

# Propuesta de Proyecto Final curso RecSys 2016

- Clustering de documentos con social tags
  - Lucas Zorich, PUC Chile
  - Mauricio Troncoso, PUC Chile

# *Clustering de documentos con social tags*

Mauricio Troncoso, Lukas Zorich

Pontificia Universidad Católica de Chile

*{mjtroncoso,lezorich}@uc.cl*

27 de septiembre de 2016



# Contexto



# Contexto

Hacer *clustering* de documentos permitiría:

- ▶ Agrupar los resultados de búsquedas en categorías
- ▶ Entregar diversidad en los resultados de las búsquedas
- ▶ Mejorar la presentación de la información al usuario



# Problema y solución propuesta

El principal problema estriba en cómo detectar el **tópico** del que trata cada documento.



# Problema y solución propuesta

Usar *tags* como información adicional:

- ▶ Los *tags* entregan información semántica importante.
- ▶ Mucho contenido en internet tiene *tags*.



# Problema y solución propuesta

Compararemos diferentes algoritmos de *clustering* en documentos utilizando *tags* como información adicional.



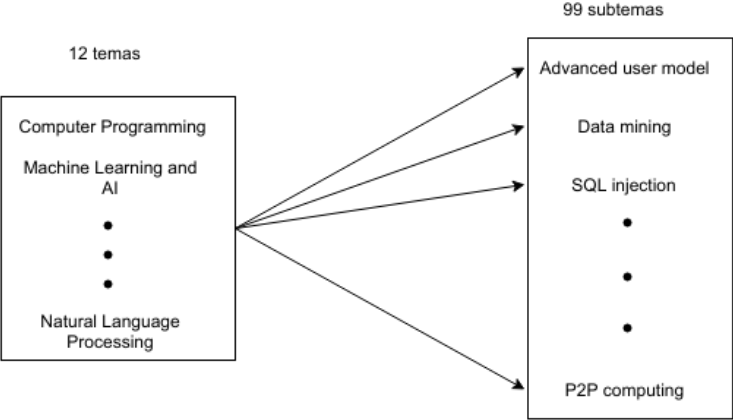
# Objetivo

¿Los *tags* de usuarios son una fuente significativa de información al hacer *clustering* de documentos?





# Dataset



# Dataset

- ▶ Cada subtema es considerado como una *query* ejecutada en CiteULike.
- ▶ Estudiantes de postgrado expertos clasificaron el resultado de la *query* como:
  - ▶ No relevante
  - ▶ Poco relevante
  - ▶ Muy relevante
- ▶ Los documentos relevantes de cada *query* será el *ground truth*.
- ▶ Los *tags* se conseguirán haciendo *scraping* de los documentos en CiteULike.



# Algoritmos a comparar

- ▶ K-Means
- ▶ Non-negative Matrix Factorization
- ▶ LDA
- ▶ MM-LDA (opcional)



# Experimentos

Por cada algoritmo, se probarán diferentes *features*:

- ▶ Solo palabras
- ▶ Solo *tags*
- ▶ *Tags* como palabras por  $n$



# Evaluación

- ▶ Siguiendo a (Ramage *et al*, 2008) ocuparemos el *Cluster-F1 score* como métrica de evaluación.



PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE  
ESCUELA DE INGENIERÍA  
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

---

PROPUESTA DE PROYECTO:  
Una comparación de diferentes técnicas de  
*clustering* de documentos

---

IIC3633 SISTEMAS RECOMENDADORES

Mauricio Troncoso    mjtroncoso@uc.cl  
Lukas Zorich        lezorich@uc.cl

## Índice

<b>1. Contexto del Problema</b>	<b>2</b>
<b>2. Problema y propuesta de solución</b>	<b>2</b>
<b>3. Objetivos</b>	<b>2</b>
<b>4. Definición de experimentos</b>	<b>2</b>
4.1. Conjunto de datos . . . . .	2
4.2. Algoritmos a comparar . . . . .	3

## 1. Contexto del Problema

Actualmente, los motores de búsqueda modernos están encargados de devolver los resultados más relevantes en el momento en el que un usuario hace una *query*, la que generalmente es bastante ambigua. Estos resultados más relevantes se tienen que encontrar de entre millones de documentos, lo que no es una tarea fácil.

La gran mayoría de las veces, para encontrar los resultados más relevantes el motor de búsqueda utiliza técnicas de *ranking* basadas en las conexiones entre *links*, textos de los *links*, *clicks* del usuario en los distintos datos, o buscar en el texto de la página. Sin embargo, el mayor desafío es manejar la ambigüedad que existe en la *query* del usuario. Esta *query* generalmente tiene pocas palabras y puede tener varios significados. Una manera de manejar esta ambigüedad es mediante *clustering* del contenido. Si una *query* ambigua puede tener varios significados, y asumiendo que un significado es un *cluster* de documentos de un mismo tópico, se le puede devolver al usuario documentos de varios de estos tópicos lo que aumentaría la diversidad de la respuesta. Por otro lado, si se sabe el significado de la *query* del usuario, se podrían obtener fácilmente documentos relevantes, ya que se sabría que el usuario quiere documentos de un tópico en particular. Por lo tanto, poder agrupar distintos documentos que pertenezcan a un mismo tópico es fundamental para que el usuario pueda obtener información relevante fácilmente.

## 2. Problema y propuesta de solución

La principal problemática que se busca atacar con este proyecto, es la de identificar tópicos de interés en documentos y poder usarlos para hacer *clusters* de documentos, lo que permitiría a los usuarios encontrar más fácilmente documentos que le interesan. El problema entonces, se representa por la pregunta: cuál es la mejor forma de hacer *clustering* de documentos?.

La solución propuesta entonces, es comparar diferentes técnicas que existen hoy en día para agrupar distintos documentos.

## 3. Objetivos

El objetivo de este proyecto es poder comparar diferentes algoritmos de *clustering* de documentos. Además, un objetivo ideal, aunque se dejará opcional, sería el de responder la pregunta: son los tags entregados por usuarios en sitios como CiteULike una fuente significativa de información al intentar hacer un *clustering* de documentos?.

## 4. Definición de experimentos

### 4.1. Conjunto de datos

Los experimentos se correrán en el *dataset* CiteEval [2]. Este *dataset* fue creado definiendo 12 áreas de interés, o categorías, dentro de la ciencia de la computación y ciencia de la información. Dentro de estas categorías se encuentran las áreas de Aprendizaje de Máquinas e IA, Bases de Datos, Arquitectura de Computadores, entre otras. Para cada una de estas categorías se definieron un subconjunto de tópicos representados como *queries*, las cuales son 99 en total. Cada una de estas 99 *queries* fueron ejecutadas en el sitio CiteULike. El resultado de cada *query* fue guardado en una colección de documentos que incluyen el id del documento, el título y el *abstract*. Para



saber la relevancia de cada documento en la *query*, se les preguntó a estudiantes de postgrado expertos en la categoría o subtópico de esa *query*. Se ocupará esta relevancia como el *ground truth* al evaluar los diferentes algoritmos.

## 4.2. Algoritmos a comparar

Los algoritmos que se compararán son:

- *K-means*.
- *Hierarchical clustering*. En particular, se utilizará `AgglomerativeClustering`.
- *Latent dirichlet allocation* (LDA).

Para el algoritmo *K-means* y *Hierarchical clustering*, se ocupará `tfidf` para representar cada palabra.

Además, dejaremos como opcional probar el algoritmo MM-LDA [1]. Para poder probar este algoritmo, será necesario extraer tags disponibles desde la página de CiteULike para los documentos que están en nuestro conjunto de datos.

## Referencias

- [1] Ramage, D., Heymann, P., Manning, C. D., & Garcia-Molina, H. (2009, February). Clustering the tagged web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining* (pp. 54-63). ACM.
- [2] Yue, Z., Harpale, A., He, D., Grady, J., Lin, Y., Walker, J., ... & Yang, Y. (2009, July). CiteEval for evaluating personalized social web search. In *SIGIR 2009 Workshop on the Future of IR Evaluation* (p. 23).

# Recomendación de artículos académicos utilizando etiquetas sociales

Lukas Zorich  
Departamento de Ciencia de la Computación  
Pontificia Universidad Católica de Chile  
lezorich@uc.cl

Mauricio Troncoso  
Departamento de Ingeniería Matemática  
Pontificia Universidad Católica de Chile  
mjtroncoso@uc.cl

## ABSTRACT

El almacenamiento digital de material científico ha dificultado considerablemente la tarea, de los investigadores, de encontrar trabajos pertinentes a su labor académica, luego, es pertinente idear métodos de recomendación que se ajusten a las nuevas necesidades del mundo de la academia. Para esto se emplea un método ya utilizado para esta labor que consiste en recomendar material nuevo y antiguo en base a, respectivamente, modelación de tópicos y filtrado colaborativo. En este trabajo se extiende la labor más reciente sobre la base de datos de CiteULike, considerando la información entregada por las etiquetas que los mismos usuarios asocian a cada artículo de su interés. Se propone el modelo MM-CTR que permite agregar etiquetas de los documentos al hacer recomendaciones. De acuerdo a los experimentos, se comprueba que el uso de etiquetas permite hacer mejores recomendaciones.

## 1. INTRODUCCIÓN

Usualmente, los investigadores realizan las búsquedas de artículos que son de su interés siguiendo las citaciones de aquellos que ya han estudiado, esto es completamente sensato pero sus resultados están inevitablemente sesgados hacia artículos altamente citados, además, de esta forma es muy poco probable que haya movimiento entre disciplinas, esto es, que desde un trabajo, por ejemplo, de biología se llegue a otro relacionado, en el campo de la estadística.

Otra alternativa es la búsqueda a través de palabras clave, que a pesar de ser una herramienta poderosa presenta problemas en los casos en los que el investigador no sabe realmente qué está esperando encontrar, además, las búsquedas son realizadas principalmente en base a contenido y no consideran que un artículo valioso es uno que también es valorado por otros usuarios. Al respecto, recientemente sitios como CiteULike<sup>1</sup> han permitido a los investigadores crear sus propias librerías con los artículos que son de su interés agregándoles *tags* descriptivos para ser compartidos

<sup>1</sup><http://www.citeulike.org>

por la comunidad. El objetivo del presente trabajo es recomendar a cada usuario un conjunto de artículos que no figuran en su librería personal y dar cuenta de qué forma las etiquetas mejoran la recomendación, ya que al ser ingresadas por los mismos usuarios representan una personalización de la evaluación hecha a cada documento.

Al momento de realizar las recomendaciones se considerarán dos importantes criterios: los usuarios quieren artículos antiguos cuando están buscando introducirse en una nueva área de estudios o cuando quieren aprender los fundamentos de ésta, al respecto, los trabajos importantes estarán en las librerías de muchos usuarios. Por otro lado, los investigadores también necesitan ser informados de las publicaciones más recientes de su disciplina que, al no tener muchas lecturas, deben ser recomendadas en base al contenido de las mismas. Así, se hará uso de dos tipos de datos: el contenido de los artículos y las librerías de otros usuarios que, inherentemente, representan una evaluación de los documentos. En ese sentido, el modelo propuesto que se explicará más adelante es un modelo híbrido, ya que ocupa filtrado colaborativo y el contenido de los documentos para hacer la recomendación. La recomendación es hecha en base a filtrado colaborativo cuando el archivo ha sido evaluado por uno o más usuarios previamente, por otro lado, para el caso de archivos sin evaluación se utilizará modelación de tópicos (*topic modeling*) para permitir recomendaciones acordes a los intereses que ya ha mostrado el usuario.

## 2. ESTADO DEL ARTE

El presente trabajo aborda dos casos en los que se realizan recomendaciones, primero cuando se tienen documentos que han sido evaluados anteriormente, y segundo cuando se tienen nuevos items sin evaluación previa, este caso es conocido como el *cold start problem*.

Para la recomendación de artículos académicos con y sin evaluaciones previas, en [1] se introduce el método seguido en este trabajo, pero sin la consideración de las etiquetas que permite introducir CiteULike. En [8] agregan las etiquetas y la variable temporal para realizar recomendaciones sobre, entre otras, una base de datos de CiteULike y obtienen un mejor comportamiento en la mayoría de las métricas. Por otra parte, en [7], introducen etiquetas para recomendar páginas web y proponen distintos espacios para la consideración de dichas etiquetas, en este ámbito, en [9] realizan recomendaciones personalizadas de páginas web utilizando la información de las etiquetas para crear los grupos que guían la recomendación.

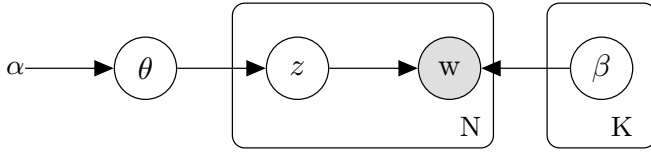


Figure 1: Modelo gráfico de LDA.

### 3. BACKGROUND

El método propuesto utiliza la factorización matricial usual explicada en [2] para el caso en el que el artículo al que se le desea predecir la evaluación ha sido evaluado con anterioridad, a este caso le llamaremos, siguiendo a [1], *in-matrix prediction*. En el otro caso, en el que no hay evaluaciones previas, *out-of-matrix prediction*, la recomendación es realizada a través de un modelamiento de tópicos como se explicará en la siguiente sección.

#### 3.1 Modelos probabilísticos de tópicos

Modelar tópicos, en este caso, tiene como objetivo dar cuenta de cómo se generan documentos, a través del descubrimiento de tópicos sobre una base de datos con muchos de ellos. El término probabilístico aparece porque cada tópico es representado como una distribución de probabilidad sobre un vocabulario que está centrada en las palabras más representativas del tópico en cuestión.

El modelo más simple de modelación probabilística de tópicos [1] es el llamado LDA por sus siglas en inglés (*Latent Dirichlet Allocation*), cuyo funcionamiento consta principalmente de dos etapas, una asignación de  $K$  tópicos para toda la base de datos, y una asignación de palabras para cada tópico, estos pasos se describen a continuación y se grafican en la Figura 1.

Para cada uno de los  $J$  documentos:

- Obtener una distribución de tópicos  $\theta_j \sim \text{Dir}(\alpha)$ , donde  $\text{Dir}(\cdot)$  es una distribución Dirichlet uniforme con parámetro  $\alpha$ .
- Para cada palabra  $w_{jn}$  en el documento:
  - Obtener un tópico específico  $z_{jn} \sim \text{multi}(\theta_j)$ , donde  $\text{multi}(\cdot)$  es una distribución multinomial de parámetro  $W$  (la cantidad total de palabras). ón de la evaluación hecha a cada documento.
  - Obtener una palabra  $w_{jn} \sim \beta_{z_{jn}}$ .

El proceso anterior clarifica de qué forma se hace que cada palabra en un documento provenga de una mezcla de tópicos.

#### 3.2 Multi-Multinomial LDA

El modelo MM-LDA (Multi-Multinomial LDA) [7], busca incluir la información que entregan las etiquetas ingresadas por los usuarios en CiteULike para realizar la recomendación, básicamente agregando una variable a LDA, explicado en la sección anterior, para la generación de etiquetas desde los  $K$  tópicos iniciales. El proceso se muestra a continuación y se grafica en la Figura 2.

Para cada uno de los  $J$  documentos:

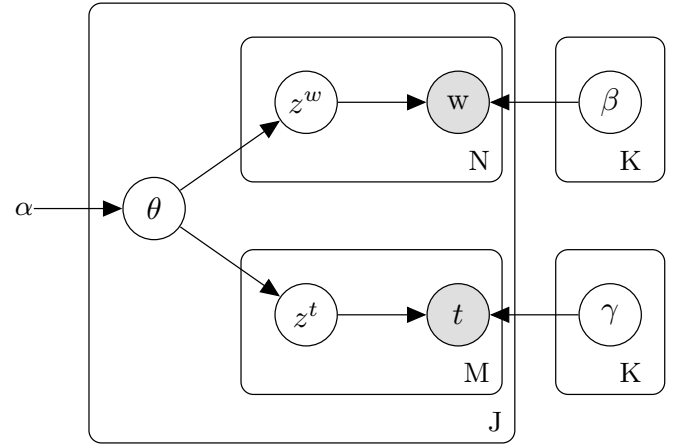


Figure 2: Modelo gráfico de MM-LDA.

- Obtener una distribución de tópicos  $\theta_j \sim \text{Dir}(\alpha)$ .
- Para cada tópico  $k = 1, \dots, K$  obtener una distribución multinomial  $\beta_k$  de tamaño  $W$ .  $\beta_k$  es la probabilidad de observar cada palabra, dado el tópico  $k$ .
- Para cada tópico  $k = 1, \dots, K$  obtener una distribución multinomial  $\gamma_k$  de tamaño  $T$  (cantidad total de etiquetas).  $\gamma_k$  es la probabilidad de observar cada etiqueta, dado el tópico  $k$ .
- Para cada palabra  $w_{jn}$  del documento  $j$ ,
  - Obtener un tópico  $z_{jn}^w \sim \text{Multi}(\theta_j)$ .
  - Obtener una palabra  $w_{jn} \sim \beta_{z_{jn}^w}$ .
- Para cada etiqueta (tag)  $t_{jm}$  del documento  $j$ .
  - Obtener un tópico  $z_{jm}^t \sim \text{Multi}(\theta_j)$ .
  - Obtener una etiqueta  $t_{jm} \sim \text{Multi}(\gamma_{z_{jm}^t})$ .

#### 3.3 Collaborative Topic Regression

El modelo *Collaborative Topic Regression* (CTR) combina filtrado colaborativo con modelamiento de tópicos. Es importante destacar que los "ítems" del filtrado colaborativo y los "documentos" de LDA se refieren a lo mismo en este trabajo, y por lo tanto se usarán ambos términos de manera de manera intercambiable.

Al igual que en LDA, en CTR a cada documento se le asigna una proporción de tópicos  $\theta_j$ , que es usado para generar las palabras. Luego, ocupa esta distribución de tópicos para generar el vector latente de los ítems  $v_j$ . El proceso generativo de CTR es el siguiente:

1. Para cada usuario  $i$ , se obtiene el vector latente del usuario  $u_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K)$ .
2. Para cada ítem  $j$ ,
  - a) Se obtiene la distribución de tópicos  $\theta_j \sim \text{Dirichlet}(\alpha)$ .
  - b) Se obtiene el *offset* latente del ítem  $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$  y se asigna el vector latente del ítem el valor  $v_j = \epsilon_j + \theta_j$ .

- c) Para cada palabra  $w_{jn}$ ,
- 1) Se obtiene el t3pico  $z_{jn}^w \sim \text{Cat}(\theta_j)$ .
  - 2) Se obtiene la palabra  $w_{jn} \sim \text{Cat}(\beta_{z_{jn}^w})$ .

3. Para cada par usuario-3tem  $(i, j)$ , se obtiene el *rating*  $r_{ij} \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1})$ , donde  $c_{ij}$  es el par3metro de precisi3n para  $r_{ij}$ .

#### 4. MULTI-MULTINOMIAL COLLABORATIVE TOPIC REGRESSION

El modelo multi-multinomial *collaborative topic regression* es una extensi3n al modelo CTR que permite agregar la informaci3n de las etiquetas de los documentos para representar de mejor manera los t3picos latentes.

Un primer *approach* para agregar informaci3n de los *tags* al modelo, es utilizar esta informaci3n como parte del mismo documento. Sin embargo, en [7] se estudio esto y se comprob3 que hay mejores formas de agregar esta informaci3n extra. Inspirado en el modelo propuesto propuesto en [7], se decidi3 tomar las palabras y los *tags* de los documentos como un conjunto de observaciones independientes. El proceso generativo de MM-CTR es el siguiente:

1. Para cada t3pico  $k$ ,
  - a) Se obtiene la distribuci3n de palabras  $\beta_k \sim \text{Dirichlet}(\eta_w)$ .
  - b) Se obtiene la distribuci3n de *tags*  $\gamma_k \sim \text{Dirichlet}(\eta_t)$ .
2. Para cada usuario  $i$ , se obtiene el vector latente del usuario  $u_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K)$ .
3. Para cada 3tem  $j$ ,
  - a) Se obtiene la distribuci3n de t3picos  $\theta_j \sim \text{Dirichlet}(\alpha)$ .
  - b) Se obtiene el *offset* latente del 3tem  $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K)$  y se asigna el vector latente del 3tem el valor  $v_j = \epsilon_j + \theta_j$ .
  - c) Para cada palabra  $w_{jn}$ ,
    - 1) Se obtiene el t3pico  $z_{jn}^w \sim \text{Cat}(\theta_j)$ .
    - 2) Se obtiene la palabra  $w_{jn} \sim \text{Cat}(\beta_{z_{jn}^w})$ .
  - d) Para cada *tag*  $t_{jm}$ ,
    - 1) Se obtiene el t3pico  $z_{jm}^t \sim \text{Cat}(\theta_j)$ .
    - 2) Se obtiene el *tag*  $t_{jm} \sim \text{Cat}(\gamma_{z_{jm}^t})$ .
4. Para cada par usuario-3tem  $(i, j)$ , se obtiene el *rating*  $r_{ij} \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1})$ , donde  $c_{ij}$  es el par3metro de precisi3n para  $r_{ij}$ .

El modelo gr3fico de la propuesta se puede ver en la Fig. 3.

##### 4.1 Inferencia

Calcular la posterior completa de  $u_i$ ,  $v_i$  y  $\theta_j$  es intratable. Por lo tanto, se maximizar3 la *log likelihood* de  $U$ ,  $V$ ,  $\Theta = \theta_{1:J}$  y  $R$ , dado  $\lambda_u$ ,  $\lambda_v$  y  $\beta$ ,

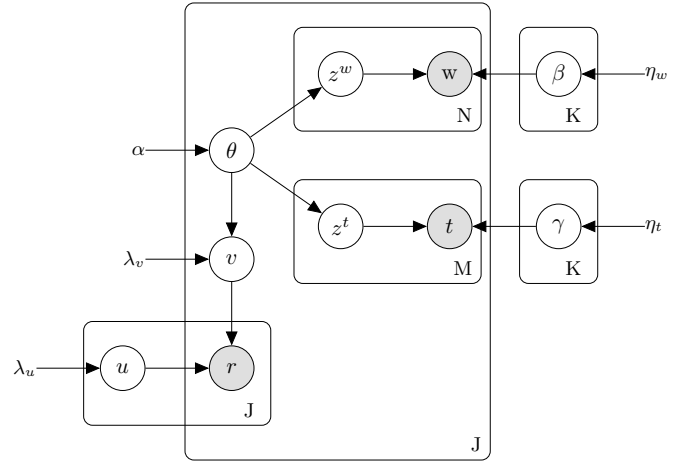


Figure 3: Modelo gr3fico de MM-CTR.

$$\begin{aligned} \mathcal{L} = & \sum_j \sum_m \log \left[ \sum_k \gamma_{kt_{jm}} \theta_{jk} \right] + \sum_j \sum_n \log \left[ \sum_k \beta_{kw_{jn}} \theta_{jk} \right] \\ & - \frac{\lambda_u}{2} \sum_i u_i^T u_i - \frac{\lambda_v}{2} \sum_j (v_j - \theta_j)^T (v_j - \theta_j) \\ & - \sum_i \sum_j \frac{c_{ij}}{2} (r_{ij} - u_i^T v_j)^2 \end{aligned} \quad (1)$$

Se omiti3 una constante y se dej3  $\alpha = 1$ . Una de las formas que se puede optimizar esta funci3n es utilizando ascenso coordinado (*coordinate ascent*), optimizando iterativamente primero  $u_i$  y  $v_i$ , y luego la proporci3n de t3picos  $\theta_j$ . Otra forma, es obtener primero un  $\theta_j$  estimado a partir de los documentos y los *tags*, y luego dejar fijo  $\theta_j$  al optimizar  $\mathcal{L}$ . Con los datos que se est3n ocupando, esto no tiene mayores p3rdidas de *performance* y ahorra tiempo de computaci3n [1]. En este trabajo se optimizar3  $\mathcal{L}$  utilizando la segunda opci3n debido a la necesidad urgente de tener resultados r3pido.

Para estimar  $\theta_j$ , se pueden utilizar varias t3cnicas. Inferencia Variacional [5] y Gibbs *sampling* [6] son las t3cnicas generales que m3s se utilizan para aprender los par3metros en este tipo de modelo. Se decidi3 ocupar Gibbs *sampling* debido a que es m3s f3cil de implementar. En Gibbs *sampling*, secuencialmente se *samplean* todas las variables latentes condicionadas en todas las dem3s. En otras palabras, la probabilidad de que un t3pico  $k$  sea asignado a una palabra  $w_n$  depende de la probabilidad de que se haya asignado ese mismo t3pico a todas las otras palabras en todos los otros documentos. Formalmente, estas probabilidades se pueden escribir de la siguiente manera

$$p(z_i^w = k | \cdot) \propto \frac{C_{w_n, k}^{WT} + \eta_w}{\sum_{w=1}^W C_{wk}^{WT} + W\eta_w} \frac{C_{d_i, k}^{DT} + \alpha}{\sum_{k=1}^K C_{d_i, k}^{DT} + K\alpha} \quad (2)$$

$$p(z_i^t = k | \cdot) \propto \frac{C_{t_n, k}^{TT} + \eta_t}{\sum_{t=1}^M C_{tk}^{TT} + M\eta_t} \frac{C_{d_i, k}^{DT} + \alpha}{\sum_{k=1}^K C_{d_i, k}^{DT} + K\alpha} \quad (3)$$

donde  $C_{wk}^{WT}$  es el número de palabras  $w$  asignadas al tópico  $k$ ,  $C_{tk}^{TT}$  es el número de *tags*  $t$  asignados al tópico  $k$  y  $C_{dk}^{DT}$  es el número de documentos  $d$  asignados al tópico  $k$ . Finalmente, la distribución de tópicos  $\theta$  se puede estimar utilizando la siguiente expresión

$$\theta_d = \frac{C_{dk}^{DT} + \alpha}{\sum_{k=1}^K C_{dk}^{DT} + K\alpha} \quad (4)$$

donde  $C_{dk}^{DT}$  es el número de documentos  $d$  asignados al tópico  $k$ . Notar que la expresión para estimar  $\theta_d$  es la misma que en [6].

Dado  $\theta_j$ , se puede estimar  $u_i$  y  $v_j$  de manera similar a como se hace al hacer factorización matricial [4]. Calculando el gradiente de  $\mathcal{L}$  con respecto a  $u_i$  y  $v_j$ , y luego igualando a cero, se obtiene

$$u_i \leftarrow (VC_iV^T + \lambda_u I_k)^{-1} VC_i R_i \quad (5)$$

$$v_j \leftarrow (UC_jU^T + \lambda_v I_k)^{-1} (UC_j R_j + \lambda_v \theta_j), \quad (6)$$

donde  $U = (u_i)_{i=1}^I$ ,  $V = (v_j)_{j=1}^J$ ,  $C_i$  es una matriz diagonal con  $c_{i,j}$   $j = 1, \dots, J$  como sus elementos diagonales y  $R_i = (r_{ij})_{j=1}^J$  para el usuario  $i$ . En la ecuación 6 se puede observar como afecta la proporción de tópicos del documento,  $\theta_j$ , al vector latente  $v_j$ , donde  $\lambda_v$  actúa como el regularizador.

## 4.2 Predicción

Después de estimar los parámetros  $U$ ,  $V$  y  $\Theta$ , la predicción se hace igual que en el modelo CTR, es decir, para *in-matrix prediction*, se puede estimar  $r_{ij}$  con

$$r_{ij} \approx (u_i)^T (\theta_j + \epsilon_j) = u_i^T v_j \quad (7)$$

donde  $v_j = \theta_j + \epsilon_j$ . Para una predicción *out-of-matrix*, el artículo es nuevo y por lo tanto no hay *ratings* disponibles. Luego,  $r_{ij}$  se calcula como

$$r_{ij} \approx u_i^T \theta_j \quad (8)$$

## 5. EXPERIMENTOS

### 5.1 Dataset

La base de datos utilizada es la misma que en [1]. Este *dataset* tiene usuarios y documentos que esos usuarios tienen en sus librerías de CiteULike<sup>2</sup>. A este *dataset*, además se

<sup>2</sup><http://www.citeulike.org/>

le agregó la información de las etiquetas entregadas por los usuarios, que fueron obtenidas entrecruzando la primera base de datos con el *snapshot*<sup>3</sup> de CiteULike.

Debido a que se contaba con pocos recursos computacionales, el *dataset* original fue reducido. El resumen del *dataset* se puede observar en la Tabla 1.

**Table 1: Resumen de la base de datos.**

Ítem	# de instancias
Documentos	5.000
Usuarios	1.749
Pares Usuario-Documento	8.745.000
<i>tags</i>	56.077

Con respecto a los artículos, se concatenó el título y el *abstract* de cada uno. Luego, se preprocesó cada artículo. En este preprocesamiento, se tokenizó el texto, eliminando los *tokens* que correspondían a caracteres de puntuación, palabras con menos de 2 caracteres, palabras que incluían un número y las *stop words* del inglés. Cada palabra se pasó a minúscula, y luego se aplicó lematización. Una vez que se aprendió el vocabulario, se aplicó el método tf-idf y se seleccionaron las 6.000 palabras más importantes. El vocabulario final consta por lo tanto, de 6.000 palabras diferentes para los documentos, lo que conforma un *corpus* de 340 mil palabras. En promedio, cada artículo tiene 69 palabras.

Con respecto a los *tags*, basado en [7], no se realizó un mayor preprocesamiento. Lo único que se hizo fue convertir los *tags* a minúscula, y eliminar los *tags* que aparecen menos de dos veces en todos los documentos. El vocabulario de los *tags* cuenta con 6.127 palabras diferentes. En promedio, cada artículo tiene 11 *tags*. En el Cuadro 2 se puede observar un cuadro resumen donde se muestran los *tags* con mayor frecuencia, donde se incluye el número de documentos distintos donde aparece cada uno.

**Table 2: Resumen de los 10 *tags* de mayor frecuencia.**

#	Tag	# de ocurrencias	# de documentos
1	networks	1903	433
2	network	1329	386
3	import	1280	1185
4	bibtex	1151	1069
5	social	1012	229
6	review	958	344
7	information	951	300
8	learning	893	256
9	theory	872	341
10	evolution	808	262

### 5.2 Evaluación

La evaluación se hará en artículos que se sacaron de las librerías de los usuarios (*held out data*). Se recomendarán  $M$  documentos a los usuarios, y se verá si esos documentos estaban o no en su librería personal.

Para poder compararse mejor con [1], se utilizó como métrica de evaluación el *recall*. La razón de esto, es que se conocen los documentos que están en la librería del usuario, y que

<sup>3</sup><http://www.citeulike.org/faq/data.adp>

por lo tanto tienen  $r_{ij} = 1$ , pero eso no quiere decir que los documentos con  $rating\ r_{ij} = 0$  no le hayan gustado a ese usuario. Quizás que el usuario simplemente no los ha leído. Por lo tanto, utilizando el *recall*, se evaluarán los verdaderos positivos. El *recall* se calcula como

$$recall@M = \frac{\text{n de docs que le gustan al usuario en top } M}{\text{total de docs que le gustan al usuario}}$$

Como se dijo anteriormente, se consideraron dos formas de recomendación, *in-matrix prediction* y *out-of-matrix prediction*.

### 5.2.1 In-matrix prediction

*In-matrix prediction* toma el caso en que un usuario tiene un conjunto de artículos que no ha visto, pero que otros usuarios sí han visto.

Esta forma de predicción es muy similar al filtrado colaborativo. Para evaluar este caso, se utilizó una validación cruzada de 5-folds. Se pusieron en el conjunto de pruebas solo los artículos que se encuentran en las librerías de más de 5 usuarios. De este modo es seguro que todos los artículos que están en el conjunto de prueba hayan aparecido en el conjunto de entrenamiento.

Se corrió el modelo utilizando el conjunto de entrenamiento y se hicieron las  $M$  mejores predicciones para el conjunto de prueba.

### 5.2.2 Out-of-matrix prediction

La predicción *out-of-matrix* es el caso en que hay un artículo nuevo que nadie ha visto. En este caso no es tan real el uso de *tags*, ya que si el artículo es nuevo no tendría ninguna etiqueta, pero se estudiará de todas formas.

Esta forma de predicción es puramente basada en contenido. Al igual que antes, se hará validación cruzada de 5-folds. En cada *fold*, se dejan 1/5 de los artículos como conjunto de prueba, y el resto de los artículos queda como el conjunto de prueba. En este caso, no se entrena con ningún artículo que está en el conjunto de pruebas, es decir, al entrenar los usuarios no tienen esos artículos en su librería.

Nuevamente se corrió el modelo utilizando el conjunto de entrenamiento y se hicieron las mejores  $M$  predicciones con el conjunto de prueba.

## 5.3 Configuración de experimentos

La implementación de Gibbs *sampling* se hizo modificando una versión ya existente de este algoritmo para LDA, la cual estaba en Cython<sup>4</sup>. La optimización de  $u_i$  y  $v_j$  se hizo en Python puro. Todos los experimentos se corrieron en un computador con un procesador Intel Xeon 2.6GHz y 16GB de memoria RAM.

Para comparar el modelo propuesto, se utilizará el modelo CTR. Además, para comparar el *performance* de manera más justa, se corrió el modelo CTR con la información

<sup>4</sup><http://cython.org/>

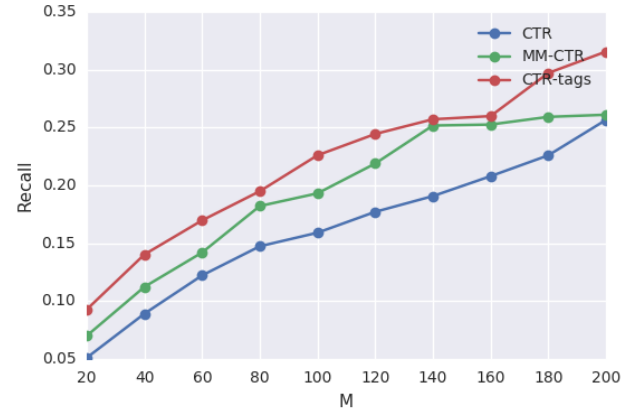


Figure 4: Comparación del *recall* en *in-matrix prediction* de los modelos CTR, MM-CTR y CTR utilizando los tags (CTR-tags) al variar el número de recomendaciones  $M$ , con  $\lambda_v = 100$ .

de las etiquetas como parte de los documentos. Para esto, se definieron dos vocabularios separados, el vocabulario del contenido del documento, y el vocabulario de las etiquetas. Luego, estos dos vocabularios se concatenaron. Se llamará a este modelo CTR-tags.

Todos los experimentos se corrieron con 80 tópicos. Además, se fijó el valor de  $\alpha = 1$ ,  $\eta_w = 0,1$  y  $\eta_t = 0,1$ . Por otro lado,  $\lambda_u = 0,01$ .

## 5.4 Resultados

En la Fig. 4 se puede observar el resultado del modelo MM-CTR, CTR y CTR-tags cuando se varía el número de recomendaciones  $M = 20, 40, \dots, 200$ . Para todos se fijó  $\lambda_v = 100$ . Como se puede observar en la figura, al añadir la información de los *tags* efectivamente se tiene una mejora en la *performance* del modelo propuesto y del modelo CTR. Sin embargo, y contrario al resultado obtenido en [7], se obtuvo mejor *recall* al añadir la información de las etiquetas como palabras nuevas del documento que al añadirlas como nuevas observaciones. Además, al igual que los resultados que se obtuvieron en [1], a medida que aumenta  $M$  se obtiene mejor un mejor *recall*.

En la Fig. 5 se estudia el efecto del número de recomendaciones  $M$  al hacer predicciones *out-of-matrix*. Como se ve en la figura, a medida que aumenta el número de recomendaciones  $M$ , el *recall* aumenta, al igual que en [1]. Además, se observa que en este caso el modelo MM-CTR es superior al modelo CTR utilizando etiquetas. En este caso, el uso de etiquetas también es superior a solo ocupar el contenido de los documentos.

En la Fig. 7 se estudió el efecto de  $\lambda_v$  en las recomendaciones. Como se explicó anteriormente,  $\lambda_v$  es el parámetro que dice cuánto pesa el contenido en comparación a los *ratings*. Cuando  $\lambda_v$  es pequeño, el vector latente  $v_j$  puede diferir bastante de la proporción de tópicos  $\theta_j$ . Por otro lado, cuando  $\lambda_v$  es grande, se tiene el efecto contrario, el vector latente  $v_j$  es más parecido a la proporción de tópicos, y por lo tanto,

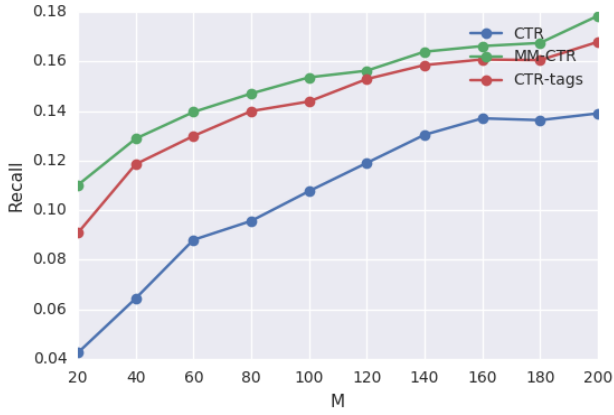


Figure 5: Comparación del *recall* en *out-of-matrix prediction* de los modelos CTR, MM-CTR y CTR utilizando los tags (CTR-tags) al variar el número de recomendaciones  $M$ , con  $\lambda_v = 100$ .

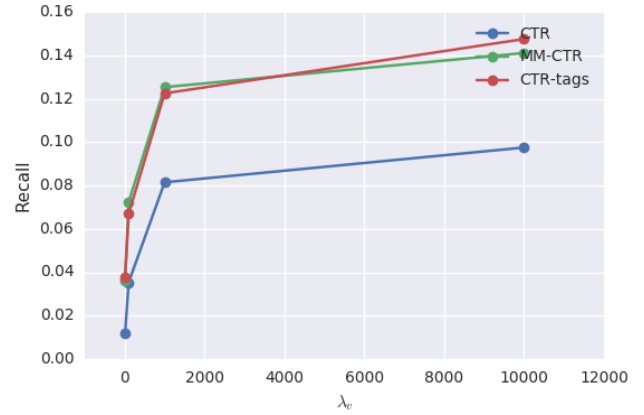


Figure 7: Comparación del *recall* en *out-of-matrix prediction* entre el modelo CTR, MM-CTR y CTR utilizando los tags (CTR-tags) al variar  $\lambda_v$ , con  $M = 50$ .

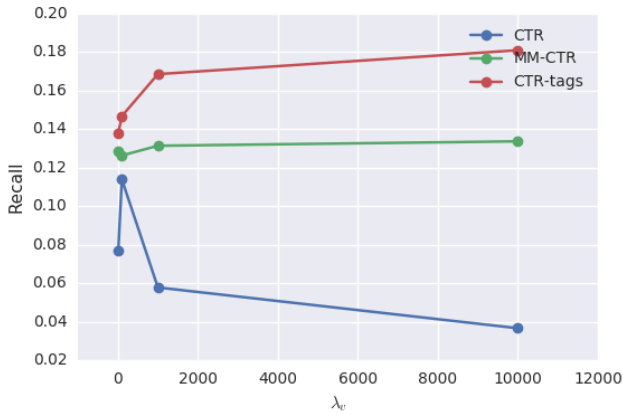


Figure 6: Comparación del *recall* en *in-matrix prediction* entre el modelo CTR, MM-CTR y CTR utilizando los tags (CTR-tags) al variar  $\lambda_v$ , con  $M = 50$ .

el contenido de los documentos tiene más peso.

Se corrieron experimentos con  $\lambda_v = 10, 100, 1000, 10000$ , con  $M = 50$ . Como se observa en la figura, a medida que aumenta  $\lambda_v$ , el modelo CTR tiene menor *performance*, lo que quiere decir que el contenido de los documentos por sí solo no basta para hacer una buena recomendación. Por otro lado, en CTR-tags y MM-CTR, el *recall* aumenta a medida que aumenta  $\lambda_v$ . Esto confirma que las etiquetas que la comunidad agrega a los documentos sí son una fuente de información importante, y pueden suplir la información de los *ratings* de los demás usuarios. Sin embargo, nuevamente añadir las etiquetas como parte del documento tiene mejor *performance* que el modelo propuesto.

Con la Tabla 3 y 4 se puede hacer un análisis más cualitativo de los resultados. En la Tabla 3 se muestran los tópicos más preferidos por el usuario U1, junto con las palabras y los *tags* que mejor describen ese tópico, y en la Tabla 4 se

muestra el resultado de una recomendación de 10 documentos para ese mismo usuario. En la tabla de los tópicos se puede observar que las palabras y los *tags* de ese tópico sí están relacionados. Por ejemplo, el tópico 1 tiene palabras como cerebro, señales, y actividad, y ese mismo tópico tiene etiquetas como neuroimagen (*neuroimaging* en inglés) y descanso. Esta última etiqueta puede tener que ver con la actividad del cerebro en períodos de descanso.

Con respecto a la recomendación que se le hizo al mismo usuario U1, no se tuvieron buenos resultados, ya que se recomendó solo 3 documentos que eran parte de la librería del usuario. Sin embargo, se puede ver que algunos documentos recomendados sí tenían que ver con los tópicos preferidos por el usuario de acuerdo al modelo MM-CTR. Por ejemplo, el documento 1 se refiere al aprendizaje en la sala de clases, y el tópico 2 incluye palabras como "estudiante", y "libro". Además, varios documentos recomendados están relacionados con la biología, y uno de los tópicos que el usuario prefiere tiene que ver con la actividad del cerebro, temas que están relacionados de alguna manera.

## 6. CONCLUSIONES

La primera conclusión del trabajo es que efectivamente las etiquetas que los usuarios agregan a los documentos son una fuente de información adicional. Al usar etiquetas se obtuvo una mejor *performance* en todos los experimentos que se corrieron.

En este trabajo se propuso un modelo que tenía la intención de ser efectivo a la hora de agregar etiquetas al modelo CTR. Sin embargo, como se vio en los resultados el modelo propuesto no fue mejor que considerar las etiquetas como parte de los documentos. En ese sentido, no se confirmaron los resultados de [7]. Es necesario hacer más experimentos y verificar que la implementación de MM-CTR fue correcta, lo que queda como trabajo a futuro. Además, es necesario ver cómo se comportan los modelos al cambiar el número de tópicos o el valor de los *priors*.

**Table 3: Top 3 tópicos del usuario U1**

Top 3 tópicos	
1	<b>palabras:</b> brain, functional, region, signal, task, subject, result, activity, suggest, human <b>tags:</b> fmri, brain, imaging, state, resting, methods, bold, meditation, mvpa, neuroimaging
2	<b>palabras:</b> book, include, topic, author, edition, chapter, new, student, introduction, cover <b>tags:</b> book, books, textbook, first, compression, computer, temporary, econometrics, textbook, skr
3	<b>palabras:</b> large, high, number, level, order, small, low, cost, however, performance <b>tags:</b> fpga, reconfigurable, rc, reconfigurable_computing, trends, reconfigurable_platforms, threads, classic, hierarchical, level

**Table 4: Top 10 documentos recomendados para usuario U1. Los documentos marcados con una Y quiere decir que ese documento sí era parte de la librería del usuario.**

Top 10 documentos	Título documento	
1	Assessment and classroom learning	X
2	Friends and neighbors on the web	X
3	Kernel methods for relation extraction	X
4	Persuasive technology: using computers to change what we think and do	Y
5	GroupLens: An Open Architecture for Collaborative Filtering of Netnews	X
6	Reverse Engineering of Biological Complexity	X
7	Sequencing and comparison of yeast species to identify genes and regulatory elements	Y
8	A database of macromolecular motions	X
9	A phylogenomic approach to bacterial phylogeny: evidence of a core of genes sharing a common history	X
10	The Stanford Microarray Database	Y

Por último, el modelo propuesto produce resultados fácilmente interpretables. Si bien no se obtuvieron los mismos resultados que en [1], al ver los tópicos que los usuarios prefieren y sus recomendaciones se ve que el modelo no funciona tan mal.

## 7. REFERENCES

- [1] Wang, C., & Blei, D. M. (2011, August). Collaborative topic modeling for recommending scientific articles. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 448-456). ACM.
- [2] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30-37.
- [3] Murphy, Kevin P. (2012). *Machine learning : a probabilistic perspective*. Cambridge, MA: MIT Press. pp. 151-152.
- [4] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, pages 263-272, Washington, DC, USA, 2008. IEEE Computer Society.
- [5] D.M. Blei and M.I. Jordan. Modeling annotated data. In SIGIR '03.
- [6] T.L. Griffiths. Finding scientific topics. Proceedings of the National Academy of Sciences, 101:5228-5235, '04
- [7] Ramage, D., Heymann, P., Manning, C. D., & Garcia-Molina, H. (2009, February). Clustering the tagged web. In Proceedings of the Second ACM International Conference on Web Search and Data Mining (pp. 54-63). ACM.
- [8] Lacic, E., Kowald, D., Seitlinger, P., Trattner, C., & Parra, D. (2014). Recommending items in social tagging systems using tag and time information. arXiv preprint arXiv:1406.7727.
- [9] Shepitsen, A., Gemmell, J., Mobasher, B., & Burke, R. (2008, October). Personalized recommendation in social tagging systems using hierarchical clustering. In Proceedings of the 2008 ACM conference on Recommender systems (pp. 259-266). ACM.