Text-like Encoding of Collaborative Information in Large Language Models for Recommendation

Diego Alejandro Quezada Campos
Diego Ignacio Campusano Schialer
Nicolás Javier Espinoza Silva

Contexto

- Creciente interés en adaptar modelos de lenguaje grandes (LLMs) para sistemas de recomendación (LLMRec)
- La información colaborativa es crucial para modelar los intereses de los usuarios, pero difiere en modalidad de los datos textuales

Marco teórico

- LLM: Modelos de Deep Learning basados en la arquitectura *transformer* (Vaswani et al., 2017) la cual les permite extraer información semántica de secuencias de palabras
- RecSys: Algoritmos que dado un usuario y una lista de ítems recomiendan uno o varios ítems que idealmente sean de preferencia para el usuario
- LLMRec: Área que estudia la aplicación y adaptación de LLMs para que sean utilizados por sistemas recomendadores

- Tokens: Unidades en las que LLMs descomponen cadenas de texto, pueden ser palabras, caracteres, signos de puntuación o combinaciones de estas
- Embeddings: Vectores de números reales que representan palabras, imágenes u otras formas de información de manera que puedan ser procesadas por modelos de Machine Learning de manera eficaz

Problema

Cómo codificar la información colaborativa en LLMs para recomendaciones

Estado del arte

Estrategia 1

- Incorporar tokens adicionales en LLMs para representar usuarios y productos, ajustando los datos de interacción para capturar información colaborativa en los embeddings (Zheng et al., 2024; Hua et al., 2023)
- Altera el espacio generativo del LLM, potencialmente comprometiendo las funcionalidades originales
- Baja eficacia debido a la naturaleza de bajo rango de la información, lo que genera redundancia en la tokenización

Estrategia 2

- Usar un modelo externo de factores latentes para capturar la información colaborativa y luego mapearla al espacio de embeddings de los tokens, evitando la necesidad de aprenderlos desde cero (Zhang et al., 2024; Li et al., 2023; Liao et al., 2024)
- Es efectivo, pero requiere entrenar un modelo adicional

Contribución

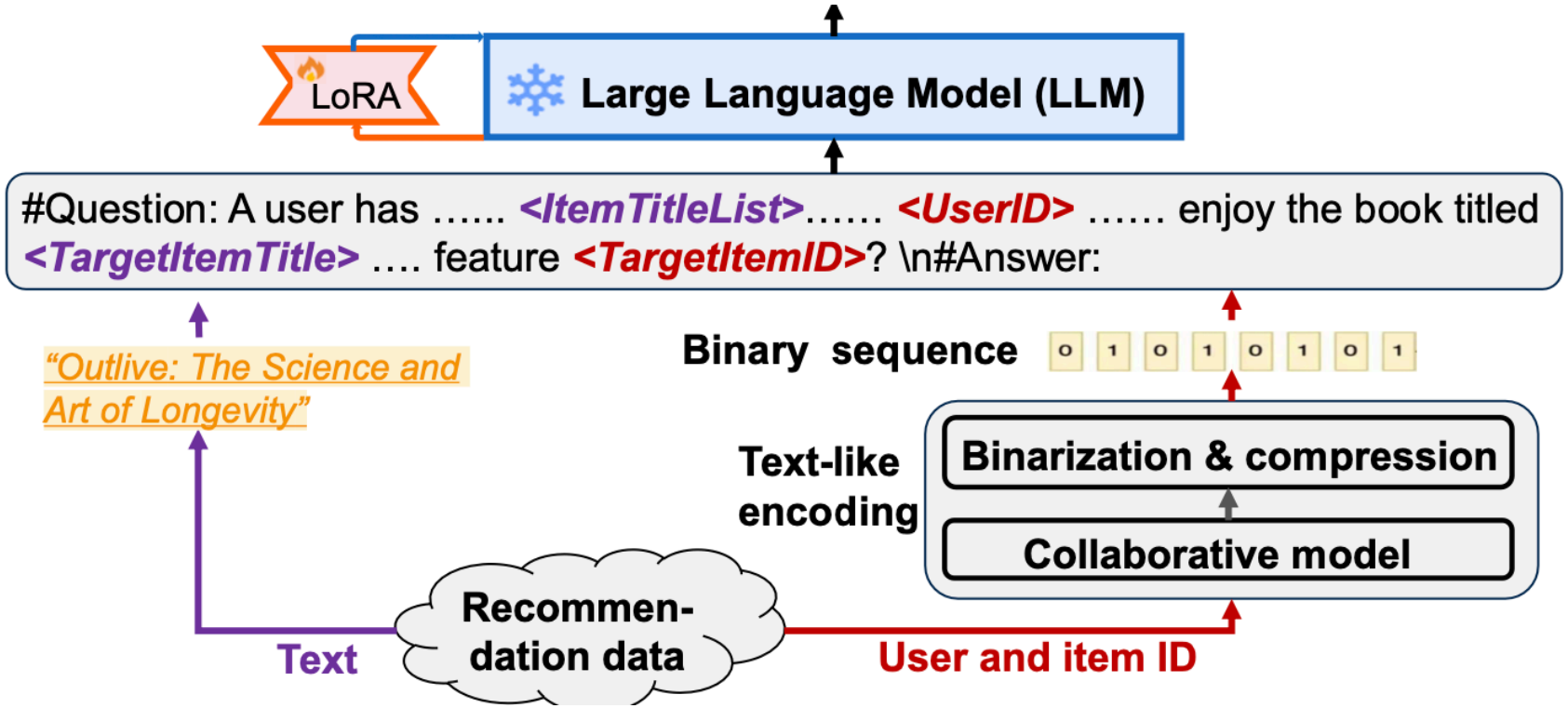
Tomando inspiración en:

- La viabilidad de binarizar embeddings colaborativos sin comprometer el rendimiento (Tan et al., 2020)
- LLMs pueden realizar operaciones a nivel de bits de forma natural o después de un ajuste por instrucciones (Savelka et al., 2023)

La contribución de los autores es:

- **BinLLM**: un enfoque innovador de LLMRec que integra información colaborativa en los LLMs al codificar los embeddings colaborativos como secuencias binarias.

BinLLM


LoRA
Large Language Model (LLM)
#Question: A user has …… <ItemTitleList>…… <UserID> …… enjoy the book titled <TargetItemTitle> …. feature <TargetItemID>? \n#Answer:
"Outlive: The Science and Art of Longevity"
Binary sequence
0 1 0 1 0 1 0 1
Text-like encoding
Binarization & compression
Collaborative model
Recommen-dation data
Text
User and item ID

Plantilla de prompt

#Question: A user has given high ratings to the following books: `ItemTitleList`. Additionally, we have information about the user's preferences encoded in the feature `UserID`. Using all available information, make a prediction about whether the user would enjoy the book titled `TargetItemTitle` with the feature `TargetItemID`? Answer with "Yes" or "No".
#Answer:

Codificación de Información Colaborativa

Dado un usuario u y un item i, obtendremos sus embeddings a través de un modelo f:

$$e_u = f(u; \theta)$$

$$e_i = f(i; \theta)$$

Para binarizar los embeddings, se utiliza una capa totalmente conectada:

$$\boldsymbol{h}_u = sign(\sigma(W e_u + b))$$

$$\boldsymbol{h}_i = sign(\sigma(W e_i + b))$$

En donde:

- $e_u \in \mathcal{R}^d$
- $e_i \in \mathcal{R}^d$
- $\boldsymbol{h}_u \in \{0,1\}^d$
- $\boldsymbol{h}_i \in \{0,1\}^d$
- $W \in \mathcal{R}^{d \times d}$
- $b \in \mathcal{R}^d$.

Configuración

- Tarea: Predicción de click
- Conjunto de datos: MovieLens-1M y Amazon-Book
- LLM: Vicuna-7B
- Largo representación binaria: 32
- Métrica de evaluación: AUC
- Función de activación: Tanh
- Función de pérdida: Binary cross-entropy

Entrenamiento

El entrenamiento se compone de dos partes fundamentales:

- **Pre-training for Text-like Encoding**
- **LoRA Tuning**

Pre-training for Text-like Encoding

En esta etapa, se entrena el módulo encargado de codificar la información colaborativa en un formato binario que pueda ser procesado por LLMs.

Se realiza de la siguiente manera:

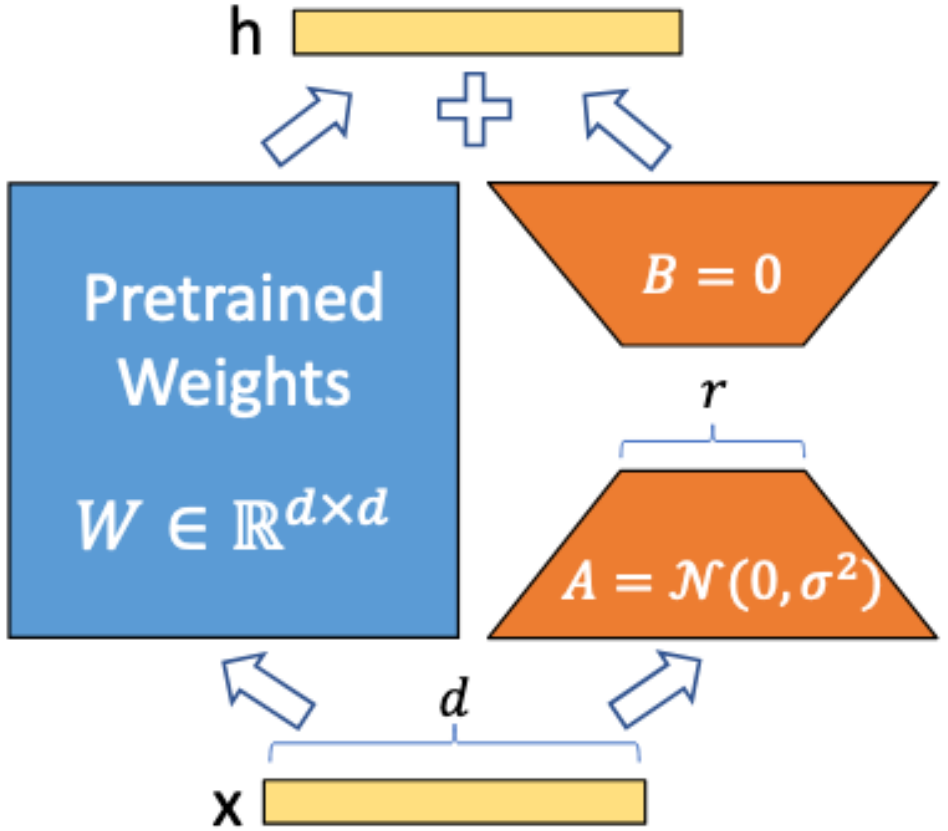
1. **Obtención de embeddings colaborativos**: A partir de usuarios e ítems, se generan vectores latentes.
2. **Binarización**: Los vectores se convierten en secuencias binarias mediante una capa completamente conectada y la función de activación `sign`:

$$h_u = \text{sign}(\sigma(We_u + b))$$

3. **Optimización**: Se minimiza una loss function definida como binary cross-entropy para ajustar las predicciones con los datos de entrenamiento. Para superar la falta de derivada de la función `sign`, se usa el Straight-Through Estimator (STE) para aproximar gradientes. El STE consiste en no derivar aquellas funciones que no tengan una derivada definida en todo el espacio.

LoRA

El método **Low-Rank Adaptation (LoRA)** adapta eficientemente los pesos de las capas densas de un modelo pre-entrenado, manteniendo los pesos originales congelados. Esto se hace optimizando matrices de baja dimensión que modelan los cambios durante la adaptación.



Profundización LoRA

En LoRA, se realiza una adaptación de los pesos de capas densas mediante dos matrices clave, (A) y (B):

Componentes del Diagrama

1. **Pesos pre-entrenados ($W \in \mathbb{R}^{d \times d}$)**:
    - Son los pesos originales del modelo pre-entrenado, que se mantienen **congelados** durante el proceso de ajuste.

2. **Matriz (A)**:
    - Representa un cambio proyectado hacia una dimensionalidad baja.
    - Se inicializa con una distribución normal (A \sim \mathcal{N}(0, \sigma^2)) porque esto facilita una exploración más eficiente del espacio de soluciones en las primeras etapas del entrenamiento.
    - Es entrenable.
3. **Matriz (B)**:
    - Proyecta de vuelta al espacio de alta dimensionalidad.
    - Se inicializa como cero ((B = 0)) para que, al inicio del entrenamiento, los cambios sean mínimos y no afecten los pesos originales (W).
    - También es entrenable.

4. **Combinación de (A) y (B)**:

   - Las matrices (A) y (B) juntas permiten capturar los cambios requeridos en los pesos del modelo sin modificar los pesos pre-entrenados (W). El cambio total se calcula como:

$$\Delta W = A \cdot B$$

5. **Entrenamiento**:

   - Solo se entrenan las matrices (A) y (B), lo que reduce significativamente la cantidad de parámetros que deben ser optimizados en comparación con un ajuste completo del modelo.
   - Esto es computacionalmente eficiente y reduce los requisitos de hardware, manteniendo el desempeño del modelo.

LoRA Tuning

Métodos

1. **Intuitive Tuning**:
    - Ajusta directamente las matrices de baja dimensión utilizando prompts enriquecidos con información colaborativa binarizada.

2. **Two-Step Tuning**:
    - **Paso 1**: Entrenamiento inicial con prompts que excluyen la información colaborativa.
    - **Paso 2**: Mayor ajuste con prompts completos que incluyen información colaborativa.
    - Este enfoque evita que el modelo dependa excesivamente de las representaciones binarias y mejora el balance con otras características del dataset.

Inferencia

Considerando un prompt p, la predicción se puede formular como:

$$\hat{y} = LLM_{\hat{\Phi}+\Phi'}(p)$$

En donde $\hat{\Phi}$ denota los parámetros del LLM y Φ' los del módulo LoRA.

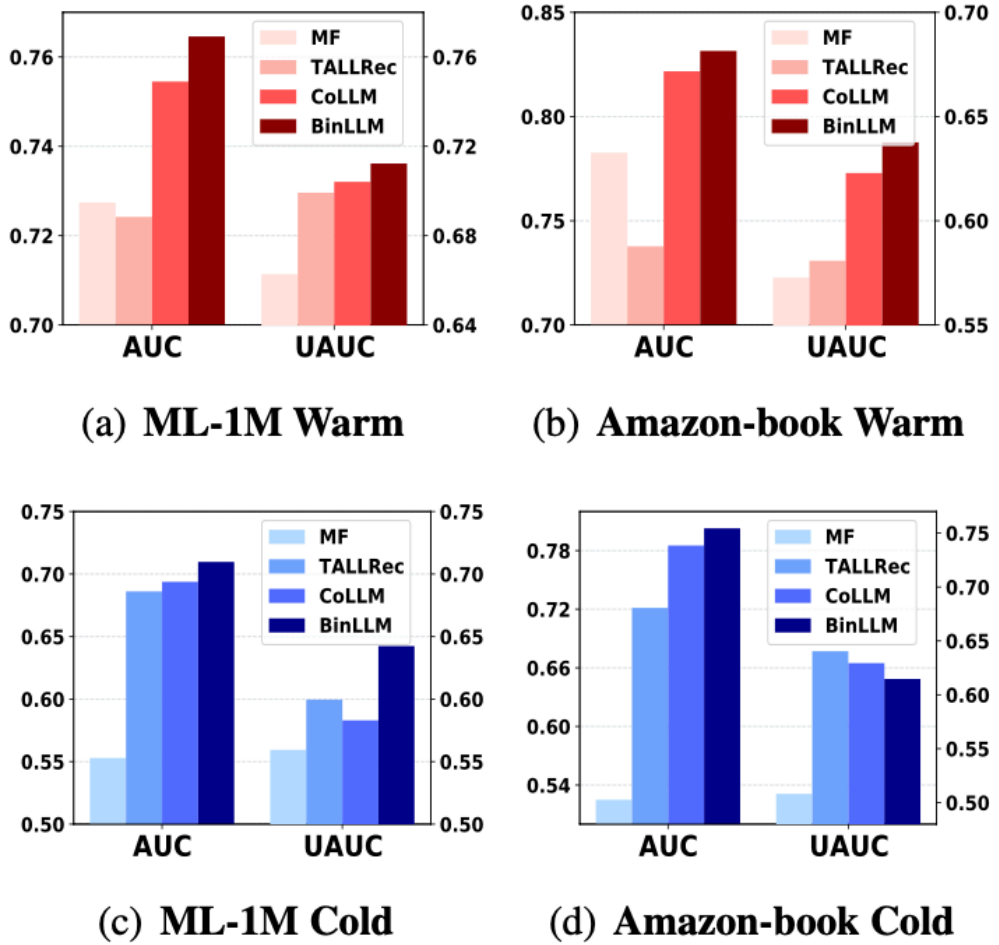
Evaluación

Dataset		ML-1M			Amazon-Book		
Methods		AUC	UAUC	Rel. Imp.	AUC	UAUC	Rel. Imp.
Collab.	MF	0.6482	0.6361	12.9%	0.7134	0.5565	14.7%
	LightGCN	0.5959	0.6499	15.8%	0.7103	0.5639	14.2%
	SASRec	0.7078	0.6884	3.0%	0.6887	0.5714	15.3%
	DIN	0.7166	0.6459	5.6%	0.8163	0.6145	2.0%
LM+Collab.	CTRL (DIN)	0.7159	0.6492	5.4%	0.8202	0.5996	3.0%
LLMRec	ICL	0.5320	0.5268	35.8%	0.4820	0.4856	50.7%
	Prompt4NR	0.7071	0.6739	4.1%	0.7224	0.5881	10.9%
	TALLRec	0.7097	0.6818	3.3%	0.7375	0.5983	8.2%
LLMRec+Collab.	PersonPrompt	0.7214	0.6563	4.5%	0.7273	0.5956	9.9%
	CoLLM-MF	0.7295	0.6875	1.5%	0.8109	0.6225	1.7%
	CoLLM-DIN	0.7243	0.6897	1.7%	0.8245	**0.6474**	-1.0%
Ours	BinLLM	**0.7425**	**0.6956**	-	**0.8264**	0.6319	-

Experimentos

Rendimiento en warm / cold

Los datos de prueba se dividen en "warm" o "cold" en base al número de interacciones.



(a) ML-1M Warm (b) Amazon-book Warm

(c) ML-1M Cold (d) Amazon-book Cold

La influencia de los componentes

BinMF: Solo las representaciones binarias y MF

BinLLM-TO: Solo los textos en el prompt

BinLLM-IO: Solo los ids en el prompt

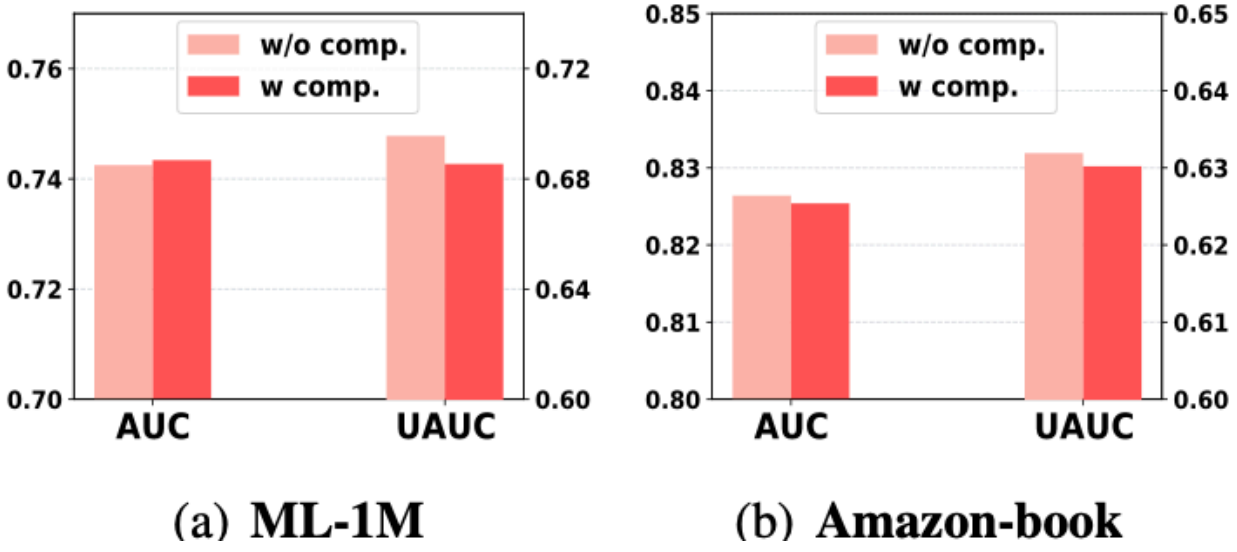
BinLLM-IT: Intuitive tuning (IT)

Datasets	ML-1M		Amazon-book	
Methods	AUC	UAUC	AUC	UAUC
BinMF	0.7189	0.6654	0.8087	0.5895
BinLLM-TO	0.7097	0.6818	0.7375	0.5983
BinLLM-IO	0.7307	0.6797	0.8173	0.5919
BinLLM-IT	0.7286	0.6842	0.8246	0.6165
BinLLM	0.7425	0.6956	0.8264	0.6319

La influencia de comprimir

Una limitación de las secuencias binarias es su largo.

Los autores evaluan el efecto de comprimirlas, convirtiendo cada ocho dígitos binarios en un número decima y utilizando el punto como carácter de separación.



(a) **ML-1M** (b) **Amazon-book**

Conclusiones

- BinLLM es un método robusto para la tarea de predicción de click
- Dada una matriz de interacción "cold", los métodos LLMRec son los más adecuados
- BinLLM aprovecha tanto la información semántica como la colaborativa para destacarse sobre los otros métodos
- La compresión puede reducir el largo de la secuencia binaria manteniendo en gran medida el desempeño de BinLLM

Referencias

1. Zheng, B., Hou, Y., Lu, H., Chen, Y., Zhao, W. X., Chen, M., & Wen, J.-R. (2024). *Adapting Large Language Models by Integrating Collaborative Semantics for Recommendation*. arXiv. https://arxiv.org/abs/2311.09049
2. Hua, W., Xu, S., Ge, Y., & Zhang, Y. (2023). *How to Index Item IDs for Recommendation Foundation Models*. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region* (SIGIR-AP '23), 195–204. ACM. https://doi.org/10.1145/3624918.3625339

3. Zhang, Y., Feng, F., Zhang, J., Bao, K., Wang, Q., & He, X. (2024). *CoLLM: Integrating Collaborative Embeddings into Large Language Models for Recommendation*. arXiv. https://arxiv.org/abs/2310.19488
4. Li, X., Chen, C., Zhao, X., Zhang, Y., & Xing, C. (2023). *E4SRec: An Elegant Effective Efficient Extensible Solution of Large Language Models for Sequential Recommendation*. arXiv. https://arxiv.org/abs/2312.02443
5. Liao, J., Li, S., Yang, Z., Wu, J., Yuan, Y., Wang, X., & He, X. (2024). *LLaRA: Large Language-Recommendation Assistant*. arXiv. https://arxiv.org/abs/2312.02445

6. Tan, Q., Liu, N., Zhao, X., Yang, H., Zhou, J., & Hu, X. (2020). *Learning to Hash with Graph Neural Networks for Recommender Systems*. arXiv. https://arxiv.org/abs/2003.01917

7. Savelka, J., Agarwal, A., An, M., Bogart, C., & Sakr, M. (2023). *Thrilled by Your Progress! Large Language Models (GPT-4) No Longer Struggle to Pass Assessments in Higher Education Programming Courses*. Proceedings of the 2023 ACM Conference on International Computing Education Research V.1. ACM. http://dx.doi.org/10.1145/3568813.3600142

8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need*. CoRR, abs/1706.03762. http://arxiv.org/abs/1706.03762

Anexos