Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC3633 - Sistemas Recomendadores

Proyect :
Data augmentation for recommender systems using LLMs

Rodrigo Pozo - Oscar Loch - Luis Miranda - Cristóbal Vásquez
Grupo 11

Índice

1. Introducción

El *cold start problem*, un desafío persistente en los sistemas de recomendación, continúa obstaculizando la entrega de recomendaciones personalizadas para elementos nuevos o poco interactuados. En la literatura se ha investigado exhaustivamente este problema. Sin embargo, sus complejidades persisten, exigiendo soluciones innovadoras que aprovechen las capacidades de las tecnologías emergentes. Al mismo tiempo, los LLMs, caracterizados por su destreza en la comprensión y generación del lenguaje natural, han demostrado una eficacia notable en diversos dominios de tareas. Específicamente, estudios recientes han explorado la utilización de LLMs como sistemas de recomendación o herramientas interactivas para elucidar recomendaciones (Gao et al. (2023), Wang and Lim (2023). Sin embargo, una frontera aún por explorar reside en la capacidad de aprovechar todo el potencial de los Modelos de Lenguaje de Gran Tamaño (LLMs) para expandir datos en el contexto

A pesar de los avances en el aprovechamiento de los modelos de lenguaje de gran escala (LLMs) para sistemas de recomendación, ningún estudio previo los ha empleado como una herramienta dedicada para la ampliación de datos. Este proyecto aborda esa notable brecha en la literatura, proponiendo un enfoque novedoso que capitaliza las capacidades únicas de los LLMs para mejorar el rendimiento de los sistemas de recomendación. Al utilizar los LLMs como una herramienta de ampliación de datos, buscamos mitigar el cold start problem, enriqueciendo la representación de elementos y usuarios dentro del sistema. A través de una exploración exhaustiva de este territorio inexplorado, nuestro objetivo es arrojar luz sobre el potencial de los LLMs para hacer frente a los desafíos planteados por el problema de arranque en frío, contribuyendo así al avance de la investigación en sistemas de recomendación y al campo más amplio de la inteligencia artificial.

2. Análisis exploratorio

Dado que en este proyecto se enfoca en realizar data augmentation para solucionar el *cold-start problem*, es que es esencial identificar los distintos tipos de usuarios que existen de acuerdo a su nivel de actividad en el dataset. Es por ello que se dividieron los usuarios en *Top*, *Middle* y *Bottom users*, como se muestra en la figura 1. Los usuarios pertenecientes a *Top* corresponden a los usuarios que poseen una cantidad de interacciones menor o igual a 200 y mayor a 100. Los usuarios de *Middle* poseen una cantidad de interacciones menor o igual a 100 y mayor a 20. Finalmente, los usuarios de *Bottom* corresponden a los usuarios con una cantidad menor o igual 20 interacciones. En la figura 1 se muestra además la línea naranja que representa como es aumentado en 20 número de interacciones para los usuarios pertenecientes a Top, Mid y Bot. Por otro lado, los usuarios con un número de interacciones superior a 200 no se les realizó data augmentation debido a que este proyecto se busca en mejorar las recomendaciones de los usuarios con menos interacciones. Y además para abaratar costes de data augmentation al utilizar la API de GPT, que será mencionada en las siguientes secciones de este informe.

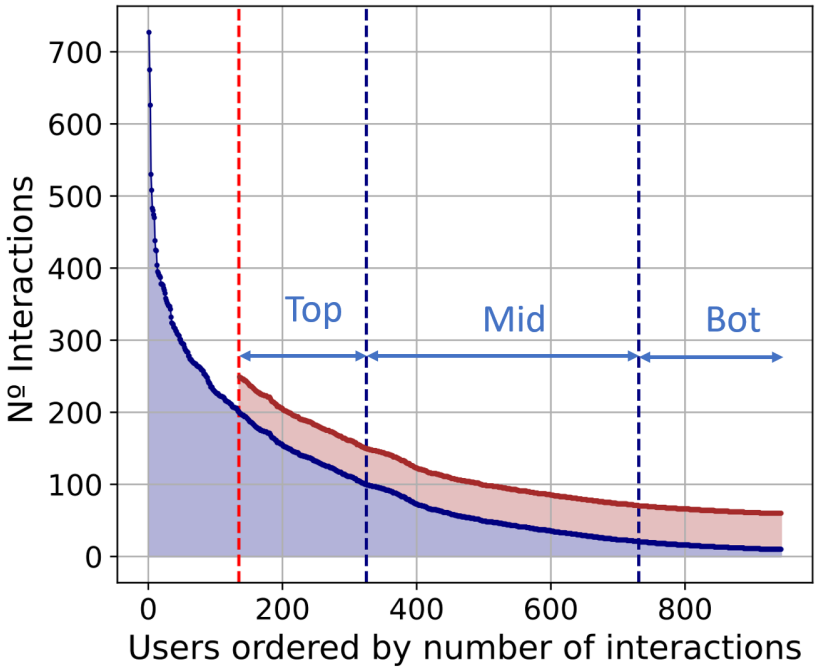

Figura 1: División de usuarios de acuerdo a su nivel de actividad (número de interacciones).

3. Metodología

3.1. Data augmentation using GPT

Para el proceso de *data augmentation* primero se llevó a cabo una fase exploratoria de LLMs con el objetivo encontrar el modelo de lenguaje que mejor se adapte a la tarea. En particular, se comparó el desempeño de `Llama 2-13b` y `GPT3.5-turbo`. Este primero se ejecutó utilizando un entorno virtual gratuito de Google Colab, mientras que para la implementación de GPT se optó por consumir el modelo mediante la API de OpenAI. Luego, tras realizar un *stress test* en ambos LLMs, se obtuvieron resultados considerablemente superiores utilizando GPT, ya que las limitaciones de *hardware* de Google Colab impedían que el modelo de lenguaje almacenara un *context* lo suficientemente extenso como para manejar todas las interacciones de usuario, las cuales para los usuarios más activos llegaban a 250.

Una vez determinado el modelo a utilizar, se utilizó la estrategia de *data augmentation* para enriquecer el conjunto de datos existente. La estrategia de data augmentation se implementó definiendo los parámetros N y K. El parámetro N corresponde a la cantidad máxima de interacciones que un usuario debe tener para que sea considerado para el *data augmentation*. Establecer un valor máximo para N ayuda a abordar el *cold start problem* al aumentar exlusivamente la cantidad de interacciones generadas para usuarios nuevos o con historiales de interacción limitados para evitar problemas de subrepresentación. Por otro lado, el parámetro K representa la cantidad máxima de interacciones que se agregarán a cada usuario que es considerado para el *data augmentation*. Esto asegura que no se genere una cantidad excesiva de datos artificiales.

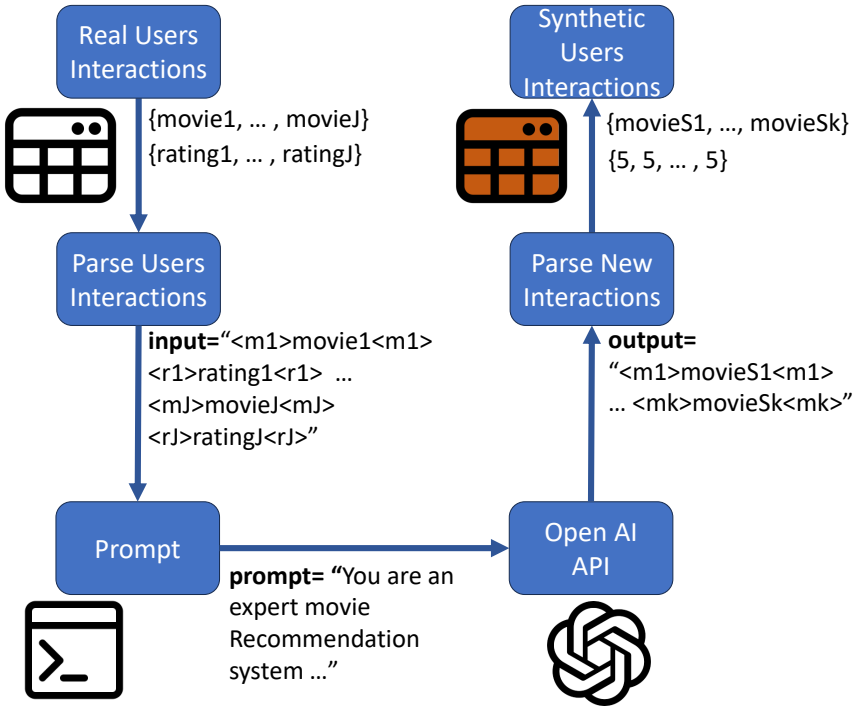

Figura 2: Estrategia de *data augmentation* implementada.

Luego, para cada usuario del *dataset* de *train* se lleva a cabo la estategia de *data augmentation* explicada en la figura 2. En esta metodología, se extraen todas las interacciones (ratings y títulos de películas) que posee dicho usuario en *train*. Luego, en el bloque *Parse User Interactions* se concatena toda esta información en un mismo `string`, utilizando un formato determinado. Luego, el siguiente bloque es alimentado con el *output* del bloque anterior, retornando el *prompt* que luego es enviado a la API de OpenAI. Finalmente, la respuesta de ChatGPT es introducida a un bloque de *parsing* para obtener una lista de películas recomendadas para el usuario en cuestión, la cual luego se filtra para que solo contenga películas con las que el usuario no ha interactuado en el set de *train* y que existen en el catálogo de *MovieLens*.

Una datalle importante es que a pesar de que MovieLens contiene ratings entre 1 y 5, pero a GPT se le entregaron solo los items relevantes para el usuario, siendo items relevantes los items a los que el usuario entregó un rating mayor o igual a 3. Esto se hizo ya que el response de GPT es una lista de títulos de peliculas relevantes para el usuario, por lo que se pueden integrar directamente esta response de GPT al resto del dataset para hacer recomendaciones usando feedback implícito.

A continuación se muestra el *prompt template* utilizado para `GPT`:

```
SYSTEM: Take a deep breath and work on this problem step by step.
You are an expert movie recommendation system.
You have programmed the following command that you will receive from USER:
```

```
<m1>MOVIE_TITLE1 (YEAR)<m1> <r1>RATING1<r1>,
<m2>MOVIE_TITLE2 (YEAR)<m2> <r2>RATING2<r2>, ...
(input can have any number of movies)

You should return {k} possible movies that the user might like based on the
movies that he/she has rated.
The format should be:

<m1>MOVIE_TITLE1 (YEAR)<m1>, <m2>MOVIE_TITLE2 (YEAR)<m2>, ...,
<m{k}>MOVIE_TITLE{k} (YEAR)<m{k}>

and nothing more than that. You should not explain the reasons for
your recommendation.
CONSIDERATION1: The movies that you return should be in MovieLens100K dataset.
CONSIDERATION2: The movies that you return should not be the movies
that the user has rated.
```

3.2. Training

Para el entrenamiento, se utilizaron los modelos de factorización de matrices ALS y BPR, por lo que se considero relevancia o no relevancia para cada item. Considerando items con un rating mayor o igual a 3 como relevantes. En primer lugar, se procedió a entrenar los modelos con el set de datos original, y se guardó la métrica MAP@10. Posteriormente, agregaron k interaccones sintéticas por usuario. Luego, los modelos fueron reentrenados con el conjunto de datos sintéticos aumentado, guardando la respectiva métrica para cada iteración. Este proceso se repitió utilizando desde k = 1 hasta k = 20.

A continuación se presentan un esquema de la metodología propuesta:

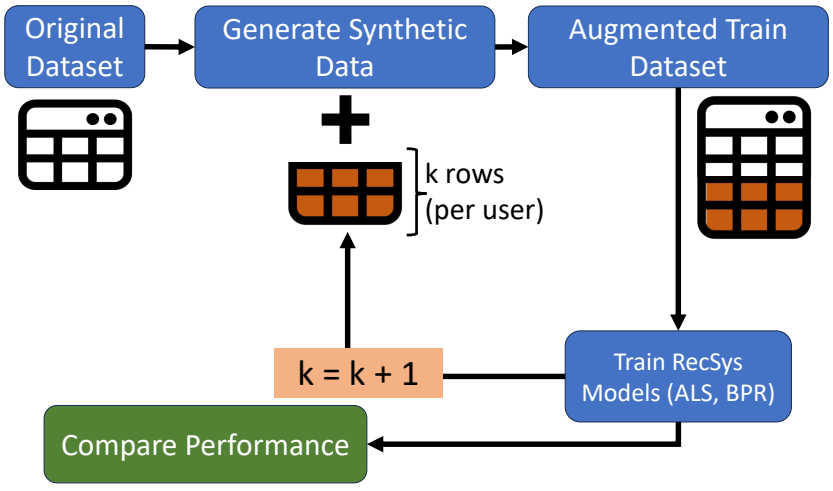

Figura 3: Enter Caption

4. Resultados

En las figuras 4 y 5 se ilustra el rendimiento para MAP@10 Y NDCG@10 respectivamente para los ALS y BPR. En estas figuras se coompara el rendimiento usando distintas random y most-popular data augmentation como baseline, es decir que se agregan de 1 a 20 interacciones sinteticas de manera aleatoria o los items más populares con los que no ha interactuado antes un usuario.

El rendimiento que la implementación de data augmentation con la metodología propuesta en este proyecto generalmente se mantiene constante, superando a consistemente a random en 3 de las 4 ploteos mostrados en las figuras y desempeñándose de manera similar al método de data augmentation most popular. El impacto es particularmente notable para los usuarios con cold start problem (bot), donde la metodología propuesta supera a random e iguala a la most popular, demostrando estabilidad. Para mid users, hay una ligera mejoría en el rendimiento, mientras que para los top users no se lográ observar un gran impacto del data augmentation, siendo esto un resultado esperado ya que la proporción de datos sintéticos agregados es considerablemente menor para los usuarios de top en comparación a los de bot.

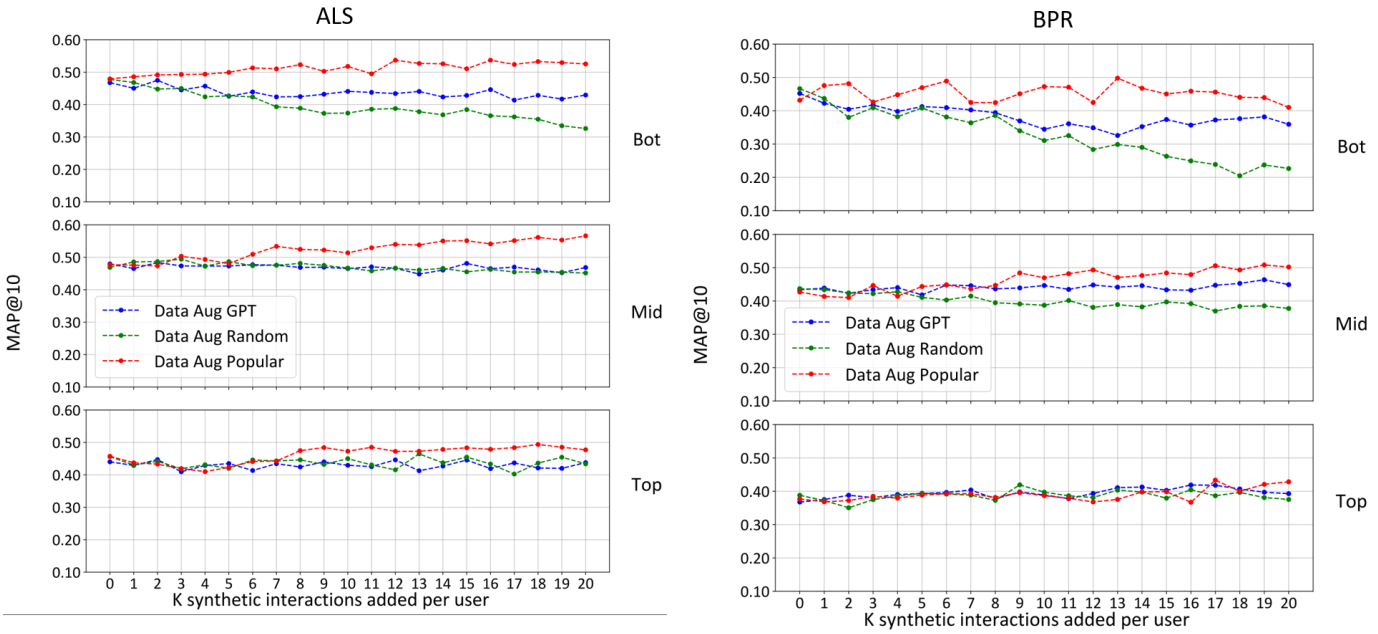

Figura 4: Resultados para MAP@10 usando ALS y BPR

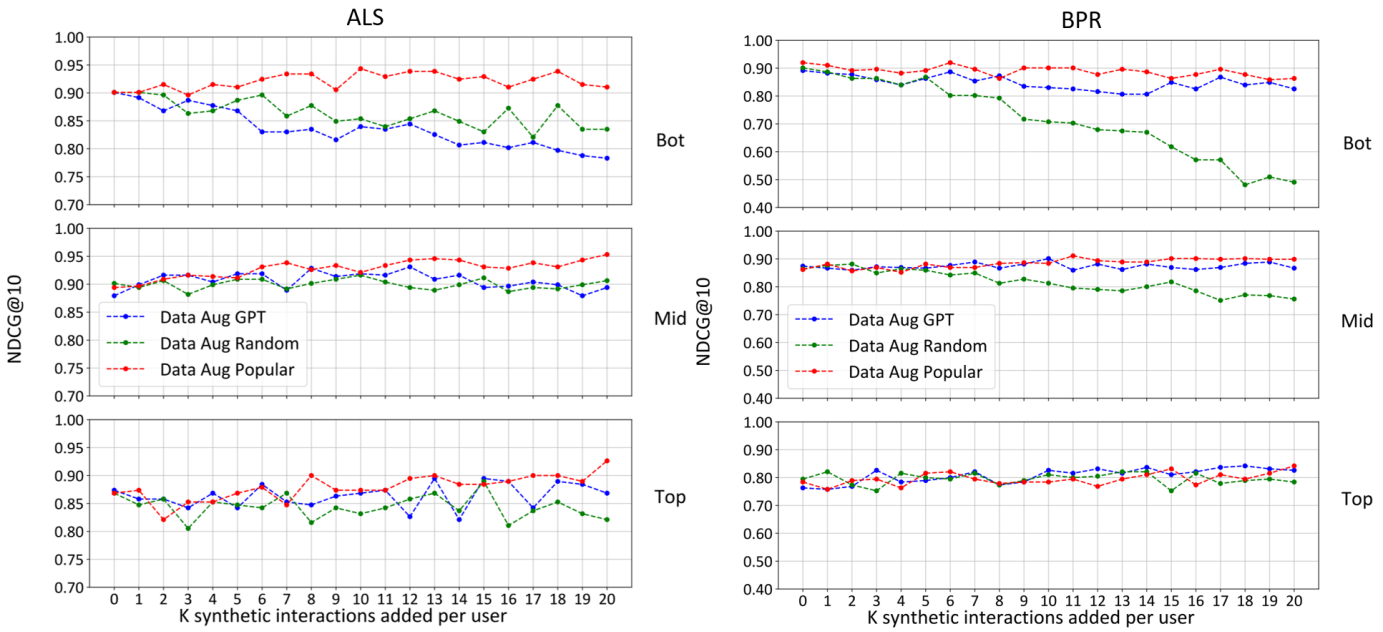

Figura 5: Resultados para NDCG@10 usando ALS y BPR

Se puede observar un comportamiento más errático para NDCG usando ALS, no se logra comprender por completo cual es la causa de este comportamiento para esta métrica y modelo recomendador.

5. Conclusiones y Trabajo futuro

Se logró la implementación de GPT como generador de datos sintéticos. Se observó que GPT supera a random, manteniendo un comportamiento estable al aumentar el número de interacciones sintéticas. Además se logra un rendimiento similar pero por debajo de most popular. Como trabajo futuro, nos proponemos ajustar el diseño del prompt para reducir el espacio de opciones de generación. Además, es posible incrementar el número de interacciones sintéticas y estudiar el comportamiento resultante. Por otro lado, aunque en la mayoría de los resultados se observa que el aumento de datos mediante GPT no supera a most popular, se plantea como hipótesis para un estudio futuro que, si bien las recomendaciones no sobrepasan en rendimiento al most popular, probablemente lo hagan en diversidad y novedad sin comprometer demasiado el rendimiento. El aumento de datos mediante GPT se convierte así en un punto intermedio entre la diversidad y el rendimiento de most popular (alto rendimiento y poca diversidad) y el random (bajo rendimiento, alta diversidad). Se plantea la integración de métricas adicionales para evaluar exhaustivamente el desempeño del sistema de recomendación. De esta forma, se proponen métricas como la Cobertura, que abarca tanto el alcance del catálogo como la distribución equitativa de elementos en las recomendaciones; Diversidad, que mide la variedad presente en las listas de recomendaciones mediante una similitud intra-lista; Novedad, que se centra en la inversa de la popularidad de los elementos, aportando una perspectiva sobre la novedad de las recomendaciones; y Serendipia, que permitirá medir que tan inusuales son las recomendaciones incorporando la distancia entre elementos recomendados y sus contenidos esperados. Estas métricas ofrecerán una evaluación más completa de la calidad de las recomendaciones generadas, ampliando la comprensión del problema. Asimismo, se propone explorar la posibilidad de utilizar modelos más avanzados, como GPT-4, para potenciar la capacidad de generación de recomendaciones y abordar de manera más eficaz este desafío.

Referencias

Gao, Y., Sheng, T., Xiang, Y., Xiong, Y., Wang, H., and Zhang, J. (2023). Chat-rec: Towards interactive and explainable llms-augmented recommender system.

Wang, L. and Lim, E.-P. (2023). Zero-shot next-item recommendation using large pretrained language models.