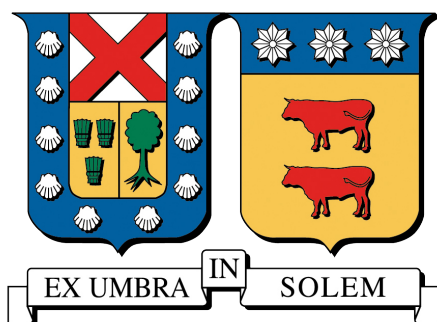


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE INFORMÁTICA

SANTIAGO - CHILE



MODELAMIENTO DE TÓPICOS INCREMENTALES CON DESCRIPTORES SENSIBLES AL TIEMPO

DANIEL IGNACIO RIVERA TOLOZA

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL INFORMATICO

PROFESOR GUÍA : MARCELO MENDOZA
PROFESOR CORREFERENTE : RICARDO ÑANCULEF

OCTUBRE - 2017

Índice general

Agradecimientos	1
Glosario	2
Resumen	3
Abstract	4
1. Introducción	5
1.1. Objetivos Principales	7
1.2. Objetivos Específicos	7
2. Marco Teórico	8
2.1. Term frequency – Inverse document frequency	8
2.2. Probabilistic Latent Semantic Indexing	9
2.3. Latent Dirichlet Allocation	9
2.3.1. Estimación parámetros LDA	11
2.4. Dynamic Topic Models	16
2.5. State Space Model	18
2.6. Kalman Filter	20
3. Descripción del problema	21
3.1. Entrenamiento Nuevos Datos	21
3.2. Representación State Space Models	22
3.3. Dynamic Topic Models Incremental	23
3.3.1. Actualización Corpus	24

3.3.2. Modificación Matrices LdaSeq	26
3.3.3. Inicialización parámetros ldaseq incremental	28
3.3.4. Algoritmo esperanza-maximización incremental (propuesta)	29
3.4. Derivación de Algoritmo Variacional (original)	29
3.5. Derivación de Algoritmo Variacional Incremental (propuesta)	31
3.6. Evaluación Coherencia de Tópicos	33
3.7. Visualización de Modelos de Tópicos	35
4. Resultados	38
4.1. Descripción Corpus	38
4.2. Dynamic Topic Model Incremental	39
4.3. Análisis Varianza DTM	42
4.4. Visualización Modelos	47
5. Conclusiones	51
6. Trabajo Futuro	53
6.1. Optimización algoritmo	53
6.2. Determinación Varianza	53
6.3. Aparición y/o Desaparación de Tópicos	54
Bibliografía	55

Agradecimientos

En primer lugar deseo agradecer a mi familia y todo el apoyo incondicional que me han brindado durante todos estos años de carrera, por estar presentes en mis éxitos y mis derrotas. Sin ellos este viaje hubiera sido casi imposible. Este trabajo va dedicado a ellos

Agradezco también a todos los profesores que estuvieron presentes en mi formación académica superior a través de los años, especialmente al profesor Marcelo Mendoza, y por la oportunidad de trabajar con el en un proyecto desafiante e interesante . Agradezco la confianza depositada en mi trabajo y todas las enseñanzas y herramientas de investigación aprendidas durante el desarrollo de esta memoria.

Finalmente , quiero agradecer a mis compañeros de Universidad por todos estos años de apoyo mutuo en los estudios, desafíos y los conocimientos compartidos. Especialmente quiero agradecer a Ignacio Espinoza y René Mujica, amigos de generación que han estado desde el primer año de esta difícil carrera.

Glosario

- **corpus** : Una gran colección de documentos
- **tópico** : Conjunto de términos relacionados capaces de describir el contenido de un documento determinado.
- **DTM** : Dynamic Topic Model
- **LDA** :Latent Dirichlet Allocation
- **pLSI** : Probabilistic Latent Semantic Indexing
- **tf-idf** : term frequency–inverse document frequency
- **ELBO** : evidence lower bound
- **VEM** : Variational Expectation Maximization
- **KL divergence** : Kullback Leibler divergence
- **SSLM** : State Space Language Model

Resumen

Cada año la cantidad de información que se sube a Internet aumenta considerablemente, siendo gran parte de estos datos creados por usuarios en diversas redes sociales o foros de conversación. Ante esto, surge la necesidad de trabajar con técnicas de clasificación y/o resumen de información que puedan adaptarse con el paso del tiempo, cuando nuevos datos se encuentren disponibles para su posterior entrenamiento .

El algoritmo Dynamic Topic Models (DTM) es un modelo generativo de texto que permite analizar la evolución de tópicos no observados en una gran colección de documentos a través del tiempo. Si bien DTM nos otorga modelos cuantitativos y predictivos sobre un corpus secuencial, este algoritmo no está pensado para actualizar modelos si se desea incorporar una nueva colección de documentos pertenecientes a un nuevo intervalo de tiempo, por lo que es necesario volver a entrenar el modelo completo, siendo este proceso costoso en tiempo y recursos.

En el siguiente trabajo se propone generar una extensión del algoritmo DTM, de modo que este permita actualizar modelos previamente entrenados con un nuevo set de documentos pertenecientes a un intervalo de tiempo subsiguiente, mejorando así el rendimiento y el análisis *online* de nuevos documentos conforme estos se van creando. A su vez, se busca analizar la identificación de nuevos términos y la evolución o desaparición de palabras, modificando parámetros del algoritmo responsables de la distribución de términos relevantes por cada tópico, tales como la varianza de cambio.

Abstract

Every year, the amount of data that is uploaded by millions of people to the Internet increases considerably, being the majority of the data created by users via various forums or social media platforms. Therefore, there is a need to work with summarization techniques that can adapt over time, when new data are available for training.

The Dynamic Topic Model (DTM) algorithm is a text generative model that can be used to analyze the evolution of unobserved topics of a large collection of documents over time. While the algorithm can give quantitative, predictive models of a sequential corpus, the algorithm is not intended to update the model if it is desired to incorporate a new collection of documents belonging to a new time interval, so it is necessary to re-train the model, being this process expensive in terms of computational time and memory management.

In this work we propose an extension for the DTM algorithm, favoring the update of previously trained models with a new set of documents belonging to a subsequent time interval, improving efficiency for *online* analysis of new documents as they are being created. At the same time, we seek to analyze the identification of new terms and the evolution or disappearance of words, modifying algorithmic parameters in charge for the distribution of terms for each topic, such as the change variance.

Capítulo 1

Introducción

Cada año, la cantidad de datos que se sube a Internet aumenta considerablemente, siendo la mayoría de estos datos creados por usuarios a través de diversas redes sociales. Esto nos da acceso a una fuente inagotable de información proveniente de diversos sitios, donde cada cual puede distribuirse en múltiples temas o tópicos. Para un usuario específico, no todos estos tópicos pueden tener información relevante, por lo que es necesario realizar búsquedas a través de documentos, palabras, o tópicos, los cuales pueden llegar a consumir una gran cantidad de recursos acorde al volumen de datos disponible.

Actualmente existen herramientas que nos facilitan la organización, búsqueda e indexación de grandes colecciones de documentos. Una de estas herramientas son los *Topic Models*, que nos permiten encontrar patrones de palabras en colecciones usando modelos jerárquicos probabilísticos, descubriendo así los tópicos subyacentes que ocurren en una colección determinada. El modelo *Latent Dirichlet Allocation* (LDA) es un modelo generativo usado para identificar tópicos en colecciones discretas de datos tales como corpus de documentos. Para una observación de palabras en un conjunto de documentos, se postula que cada documento es producido desde una distribución multinomial sobre un número definido de tópicos, y que la aparición de cada palabra puede atribuirse a la probabilidad de ocurrencia de la palabra en los tópicos de dicho documento. La distribución de tópicos nos otorga una representación explícita de un documento, el cual permite al reducir la dimensionalidad de este, realizar operaciones de búsqueda, clasificación y organización con mayor facilidad.

Una de las propiedades que el modelo LDA no logra capturar es la evolución de tópicos a través del tiempo. Por ejemplo, si el contenido de un corpus son reclamos de clientes ante un servicio determinado, el contenido de estos tópicos puede diferir dependiendo si es un período electoral o si es durante un periodo de ofertas de sitios web tales como Black Friday. Los temas en una colección de documentos evolucionan a través del tiempo, y también el uso términos, por lo que es necesario modelar explícitamente la dinámica de estos tópicos para capturar tendencias, surgimiento de nuevos términos o desaparición de otros.

El modelo generativo *Dynamic Topic Models* (DTM) es una extensión del modelo LDA que puede ser usado para capturar la evolución de tópicos a través del tiempo. El modelo desarrollado por David M. Blei y John D. Lafferty [3] implementa una distribución Gaussiana sobre los intervalos de tiempo en los parámetros del modelo, de modo que cada tópico en la secuencia evoluciona *suavemente* desde el intervalo de tiempo previo.

Si bien el modelo DTM nos permite analizar tópicos en una colección de documentos y la evolución de estos a través del tiempo, el algoritmo no está pensado para actualizar el modelo si se desea incorporar una nueva colección de documentos (model updating). Si se desea modelar un nuevo intervalo de tiempo, es necesario reentrenar el modelo completo, siendo este proceso costoso en términos de tiempo. Otro fenómeno que el modelo no logra capturar es la medida de dispersión entre intervalos de tiempo, por lo que no se logra identificar si en la evolución entre intervalos, la transformación de estos es suave o si los términos cambian drásticamente. Esto se debe a que DTM define un parámetro con un valor constante (estático) que determina la dinámica entre intervalos.

En este trabajo se propone extender el algoritmo DTM, de modo que este pueda incorporar en un modelo ya entrenado, nuevos documentos correspondientes a nuevos intervalos de tiempos. La idea es usar una propiedad de los *State Space Models* que señalan que la evolución de un tópico solo es dependiente del tópico precedente (no se modela la correlación entre los tópicos) por lo tanto solo se necesitan los parámetros del intervalo de tiempo anterior para modelar el nuevo set de documentos.

En conjunto con la propuesta presentada, se busca también analizar la evolución de los tópicos y la identificación de términos descriptivos emergentes en conjunto con la dependencia de esta evolución al parámetro que regula la medida de dispersión entre intervalos de tiempo, comparando distintos modelos entrenados con distintos valores de varianza de cambio.

Los objetivos principales y específicos de este trabajo se declaran a continuación:

1.1. Objetivos Principales

El objetivo principal de esta memoria consiste en crear una versión incremental del modelo generativo Dynamic Topic Model que permita el entrenamiento de nuevos documentos pertenecientes a un nuevo intervalo de tiempo, manteniendo el resto de los intervalos de tiempo entrenados fijos.

1.2. Objetivos Específicos

- Analizar y modificar el algoritmo *ldaseq* de modelamiento dinámico de tópicos de la librería de python *Gensim*, con el objetivo de poder entrenar nuevos documentos pertenecientes a un nuevo intervalo de tiempo, usando los parámetros pertenecientes al intervalo de tiempo precedente.
- Observar la aparición de nuevos términos y/o desaparición de términos relevantes en la evolución de los tópicos modelados al modificar la varianza marginal usada en la fase de inferencia (usando filtro de Kalman).
- Comparar modelos usando el modelo DTM con y sin versión incremental, usando como métrica de comparación "Topic Coherence Measures"
- Visualizar y analizar modelos entrenados usando librerías interactivas de visualización de modelos de tópicos.

Capítulo 2

Marco Teórico

2.1. Term frequency – Inverse document frequency

En el área de recuperación de información (IR), *Term frequency – Inverse document frequency* (tf-idf) es una función de pesos (*weighting function*) que indica que tan relevante es una palabra para un documento perteneciente a un corpus [13]. En este esquema, un vocabulario de palabras es usado y para cada documento en el corpus, se genera un recuento del número de ocurrencias de cada palabra. Este recuento es comparado luego con una lista inversa de frecuencia de documentos, en donde se mide el número de ocurrencias de cada palabra en el corpus, obteniendo así una matriz *términos-por-documento*.

El factor **Term Frequency**(tf) determina el peso de un término t_i en un documento d_j , basado en la frecuencia relativa de este. Siendo $f_{i,j}$ la frecuencia del término t_i en el documento d_j y $\max f_{i,j}$ la frecuencia del término más frecuente en d_j , se define $tf_{i,j}$ como:

$$tf_{i,j} = \frac{f_{i,j}}{\max f_{i,j}} \quad (2.1)$$

Uno de los problemas de este enfoque es que no considera la distribución de todos los términos en la colección de documentos. Si un término aparece frecuentemente en todos los documentos, este posee una menor capacidad descriptiva, como ejemplo el término *el, la, y*, etc. El factor **Inverse Term Frequency**(idf) se incorpora al factor *tf* con el objetivo de disminuir el peso de un término si este ocurre con mucha frecuencia en la colección de documentos y aumentar el peso si este ocurre

con poca frecuencia en la colección. Siendo N el número total de documentos en la colección y n_i la cantidad de documentos donde aparece el término i , se define idf como:

$$idf_{t(i)} = \log \frac{N}{n_i} \quad (2.2)$$

Finalmente, el modelo $tf-idf$ para un término t se define como:

$$tfidf_{t,d} = tf_{i,j} \times idf_{t(i)} \quad (2.3)$$

Las ventajas de este modelo es que es posible identificar fácilmente un set de palabras que son discriminantes para cada documento (que mejor lo definen) pero otorgan poca información de las relaciones estructurales entre documentos (como los documentos se relacionan entre sí). Otra desventaja de este enfoque es que entrega poca información analítica de la distribución interna de cada documento (no logra identificar la posición cada término en el documento, o la co-ocurrencia de palabras en otros documentos).

2.2. Probabilistic Latent Semantic Indexing

El modelo $pLSI$, también conocido como *aspect model*, es una extensión de $textit{LSI}$ [8], propuesto por Hoffmann (1999) el cual genera cada palabra en un documento como una muestra de un modelo de mezcla, donde cada componente es una variable multinomial aleatoria que puede ser vista como una representación de **tópicos**. Así, cada palabra es generada por un solo tópico, y diferentes palabras en un documento pueden ser generadas por diferentes tópicos.[11]

Si bien este modelo constituye un aporte a lo que es un modelo probabilístico de términos, este es un modelo incompleto, dado que cada documento es representado como una lista de números que determinan la distribución de tópicos, pero no existe un modelo probabilístico generativo a priori de estos números.

2.3. Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) consiste en una técnica de Bayesiana de inferencia no paramétrica en donde un conjunto de observaciones son explicadas o definidas por un conjunto de

variables no observadas o *latente*, interpretando así el porqué ciertos elementos de los datos son similares. En este modelo, cada documento de un corpus es representado como una distribución de probabilidad sobre un número finito de tópicos, mientras que estos tópicos son representados como una distribución de probabilidad sobre un vocabulario.[4]

Un ejemplo práctico de los resultados de LDA es el siguiente. Suponga que tenemos las siguientes sentencias:

1. Debo estudiar Matemática y Física para la siguiente semana.
2. El certamen de Física Cuántica estuvo mas difícil de lo que creía.
3. Prefiero comer Manzanas sobre Naranjas, pero nada reemplaza los Chocolates.
4. Antes del Certamen de Matemática, debo comer una barra de chocolate para tener energías.

El algoritmo LDA permite descubrir tópicos que estas frases pueden contener. Dado el supuesto que tenemos 2 tópicos, el algoritmo generará la siguiente distribución:

Distribución Texto 1 : 100 % Tópico A

Distribución Texto 2 : 100 % Tópico A

Distribución Texto 3 : 100 % Tópico B

Distribución Texto 4 : 50 % Tópico A 50 % Tópico B

Tópico A : 30 % matemática, 15 % física, 10 % ciencias, 10 % estadística, ... (con las primeras palabras se puede inferir que el tópico habla sobre materias de estudio, o campos de la ciencia)

Tópico B : 20 % chocolate, 15 % manzana, 25 % brocoli, 10 % naranja, ... (con las primeras palabras se puede inferir que el tópico habla sobre comida)

El modelo generativo de LDA para cada documento es el siguiente:

1. Escoger el número de palabras N en cada documento $\sim \text{Poisson}(\zeta)$
2. Escoger la distribución multinomial de tópicos $\theta \sim \text{Dirichlet}(\alpha)$
3. Para cada palabra w_n en el conjunto N :

- a) Escoger un tópico $z_m \sim \text{Multinomial}(\theta)$
- b) Escoger una palabra w_n de la probabilidad $p(w_n|z_n, \beta)$, una probabilidad multinomial condicionada por el tópico z_n

Los hiperparámetros α y β condicionan la distribución de tópicos por documentos y de palabras por tópicos, respectivamente. Un valor alto de α indica que todos los documentos contienen probablemente una distribución de probabilidad de la mayoría de los tópicos, mientras que un valor bajo indica que los documentos está representado por solamente una cantidad pequeña de tópicos. El mismo criterio es aplicado en β

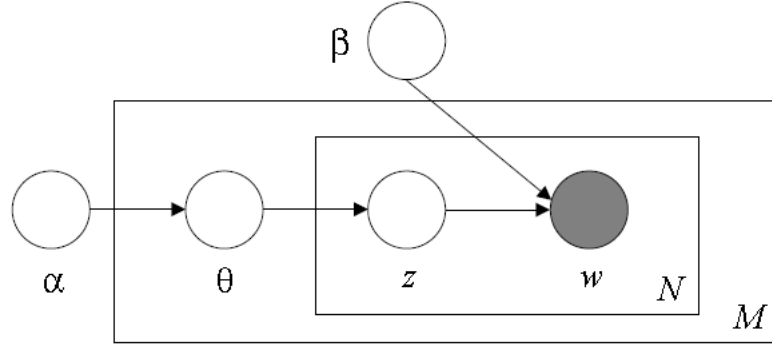


Figura 2.1: Representación gráfica del modelo generado por LDA, en donde cada rectángulo representa un replicado. El rectángulo exterior representa los documentos, mientras que el rectángulo interior representa la selección de tópicos y palabras en un documento

2.3.1. Estimación parámetros LDA

Resolver el problema de inferencia corresponde a calcular la distribución *posterior*, es decir, la distribución más probable respecto a una serie de observaciones (evidencia) de las variables latentes dado un documento determinado, es decir $\mathbf{z}$ y θ . Lamentablemente, esta distribución es imposible de calcular dado el acoplamiento de las variables θ, β al momento de sumar los tópicos latentes, por lo que no es posible obtener una inferencia exacta.

Inferencia Variacional

Una propuesta para la determinación de dichas variables es usar **inferencia variacional**. Los métodos de inferencia variacional son una familia de técnicas usadas frecuentemente en modelos estadísticos complejos en donde existen variables observables y variables latentes, por lo que la determinación de la distribución a posteriori se vuelve en muchos casos computacionalmente intratable.[1]

Si se tiene la siguiente configuración:

$\vec{x} : x_{1:n}$ set de observaciones.

$\vec{z} : z_{1:m}$ set de variables latentes.

α : set de parámetros fijos.

- Nuestro objetivo es encontrar la distribución a posteriori de:

$$p(z \mid x, \alpha) = \frac{p(z, x \mid \alpha)}{\int_z p(z, x \mid \alpha)} \quad (2.4)$$

Una propuesta para determinar esta distribución intratable, es crear una **distribución variacional** q sobre todas las variables latentes, parametrizadas por variables variacionales μ . de modo que el objetivo es encontrar la configuración de parámetros μ que ajusten la distribución variacional q a la distribución posterior.

$$q(z_{1:m} \mid \mu) \quad (2.5)$$

La idea de *cercanía* entre dos distribuciones de probabilidad se observa mediante el uso de la divergencia de **Kullback-Leibler**(KL). La divergencia KL es una medida no simétrica entre dos distribuciones denominadas P y Q , en donde en general, P representa la verdadera distribución de los datos u observaciones, mientras que Q representa en general el modelo o teoría que busca aproximarse a la distribución P , de esta forma, $D_{KL}(p \parallel Q)$ busca medir la cantidad de información perdida cuando la distribución Q es usada para aproximar P (entropía relativa).

$$KL(p \parallel q) \equiv \mathbf{E}[\log \frac{q(z)}{\int_z p(Z \mid x)}] \quad (2.6)$$

El valor de $D_{KL}(p \parallel Q)$ es siempre no negativo e igual a cero solo si $p = q$ (en este caso estaríamos en un caso común de Algoritmo esperanza-maximización). Dado que no es posible minimizar D_{KL} exactamente, buscamos minimizar una función que sea igual hacia arriba a una constante. Este método se llama **evidence lower bound** o ELBO

$$\begin{aligned}
 \log p(x) &= \log \int_x p(x, z) \\
 &= \log \int_x p(x, z) \frac{q(z)}{q(z)} \\
 &= \log(\mathbf{E}_q \frac{p(x, Z)}{q(Z)}) \\
 &= \log(\mathbf{E}_q [\frac{p(x, Z)}{q(Z)}]) \\
 &\geq (\mathbf{E}_q[\log p(x, Z)] - \mathbf{E}_q[\log q(Z)])
 \end{aligned} \tag{2.7}$$

Esta ecuación representa el **evidence lower bound**. La idea es escoger una familia de distribuciones variacionales con el objetivo de lograr computar la expectación de estas. Luego, maximizamos ELBO para encontrar los parámetros que nos otorgan una cota para la probabilidad marginal de la variable x . El segundo término en la ecuación corresponde a la entropía, otra variable usada en el campo de teoría de la información.

Volviendo a la divergencia KL, observamos que:

$$\begin{aligned}
 D_{KL}(q(z) \parallel p(z \mid x)) &= \mathbf{E}_q[\log \frac{q(Z)}{p(Z \mid x)}] \\
 &= \mathbf{E}_q[\log q(Z)] - \mathbf{E}_q[\log p(Z \mid x)] \\
 &= \mathbf{E}_q[\log q(Z)] - \mathbf{E}_q[\log p(Z, x)] + \log p(x) \\
 &= -\mathbf{E}_p[\log q(Z, x)] - \mathbf{E}_q[\log q(Z)] + \log p(x)
 \end{aligned} \tag{2.8}$$

Este resultado corresponde al ELBO negativo mas la probabilidad marginal logarítmica de la variable x . Dado que $\log p(x)$ no depende de q , se concluye que el proceso de minimizar la divergencia KL es el mismo proceso de maximizar ELBO.

Para determinar la distribución q , es necesario considerar que la complejidad de la familia de distribuciones de Q escogida determina la complejidad del proceso de optimización; es más complejo computacionalmente optimizar usando una familia de distribuciones compleja en comparación a familias mas simples.

La familia de distribuciones denominadas *mean-field variational family* tiene la propiedad de que las variables latentes son independientes entre sí y cada una esta condicionada por un factor distinto en la densidad variacional:

$$q(z_1, z_2, \dots, z_m) = \prod_{j=1}^m q(z_j) \quad (2.9)$$

Cada variable latente z_j es condicionada por su propio factor variacional, la función de densidad $q(j)$. En el proceso de optimización, se escogen los factores variacionales que busquen maximizar el ELBO de la ecuación.

En resumen, los pasos a seguir para determinar los parámetros mediante inferencia variacional son los siguientes:

1. Escoger familia de distribuciones q
2. Determinar ELBO
3. Usar coordinate ascent para cada q_i
4. Repetir hasta cumplir criterio de convergencia.

Volviendo al caso de LDA, dado un corpus $D = \{w_1, w_2, \dots, w_M\}$, deseamos encontrar los parámetros α y β que maximicen de verosimilitud marginal de los datos:

$$l(\alpha, \beta) = \sum_{d_1}^M \log p(w_d | \alpha, \beta) \quad (2.10)$$

Dado que $p(w | \alpha, \beta)$ no puede ser calculado, se propone usar *Variational Expectation Maximization* (VEM) para obtener una estimación de la máxima verosimilitud. Este algoritmo ejecuta iterativamente un *E-step* y luego un *M-step* que actualiza los parámetros del modelo usando los resultados de la inferencia. Se implementan métodos de inferencia Bayesiana en el *E-step* para obtener una aproximación.

Otro método propuesto para realizar la inferencia de los parámetros de LDA es usar muestreo de Gibbs o **Gibbs Sampling**[14]. Gibbs Sampling es una método de cadena de Markov de Monte Carlo (MCMC) usado para realizar inferencias, asumiendo una distribución a priori de Dirichlet β para el set de tópicos $\Phi = \{\phi_k\}$, este set puede ser integrado y extraído de la ecuación usando la conjugación de Dirichlet multinomial. La distribución a posteriori $P(Z|W)$ puede estimarse usando el algoritmo *Collapsed Gibbs Sampling*, el cual en cada iteración, actualiza cada tópico $z_{d,i} \in Z$ realizando un muestreo a toda la distribución:

$$p(z_{d,i} = k | Z_{-(d,i)}, w_{d,i}=v, W_{(d,i)}) \propto (C_{d,k}^{doc} + \alpha) \frac{C_{v,k}^{word} + \beta}{\sum_{v'} C_{v',k}^{word} + V\beta} \quad (2.11)$$

En esta ecuación, $k \in [1, K]$ es un tópico, $v \in [1, V]$ es una palabra en el vocabulario, $w_{d,i}$ corresponde a la i -ésima palabra en el documento d y $z_{d,i}$ es la asignación de tópico a la palabra $w_{d,i}$. $W_{-(d,i)}$ es la palabra en el set de entrenamiento con $w_{d,i}$ excluido, y $Z_{-(d,i)}$ es la asignación de tópico correspondiente a $W_{-(d,i)}$. El parámetro $C_{v,k}^{word}$ denota el número de veces que la palabra v es asignada al tópico k sin incluir los valores actuales de las instancias $w_{d,i}$ y $z_{d,i}$, mientras que $C_{d,k}^{doc}$ corresponde al número de veces que el tópico k a ocurrido en el documento d sin incluir $w_{d,i}$ y $z_{d,i}$. Cada vez que $z_{d,i}$ es asignado en un muestreo, las matrices C^{word} y C^{doc} son actualizadas. Después de una cierta cantidad de iteraciones considerable en la cadena de Markov, los parámetros $\Theta = \{\theta_d\}_{d=1}^D$ y $\Phi = \{\phi_k\}_{k=1}^K$ pueden estimarse con las siguientes ecuaciones.

$$\theta_{d,k} = \frac{C_{d,k}^{doc} + \alpha}{\sum_{k'=1}^K C_{d,k'}^{doc} + K\alpha} \quad (2.12)$$

$$\phi_{v,k} = \frac{C_{v,k}^{word} + \beta}{\sum_{v'=1}^V C_{v',k}^{word} + V\beta} \quad (2.13)$$

Griffiths y Steyvers, los investigadores que usaron por primera vez el método de Gibbs Sampling para inferencia en LDA [9], realizaron un estudio empírico para comparar el rendimiento de VEM y Gibbs sampling, llegando a resultados que demuestran que Gibbs Sampling converge en menos iteraciones que VEM o EP (Expectation Propagation).

2.4. Dynamic Topic Models

Si bien considerar que las palabras dentro de un modelo siguen un modelo de bolsa de palabras, o *Bag of Words*, esta suposición es una simplificación que es coherente solo con el objetivo de identificar patrones semánticos en un documento, pero no es adecuada para la mayoría de las colecciones considerar que la probabilidades de los documentos sea intercambiable (que el orden no sea de importancia). Por ejemplo, si se está analizando una colección de artículos científicos respecto a Inteligencia Artificial, algunos conceptos tales como “Deep Learning” comienzan a ser mas relevante dentro de los últimos años, mientras que artículos respecto a “Lógica Difusa” comienzan a tener mas presencia durante los años 60'. Los tópicos en una colección de documentos evolucionan en el tiempo, y es de interés modelar explícitamente la dinámica de estos tópicos latentes.

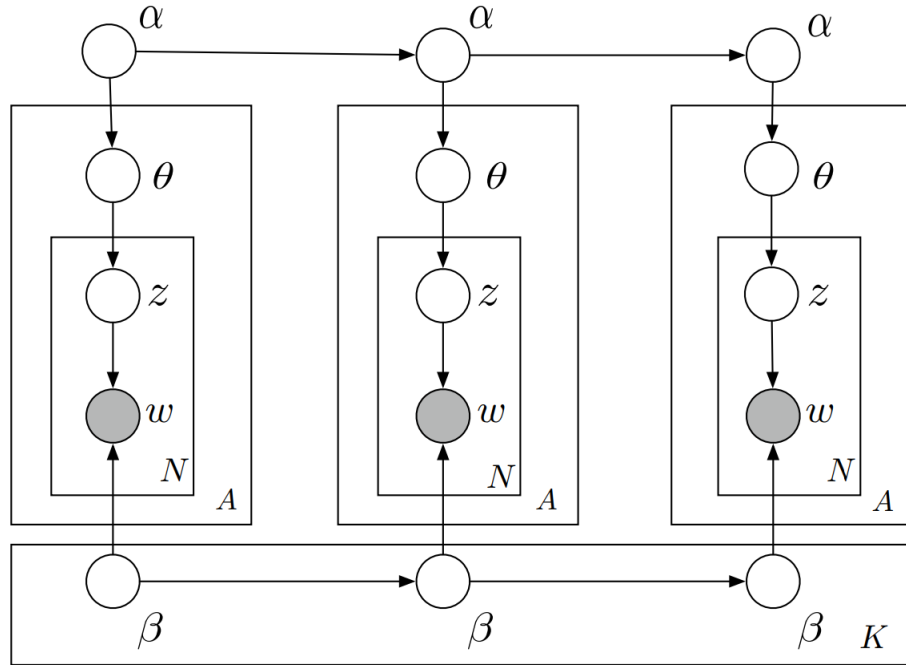


Figura 2.2: Representación del modelo DTM

El modelo generativo **Dynamic Topic Models**(DTM) permite analizar la evolución de tópicos latentes sobre una colección de documentos ordenados secuencialmente[3]. El enfoque consiste en usar una **Representación de Espacio de Estados** o *state space models* (ssm) sobre los parámetros de la distribución que representa los tópicos, y también sobre los parámetros de la distribución

normal usada para el modelamiento de la proporción documento-tópico (específica para cada documento).

Al enlazar los parámetros de cada tópico $\beta_{t,k}$ en un modelo de espacio de estados, el cual evoluciona con ruido Gaussiano, podemos obtener los parámetros en un tiempo determinado:

$$\beta_{t,k} \mid \beta_{t-1,k} \sim \mathcal{N}(\beta_{t-1,k}, \sigma^2 I) \quad (2.14)$$

Por otro lado, la proporción documento-tópico relativo a cada documento específico, el cual era extraído de la distribución de Dirichlet en el modelo LDA, usa una distribución normal con media α para representar incertidumbre sobre las proporciones.

$$\alpha_{t,k} \mid \alpha_{t-1,k} \sim \mathcal{N}(\alpha_{t-1,k}, \sigma^2 I) \quad (2.15)$$

Al enlazar un conjunto tópicos y distribución documento-tópico, hemos vinculado una colección de modelos de tópicos, en donde los tópicos pertenecientes al intervalo de tiempo t evolucionan suavemente de los tópicos en el tiempo $t - 1$. El proceso generativo para un intervalo t de un corpus secuencial es el siguiente:

1. Extraer tópicos $\beta_t \mid \beta_{t-1} \sim \mathcal{N}(\beta_{t-1}, \sigma^2 I)$
2. Extraer $\alpha_t \mid \alpha_{t-1} \sim \mathcal{N}(\alpha_{t-1}, \sigma^2 I)$
3. Por cada documento:
 - a) Extraer $\eta \sim \mathcal{N}(\alpha_{t-1}, \sigma^2 I)$
 - b) Por cada término:
 - 1) Extraer $Z \sim Mult(\pi(\eta))$.
 - 2) Extraer $W_{t,d,n} \sim Mult(\pi(\beta_{t,z}))$.

Respecto al cálculo de la inferencia del modelo, dado que la representación de evolución de tópicos en el tiempo permite el uso de modelos Gaussianos, no es posible calcular la inferencia posterior de este, por lo tanto, se obtienen algoritmos de inferencia variacional para aprovechar así técnicas

usadas en datos secuenciales. Se implementan aproximaciones variacionales basados en el filtro de Kalman para calcular aproximaciones de la inferencia posterior sobre los tópicos.

2.5. State Space Model

Una “Representación de Espacio de Estados” (State Space Model) es un modelo probabilístico que usa variables de estado para describir un sistema mediante un conjunto de ecuaciones diferenciales de primer orden y ecuaciones algebraicas de salida. Este modelo permite describir la dependencia probabilística entre variables latentes de estado y las mediciones observadas. Este modelo otorga un marco general para analizar sistemas dinámicos determinísticos y estocásticos los cuales son medidos u observados mediante procesos estocásticos.[2]

El concepto de *estado* en teoría de sistemas, puede interpretarse como el resumen o resultado de comportamientos pasados del sistema. El estado en conjunto con la variable de entrada perteneciente a un estado futuro determinan todos los estados futuros y sus correspondientes variables de salidas. Además, las variables de salida de un estado determinado son una función perteneciente únicamente al estado y sus variables de entrada.

Un modelo de espacio de estados lineal genérico, con $\mathbf{p}$ entradas, $\mathbf{q}$ salidas y $\mathbf{n}$ variables de estado, puede representarse como:

$$\begin{aligned} x_n &= A_n z_{n-1} + B_n w_n \\ y_n &= C_n z_n + \delta_n \end{aligned} \tag{2.16}$$

donde x_n es el estado del sistema en el tiempo n , $x_n \in \mathbb{R}^n$, w_n es el vector de entrada, $w_n \in \mathbb{R}^p$ y y_n es el vector de salida, $y_n \in \mathbb{R}^q$. A_n es la matriz de estados, B_n es la matriz de entrada y C_n es la matriz de salida. Por temas de simplicidad y conveniencia, la matriz de transmisión $D(w_n, \delta_n) = 0$. La ecuación de x_n representa la ecuación de estado del sistema, mientras que la ecuación y_n corresponde la ecuación de observación.

Kalman y Bucy (1961) [12] introdujeron métodos estocásticos de espacio de estados en el espectro de teoría de sistemas, mientras que Ho y Lee (1964) observaron que si el estado inicial x_0 se encuentra distribuido mediante una Gaussiana, y si la densidad condicional son Gaussianas, la media y la varianza pueden ser calculadas mediante fórmulas algebraicas y propagadas acorde a la

ecuación de Filtro de Kalman y que este Filtro de Kalman es de hecho la solución Bayesiana para dicho modelo[10]. De este modo, es posible usar representaciones de espacio de estados y el filtro de Kalman para calcular funciones de verosimilitud para valores dados de parámetros desconocidos asumiendo una distribución Gaussiana.

El modelo de espacios de estado lineal Gaussiano puede observar una secuencia de datos $x_0, x_1, \dots, x_T \in \mathbb{R}^p$ como una extracción aleatoria correspondiente al siguiente proceso generativo:

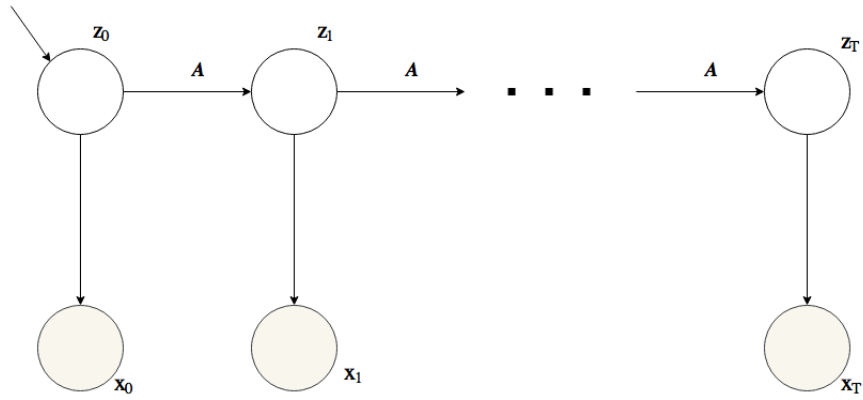


Figura 2.3: Representación gráfica del modelo Linear Gaussian State Space Model

1. $z_n \sim \mathcal{N}(0, \Sigma_0)$ estado inicial en $\mathbb{R}^q$.
2. $z_n = Az_{n-1} + w_{n-1}$, donde $w_{n-1} \sim \mathcal{N}(0, Q)$, es decir, z_t se obtiene de una dinámica gaussiana lineal dado el estado precedente z_{t-1} a través de una matriz de estado A y una matriz de covarianza $Q \in \mathbb{R}^{q \times q}$
3. $x_t = Cz_t + v_t$, donde $v_t \sim \mathcal{N}(0, R)$, es decir, x_t es la observación de la muestra dado un estado z_t , normalmente distribuido con media Cz_t , donde C es la matriz de salida y covarianza $R \in \mathbb{R}^{p \times p}$

Los usos más comunes de este modelo al buscar generar inferencias son:

1. *Filtering*. Se busca inferir el estado actual dado un historial de observaciones $P(z_t | x_0, \cdot, x_t)$. Un ejemplo es querer buscar el estado actual de un misil dado su posición en tiempos precedentes.

2. *Smoothing*. Infere un estado pasado dada las observaciones $P(z_s|x_0 \cdots, x_t)$, donde $s < t$. Un ejemplo es calcular el origen del lanzamiento de un misil, dado la observación de sus estados siguientes a este.
3. *Prediction*. Se busca predecir el estado futuro dada las observaciones $P(z_u|x_0 \cdots, x_t)$, donde $u > t$. Un ejemplo es predecir la ubicación de un misil desde el tiempo presente a un tiempo futuro.

2.6. Kalman Filter

El filtro de Kalman es un algoritmo de filtro para modelos de espacios de estado lineal Gaussiano que sirve para poder identificar el estado oculto de un sistema donde los parámetros son conocidos, es decir, es implementado cuando se desea buscar la distribución de probabilidad de un estado actual dado su historial de observaciones[12]. Dado que los estados y las distribuciones distribuyen según densidad gaussiana, solo es necesario encontrar la media y la covarianza de la distribución condicional, la cual también debe ser una distribución gaussiana. La notación para calcular la media y la covarianza respectivamente es:

$$\begin{aligned}\hat{z}_{t|t} &= \mathbf{E}[z_t|x_{0:t}] \\ P_{t|t} &= \mathbf{E}[(z_t - \hat{z}_{t|t})(z_t - \hat{z}_{t|t})^T|x_{0:t}]\end{aligned}\tag{2.17}$$

El Filtro de Kalman puede definirse en dos pasos recursivos:

1. *Time Update*. Calcular la predicción de la distribución $P(z_{t+1}|x_{0:t})$
2. *Measurement Update*. Calcular la nueva distribución $P(z_{t+1}|x_{0:t+1})$ dada la distribución predicha $P(z_{t+1}|x_{0:t})$

El filtro de Kalman tiene numerosas aplicaciones en tecnología, tales como navegación y control de vehículos. Una aplicación común es la guía, navegación y control de vehículos, especialmente naves espaciales. Además el filtro es ampliamente usado en campos como procesamiento de señales y econometría.

Capítulo 3

Descripción del problema

3.1. Entrenamiento Nuevos Datos

La capacidad de analizar la evolución de tópicos en el tiempo permite observar en un set de documentos propiedades tales como tendencias en un intervalo de tiempo determinado, transformación del vocabulario dentro de un mismo tópico o aparición de nuevos términos, entre otros fenómenos de interés.

Cada año, la cantidad de información que se sube a Internet aumenta considerablemente, siendo gran parte de esta data creada por usuarios de redes sociales. Por ejemplo, para el año 2016 se tiene un estimado que en la red social Twitter, se suben cerca de 500 millones de tweets al día. Ante tal volumen de contenido, surge la necesidad de analizar, resumir y visualizar colecciones de gran volumen con el fin de extraer información de utilidad.

El modelo generativo DTM nos permite visualizar la evolución de tópicos en un corpus de documentos organizados secuencialmente, pero la necesidad de reentrenar el corpus completo para cada nuevo set de documentos perteneciente a un nuevo intervalo de tiempo, dificulta la implementación de este modelo cuando se busca un análisis y visualización *online* (ir entrenando los documentos conforme estos estén accesibles). Como fue descrito en la sección de Marco Teórico, como los parámetros de cada tópico se encuentran enlazados en un *state space model*, cuyos parámetros evolucionan según un proceso Gaussiana, es posible incorporar un nuevo intervalo de tiempo, para el cual el tópico k en el nuevo intervalo $T + 1$ evolucione suavemente desde el tópico k en el intervalo T .

El algoritmo DTM implementado en la librería de Python *Gensim* se llama *ldaseq* y necesita como entrada el corpus, ordenado secuencialmente, en formato *Blei's LDA-C*, el diccionario que representa al corpus, el cual corresponde a un mapeo entre los términos y su correspondiente identificador y un arreglo de largo T que indique cuantos documentos pertenecen a cada intervalo de tiempo. Una vez entrenado el modelo, este no puede utilizarse para entrenar nuevos documentos pertenecientes a intervalos de tiempos subsecuentes. Volver a entrenar todo el corpus es un proceso considerablemente costoso en términos de tiempo de ejecución.

El corpus debe estar en formato *Blei's LDA-C*, el donde el corpus en su completitud es interpretado en un solo archivo, donde cada linea representa un documento con la siguiente estructura:

$$N \text{ fieldId1:fieldValue1 fieldId2:fieldValue2 ... fieldIdN:fieldValueN}$$

En el ejemplo, N representa la cantidad de términos de dicho documento específico y los elementos subsiguientes son los términos representados por su identificador (ID) y el término (Value).

Otro problema que se encuentra en el proceso consiste en la identificación de nuevos términos que no hayan sido observados en los intervalos de tiempo ya modelados, dado que estos deben tener una representación en la matriz *topic-word*, el cual no puede ser un valor nulo. Este problema se repite para todas las variables condicionadas al nuevo término.

3.2. Representación State Space Models

En el algoritmo *ldaseq* de *gensim*, por cada tópico en el modelo, existe una representación de espacio de estados con T estados. En cada representación de tópico, se encuentra la siguiente información para todos los intervalos de tiempo. Siendo T el número de intervalos de tiempo y V la cantidad de términos en el vocabulario:

- Variable de las observaciones ,obtenidas inicialmente de los *sstats* del modelo LDA entrenado previo en el corpus, representado como una matriz de dimensionalidad $T \times V$
- Distribución *topic-word* para el tópico específico, representado como una matriz de dimensionalidad $T \times V$.

- Valores de la media y la varianza de cada término, usados en el proceso de inferencia de cada palabra en el intervalo de tiempo. Tanto la media y varianza están representados por una matriz de dimensionalidad $(T + 1) \times V$.
- Parametro variacional extra ζ con un valor único para cada intervalo de tiempo, de dimensionalidad $1 \times T$.

Es importante destacar que el algoritmo *ldaseq* no posee una representación por cada intervalo de tiempo, sino que posee una representación de cada tópico, modelado como un State Space Models con estados encadenados donde cada estado representa el tópico en dicho intervalo de tiempo. Todos los tópicos se encuentran almacenados en un array llamado *topic_chains*.

3.3. Dynamic Topic Models Incremental

Como fue mencionado anteriormente, el objetivo central es lograr entrenar un nuevo set de documentos, pertenecientes a un nuevo intervalo de tiempo, usando un modelo previamente entrenado usando el algoritmo Dynamic Topic Models, manteniendo los parámetros de los intervalos anteriores estáticos (model updating). La construcción del algoritmo propuesto sigue los siguientes pasos:

1. Actualizar corpus con nuevo set de documentos.
2. Extender matrices en los state space models de cada tópico para aceptar parámetros correspondientes al nuevo intervalo de tiempo $T + 1$.
3. Inicializar valores correspondientes al nuevo intervalo de tiempo $T + 1$, para cada tópico en la variable *topic_chains*.
4. Ajustar el modelo usando algoritmos esperanza-maximización (*EM-STEP*) para encontrar estimadores de máxima verosimilitud.

3.3.1. Actualización Corpus

Siguiendo la estructura de un corpus como un conjunto ordenado secuencialmente de documentos, los nuevos documentos deben incorporarse en la cola de la secuencia, simulando así el nuevo intervalo de tiempo. Dado que el formato de ambos conjuntos de documentos (el usado para entrenar el modelo y el nuevo set que busca incorporarse) deben estar en el formato *Blei's LDA-C*, puede ocurrir que para un mismo término, este tenga un identificador distinto. Por ejemplo, la palabra *people* puede tener el identificador #1 mientras que en el nuevo set de documentos, *people* tenga como identificador #5. Es necesario actualizar el mapeo de los términos en el nuevo set de documentos previo a incorporarlos en el corpus.

El siguiente paso consiste en identificar el set de términos nuevos que no hayan sido modelados y asignarle un identificador compatible con el diccionario original. Dado un corpus con un vocabulario de largo V y un nuevo set de documentos con N términos nuevos (no presentes en el corpus original), entonces se asigna para cada uno de estos términos el identificador: $V + 1, V + 2, \dots, V + N$. El orden y frecuencia de estos nuevos términos no es relevante para el mapeo identificador:valor.

```

def update_corpus(self, new_corpus=None, new_id2word=None):
    """
    Update Method
    Takes a new corpus and its new tokens and update the actual corpus
    """
    # Get New Tokens from new corpus set
    new_tokens = set(new_id2word.values()).difference(set(self.id2word.values()))
    new_tokens_len = len(new_tokens)
    vocab_len = self.vocab_len

    # Updating values
    for token in new_tokens:
        self.id2word.token2id[token] = vocab_len
        vocab_len += 1

    new_corpus_update = []
    for i, document in enumerate(new_corpus):
        new_corpus_update.append([])
        for j, word in enumerate(document):
            new_corpus_update[i].append(
                (self.id2word.token2id[new_id2word[new_corpus[i][j][0]]],
                 new_corpus[i][j][1]))

    new_corpus = new_corpus_update
    self.new_corpus = new_corpus
    del new_corpus_update

```

Listing 3.1: Función implementada para actualizar el corpus con los nuevos términos encontrados en el nuevo corpus a entrenar.

3.3.2. Modificación Matrices LdaSeq

El siguiente paso consiste en extender las matrices pertenecientes al modelo de espacios de estados para cada tópico en el modelo, para representar así el nuevo intervalo de tiempo. Este procedimiento consta de 2 fases : extensión del vocabulario (columnas) y extensión de nuevo intervalo de tiempo (filas).

- Al extender el **vocabulario**, para cada matriz extienden las filas equivalentes a la cantidad de términos nuevos identificados. Ese procedimiento no se realiza para la matriz *zeta*
- Al extender el **intervalo**, para cada matriz indexada por T se agrega una nueva columna, para representar así el nuevo intervalo de tiempo.

Uno de los problemas al momento de extender las matrices de cada modelo de espacio de estado, es la representación matricial de los nuevo términos identificados en los intervalos de tiempos anteriores. Una propuesta consiste en identificar el comportamiento de términos en el corpus que tengan una frecuencia mínima y simular la distribución de los nuevos términos , asumiendo que previamente aparecieron a lo más, una sola vez.

Para el modelo entrenado con el dataset de noticias original, se analiza un set de 10 palabras que se encuentran al final de la cola del vocabulario V con frecuencia entre 1 y 2 apariciones en el corpus. A su vez, previo al entrenamiento del corpus, se incorpora el término *dummy* con frecuencia 1 al dataset, para analizar así el comportamiento en comparación con los otros términos.

Word	Freq	topic-word T_0	topic-word T_1	topic-word T_2	Position T_0	Position T_1	Position T_2
convert	3	2.2310e-07	2.2176e-07	2.2103e-07	6066	6066	6066
liveshot	3	2.0867e-07	2.0741e-07	2.0672e-07	9782	9782	9782
sectarianism	3	2.0869e-07	2.0745e-07	2.0675e-07	9631	9631	9631
bohemian	3	2.0859e-07	2.0732e-07	2.0664e-07	10329	10329	10329
appleton	3	2.6350e-07	2.6194e-07	2.6110e-07	4543	4543	4543
nebraska	3	2.08863e-07	2.0759e-07	2.0691e-07	8759	8759	8759
vicious	3	2.08696e-07	2.0742e-07	2.0674e-07	9656	9656	9656
garment	3	2.08911e-07	2.0764e-07	2.0696e-07	8616	8616	8616
essentially	3	2.08875e-07	2.0760e-07	2.0692e-07	8717	8717	8717
marginal	3	2.08644e-07	2.07378e-07	2.0669e-07	10011	10011	10011

Cuadro 3.1: Distribución Topic-Words para 10 términos con frecuencia mínima en el corpus

Word	Freq	obs T_0	obs T_1	obs T_2
convert	3	-9.88396	-9.88396	-9.88396
liveshot	3	-9.97312	-9.97312	-9.97312
sectarianism	3	-9.97298	-9.97298	-9.97298
bohemian	3	-9.97364	-9.97364	-9.97364
appleton	3	-9.66201	-9.66201	-9.66201
nebraska	3	-9.97193	-9.97193	-9.97193
vicious	3	-9.97300	-9.97300	-9.97300
garment	3	-9.97162	-9.97162	-9.97162
essentially	3	-9.97185	-9.97185	-9.97185
marginal	3	-9.97333	-9.97333	-9.97333

Cuadro 3.2: Observaciones, o sstats de un término en cada intervalo de tiempo

Existen 2 observaciones : el comportamiento de los términos con frecuencia mínima es similar y estos no cambian en los intervalos de tiempos, ni en su posición en la distribución *topic-words*, ni en su valor de $\beta_{t,k}$. Este comportamiento nos permite simular para los nuevos términos identificados en el nuevo corpus a entrenar, su presencia en los intervalos de tiempo anteriores, con una frecuencia mínima, sin alterar los valores previamente entrenados en los modelos de espacio de estado.

En resumen: para inicializar los valores de los nuevos términos en los intervalos de tiempos pasados, se busca para cada nuevo término identificado simular el comportamiento de un término que haya sido modelado y que tenga la frecuencia mínima dentro del corpus.

Otra variable que se requiere para inicializar el modelo son los estadísticos suficientes (*sstats*) necesarios para la actualización de la distribución tópico-palabra en el modelo. En el algoritmo *ldaseq* original hay 3 formas de obtener los estadísticos suficientes: se obtienen durante el entrenamiento, entrenando un modelo LDA convencional sobre todo el corpus (ignorando los intervalos de tiempo) y extrayendo los sstats entrenados; usando los sstats de un modelo lda ya entrenado (correspondiente al mismo corpus); entregando sólo una matriz de dimensiones $V \times T$.

3.3.3. Inicialización parámetros *ldaseq* incremental

Usando el mismo procedimiento que implementa el algoritmo original *ldaseq*, el modelo para el nuevo intervalo de tiempo se inicializa para cada tópico las variables descritas en el punto 4.3 usando los estadísticos suficientes obtenidos usando el algoritmo *LDA* sobre el nuevo corpus.

- **Varianza:** La propuesta de inferencia usando Filtro de Kalman en parámetros encadenados de DTM [3] declara lo siguiente. La varianza V_t y la varianza marginal $\hat{V}_{t-1}$ se calculan de la siguiente manera:

$$\begin{aligned} V_t &\equiv \mathbb{E}((\beta_t - m_t)^2 | \hat{\beta}_{1:t}) \\ &= \left(\frac{\hat{v}_t^2}{V_{t-1} + \sigma^2 + \hat{v}_t^2} \right) (V_{t-1} + \sigma^2) \end{aligned} \quad (3.1)$$

$$\begin{aligned} \hat{V}_{t-1} &\equiv \mathbb{E}((\beta_{t-1} - \hat{m}_{t-1})^2 | \hat{\beta}_{1:T}) \\ &= V_{t-1} + \left(\frac{V_{t-1}}{V_{t-1} + \sigma^2} \right)^2 (\hat{V}_t - (V_{t-1} + \sigma^2)) \hat{\beta}_t \end{aligned} \quad (3.2)$$

- **Media:** La propuesta de inferencia usando Filtro de Kalman en parámetros encadenados de DTM [3] declara lo siguiente. La media m_t y la varianza marginal $\hat{m}_{t-1}$ se calculan según:

$$\begin{aligned} m_t &\equiv \mathbb{E}(\beta_t | \hat{\beta}_{1:t}) = \\ &\left(\frac{\sigma^2}{V_{t-1} + \sigma^2} \right) m_{t-1} + \left(1 - \frac{\sigma^2}{V_{t-1} + \sigma^2 + \sigma^2} \right) \end{aligned} \quad (3.3)$$

$$\begin{aligned} \hat{m}_{t-1} &\equiv \mathbb{E}(\beta_{t-1} | \hat{\beta}_{1:T}) = \\ &\left(\frac{\hat{v}_t^2}{V_{t-1} + \sigma^2 + \hat{v}_t^2} \right) m_{t-1} + \left(1 - \frac{\hat{v}_t^2}{V_{t-1} + \sigma^2 + \hat{v}_t^2} \right) \hat{m}_t \end{aligned} \quad (3.4)$$

- **Zeta:** el parámetro zeta corresponde a un parámetro variacional usado para maximizar la cota inferior en la etapa de derivación del algoritmo variacional.

$$\hat{\zeta}_t = \sum_w \exp(\hat{m}_{tw} + \frac{\hat{V}_{tw}}{2}) \quad (3.5)$$

■ `Expected_log_prob` :

$$\sum_{t_1}^T \mathbb{E}_q \log p(d_t | \beta_t) = \hat{m}_{tw} - \log \hat{\zeta}_t \quad (3.6)$$

3.3.4. Algoritmo esperanza-maximización incremental (propuesta)

El objetivo de la función incremental del modelo DTM es entrenar nuevos documentos pertenecientes a un nuevo intervalo de tiempo, preservando el modelo de los intervalos de tiempos anteriores, por lo que al momento de optimizar los parámetros, los intervalos de tiempos anteriores no deben verse alterados. Durante el proceso de esperanza-maximización para la inferencia de los nuevos parámetros, se realizará un *freezing* de los parámetros pertenecientes a intervalos de tiempo anteriores, considerándolos como constantes y no variables a ser optimizadas. Durante el paso de esperanza (E-Step) solo se calculará la verosimilitud de los documentos pertenecientes al nuevo intervalo de tiempo, mientras que en el paso de maximización (M-Step) los parámetros que se actualizarán solo corresponden a los parámetros del nuevo intervalo de tiempo. Los detalles respecto al paso de maximización serán detallados en la siguiente sección.

3.4. Derivación de Algoritmo Variacional (original)

Usando Filtro de Kalman, se aproxima la distribución posterior $p(\beta_{1:T} | w_{1:T})$ usando el state space posterior $q(\beta_{1:T} | \tilde{\beta}_{1:T})$. Usando la desigualdad de Jensen, el *log-likelihood* está acotado por abajo como:

$$\begin{aligned} \log p(d_{1:T}) &\geq \\ \int q(\beta_{1:T} | \tilde{\beta}_{1:T}) \log \left(\frac{p(\beta_{1:T}) p(d_{1:T} | \beta_{1:T})}{q(\beta_{1:T} | \tilde{\beta}_{1:T})} \right) d\beta_{1:T} \end{aligned} \quad (3.7)$$

$$= \mathbf{E}_q \log_p(\beta_t | \beta_{t-1}) + \sum_{t=1}^T \mathbf{E}_q \log_p(d_t | \beta_t) + H(q)$$

El objetivo es calcular la distribución $q(\beta_{1:T} | \tilde{\beta}_{1:T})$ para maximizar así la cota inferior de $\log p(d_{1:T})$. El primer término del algoritmo variacional (1) es:

$$\begin{aligned}
\sum_{t=1}^T \mathbf{E}_q \log_p(\beta_t | \beta_{t-1}) &= -\frac{VT}{2} (\log \sigma^2 + \log(2\pi^2)) \\
&- \frac{1}{2\sigma^2} \sum_{t=1}^T \mathbf{E}_q (\beta_t - \beta_{t-1})^T (\beta_t - \beta_{t-1}) \\
&= \frac{VT}{2} (\log \sigma^2 + \log 2\pi) - \frac{1}{2\sigma} \sum_{t=1}^T \|\tilde{m}_t - m_{t-1}\|^2 \\
&- \frac{1}{\sigma^2} \sum_{t=1}^T \text{Tr}(\tilde{V}_t) + \frac{1}{2\sigma^2} (\text{Tr}(\tilde{V}_0) - \text{Tr}(\tilde{V}_t))
\end{aligned} \tag{3.8}$$

El segundo término del algoritmo variacional (1) es:

$$\begin{aligned}
\sum_{t=1}^T \mathbf{E}_q \log_p(d_t | \beta_t) &= \\
&\sum_{t=1}^T \sum_w n_{tw} \mathbf{E}_q (\beta_{tw} - \log \sum_w \exp(\beta_{tw})) \\
&\geq \sum_{t=1}^T \sum_w n_{tw} \tilde{m}_{tw} - n_t \tilde{\zeta}_t^{-1} \sum_w \exp(\tilde{m}_t + \tilde{V}_{tw}/2) \\
&+ \sum_{t=1}^T n_t - n_t \log \tilde{\zeta}_t^{-1}
\end{aligned} \tag{3.9}$$

donde $n_t = \sum_w n_{tw}$ es la suma de los sstats, introduciendo el parámetro variacional adicional $\tilde{\zeta}_{1:T}$. El tercer término es la entropía:

$$\begin{aligned}
H(q) &= \sum_{t=1}^T (\frac{1}{2} \log |\tilde{V}_t| + \frac{T}{2} \log 2\pi) \\
&= \frac{1}{2} \sum_{t=1}^T \sum_w \log \tilde{V}_{tw} + \frac{TV}{2} \log 2\pi
\end{aligned} \tag{3.10}$$

Para maximizar la cota inferior como una función de los parámetros variacionales, usamos el algoritmo de gradiente conjugado. Primero, se busca maximizar con respecto a $\tilde{\zeta}$; La derivada es:

$$\frac{\partial \ell}{\partial \tilde{\zeta}_t} = \frac{n_t}{\tilde{\zeta}_t^2} \sum_w \exp(\tilde{m}_{tw} + \tilde{V}_{tw}/2) - \frac{n_t}{\tilde{\zeta}_t} \tag{3.11}$$

Ajustando a cero y resolviendo para $\tilde{\zeta}_t$ se obtiene:

$$\tilde{\zeta}_t = \sum \exp(\tilde{m}_{tw} + \tilde{V}_{tw}/2) \tag{3.12}$$

A continuación, se busca maximizar con respecto a $\tilde{\beta}_s$:

$$\begin{aligned}
\frac{\partial \ell(\tilde{\beta}, \tilde{v})}{\partial \tilde{\beta}_{sw}} = & \\
& -\frac{1}{\sigma^2} \sum_{t=1}^T (\tilde{m}_{tw} - \tilde{m}_{t-1,w}) \left(\frac{\partial \tilde{m}_{tw}}{\partial \tilde{\beta}_{sw}} - \frac{\partial \tilde{m}_{t-1,w}}{\partial \tilde{\beta}_{sw}} \right) \\
& + \sum_{t=1}^T (n_{tw} - n_{tw} \zeta^{-1} \exp(\tilde{m}_{tw} - \tilde{V}_{tw}/2)) \frac{\partial \tilde{m}_{tw}}{\partial \tilde{\beta}_{sw}}
\end{aligned} \tag{3.13}$$

La ecuación *forward-backward* para $\tilde{m}_t$ puede ser usada para derivar la recurrencia de $\partial \tilde{m}_t / \partial \tilde{\beta}_{sw}$. La recurrencia hacia adelante es:

$$\begin{aligned}
\frac{\partial \tilde{m}_t}{\partial \tilde{\beta}_s} = & \left(\frac{\tilde{v}^2}{v_{t-1} + \sigma^2 + \tilde{v}^2} \right) \frac{\partial \tilde{m}_{t-1}}{\partial \tilde{\beta}_s} + \\
& \left(1 - \frac{\tilde{v}^2}{v_{t-1} + \sigma^2 + \tilde{v}^2} \right) \delta_{s,t}
\end{aligned} \tag{3.14}$$

Con condiciones iniciales $\partial \tilde{m}_0 / \partial \tilde{\beta}_{sw} = 0$. La recurrencia hacia atrás es:

$$\begin{aligned}
\frac{\partial \tilde{m}_{t-1}}{\partial \tilde{\beta}_s} = & \left(\frac{\tilde{\sigma}^2}{V_{t-1} + \sigma^2 +} \right) \frac{\partial \tilde{m}_t}{\partial \tilde{\beta}_s} + \\
& \left(1 - \frac{\tilde{\sigma}^2}{V_{t-1} + \sigma^2 +} \right) \frac{\partial \tilde{m}_t}{\partial \tilde{\beta}_s}
\end{aligned} \tag{3.15}$$

3.5. Derivación de Algoritmo Variacional Incremental (propuesta)

Para este caso, solo se busca calcular el término para el último intervalo de tiempo T . El primer término de la cota es:

$$\begin{aligned}
\mathbf{E}_q \log_p(\beta_T | \beta_{T-1}) &= -\frac{V}{2} (\log \sigma^2 + \log(2\sigma^2)) \\
&- \frac{1}{2\sigma^2} \mathbf{E}_q(\beta_T - \beta_{T-1})^T (\beta_T - \beta_{T-1}) \\
&= -\frac{V}{2} (\log \sigma^2 + \log 2\pi) - \frac{1}{2\sigma} \|\tilde{m}_T - \tilde{m}_{T-1}\|^2 \\
&- \frac{1}{\sigma^2} \text{Tr}(\tilde{V}_T) + \frac{1}{2\sigma^2} (\text{Tr}(\tilde{V}_0) - \text{Tr}(\tilde{V}_T))
\end{aligned} \tag{3.16}$$

El segundo término es:

$$\begin{aligned}
\mathbf{E}_q \log_p(d_T | \beta_T) &= \\
\sum_w n_{tw} \mathbf{E}_q(\beta_w - \log \sum_w \exp(\beta_w)) & \\
\geq \sum_w n_w \tilde{m}_w - n_w \zeta_T^{-1} \sum_w \exp(\tilde{m}_T + \tilde{V}_w/2) & \\
+n_T - n_T \log \zeta_T^{-1} &
\end{aligned} \tag{3.17}$$

donde $n_T = \sum_w n_w$. El tercer término corresponde a la entropía.

$$H(q) = \frac{1}{2} \sum_w \log \tilde{V}_w + \frac{V}{2} \log 2\pi \tag{3.18}$$

La variable $\frac{V}{2} \log 2\pi$ se cancela entre el término 1 de la cota y la entropía. En el segundo término:

$$n_T \zeta_T^{-1} \sum_w \exp(\tilde{m}_T + \tilde{V}_w/2) = n_T \zeta_T^{-1} \zeta_T = n_T \tag{3.19}$$

el nuevo término $-n_T$ es cancelado con el n_T correspondiente en la misma ecuación. Finalmente, la cota puede obtenerse como :

$$\begin{aligned}
&= -\frac{V}{2}(\log \sigma^2) - \frac{1}{2\sigma} \|\tilde{m}_T - m_{T-1}\|^2 - \frac{1}{\sigma^2} Tr(\tilde{V}_T) \\
&+ \frac{1}{2\sigma^2} (Tr(\tilde{V}_0) - Tr(\tilde{V}_T)) + \sum_w n_w \tilde{m}_w - n_T \log \zeta_T \\
&+ \frac{1}{2} \sum_w \log \tilde{V}_w
\end{aligned} \tag{3.20}$$

Esta última ecuación, en comparación a la ecuación propuesta en 3,13 , reduce el número de variables a optimizar a solo una (estamos actualizando un solo intervalo de tiempo), en comparación con el modelo original, que optimiza por cada término, todos los intervalos de tiempo a la vez, es decir, una optimización sobre una función de T variables. Esta diferencia reduce drásticamente el tiempo computacional durante el paso de maximización.

3.6. Evaluación Coherencia de Tópicos

Para cada modelo entrenado se debe tener una medida que nos permita cuantificar la calidad de cada modelo. Se entiende que un conjunto de sentencias es *coherente* si estos se sostienen entre sí. En consecuencia, un conjunto de hechos coherentes puede interpretarse en un contexto en el cual es posible cubrir todas o la mayoría de los hechos pertenecientes al conjunto. Un ejemplo de conjuntos coherentes es “*el juego es un deporte en equipos*”, “*el juego es jugado con una pelota*”, “*el juego requiere de un gran esfuerzo físico*”.

Los modelos de tópicos pueden capturar efectivamente una interpretabilidad intuitiva a nivel humano, pero estos no otorgan necesariamente garantías de interpretabilidad cuantitativa respecto los modelos generados[6], por lo que surge la necesidad de crear métricas de coherencias para estos modelos. Para este trabajo, se usará un framework que es capaz de representar medidas de coherencia como una composición de fases o dimensiones que pueden ser combinadas libremente[15]. Cada dimensión está representada por un set de componententes intercambiables, permitiendo crear nuevas medidas de coherencias adaptables al modelo requerido. Las dimensiones son : *Segmentación*, *Cálculo de Probabilidad*, *Medidas de confirmación* y *Agregación*.

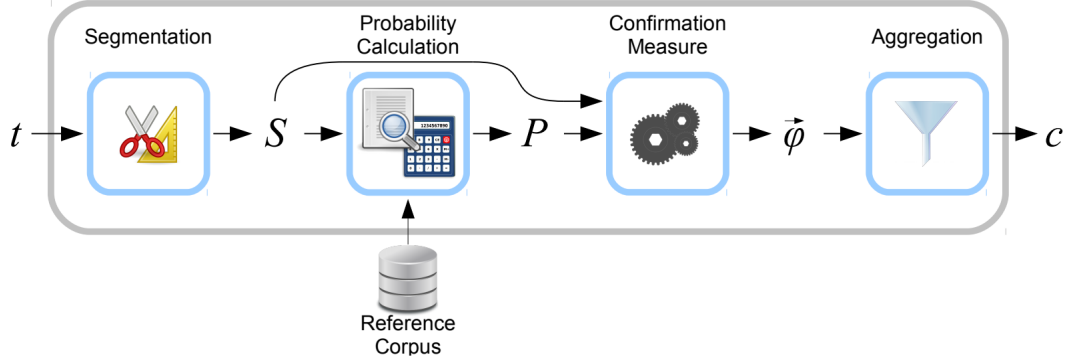


Figura 3.1: Representación gráfica del framework unificado para cuantificar la coherencia de un modelo, con sus cuatro componentes y sus resultados intermedios

La configuración del framework para este trabajo será una representación de la medida *UMass* Coherence, la cual puede ser descrita como la composición de los cuatro criterios anteriores:

$$C_{UMass} = (P_{bd}, S_{pre}^{one}, m_{lc}, \sigma_a) \quad (3.21)$$

- En la fase de **segmentación** se tiene como resultado, dado un conjunto de palabras W , un set de pares pertenecientes a un subconjunto de W . En la mayoría de las medidas de coherencias propuestas para evaluación de tópicos, se suele comparar pares de términos, en donde todos los términos son emparejados con todos los términos, siguiendo un criterio determinado. La segmentación S_{pre}^{one} empareja cada palabra con todas las palabras que le preceden y se define como:

$$S_{pre}^{one} = \left\{ (W', W^*) \mid W' = \{w_i\}; \right. \\ \left. W^* = \{w_j\}; w_i, w_j \in W; i \neq j \right\} \quad (3.22)$$

- En la fase de **cálculo de probabilidad**, se define el modo en que las probabilidades son derivadas de los datos subyacentes (corpus de referencia). Para esta configuración, P_{bd} se denomina *Boolean document*, y estima la probabilidad de un término como el número de documentos en el cual la palabra aparece, dividido por el número total de documentos.
- La fase de **Medidas de Confirmación** toma un par $S_i = (W', W^*)$ de subconjunto de términos, así como las correspondientes probabilidades, para calcular que tan fuerte el conjunto

de términos condicionadores W_* soporta el conjunto W' . La medida de confirmación directa m_{cl} se define como:

$$m_{lc}(S_i) = \log \frac{P(W', W_*) + \epsilon}{P(W_*)} \quad (3.23)$$

- La fase de **agregación** toma todas las medidas de confirmación $\varphi = \{\varphi_1, \varphi_2, \dots, \varphi_{|S|}\}$ del subconjunto S_i y son agregadas a una sola calificación de coherencia. La evaluación σ_a corresponde a la media aritmética de las agregaciones. En este modelo, cada tópico tendrá su valor correspondiente otorgado por la medida de confirmación implementada, por lo que estos son los valores que serán finalmente agregados a un único score, implementando así la media aritmética.

En resumen: la configuración del framework presentada para evaluar y cuantificar la coherencia de un modelo de tópicos, nos entregará un score que se encontrará en el dominio de los números negativos, siendo un mejor modelo mientras más cercano a 0 sea el score calculado.

3.7. Visualización de Modelos de Tópicos

Generar visualizaciones para modelos de tópicos constituye un desafío debido a la alta dimensionalidad de estos, la cual puede constituir miles de documentos representados como una distribución multinomial de cientos de tópicos. Diversos trabajos y propuestas se han desarrollado en torno a la visualización de modelos de tópicos, enfocados en el modelo LDA convencional, en donde una de las propiedades que más se destacan es la interactividad y la capacidad de navegar sobre los distintos niveles de granularidad del modelo (navegar a nivel de tópico, a nivel de documentos, entre otras alternativas)[5].

Para los modelos desarrollados en este trabajo, se usará la librería de visualización de tópicos *pyLDavis*, el cual provee una mirada global de los tópicos y como estos difieren entre ellos en un intervalo de tiempo determinado (*pyLDavis* no permite obtener una visualización global de los intervalos de tiempo, por lo que es necesario generar varias visualizaciones para realizar comparaciones).

Una de las propiedades de *pyLDavis* es la capacidad de determinar y modificar la *relevancia* de un término en un tópico determinado. Sea $\phi_{k,w}$ la probabilidad del término $w \in \{1, \dots, V\}$ en el tópico $k \in \{1, \dots, K\}$ con V el número total de términos en el corpus y K el número total de tópicos en el modelo, y sea también p_w la probabilidad marginal del término en el corpus. Definimos la **relevancia** del término w en el tópico k dado el parámetro de peso $\lambda (0 \leq \lambda \leq 1)$ como:

$$r(w, k|\lambda) = \lambda \log(\phi_{kw}) + (1 - \lambda) \log\left(\frac{\phi_{kw}}{p_w}\right) \quad (3.24)$$

El parámetro λ determina el peso otorgado por la probabilidad del término w en el tópico k relativo a su factor *lift*. El término *lift*, acuñado por M.Taddy(2012)[16] determina la proporción entre la probabilidad de un término en un tópico con su probabilidad marginal a través de corpus. Este factor es útil para evitar ranking de términos en orden decreciente basado en su frecuencia absoluta, pero puede generar ruido al otorgar mayor relevancia a términos poco ocurrentes que solo son frecuentes en un solo tópico. Si $\lambda = 1$ la distribución de términos en un tópico corresponde a la distribución original entrenada por el modelo en orden decreciente, mientras que si $\lambda = 0$ el ranking es construido basado en su factor *lift*. *pyLDavis* nos permite modificar el parámetro λ en tiempo real y ver la redistribución de los términos en los tópicos basados en la nueva proporción de pesos seleccionada.

La librería *pyLDavis* permite determinar la **prominencia** global de los top-N términos dentro del corpus entrenado [7]. Sea $P(k|w)$ la probabilidad de que el término w sea observado por el tópico k y sea p_k la probabilidad de que una término al azar w' perteneciente al vocabulario haya sido generado por el tópico k . Se define la peculiaridad del término w como la divergencia KL entre $P(k|w)$ y p_k :

$$peculiaridad(w) = \sum_k P(k|w) \log\left(\frac{P(k|w)}{p_k}\right) \quad (3.25)$$

La peculiaridad de un término describe que tan informativo el término w es para generar un tópico, en comparación a un término seleccionado aleatoriamente del vocabulario. Si un término ocurre con frecuencia en todos los tópicos, observar dicho término nos otorga poca información acerca del documento, por lo que se le otorga un score bajo. La peculiaridad de un término cumple un rol similar al factor idf mencionado en el modelo *tf-idf*, con la diferencia de que éste determina el peso de un término en un tópico en vez de un documento.

La prominencia de un término se define:

$$prominencia(w) = peculiaridad(w) \times P(w) \quad (3.26)$$

Al multiplicar la peculiaridad de un término con la probabilidad de obtener el término w en todo el corpus, el resultado final son los términos mas descriptivos del corpus a nivel global. Los top N términos mas descriptivos, o mas salientes, también serán aquellos que logren describir con mayor probabilidad los tópicos modelados.

Capítulo 4

Resultados

4.1. Descripción Corpus

Los experimentos y análisis se realizarán sobre un corpus de noticias recogidos de sitios web durante los meses de Marzo, Abril y Mayo del año 2016. El corpus posee un total de 1236 artículos con los siguientes temas: Deportes, Tecnologías, Política, Economía y Entretenimiento. A su vez, se tiene un cuarto intervalo de tiempo con 224 documentos que será usado para analizar la efectividad del algoritmo incremental. La configuración del modelo generado con el algoritmo *ldaseq* es la siguiente:

- Intervalos de Tiempo $T = 3$
- Número de Tópicos usados $K = 5$
- Número de documentos en Corpus $D = 1236$
- Largo Vocabulario del Corpus $V = 10734$

En la experimentación de la función incremental, todos los corpus pertenecientes a nuevos intervalos de tiempos, corresponden a una distribución aleatoria de los 1236 documentos mencionados anteriormente, especificando explícitamente si estos intervalos poseen una propiedad (por ejemplo, ausencia de cierto tópico en el conjunto de documentos).

4.2. Dynamic Topic Model Incremental

Vamos a comparar diferentes modelos creados con y sin el algoritmo incremental propuesto, con el objetivo de analizar los resultados con una representación visual de la distribución de términos relevantes por cada tópico y una cuantificación numérica de la coherencia de los tópicos observados.

Para el nuevo intervalo de tiempo que se busca incorporar al modelo, se tienen dos colecciones con propiedades específicas, las cuales se espera observar en la distribución *topic-word* final:

- **Colección 1:** posee una colección de artículos de noticias con temas de entretenimiento, economía, política y tecnología, pero no posee noticias sobre deportes.
- **Colección 2 :** esta colección solo posee artículos respecto a entretenimiento, tecnología y deportes.

Para los siguientes experimentos, el modelo previamente entrenado con 3 intervalos de tiempos y 5 tópicos definidos, al cual se busca incorporar un nuevo intervalo de tiempo, es el siguiente:

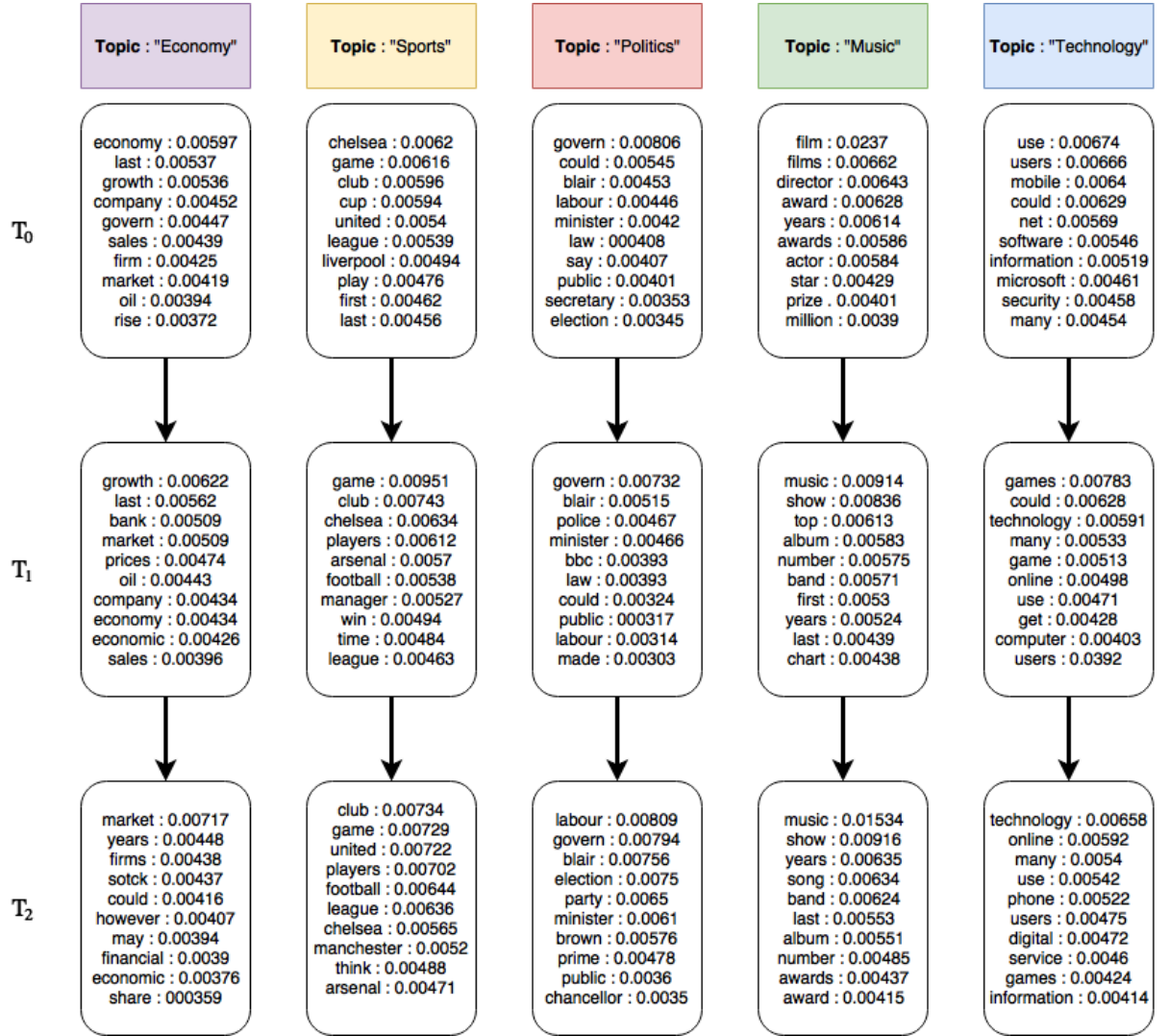


Figura 4.1: Representación gráfica de un modelo DTM entrenado con 3 intervalos de tiempos. Como se observa, los tópicos encontrados son coherentes con la descripción preliminar del corpus usado.

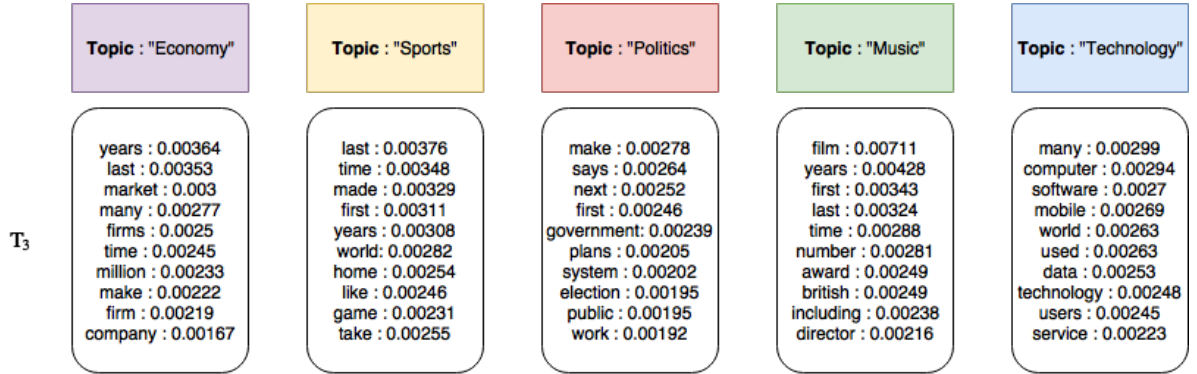


Figura 4.2: Representación gráfica de los resultados al implementar un nuevo intervalo de tiempo sobre el modelo de la figura 4.1. La colección usada corresponde a la colección 1, donde no hay noticias relacionadas con deportes

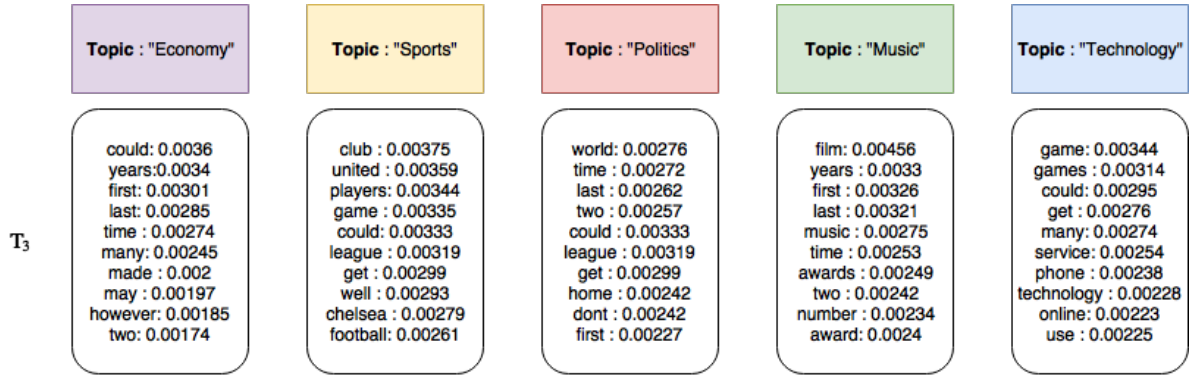


Figura 4.3: Representación gráfica de los resultados al implementar un nuevo intervalo de tiempo sobre el modelo de la figura 4.1. La colección usada corresponde a la colección 2, donde no hay noticias relacionadas con política y economía.

Dado que en la colección 1 los documentos pertenecientes al nuevo intervalo de tiempo no poseen artículos relacionados con deportes, se espera que este tópico posea un cambio brusco en su distribución de palabras más relevantes. Como se observa en la Figura 4,2, la nueva distribución de tópicos preserva el contenido de los tópicos relacionado con deportes, música y tecnología, mientras que el tópico de deportes pierde coherencia respecto a los intervalos anteriores. En el caso de la figura 4,3, donde fue usada la colección 2, tanto el tópico de Economía como de Política pierden interpretabilidad de las palabras más relevantes.

A continuación se cuantificará la coherencia de los modelos entrenados bajo 2 configuraciones: una usando DTM sobre los cuatro intervalos de tiempo, sin usar el proceso incremental, y otra configuración usando DTM sobre los 3 primeros intervalos de tiempo, y actualizando el modelo usando el algoritmo incremental propuesto. Para cada experimento se reentrenará el modelo 10 veces y se calculará la media y la varianza del valor de topic coherence. Los resultados se encuentran en el Cuadro 4.1 :

T	1	2	3	4	5	6	7	8	9	10	Media	Var
T_1	-1.6386	-1.5710	-1.6060	-1.5388	-1.6744	-1.7172	-1.5327	-1.6194	-1.5895	-1.5720	-1.6061	0.0034
T_2	-1.5170	-1.5261	-1.5906	-1.5331	-1.7362	-1.7718	-1.5216	-1.6730	-1.5055	-1.5341	-1.5909	0.00987
T_3	-1.5058	-1.5096	-1.5933	-1.5271	-1.5412	-1.7468	-1.5135	-1.5517	-1.5338	-1.6246	-1.5647	0.00553
T_4	-1.5191	-1.5318	-1.5919	-1.5583	-1.5847	-1.7172	-1.5327	-1.6194	-1.5895	-1.5531	-1.5747	0.0047

Cuadro 4.1: Media y Varianza del cálculo de Tópico Coherence sobre el corpus usando el algoritmo ldaseq sin componente incremental.

T	1	2	3	4	5	6	7	8	9	10	Media	Var
T_4	-1.5843	-1.5229	-1.6030	-1.5931	-1.5318	-1.5294	-1.4409	-1.4235	-1.4297	-1.6142	-1.5373	0.00631

Cuadro 4.2: Media y Varianza del cálculo de Tópico Coherence sobre el corpus usando el algoritmo ldaseq sin componente incremental.

El intervalo de tiempo entrenado con el algoritmo incremental poseen mejores resultados respecto al mismo intervalo en el Cuadro 4.1, esto puede interpretarse por la capacidad de capturar mejor aquellos términos que aparecen con mayor o menor frecuencia, obteniéndose mejores resultados en la fase de cálculo de probabilidad. La máquina usada para estos experimentos fue un Linux Ubuntu 3.40[Ghz] $\times$ 8, 62.8 [GiB]. El modelo entrenado sin el componente incremental se demoró un tiempo promedio de 2 horas, mientras que la fase incremental (solo incorporar un intervalo de tiempo) demoró un tiempo medio de 15 minutos.

4.3. Análisis Varianza DTM

En la siguiente sección, se entrenará el corpus con el modelo DTM usando diferentes valores de change variance, con el objetivo de comparar la distribución *word-topic* y la coherencia de cada modelo. Los valores de varianzas usados para el entrenamiento son 0,1 , 0,01 y 0,001. El valor default en Gensim es 0,005.

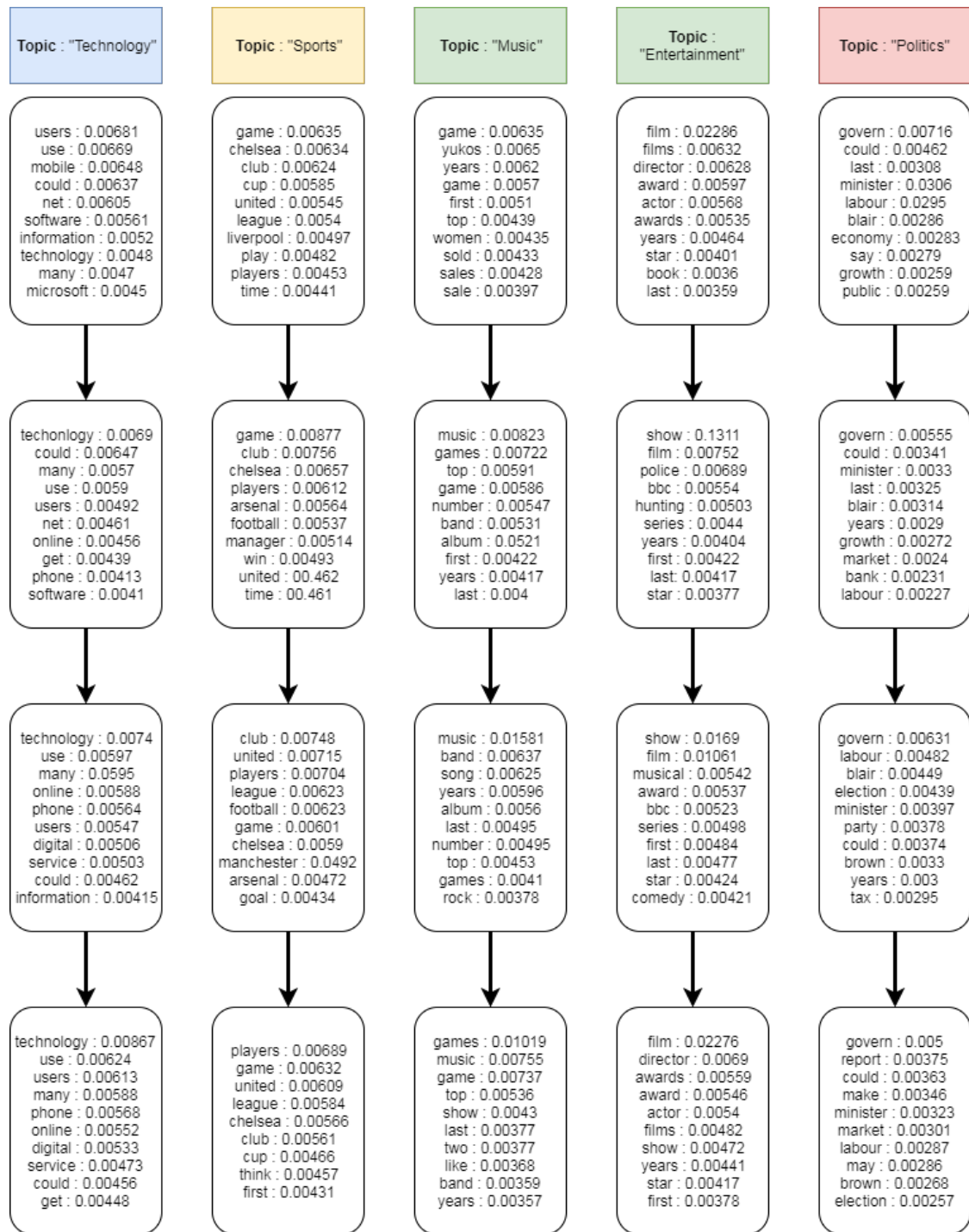


Figura 4.4: Representación gráfica de un modelo DTM entrenado con 3 intervalos de tiempos. Como se observa, los tópicos encontrados son coherentes con la descripción preliminar del corpus usado.

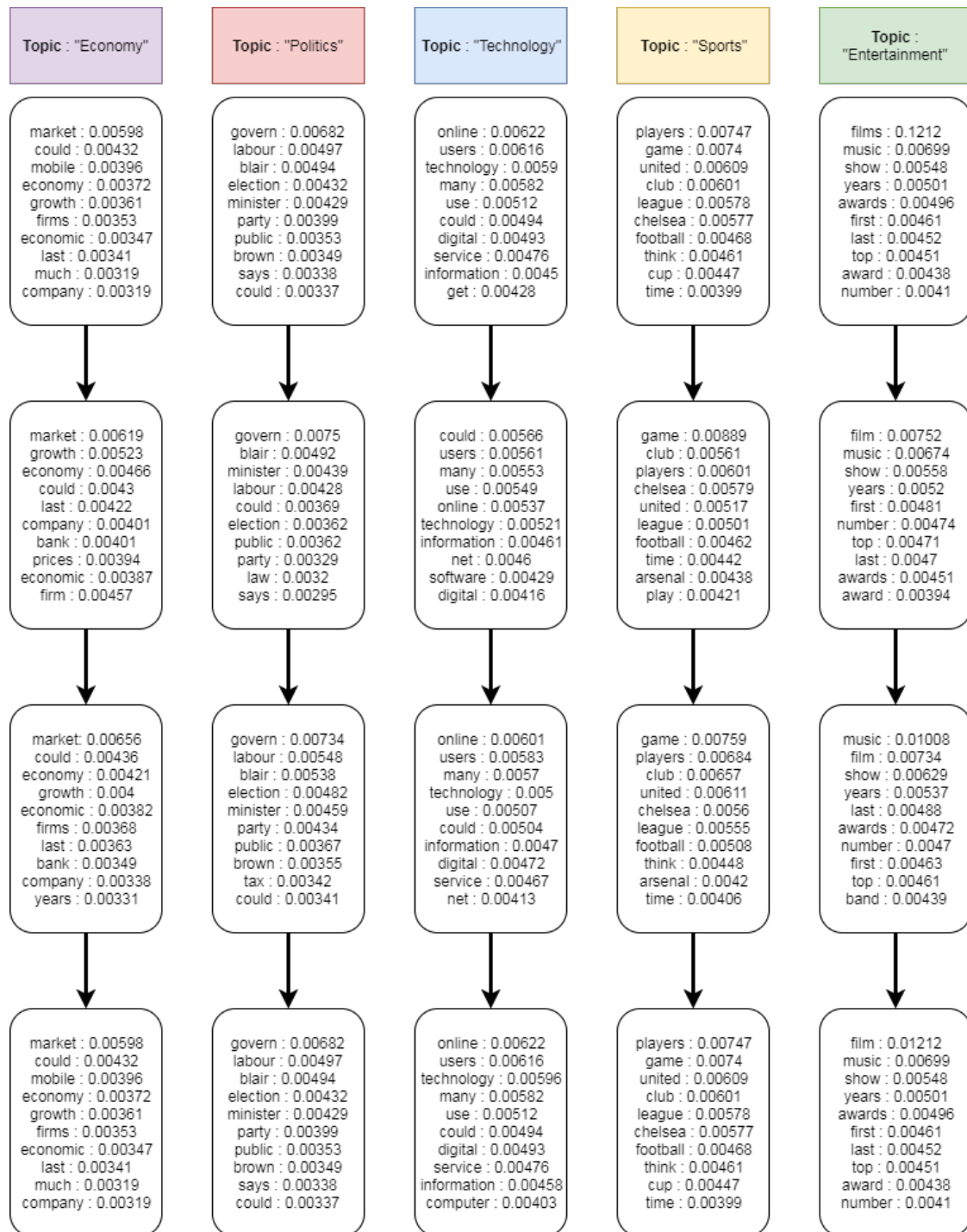


Figura 4.5: Representación gráfica de un modelo DTM entrenado con 3 intervalos de tiempos. Como se observa, los tópicos encontrados son coherentes con la descripción preliminar del corpus usado.

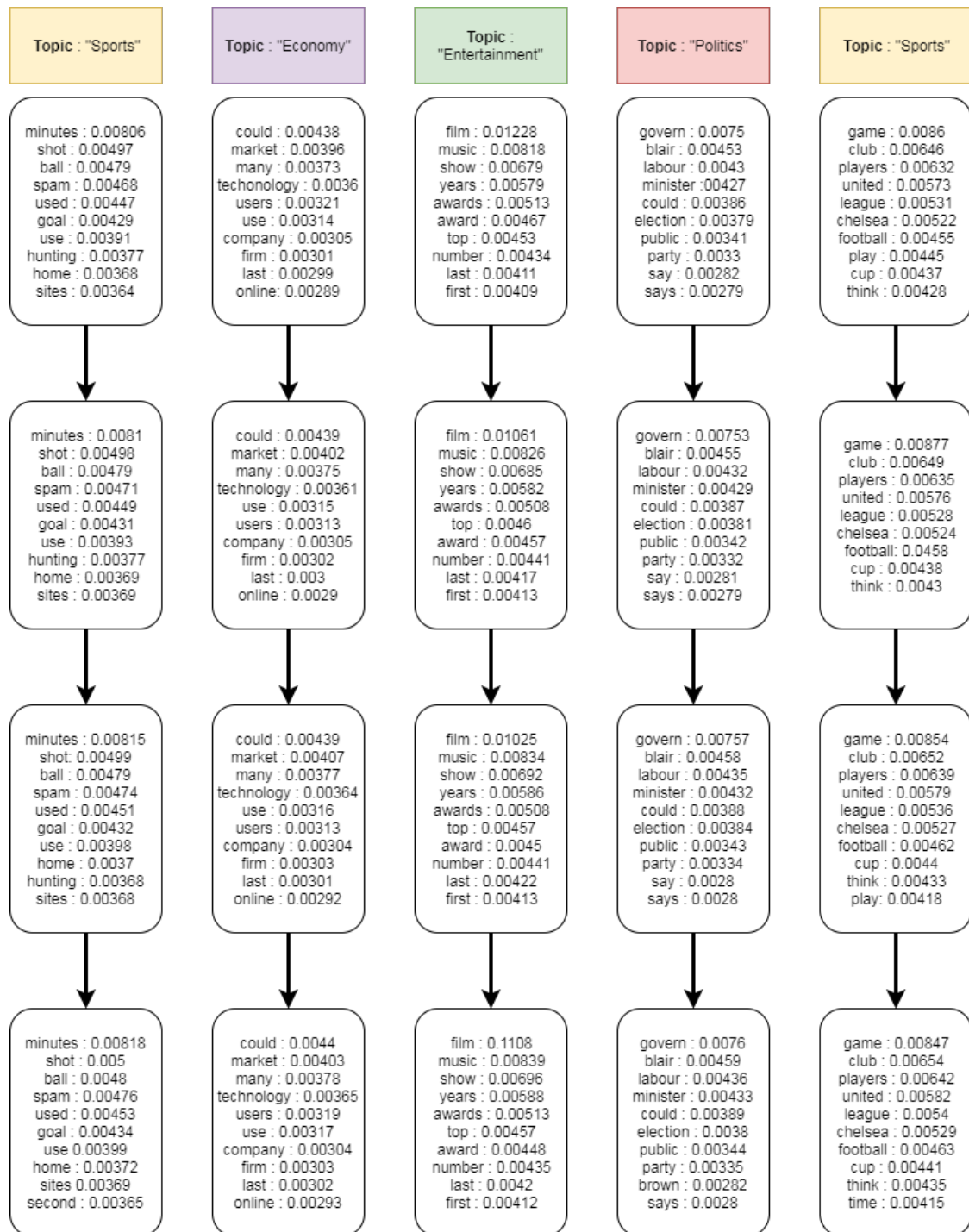


Figura 4.6: Representación gráfica de un modelo DTM entrenado con 3 intervalos de tiempos. Como se observa, los tópicos encontrados son coherentes con la descripción preliminar del corpus usado.

Como se observa en las representaciones gráficas de los modelos en las figuras 4.4 , 4.5 y 4.6, a mayor varianza, mayor es la dispersión de la distribución de palabras entre cada intervalo de tiempo para un tópico determinado, mientras que a menor varianza, la evolución de los tópicos es casi nula, manteniendo una distribución de palabras estática a lo largo del tiempo. Es importante destacar que este valor definido de varianza es global estático, por lo que no se puede determinar distintos valores para cada evolución entre intervalos de tiempo. El mayor problema con esta configuración es que si se define una evolución suave en la distribución de palabras, puede ocurrir que ciertas palabras que tengan frecuencia cero en el siguiente intervalo aun sean consideradas como una palabra relevante para un determinado tópico.

Ademas de la interpretación de los modelos basados en su distribución de términos relevantes, se realizó un análisis cuantitativo comparativo de los modelos, usando la métrica Topic Coherence. Para cada valor de varianza definido, se entrenaron 10 modelos, y se ponderó la media y la varianza de los resultados obtenidos:

T	1	2	3	4	5	6	7	8	9	10	Media	Var
T_1	-1.8743	-1.6474	-1.5050	-1.4786	-1.6842	-1.5692	-2.1548	-1.4795	-1.7635	-1.8731	-1.7101	0.0488
T_2	-2.0141	-1.7812	-1.5848	-1.5765	-1.6795	-1.6058	-1.5854	-1.5312	-1.6984	-1.7328	-1.67900	0.0201
T_3	-1.5206	-1.4685	-1.4872	-1.4754	1.5080	-1.8053	-1.6690	-1.4752	-1.7924	-1.7350	-1.5937	0.01974
T_4	-1.5747	-1.7683	-2.0285	-1.6344	-1.6238	-1.6035	-1.5565	-1.7217	-2.4152	-1.6581	-1.7585	0.0719

Cuadro 4.3: Resultados de análisis de Topic Coherence usando la medida UMass con varianza 0.1

T	1	2	3	4	5	6	7	8	9	10	Media	Var
T_1	-1.6813	-1.8494	-1.4768	-2.1339	-1.7271	-1.5313	-1.4549	-1.5092	-1.4690	-1.5576	-1.7737	0.0469
T_2	-1.8066	-1.7222	-1.5515	-2.1940	-1.8060	-1.5521	-1.5072	-1.5142	-1.5584	-1.5362	-1.8161	0.0470
T_3	-1.5039	-1.5799	-1.5511	-2.3461	-1.5982	-1.5036	1.5133	-1.4842	-1.5251	-1.4777	-1.7158	0.0687
T_4	-1.5720	-1.8739	-1.6185	-2.0688	-1.6300	-1.5510	-1.5459	-1.5213	-1.5440	-1.6780	-1.7526	0.0312

Cuadro 4.4: Resultados de análisis de Topic Coherence usando la medida UMass con varianza 0.01

T	1	2	3	4	5	6	7	8	9	10	Media	Var
T_1	-1.8453	-2.2140	-1.8386	-2.3824	-1.84342	-2.4328	-2.1711	-2.2143	-1.976	-2.0732	-2.0247	0.0595
T_2	-1.88345	-2.2549	-1.9732	-2.4376	-1.8294	-2.0582	-1.9874	-1.9249	-2.1672	-2.0643	-2.0757	0.0341
T_3	-1.9693	-2.2549	-1.9984	-2.1436	-1.9063	-2.013	-2.0103	-1.4842	-2.3709	-2.143	-2.0247	0.0172
T_4	-1.8342	-2.1359	-1.9587	-2.6745	-1.8002	-2.0345	-1.9943	-1.5213	-1.8923	-1.6423	-2.0807	0.0832

Cuadro 4.5: Resultados de análisis de Topic Coherence usando la medida UMass con varianza 0.001

Como se observa en los cuadros 4.3, 4.4 y 4.5 los mejores resultados se obtienen con una varianza igual a 0,01. A mayor varianza se obtienen resultados mas dispersos entre intervalos de tiempos. Por el otro lado, a menor varianza, el score obtenido es en la mayoría de los casos bajo y similar entre

intervalos de tiempo, dado que la poca flexibilidad otorgada no permite modelar los intervalos de tiempo adecuadamente.ubu

4.4. Visualización Modelos

A continuación se visualizarán los resultados obtenidos usando la librería interactiva de visualización de modelos de tópicos *pyLDavis*. El modelo seleccionado corresponde a uno de los resultados usando la función incremental con la colección 2 (sin noticias de política y economía). En la imagen 4.7, los tópicos y sus índices son los siguientes: 1: Deportes, 2: Economía, 3: Entretenimiento, 4: Tecnología y 5: Política.

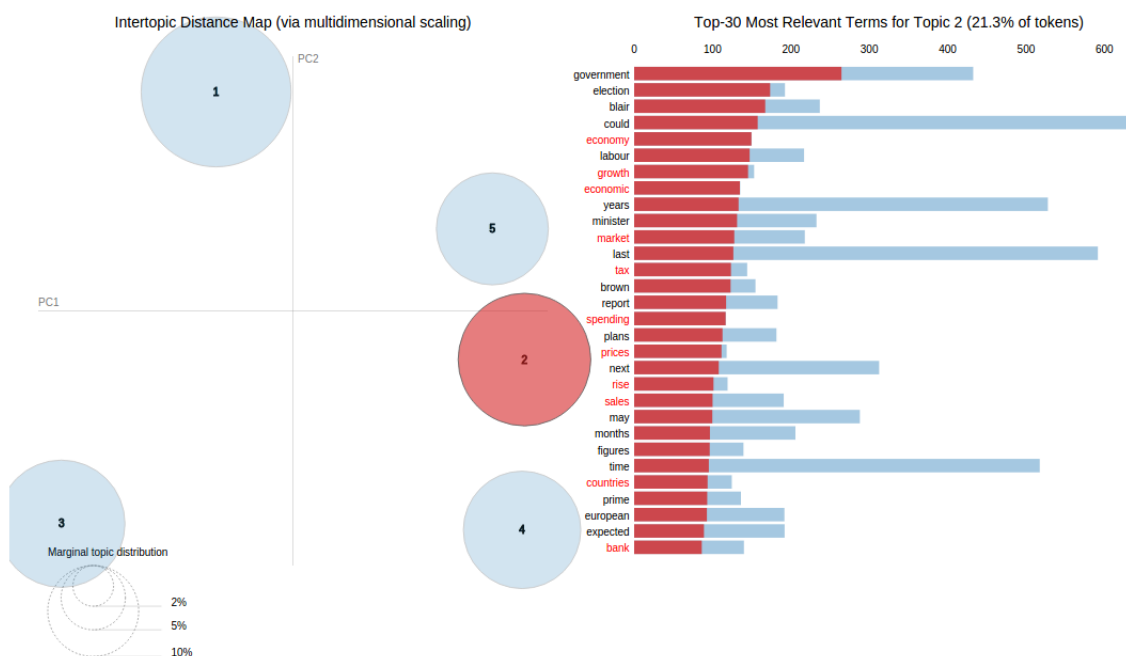


Figura 4.7: Visualización de un modelo DTM entrenado sobre un intervalo de tiempo, con énfasis en el tópico 2 relacionado con economía.

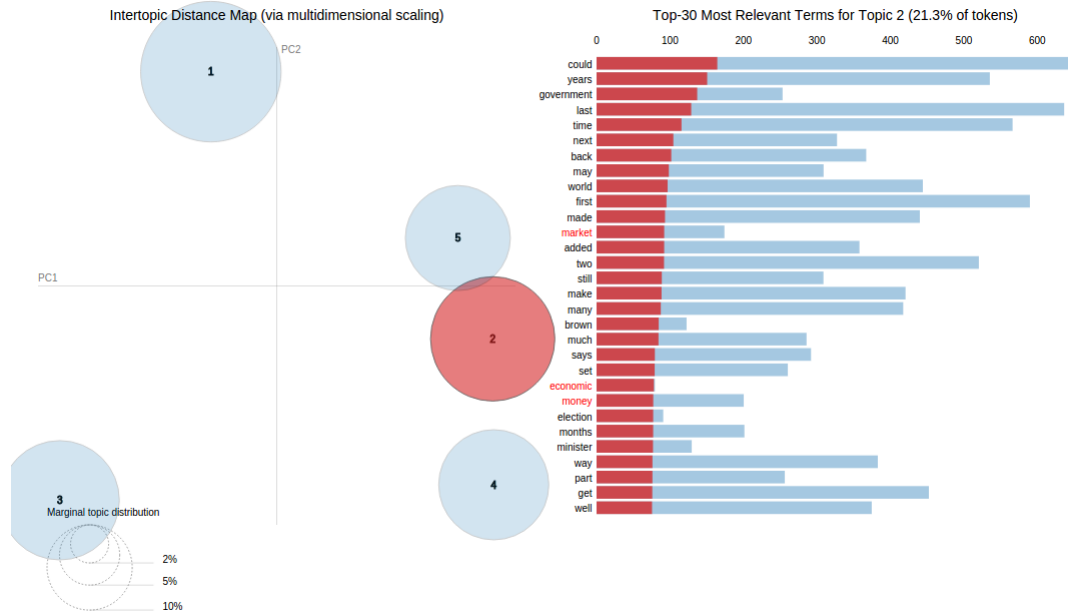


Figura 4.8: Visualización de un modelo DTM entrenado sobre el último intervalo de tiempo, el cual fue entrenado usando la función incremental.

Comparando las imágenes 4.7 y 4.8 sobre el tópico 2 relacionado con economía, se observa que la distribución de términos relevantes en el último ejemplo pierde interpretabilidad, siendo los términos más relevantes poco descriptivos dada la alta frecuencia de uso a través del tiempo, y dado que la colección de documentos usada para la función incremental no posee documentos ni de economía ni política. En consecuencia, la distancia entre estos decrece, dando a entender que la distribución Tópico-Término entre ambos tópicos comienzan a igualarse. Este evento puede interpretarse como una desaparición de tópicos a través del tiempo, por lo que puede usarse como ayuda para una determinación óptima en el número de tópicos del modelo.

Otra observación que puede realizarse entre ambas imágenes es la evolución *suave* de los tópicos y sus distancias entre tópicos. Los tres tópicos que se encuentran más cercanos entre sí son los correspondientes a economía, política y tecnología, mientras que los tópicos relacionados con deportes y entretenimiento se encuentran más alejados del resto de todos los tópicos, atribuyendo cierta singularidad en los términos captados. Al usar la función incremental, el nuevo intervalo de tiempo conserva las proporciones de distancia de los tópicos, cambiando ligeramente la relación entre ellos, por lo que se puede atribuir que el algoritmo incremental logra modelar correctamente la evolución de los tópicos desde el intervalo de tiempo anterior.

pyLDavis también nos permite modificar el parámetro λ que determina el peso entre la distribución tópico-termino y el factor *lift*. Al otorgar pesos equivalentes con $\lambda = 0.5$, las distribuciones para el tópico 2 mencionadas en las imágenes 4.7 y 4.8, se logra recuperar términos descriptivos para un tópico determinado, dado que se considera la probabilidad del término en el corpus proporcional a la probabilidad marginal del término en todo el corpus. Modificar el parámetro λ nos permite analizar y modificar la relevancia de la distribución de términos relevantes para un determinado tópico en un intervalo de tiempo y compararlo con la distribución del término en el corpus total.

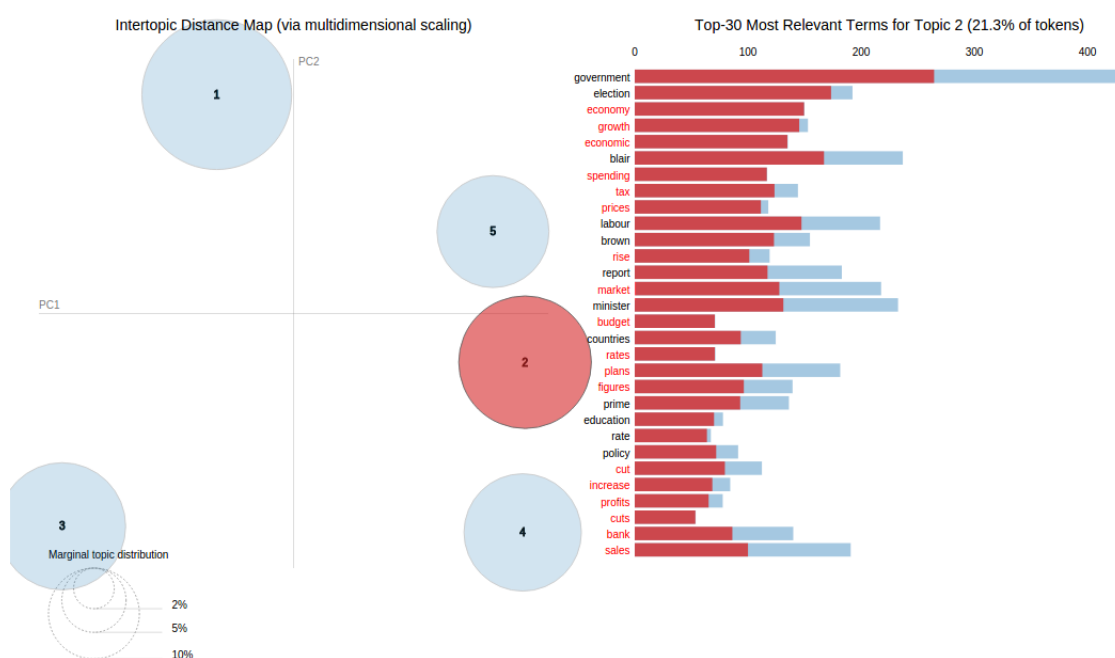


Figura 4.9: Visualización de un modelo DTM entrenado sobre el último intervalo de tiempo, el cual fue entrenado sin el uso de la función incremental.

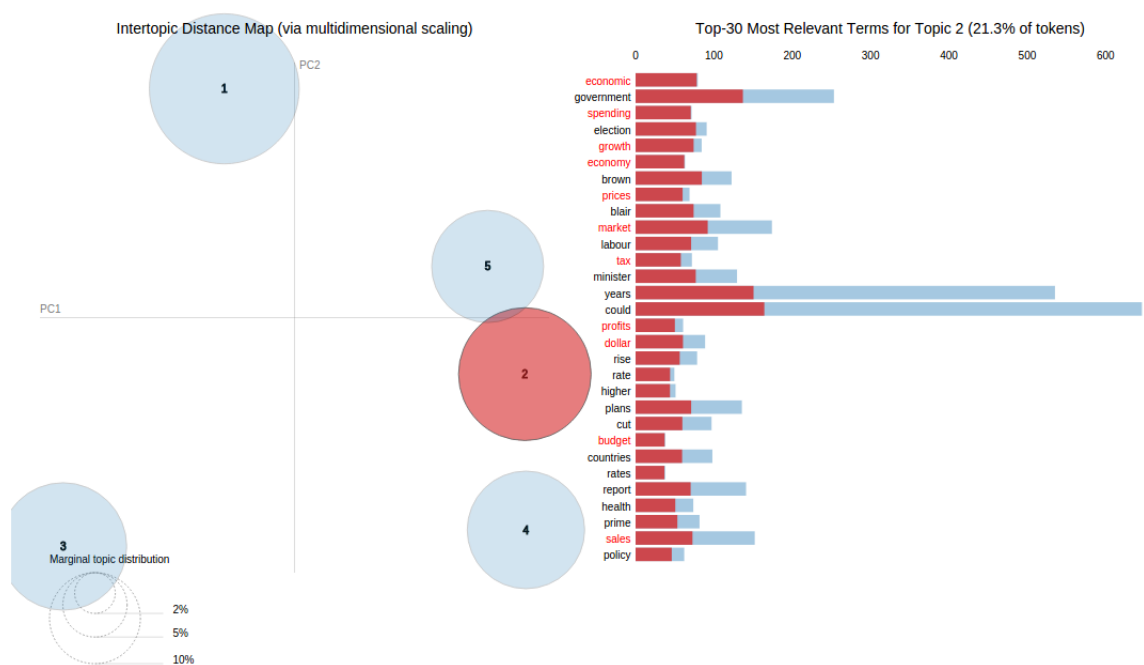


Figura 4.10: Visualización de un modelo DTM entrenado sobre el último intervalo de tiempo, el cual fue entrenado usando la función incremental.

Capítulo 5

Conclusiones

Se ha desarrollado en este trabajo una extensión del algoritmo Dynamic Topic Models, con el fin de modelar nuevos intervalos de tiempos dentro de un modelo previamente entrenado, sin necesidad de reconstruir el modelo en su totalidad, optimizando la actualización del modelo en términos de tiempo. Los resultados obtenidos nos permiten verificar que la función incremental de este modelo nos permite tener una representación más fiel de los documentos en los nuevos intervalos, dado que al poder modificar la varianza para cada nueva actualización, se tiene el control sobre la evolución de las distribuciones de términos en cada tópico.

Uno de los resultados más relevantes fue observar la desaparición de tópicos en nuevos intervalos de tiempos, al no incluir documentos relevantes a dichos tópicos, estos pierden la interpretación semántica de los términos mas relevantes. Respecto a la aparición de nuevos tópicos, la interpretación de este fenómeno se complica dado que el modelo sigue trabajando con un número fijo predeterminado de tópicos. Al incluir artificialmente documentos pertenecientes a un nuevo tópico en los intervalos de tiempo, se logra observar la aparición de nuevos términos relevantes y los tópicos comienzan a poseer semejanzas en la distribución de términos. Encontrar un patrón dentro de este fenómeno para determinar la aparición de nuevos tópicos es un factor clave para la extensión de un modelo dinámico con un número de tópicos variable en el tiempo.

Por otro lado, se ha usado un framework unificador de medidas de coherencia, con el objetivo de cuantificar y comparar la coherencia de los tópicos generados. Si bien la interpretabilidad del sentido de coherencia es difícil de modelar en tópicos, este framework nos permite evaluar todas las posibles medidas de confirmación existentes en un set de términos, ya sea en sentido de correlación, co-ocurrencia, frecuencia relativa, etc. Los resultados obtenidos sobre los modelos generados usando la medida intrínseca *UMass* revelan resultados acordes a lo esperado en la evaluación de distintos valores de varianza, donde modelos con una evolución menos pronunciada en la distribución de términos poseen un bajo score, dado que el modelo fijo no logra representar la evolución los documentos en el corpus.

Finalmente, el uso e implementación de librerías interactivas de modelos de tópicos nos permiten visualizar grandes colecciones de documentos, facilitando así la interpretación de modelos complejos debido a su alta dimensionalidad. La librería *pyLDavis* es un navegador de modelos de tópicos que, si bien está desarrollado para modelos entrenados usando LDA, tiene la flexibilidad de permitir ser usada para visualizar intervalos de tiempos por separado del modelo DTM, para su posterior análisis y comparación, logrando captar la evolución de los tópicos a través del tiempo. A través de *pyLDavis*, es posible visualizar la evolución de la distribución de los términos mas relevantes de un tópico y la relación entre tópicos (que tan semejante son entre ellos):

Capítulo 6

Trabajo Futuro

6.1. Optimización algoritmo

Una de las áreas donde se requiere más trabajo es en la optimización del proceso de inferencia de los parámetros. Actualmente el algoritmo implementado en python realiza la optimización de los parámetros usando un método del gradiente conjugado basado en el trabajo Polak-Ribiere, mientras que el algoritmo original desarrollado por Blei en el lenguaje C++ implementa el método de gradiente conjugado basado en el trabajo de Fletcher-Reeves[17]. Actualmente la librería en python no posee una implementación eficiente del método usado en el algoritmo original, por lo que una propuesta para la optimización es crear un *wrapper* usando el compilador Cython para realizar la optimización de los parámetros y evaluar tiempos de ejecución.

6.2. Determinación Varianza

El proceso incremental desarrollado para el modelo, nos permite modificar el valor de la varianza para cada actualización. Si bien este proceso nos permite tener más control de la evolución de la distribución de términos, este proceso es manual, por lo que la determinación del mejor valor para la varianza depende de una observación cualitativa del usuario. Una posibilidad es determinar el mejor valor para la varianza realizando un proceso previo al entrenamiento de cada intervalo de tiempo en el modelo acorde a la comparación de frecuencia de términos, logrando la auto sintinización de este parámetro.

6.3. Aparición y/o Desaparición de Tópicos

En los experimentos realizados, fue posible observar como los tópicos pierden poder interpretativo a través del tiempo si ciertos términos y/o documentos desaparecen, pero el modelo solo está implementado con un número fijo de tópicos. Una mejora consiste en incorporar en el modelo un sistema dinámico respecto al número de tópicos a través del tiempo. Para lograr tal objetivo, es necesario considerar métricas y criterios para determinar la aparición y/o desaparición de tópicos a través del tiempo. Una posibilidad propuesta por Blei y Lafferty es incorporar el proceso estocástico de Galton-Watson sobre la presencia de los tópicos en un intervalo de tiempo determinado [3].

Bibliografía

- [1] Christopher M Bishop. Approximate inference. In Jon Kleinberg Michael Jordan and Bernhard Scholkopf, editors, *Pattern recognition and machine learning*, chapter 10, pages 461–522. springer, 2006.
- [2] Christopher M Bishop. Sequential data. In Jon Kleinberg Michael Jordan and Bernhard Scholkopf, editors, *Pattern recognition and machine learning*, chapter 13, pages 605–652. springer, 2006.
- [3] David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120. ACM, 2006.
- [4] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
- [5] Allison June-Barlow Chaney and David M Blei. Visualizing topic models. In *ICWSM*, 2012.
- [6] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296, 2009.
- [7] Jason Chuang, Christopher D Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the international working conference on advanced visual interfaces*, pages 74–77. ACM, 2012.
- [8] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

- [9] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.
- [10] YC Ho and Robert Lee. A bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, 9(4):333–339, 1964.
- [11] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.
- [12] Rudolph Emil Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.
- [13] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge university press, 2014.
- [14] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 569–577. ACM, 2008.
- [15] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408. ACM, 2015.
- [16] Matt Taddy. On estimation and selection for topic models. In *International Conference on Artificial Intelligence and Statistics*, pages 1184–1193, 2012.
- [17] Stephen J Wright and Jorge Nocedal. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.