Exploración de recomendadores híbridos para música mediante el uso de letras

Roshad Alipanah, Sebastián Carreño, Lucas Suárez
Departamento de Ciencias de la Computación
Pontificia Universidad Católica de Chile, Santiago, Chile

Abstract

En el presente trabajo se presentan los resultados de estudiar el efecto que tienen las letras de las canciones en la tarea de recomendación de música a usuarios. En nuestros experimentos se compararon modelos híbridos de recomendación con modelos basados en filtrado colaborativo. Los resultados obtenidos mostraron que nuestro enfoque permitió mejorar en un 3% según MAP@20 las recomendaciones.

Keywords: music recommendation, spotify, lyrics, hybrid recommendation.

1 Introducción

Durante los últimos años, los servicios de streaming han adquirido una enorme popularidad. Dentro de estos, se encuentran plataformas musicales, como Spotify o Apple Music, cuya cantidad de usuarios, hasta el día de hoy, está en un constante aumento. Sin embargo, a medida que estas plataformas van creciendo, es necesario para ellas ir creando y adquiriendo nuevas tecnologías que les permitan aumentar la calidad en cuanto a la experiencia de uso que ofrecen, ya sea para atraer nuevos usuario, o bien para mantener satisfechos a los que ya utilizan sus servicios.

Para alcanzar estos objetivos, es vital que se haga un buen uso de sistemas recomendadores, ya que además de poder acceder al audio de las canciones, el usuario requiere que el sistema le realice recomendaciones para descubrir y escuchar música que pueda resultar interesante para él. Si una recomendación resulta ser inadecuada para un usuario, este se verá disconforme, y su nivel de satisfacción al utilizar la plataforma disminuirá.

De muchas formas se ha intentado aumentar la calidad de estas recomendaciones, obteniendo interesantes resultados año tras año. Incluso existen desafíos, como el Spotify Challenge, que se realizan con el fin principal de mejorar los algoritmos recomendadores que Spotify utiliza en sus servicios. Específicamente, en el RecSys Challenge 2018, la tarea propuesta en la competencia fue la de sugerir canciones para continuación automática de *playlists* [**1**].

En el desafío mencionado anteriormente, una de las ideas para investigación futura fue la estudiar el impacto de las letras de las canciones en las recomendaciones. Este aspecto de una canción, a pesar de no reflejar su ritmo, melodía o estilo, puede representar estados de ánimo, o situaciones con las que un usuario pueda sentirse identificado, por lo que es natural pensar en utilizar esta fuente de información para realizar recomendaciones.

Con respecto a la recomendación de música que utiliza el contenido de esta, existen distintas fuentes que ya han sido utilizadas anteriormente, por ejemplo, género, nombres de playlists e incluso el propio sonido de las canciones [**2**, **3**]. Sin embargo, en el caso de la letra, no existen estudios que permitan afirmar que se pueda sacar provecho de esa fuente de información para mejorar una recomendación inicial basada en filtrado colaborativo.

Por estos motivos se llevó a cabo un estudio que intentó responder la pregunta de si es posible utilizar las letras de las canciones para mejorar las recomendaciones a usuarios. El enfoque adoptado para cumplir este objetivo fue el de comparar modelos basados exclusivamente en filtrado colaborativo y modelos híbridos que combinan los anteriores modelos con el contenido de las letras.

2 Dataset

Los datos utilizados en este trabajo fueron escogidos a partir de una colección de aproximadamente 12 millones de *tweets* de usuarios que compartieron las canciones que escuchaban en Spotify[1]. Debido a esto, sólo se tiene registró de si un usuario escuchó o no una canción y no de ratings propiamente tal, por lo que este estudio está enfocado en implicit feedback.

De esa colección de canciones se intentó conseguir la mayor cantidad de letras utilizando la API de Genius, proceso que se llevó a cabo mediante un *script* que hace requests a la API sobre la información de cada canción. En caso de que no se encontrara la letra simplemente se continuaba con el siguiente *tweet*, además, cabe destacar que entre cada request debe esperarse algunos segundos por política de la API, por lo que el proceso de recolección es lento. Finalmente, se logró obtener las letras de 10000 canciones escuchadas por 14606 usuarios, con lo que al filtrar el dataset inicial alcanzan un total de 917901 interacciones, que es el dataset con el cual se llevó a cabo este estudio.

Para la separación de test/training set se consideró un split 80/20. Además, el set de validación para obtener los hiperparámetros óptimos de cada modelo se escogió tomando el 10% del set de entrenamiento. Cada uno de estos subconjuntos de datos contienen interacciones para cada usuario y para cada canción del conjunto antes del split.

3 Metodología

Para estudiar si la utilización de letras es o no capaz de mejorar el rendimiento de los algoritmos se propuso la tarea de generar algoritmos híbridos que combinen algún método similar a los utilizados en el Spotify Challenge 2018 con el contenido de las letras. Dentro de las técnicas de filtrado colaborativo que se utilizaron en ese concurso, este estudio se enfocó en Factorización Matricial (ALS) [**4**] y su versión generalizada, el perceptrón multicapa (MLP) cuya formulación es la adoptada en el paper Neural Collaborative Filtering (NCF) [**5**].

En el caso de Factorización Matricial, dado un usuario u y un item i el score s_{ui} del modelo viene dado por el producto interno entre la representación de u con la representación de i en el espacio de factores latentes, que corresponden a filas y columnas de las matrices de usuarios e items resultantes de la factorización matricial.

La red neuronal utilizada en la implementación de NCF recibe como input un vector (x_u, y_i), donde x_u representa al usuario u como un vector de la forma $(0, 0, \ldots 1 \ldots 0)$, con el valor 1 en la posición del id de u, y de manera análoga para y_i como representación de la canción i.

La primera capa de la red consta de dos embeddings separados para items y usuarios que consta de tantas neuronas como usuarios e items respectivamente y cuyo output tiene 8 dimensiones. Lo que en total da origen a 114000 y 80000 parámetros aproximadamente para estos embeddings que permiten capturar la representación de cada usuario.

Las dos capas siguientes son capas densas con función de activación ReLU, donde la primera recibe un input de 16 dimensiones (el resultado de la capa de embeddings) y entrega una salida de 64, para posteriormente entrar a la segunda capa ReLU que entrega un vector de 32 dimensiones. Finalmente, a este vector se le agrega un *bias* y el resultado se obtiene al procesar esto con una neurona con función de activación sigmoidal. El output es la probabilidad de que el usuario u haya escuchado la canción i, cantidad que fue considerada como el *score* s_{ui} asociado a este modelo.

El entrenamiento de esta red es fundamentalmente bayesiano: se busca maximizar la probabilidad de predecir correctamente dado que se tiene un conjunto de datos empíricamente observado de canciones escuchadas. En este punto, también es necesario incluir items negativos, es decir, items que los usuarios no hayan visto. Como nuestro dataset sólo contiene items positivos (que los usuarios efectivamente vieron) en nuestros experimentos se escogieron aleatoriamente 4 items negativos para la fase de entrenamiento.

Para vectorizar las letras de las canciones se utilizó BERT large multilenguaje[2] preentrenado que generó embeddings de 768 dimensiones. El modelo basado en contenido BERT opera de la siguiente manera: dado un usuario u y un item i se calcula el score de s_{ui} como la máxima similaridad [**6**] de coseno (entre las features de contenido o embeddings) obtenida con las canciones o items que el usuario ya escuchó. Se intentaron otras maneras de calcular el *score* usando similaridad de coseno, por ejemplo, tomando el promedio de las similaridades con los items que el usuario ya consumió, sin embargo, eso resultó ser demasiado costoso computacionalmente, por lo que fue descartado.

Además, se utilizó Most Popular para corroborar si efectivamente se obtienen buenos resultados, pues, en caso de que los modelos híbridos efectivamente obten-

[1] Disponible en `https://www.kaggle.com/ashwinik/spotify-playlist`

[2] Disponible en `https://github.com/google-research/bert`

gan mejores resultados que los que utilizan exclusivamente filtrado colaborativo, es importante que también sean mejores que uno de los algoritmos más simples, y así se pueda establecer o no la utilidad de las letras bajo el enfoque adoptado.

La manera en que se combinaron los modelos es mediante *weighted scoring* [7], esto es, incorporando un peso que representa que tanta importancia tiene un modelo en la hibridación. Técnicamente, dado un item i, un usuario u y dos modelos M_1 y M_2 con scores s^1_{ui} y s^2_{ui} respectivamente para ese item, el score s^H_{ui} del modelo híbrido M_H se calcula según la expresión

$$s^H_{ui} = w_1 \cdot s^1_{ui} + w_2 \cdot s^2_{ui} \quad (1)$$

Sin pérdida de generalidad, y como la única tarea es ordenar los scores para recomendar, por lo que podemos escalarlos sin alterar dicho orden, consideramos $w_1 = 1$ y sólo nos concentramos en la elección óptima de w_2 para cada modelo híbrido, donde M_2 fue el modelo basado en contenido con BERT embeddings descrito anteriormente.

4 Análisis de parámetros

Para cada modelo se encontró el mejor conjunto de hiperparámetros midiendo el rendimiento en MAP@20 que obtuvo en el set de validación. En el caso de los modelos basados en filtrado colaborativo, para ALS se obtuvo los valores de factores, regularización e iteraciones, mientras que para NCF se consideró la cantidad de épocas de entrenamiento.

Por otro lado, para el modelo basado en contenido que utiliza BERT embeddings, se obtuvo el valor óptimo de reducción de dimensionalidad con *Principal Component Analysis* de la librería Scikit-learn .

Los resultados para cada hiperparámetro se encuentran detallados a continuación

- Factorización Matricial (ALS)
  - Factores: 256
  - Regularización: 0.025
  - Iteraciones: 20
- Factorización Matricial (ALS) + BERT
  - Factores: 128
  - Regularización: 0.025
  - Iteraciones: 20
  - PCA: 20
  - Weight: 0.01
- Neural Collaborative Filtering
  - Épocas: 25
- Neural Collaborative Filtering + BERT
  - Épocas: 25
  - PCA: 30
  - Weight: 0.01
- BERT
  - PCA: 100

Con respecto a NCF se encontró que entre más grande la cantidad de épocas mejor era el rendimiento en MAP@20, sin embargo, el tiempo de entrenamiento se incrementaba sustancialmente, motivo por el que se acotó en 25 la cantidad de épocas para mantenerlo manejable dentro de los tiempos que permiten las sesiones de Google Colab. Algo similar ocurrió con el modelo basado en contenido BERT con la reducción de dimensionalidad. Incluso teniendo estas consideraciones fue necesario adquirir una cuenta Pro en Google Colab.

En la siguiente sección se reportan los resultados obtenidos sobre el set de test.

5 Resultados

Una vez que se encontraron los valores óptimos de cada hiperparámetro se realizaron las evaluaciones de los modelos optimizados en las métricas MAP@n y nDCG@n para $n = 20$ y $n = 50$.

	MAP@20	NDCG@20	MAP@50	NDCG@50
MostPopular	0.1444	0.3015	0.2039	0.4718
BERT	0.0133	0.0635	0.0322	0.1396
ALS	0.3053	**0.7198**	0.3943	0.7997
ALS + BERT	**0.3102**	0.7142	**0.4080**	**0.8000**
NCF	0.1386	0.3709	0.2281	0.5391
NCF + BERT	0.1451	0.3679	0.2277	0.5388

Table 1: Resultados en set de *test*

Se observó que ambos modelos basados en filtrado colaborativo ven mejorado su rendimiento en MAP@20 al ser combinados con BERT mediante el enfoque propuesto. La mejora de rendimiento reportada en esta métrica es aproximadamente de un 3%. En el caso de MAP@50 para NCF no se encontró la misma tendencia, lo que posiblemente se debió a que el modelo no fue optimizado con esta función objetivo si no que con MAP@20.

Con respecto a la métrica nDCG@n la evaluación mostró que la variación es menos significativa que con MAP@n. Aún más, solamente en el caso de para ALS y ALS + BERT hubo una mejoría de nDCG@20 con la

hibridación, sin embargo, tal mejora en alcanzó apenas un 0.04%.

Pese a que NCF mejoró sus resultados al combinarse con BERT, el híbrido no es capaz de establecer una superioridad significativa frente a Most Popular en la métrica en la que fue optimizado. Este hecho, fuera del enfoque de este trabajo, vuelve a generar evidencia acerca de que modelos complejos no necesariamente permiten obtener mejores resultados que modelos más simples [**8**].

En el caso del modelo basado en contenido BERT se observó un rendimiento muy por debajo de los demás algoritmos, sin embargo, pudo ser utilizado para mejorar el rendimiento de los otros modelos. Este curioso hecho puede hallar su explicación en que lo que se ponderó con pesos no son los rendimientos, si que no los *scores*, y fundamentalmente lo que cambia son las listas de recomendación al ordenar dichos *scores*. Tras este reordenamiento puede ocurrir que items que en el modelo BERT tengan una similaridad muy alta en contenido queden en posiciones más bajas al agregar el *score* del modelo basado en filtrado colaborativo, y de esta manera el modelo híbrido podría hallar correcciones cuando uno de sus componentes se equivoca. Cabe destacar que los pesos óptimos encontrados para el modelo basado en contenido son pequeños ($w = 0.01$ en ambos casos), lo que indicaría que la contribución de este modelo es menor en el híbrido y que el aporte o corrección que realiza es pequeño.

6 Conclusiones

En el Spotify Challenge de 2018 se propuso la tarea de continuación de playlists. El track principal de esta competencia contemplaba un dataset de estilo implicit feedback, donde los grupos de trabajo que participaron utilizaron principalmente métodos basados en filtrado colaborativo. Uno de los lineamientos que se proponen para continuar con esa tarea es la posibilidad de utilizar otras fuentes de contenido (además del nombre de la playlist) para mejorar las recomendaciones.

En el contexto de recomendación de música basada en contenido existen diversas fuentes que pueden ser utilizadas, por ejemplo: autor, álbum, género, letra e incluso el propio sonido de la canción [**2**, **3**]. Este estudio se concentró en el uso de la letra de las canciones para generar recomendaciones a usuarios.

Para estudiar si realmente se podía sacar provecho de la letra propusimos utilizar modelos híbridos que combinan modelos basados en contenido con modelos basados en filtrado colaborativo a través de pesos en los *scores*.

Los resultados obtenidos en los experimentos muestran que la letra de las canciones efectivamente puede ser incluida a través del enfoque propuesto para mejorar las recomendaciones de un modelo basado en filtrado colaborativo por sí solo. La mejora alcanzada tras la hibridación alcanza aproximadamente un 3% en MAP@20, cantidad que puede ser significativa a la hora de participar en competencias masivas donde los equipos compiten por cada punto de rendimiento. Los mejores resultados fueron alcanzados al combinar el modelo de contenido con el modelo de filtrado colaborativo basado en factorización matricial ALS.

Un hallazgo interesante fue que el enfoque neuronal Neural Collaborative Filtering, no logró resultados competitivos y prácticamente no superó a Most Popular incluso tras incluir la hibridación. Esto concluye lo mismo que el análisis del paper *Are we really making much progress? A worrying analysis of recent neural recommendation approaches* del 2019 [**8**]. Debido a esto, una futura línea de investigación posible sería replicar este mismo estudio pero utilizando otro tipo de modelo moderno, como por ejemplo, Mult - VAE que ha mostrado ser capaz de obtener resultados competitivos como muestra el paper mencionado anteriormente.

Agradecimiento

Agradecemos al equipo docente de Sistemas Recomendadores por los conocimientos entregados. Este trabajo no sólo sirvió para aprender acerca de sistemas recomendadores si no que también para aproximarnos a lo que significa hacer un estudio académico riguroso.

7 Referencias

[1] Zamani, H., Schedl, M., Lamere, P., & Chen, C. W. (2019). An analysis of approaches taken in the ACM recsys challenge 2018 for automatic music playlist continuation. ACM Transactions on Intelligent Systems and Technology (TIST), 10(5), 1-21.

[2] Pichl, M., Zangerle, E., & Specht, G. (2015, November). Towards a context-aware music recommendation approach: What is hidden in the playlist name?. In 2015 IEEE International Conference on Data Mining Workshop (ICDMW) (pp. 1360-1365). IEEE.

[3] Van den Oord, A., Dieleman, S., & Schrauwen, B. (2013). Deep content-based music recommendation. Advances in neural information processing systems, 26, 2643-2651.

[4] Hu, Y., Koren, Y., & Volinsky, C. (2008, December). Collaborative filtering for implicit feedback datasets. In 2008 Eighth IEEE International Confer-

ence on Data Mining (pp. 263-272). Ieee.

[5] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017, April). Neural collaborative filtering. In Proceedings of the 26th international conference on world wide web (pp. 173-182).

[6] Messina, P., Dominguez, V., Parra, D., Trattner, C., Soto, A. (2019). Content-based artwork recommendation: integrating painting metadata with neural and manually-engineered visual features. User Modeling and User-Adapted Interaction, 29(2), 251-290.

[7] Burke, R. (2007). Hybrid web recommender systems. In The adaptive web (pp. 377-408). Springer, Berlin, Heidelberg.

[8] Dacrema, M. F., Cremonesi, P., & Jannach, D. (2019, September). Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In Proceedings of the 13th ACM Conference on Recommender Systems (pp. 101-109).