ el úmbral viene siendo 0.21.

Rehacer

q	**zeta**	**recall**	**precision**	**f-score**
0.2	0.9	0.87	0.71	0.78
0.4	0.95	0.61	1	0.76
0.6	0.95	0.61	1	0.76
0.8	0.95	0.61	1	0.76
0.9	0.95	0.61	1	0.76
0.95	0.95	0.61	1	0.76

Tabla 3.3: Configuración de ζ para cada q que máximiza el *F-score*.

Rehacer

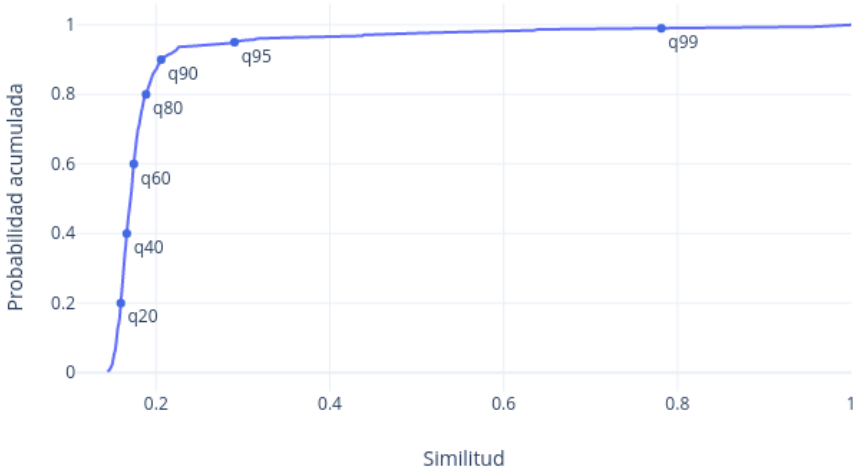


Figura 3.5: Estimación empírica de la función de distribución acumulada (cdf) de la similitud entre tópicos correspondiente al grafo temporal completamente conectado para la configuración óptima $(q, \lambda) = (0.2, 1.0)$.

Rehacer

En la figura 3.6 se observa que la configuración óptima es en promedio 184 veces más eficiente que $q = 0.95$, esto se debe a que $q = 0.2$ es un 0.3 % del vocabulario (6 palabras en promedio) y $q = 0.95$ alrededor de un 21 % (488 palabras en promedio).

Rehacer

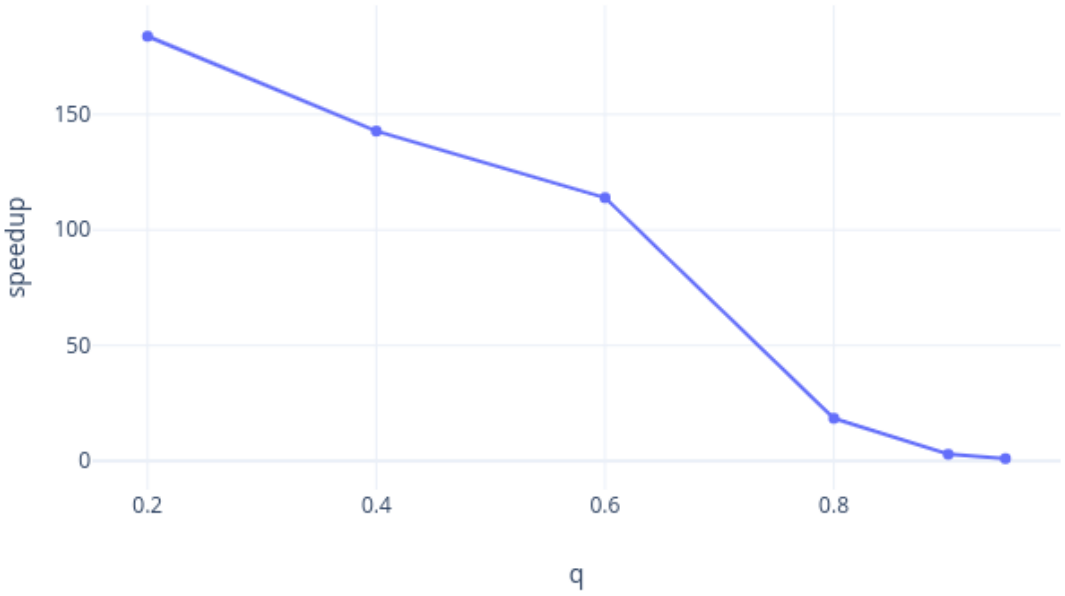


Figura 3.6: Speedup promedio de la construcción del grafo en función de q. El speedup 1 equivale al tiempo más lento el cual está asciado a $q = 0.95$ que es el valor de q más grande y por ende con menor reducción de vocabulario de los tópicos a la hora de computar WMD.

Rehacer

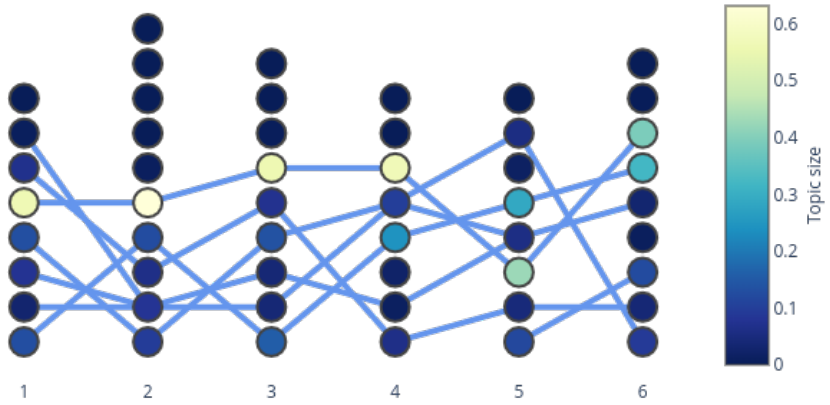


Figura 3.7: Grafo temporal etiquetado.

Rehacer

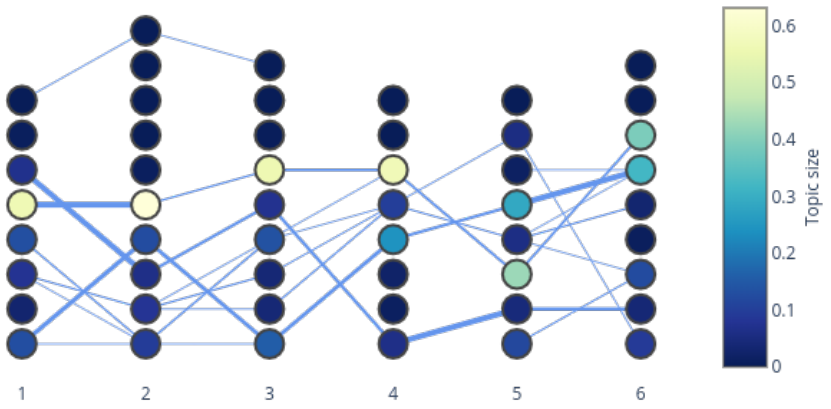


Figura 3.8: Grafo temporal obtenido a partir de la configuración óptima de parámetros $(q, \lambda, \zeta) = (0.2, 1.0, 0.9)$.

Rehacer

3.4. Análisis cualitativo de resultados

Análisis cualitativo de tópicos

Capítulo 4

Conclusiones y trabajo futuro

Conclusiones

Otras aplicaciones

Trabajo futuro

Bibliografía

Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273, 2003.

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

Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in neural information processing systems*, pages 1385–1392, 2005.

Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. Exploring topic coherence over many models and many topics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 952–961. Association for Computational Linguistics, 2012.

Xuerui Wang and Andrew McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433, 2006.

Amr Ahmed and Eric P Xing. Timeline: A dynamic hierarchical dirichlet process model for recovering birth/death and evolution of topics in text stream. *arXiv preprint arXiv:1203.3463*, 2012.

Andrew T Wilson and David G Robinson. Tracking topic birth and death in lda. *Sandia National Laboratories*, 2011.

Adham Beykikhoshk, Ognjen Arandjelović, Dinh Phung, and Svetha Venkatesh. Discovering topic structures of a temporally evolving document corpus. *Knowledge and Information Systems*, 55(3):599–632, 2018.

Thomas Minka. Estimating a dirichlet distribution, 2000.

Thomas S Ferguson. A bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.

Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650, 1994.

Erik Blaine Sudderth. *Graphical models for visual object recognition and tracking*. PhD thesis, Massachusetts Institute of Technology, 2006.

David J Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII—1983*, pages 1–198. Springer, 1985.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3): 103–134, 2000.

Chong Wang and David Blei. HDP: Hierarchical dirichlet process C++, 2010. URL https://github.com/blei-lab/hdp.

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

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Dominik Maria Endres and Johannes E Schindelin. A new metric for probability distributions. *IEEE Transactions on Information theory*, 49(7):1858–1860, 2003.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966, 2015.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Vlad Niculae. Word mover's distance in python, 2015. URL http://vene.ro/blog/word-movers-distance-in-python.html.

Gary Doran. PyEMD: Earth mover's distance for Python, 2014. URL https://github.com/garydoranjr/pyemd.

Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *2009 IEEE 12th International Conference on Computer Vision*, pages 460–467. IEEE, 2009.

José Cañete. Fasttext embeddings from SUC. https://github.com/BotCenter/spanishWordEmbeddings, 2019a.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

José Cañete. Spanish Unannotated Corpora, 2019b. URL https://github.com/josecannete/spanish-corpora.

Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *Introduction to information retrieval*. Cambridge university press, 2008.

Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

Juan Cao, Tian Xia, Jintao Li, Yongdong Zhang, and Sheng Tang. A density-based method for adaptive lda model selection. *Neurocomputing*, 72(7-9):1775–1781, 2009.

Romain Deveaud, Eric SanJuan, and Patrice Bellot. Accurate and effective latent concept modeling for ad hoc information retrieval. *Document numérique*, 17(1):61–84, 2014.

Rajkumar Arun, Venkatasubramaniyan Suresh, CE Veni Madhavan, and MN Narasimha Murthy. On finding the natural number of topics with latent dirichlet allocation: Some observations. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 391–402. Springer, 2010.

Wen Zhang, Yangbo Cui, and Taketoshi Yoshida. En-lda: An novel approach to automatic bug report assignment with entropy optimized latent dirichlet allocation. *Entropy*, 19(5): 173, 2017.

George Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of machine learning research*, 3(Mar):1289–1305, 2003.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. .°'Reilly Media, Inc.", 2009.

Martin F Porter et al. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.