August 3-7, 2025 KDD2025

KDD 2025 – ACM SIGKDD Conference on Knowledge Discovery and Data Mining

Lost in Sequence: Do Large Language Models Understand Sequential Recommendation ?

Sein Kim[1]*, Hongseok Kang[1]*, Kibum Kim[1], Jiwan Kim[1], Donghyun Kim[2], Minchul Yang[2], Kwangjin Oh[2], Julian McAuley[3],Chanyoung Park[1]†

[1]KAIST, [2]NAVER Corporation, [3]University of California San Diego

*: these authors contributed equally to this work

†: corresponding author

KAIST DSAIL Data Science & Artificial Intelligence NAVER UC San Diego

LLM-based Recommendation

Table 1: An example prompt for various LLM4Rec models (Next Item Title Generation approach).

	(a) TALLRec	(b) LLaRA	(c) CoLLM/A-LLMRec
Inputs ($\mathcal{P}^u$)	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles].	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles, Item Embeddings].	This is user representation from recommendation models: [User Representation], and this user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles, Item Embeddings].

LLM-based Recommendation

- Many LLM-based recommendation studies, like TALLRec, LLaRA, and A-LLMRec, **leverage item metadata and user interaction history** to generate effective recommendations.
- These studies typically incorporate **user interaction history** into the input prompt **by serializing previous interaction as sentence.**

LLM-based Recommendation

Table 1: An example prompt for various LLM4Rec models (Next Item Title Generation approach).

	(a) TALLRec	(b) LLaRA	(c) CoLLM/A-LLMRec
Inputs ($\mathcal{P}^u$)	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles].	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles, Item Embeddings].	This is user representation from recommendation models: [User Representation], and this user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles, Item Embeddings].

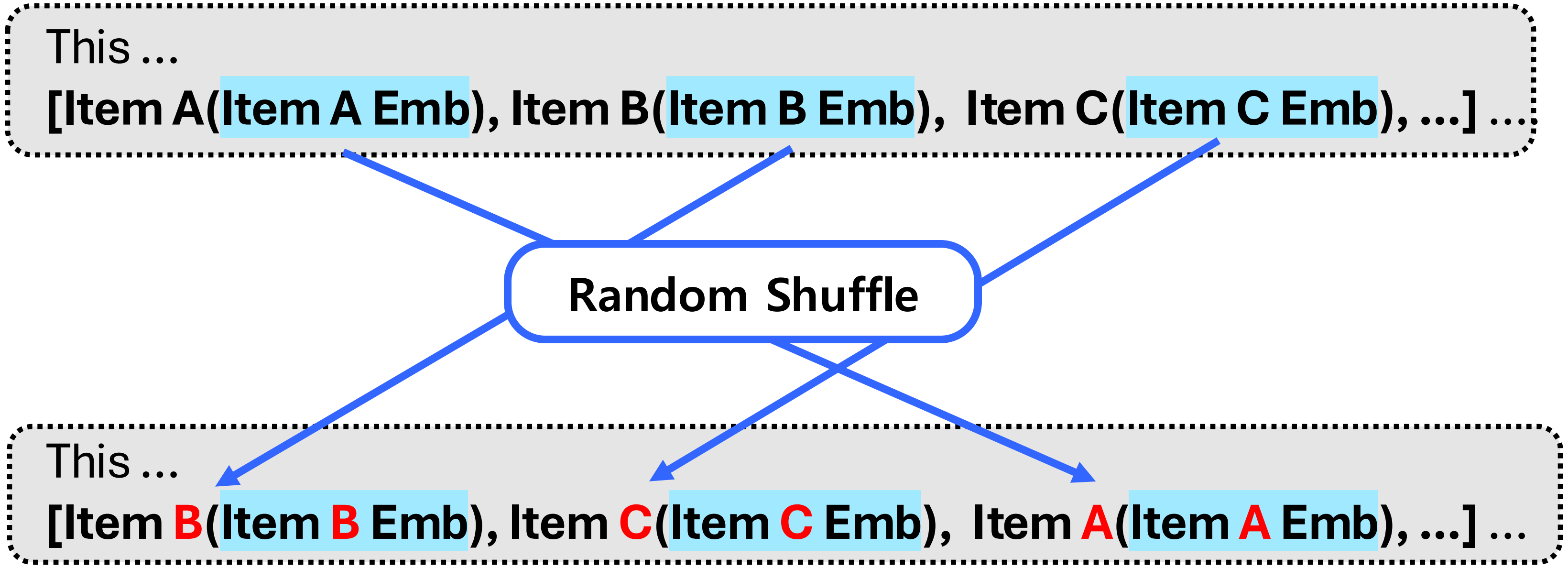
LLM-based Recommendation

- Many LLM-based recommendation studies, like TALLRec, LLaRA, and A-LLMRec, **leverage item metadata and user interaction history** to generate effective recommendations.
- These studies typically incorporate **user interaction history** into the input prompt **by serializing previous interaction as sentence.**

→ Question: Do LLMs understand sequential information in user interaction history ?

Preliminary Analysis (PA): Shuffle

<Example of **Original** Sequence>



<Example of **Shuffled** Sequence>

Hypothesis: Performance of models that understand the sequential information would deteriorate when the sequence is disrupted

PA: Prompt Setting

Table 1: An example prompt for various LLM4Rec models (Next Item Title Generation approach).

	(a) TALLRec	(b) LLaRA	(c) CoLLM/A-LLMRec
Inputs ($\mathcal{P}^u$)	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles].	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles, Item Embeddings].	This is user representation from recommendation models: [User Representation], and this user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Choose one "Title" to recommend for this user to buy next from the following item "Title" set: [Candidate Item Titles, Item Embeddings].
Outputs ($\text{Text}(i^{(u)}_{n_u+1})$)	Item Title	Item Title	Item Title

Table 2: An example prompt for various LLM4Rec models (Next Item Retrieval approach).

	(a) TALLRec	(b) LLaRA/LLM-SRec (Ours)	(c) CoLLM/A-LLMRec
User ($\mathcal{P}^u_{\mathcal{U}}$)	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title]). Based on this sequence of purchases, generate user representation token: [UserOut].	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Based on this sequence of purchases, generate user representation token: [UserOut].	This is user representation from recommendation models: [User Representation], and this user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title, Item Embedding]). Based on this sequence of purchases and user representation, generate user representation token: [UserOut].
Item ($\mathcal{P}^i_{\mathcal{I}}$)	The item title is as follows: "Title": Item Title, then generate item representation token: [ItemOut].	The item title and item embedding are as follows: "Title": Item Title, Item Embedding, then generate item representation token: [ItemOut]	

- Recent LLM-based recommendation methods can be broadly categorized into two approaches
  - **Next Item Title Generation**: Given a candidate set, generates the next item title from that set
  - **Next Item Retrieval**: Extract user and item representation using the LLM, and recommendations are made based on the similarity between these representations

$$\mathbf{h}^u_{\mathcal{U}} = \text{LLM}(\mathcal{P}^u_{\mathcal{U}}, \mathcal{D}'^u), \quad \mathbf{h}^i_{\mathcal{I}} = \text{LLM}(\mathcal{P}^i_{\mathcal{I}}, \mathcal{D}'^i) \qquad s(u,i) = f_{item}(\mathbf{h}^i_{\mathcal{I}}) \cdot f_{user}(\mathbf{h}^u_{\mathcal{U}})^T$$

PA 1: Train Stage – Sequence Shuffling

This …
[Item A(Item A Emb), Item B(Item B Emb), Item C(Item C Emb), …] ….

<Example of **Original** Sequence>

This …
[Item B(Item B Emb), Item C(Item C Emb), Item A(Item A Emb), …] …

<Example of **Shuffled** Sequence>

- **Train** the same model on both the **original sequence** and **shuffled sequence.**
- Perform **inference** on these models, using the **original sequence.**

The sequential patterns in the original and shuffled sequences are different !

→The model **trained on the shuffled sequence** did not learn the original sequential pattern, so its **performance will be worse** !

PA 1: Train Stage – Sequence Shuffling

		Scientific	Electronics	CDs
SASRec	Original	0.3171	0.2390	0.3662
	Shuffle	0.2821	0.2158	0.3386
	Change ratio	(-11.04%)	(-9.71%)	(-7.54%)
TALLRec	Original	0.2221	0.1787	0.2589
	Shuffle	0.2181	0.1815	0.2728
	Change ratio	(-1.81%)	(+1.57%)	(+5.37%)
LLaRA	Original	0.3022	0.2616	0.3142
	Shuffle	0.2996	0.2650	0.3530
	Change ratio	(-0.86%)	(+1.30%)	(+12.35%)
CoLLM	Original	0.3010	0.2311	0.3447
	Shuffle	0.3165	0.2323	0.3763
	Change ratio	(+5.15%)	(+0.52%)	(+9.17%)
A-LLMRec	Original	0.2804	0.2672	0.3319
	Shuffle	0.2796	0.2684	0.3528
	Change ratio	(-0.29%)	(+0.45%)	(+6.30%)

<Next Item Title Generation Task>

SASRec

Performance Change: -11% to -7 %

Collaborative-based

LLM-based

LLM-based Models

Performance Change: -1% to +N %

- **SASRec**, which **learns the sequential patterns**, training on the shuffled sequence leads to a significant performance drop.
- However, **LLM-based models show almost no performance drop** when trained on a shuffled sequence.

This suggests that LLMs do not effectively capture user-item interaction sequential information.

PA 2: Inference Stage – Sequence Shuffling

This ...
[Item A(Item A Emb), Item B(Item B Emb), Item C(Item C Emb), ...] ....

<Example of **Original** Sequence>

This ...
[Item B(Item B Emb), Item C(Item C Emb), Item A(Item A Emb), ...] ...

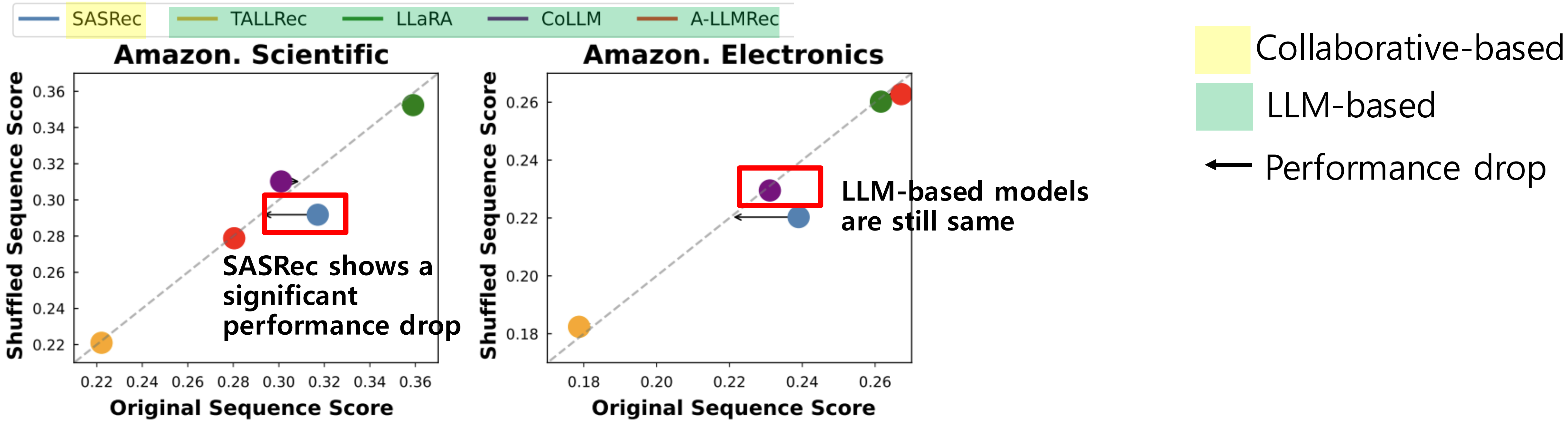
<Example of **Shuffled** Sequence>

- **Train** one model on the original sequence data.
- Perform **inference** on this model using the original sequence and shuffled sequence.

Since the **shuffled sequence contains different sequential information** compared to what the model has learned.

→There will be a performance difference between those sequences.

PA 2: Inference Stage – Sequence Shuffling



- **SASRec**, which uses only the sequential pattern (blue), **shows a significant performance drop**, when performing inference on a shuffled sequence
- In contrast, the **LLM-based recommender shows almost the same performance regardless of the shuffled sequence**

This suggests that LLMs do not understand the sequential information in the interaction history

PA 3: User Representation Similarity

	Movies	Scientific	Electronics	CDs
SASRec	0.6535	0.7375	0.7083	0.7454
TALLRec	0.9731	0.9326	0.9678	0.9570
LLaRA	0.9639	0.9424	0.9800	0.9624
CoLLM	0.9067	0.9263	0.8921	0.9526
A-LLMRec	0.8872	0.8911	0.8623	0.8706
LLM-SRec	0.6128	0.7852	0.7393	0.8589

LLM-based models exhibit high cosine similarity

	(a) **TALLRec**
User ($\mathcal{P}^u_{\mathcal{U}}$)	This user has made a series of purchases in the following order: (History Item List: [No.# Time: YYYY/MM/DD Title: Item Title]). Based on this sequence of purchases, generate user representation token: [UserOut].

<Example of User representation extraction prompt>

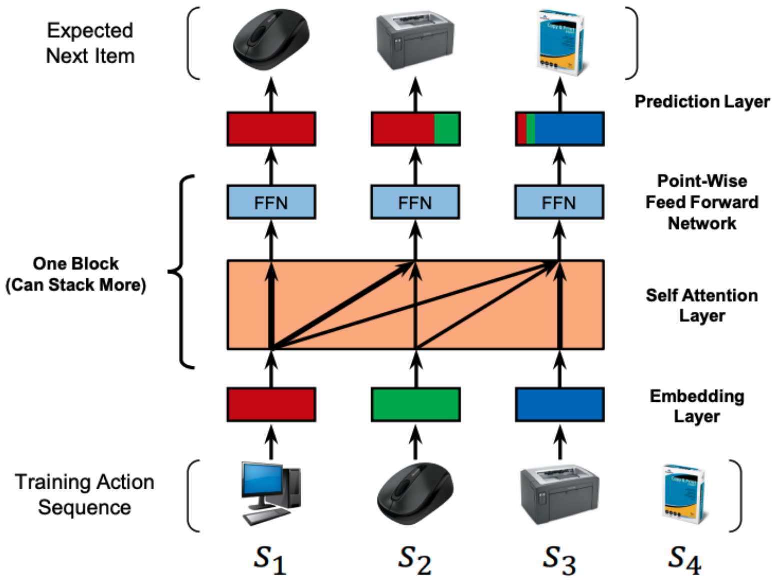
Collaborative-based

LLM-based

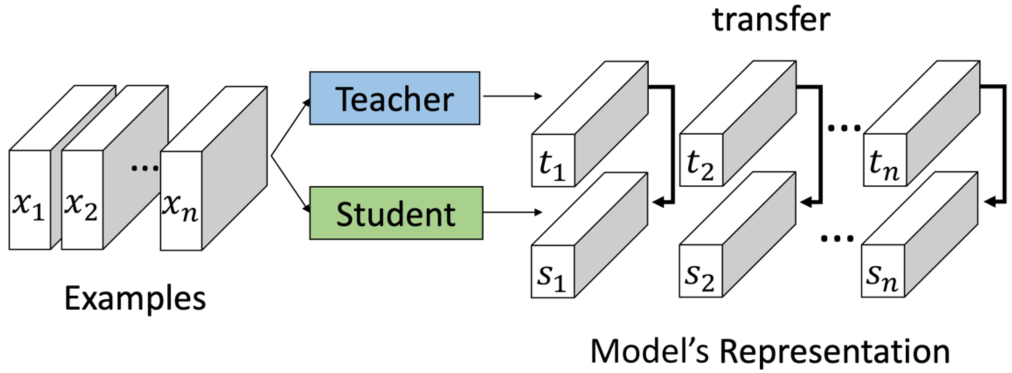
- Extract user representations from both original and shuffled sequences and compute their cosine similarity
  - If the model understands sequence information, the representations should differ due to the change in sequential patterns
- While SASRec shows low similarity between the two representations, LLM-based models show high similarity

This suggests that LLMs have not yet fully captured sequential patterns

How to Integrate Sequential Information to LLMs



<SASRec architecture>



<Knowledge distillation>

- Collaborative filtering-based sequential recommenders (CF-SRec), such as SASRec, effectively capture sequential patterns
- Using knowledge distillation, we can easily transfer this sequential knowledge from CF-SRec to LLMs

Relational Knowledge Distillation, CVPR 2019

How to Integrate Sequential Information to LLMs

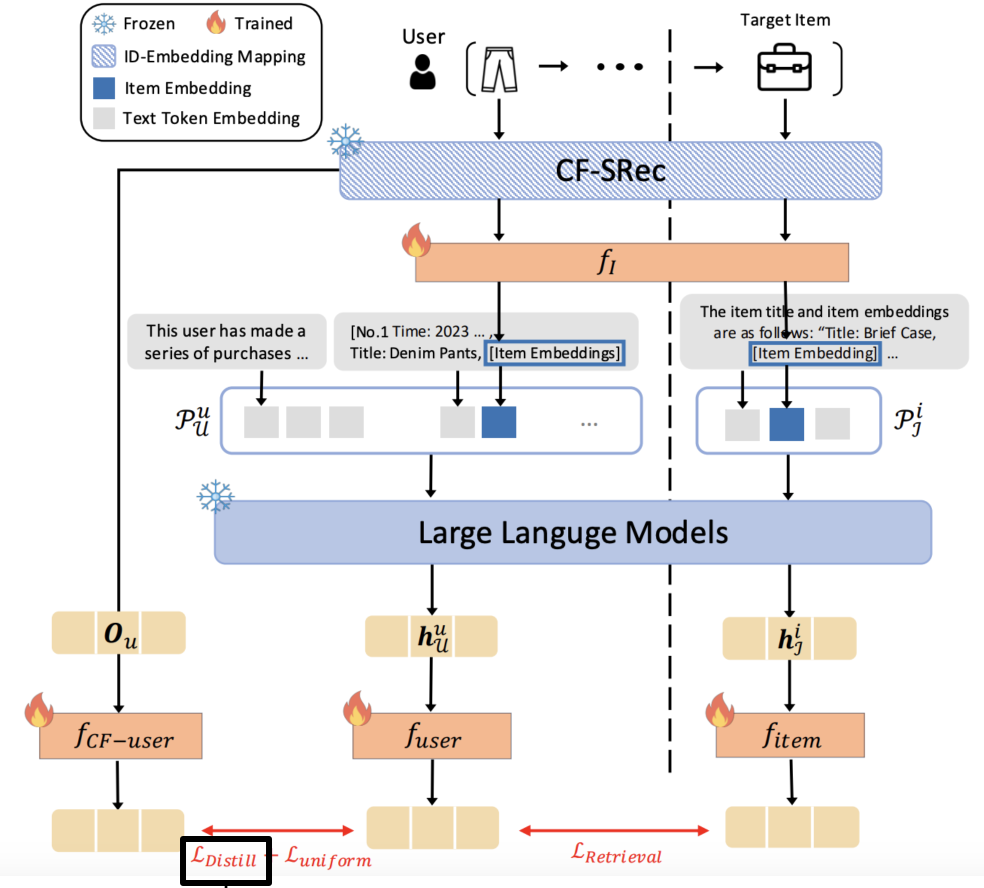
- We perform knowledge distillation by aligning the user representations from a pre-trained CF-SRec (e.g., SASRec) with user representations generated by the LLMs

2-layer MLP

$$\mathcal{L}_{\text{Distill}} = \mathop{\mathbb{E}}_{u \in \mathcal{U}} [\text{MSE}(f_{CF-user}(\mathbf{O}_u), f_{user}(\mathbf{h}^u_{\mathcal{U}}))]$$

User representation from CF-SRec (e.g., SASRec)

User representation from LLM



How to Integrate Sequential Information to LLMs

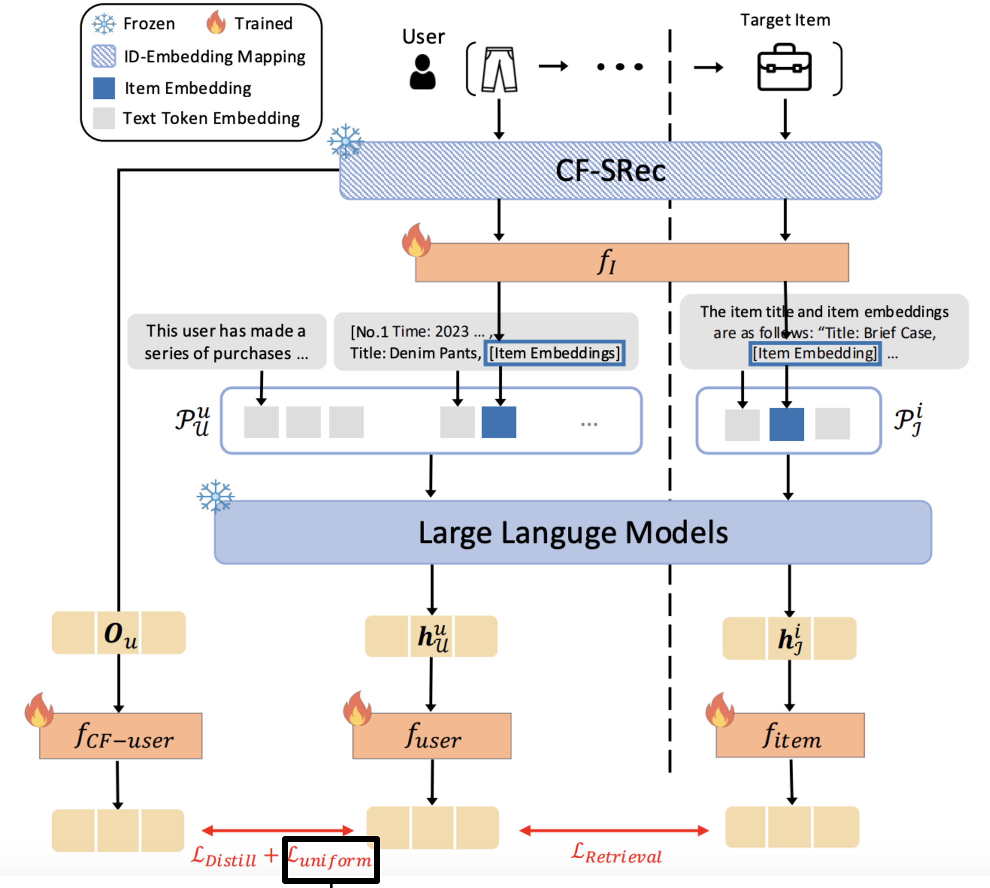
- Using MSE for the distillation can lead to over-smoothing
- We construct uniformity loss to prevent over-smoothing

Separate user representation from CF-SRec

$$\mathcal{L}_{\text{Uniform}} = \mathop{\mathbb{E}}_{u \in \mathcal{U}} \Big[\mathop{\mathbb{E}}_{u' \in \mathcal{U}} \big[e^{-2\| f_{CF-user}(\mathbf{O}_u) - f_{CF-user}(\mathbf{O}_{u'}) \|_2^2} \big] \Big]$$

$$+ \mathop{\mathbb{E}}_{u \in \mathcal{U}} \Big[\mathop{\mathbb{E}}_{u' \in \mathcal{U}} \big[e^{-2\| f_{user}(\mathbf{h}^u_{\mathcal{U}}) - f_{user}(\mathbf{h}^{u'}_{\mathcal{U}}) \|_2^2} \big] \Big]$$

Separate user representation from LLMs



- Final objective: $\mathcal{L} = \mathcal{L}_{\text{Retrieval}} + \mathcal{L}_{\text{Distill}} + \mathcal{L}_{\text{Uniform}}$

Experiments: Dataset & Baseline

- Four public dataset (AMAZON 2023 dataset)
  - 5-core (All users and items have at least 5 interactions)

Dataset	Movies	Scientific	Electronics	CDs
# Users	11,947	23,627	27,601	18,481
# Items	17,490	25,764	31,533	30,951
# Interactions	144,071	266,164	292,308	284,695

- Baseline
  - CF-SRec: GRU4Rec, BERT4Rec, NextItNet, SASRec
  - LM-based: CTRL, RECFORMER
  - LLM-based: TALLRec, LLaRA, CoLLM, A-LLMRec

Experiments: Overall Performance

Dataset	Metric	CF-SRec				LM-based		LLM4Rec				
		GRU4Rec	BERT4Rec	NextItNet	SASRec	CTRL	RECFORMER	TALLRec	LLaRA	CoLLM	A-LLMRec	LLM-SRec
Movies	NDCG@10	0.3152	0.2959	0.2538	0.3459	0.2785	0.2068	0.1668	0.1522	0.3223	0.3263	**0.3560**
	NDCG@20	0.3494	0.3303	0.2879	0.3745	0.3099	0.2337	0.2126	0.1944	0.3577	0.3629	**0.3924**
	HR@10	0.4883	0.4785	0.4221	0.5180	0.4264	0.3569	0.3234	0.2914	0.5089	0.5127	**0.5569**
	HR@20	0.6245	0.6213	0.5522	0.6310	0.5429	0.5264	0.5060	0.4599	0.6491	0.6577	**0.7010**
Scientific	NDCG@10	0.2642	0.2576	0.2263	0.2918	0.2152	0.2907	0.2585	0.2844	0.3111	0.2875	**0.3388**
	NDCG@20	0.2974	0.2913	0.2657	0.3245	0.2520	0.3113	0.3048	0.3265	0.3489	0.3246	**0.3758**
	HR@10	0.4313	0.4437	0.3908	0.4691	0.3520	0.4506	0.4574	0.4993	0.5182	0.4957	**0.5532**
	HR@20	0.5524	0.5822	0.5356	0.5987	0.4882	0.5710	0.6276	0.6658	0.6676	0.6427	**0.6992**
Electronics	NDCG@10	0.2364	0.1867	0.1712	0.2267	0.1680	0.2032	0.2249	0.2048	0.2565	0.2791	**0.3044**
	NDCG@20	0.2743	0.2172	0.2069	0.2606	0.2003	0.2441	0.2670	0.2454	0.2948	0.3173	**0.3424**
	HR@10	0.3843	0.3325	0.3017	0.3749	0.2861	0.3586	0.3802	0.3441	0.4236	0.4622	**0.4885**
	HR@20	0.5196	0.4740	0.4324	0.5096	0.4152	0.5213	0.5476	0.5032	0.5741	0.6137	**0.6385**
CDs	NDCG@10	0.2155	0.3019	0.2207	0.3451	0.2968	0.3238	0.3100	0.2464	0.3152	0.3119	**0.3809**
	NDCG@20	0.2530	0.3386	0.2562	0.3795	0.3316	0.3642	0.3493	0.2951	0.3557	0.3526	**0.4158**
	HR@10	0.3712	0.5018	0.3842	0.5278	0.4574	0.5140	0.5052	0.4665	0.5290	0.5300	**0.6085**
	HR@20	0.5092	0.6605	0.5422	0.6635	0.5957	0.6739	0.6633	0.6590	0.6895	0.6914	**0.7461**

- LLM-SRec achieves the superior performance by leveraging textual, collaborative, and sequential information, highlighting the importance of sequential modeling for LLMs
- In LLM4Rec models, incorporating collaborative information is crucial (TALLRec vs. Other LLM4Rec)

Experiments: Transition vs. Non-Transition

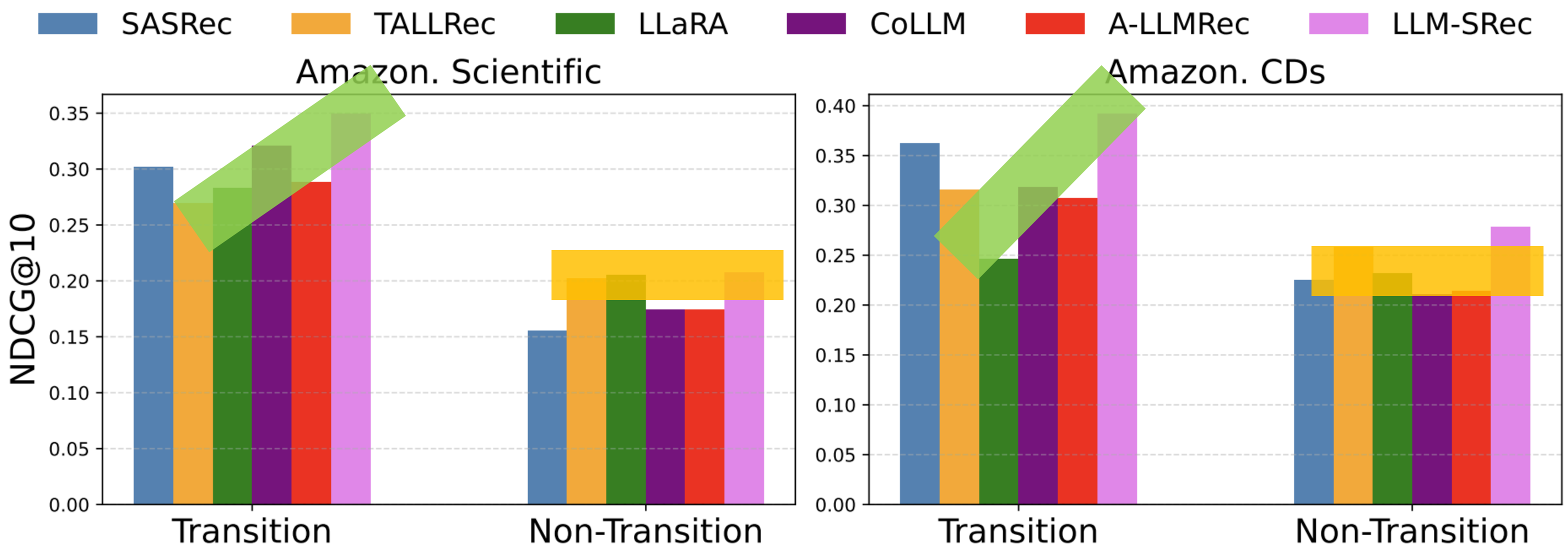
- **Transition set**: Sequences where order matters
- **Non-Transition set**: Sequence with weak or no sequential patterns

  - Count item-to-item transitions (Item i → Item j) across the entire training dataset
  - Based on these counts, compute a transition score (t-score) for each user

$$\text{t-score}^{u} = \left(\sum_{t=1}^{n_u - 1} Count\left(i_t^{(u)} \rightarrow i_{t+1}^{(u)}\right)\right) / \left(n^{u} - 1\right).$$

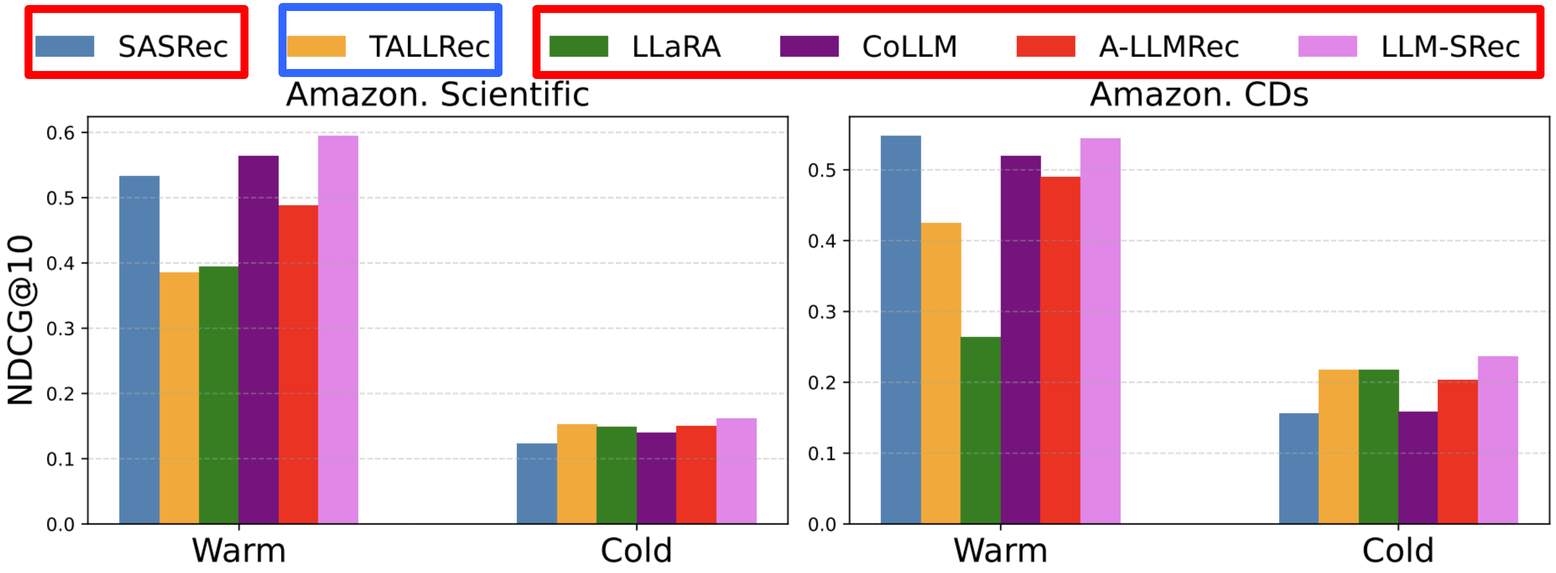
  - Transition set: Users in top 50 % t-score
  - Non-Transition set: Remaining users

Experiments: Transition vs. Non-Transition



- LLM-SRec achieves the best overall performance, particularly large performance gap over baselines in the Transition set, where sequential information is matter
- While the performance difference between LLM-SRec and LLM4Rec baselines is small in the Non-Transition Set, it becomes significant in the Transition set
  - This highlights the LLM4Rec baselines struggle to modeling sequential patterns
  - Underscores the importance of effective sequential modeling in LLM-based recommenders

Experiments: Warm/Cold Scenarios



- LLM-SRec shows superior performance in both warm and cold scenarios
  - By leveraging collaborative and textual information, LLM-SRec generalizes across warm/cold scenarios and further improves performance by incorporating sequential signal

- In cold setting, LLM4Rec models perform well, while in warm settings, models that incorporate collaborative knowledge show better results
  - These findings highlight the importance of effectively modeling both textual and collaborative information for recommendation task

Experiments: Ablation Study

Row	Ablation		Movies NDCG@10	Movies NDCG@20	Scientific NDCG@10	Scientific NDCG@20	Electronics NDCG@10	Electronics NDCG@20	CDs NDCG@10	CDs NDCG@20
(a)	w.o. $\mathcal{L}_{\text{Distill}}$, $\mathcal{L}_{\text{Uniform}}$	Original	0.3204	0.3569	0.3088	0.3450	0.2659	0.3066	0.2278	0.2701
		Shuffle	0.3176	0.3557	0.3013	0.3379	0.2589	0.2990	0.2224	0.2649
		Change ratio	(-0.87%)	(-0.34%)	(-2.33%)	(-2.06%)	(-2.63%)	(-2.48%)	(-2.37%)	(-1.92%)
(b)	w.o. $\mathcal{L}_{\text{Uniform}}$	Original	0.3339	0.3700	0.3283	0.3653	0.2895	0.3285	0.3622	0.4013
		Shuffle	0.3089	0.3456	0.3164	0.3536	0.2732	0.3110	0.3478	0.3885
		Change ratio	(-7.49%)	(-6.59%)	(-3.62%)	(-3.20%)	(-5.63%)	(-5.33%)	(-3.98%)	(-3.19%)
(c)	LLM-SRec	Original	**0.3560**	**0.3924**	**0.3388**	**0.3758**	**0.3044**	**0.3424**	**0.3809**	**0.4158**
		Shuffle	0.3263	0.3624	0.3224	0.3591	0.2838	0.3210	0.3614	0.3981
		Change ratio	(-8.34%)	(-7.65%)	(-4.84%)	(-4.44%)	(-6.77%)	(-6.25%)	(-5.11%)	(-4.26%)

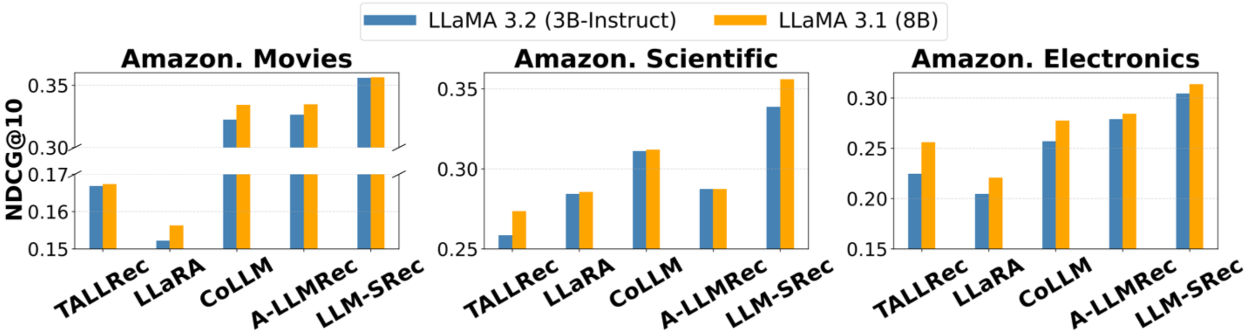
- Without the distillation loss (a), the model shows little performance difference between original and shuffled sequence, indicating a lack of understanding of sequential information
  - This demonstrates that the distillation loss effectively transfers sequence knowledge to the LLMs

- Without the uniformity loss (b), performance drops due to over-smoothing

Experiments: Train/Inference Time & LLM sizes

	Scientific		Electronics	
	Train (min/epoch)	Inference (min)	Train (min/epoch)	Inference (min)
TALLRec	194.43	37.04	236.73	29.04
LLaRA	202.20	38.79	241.17	30.62
CoLLM	214.12	39.86	251.51	32.58
A-LLMRec	190.94	35.01	235.02	28.14
LLM-SRec	**185.91**	**34.17**	**218.21**	**27.57**

<Train/Inference Time>

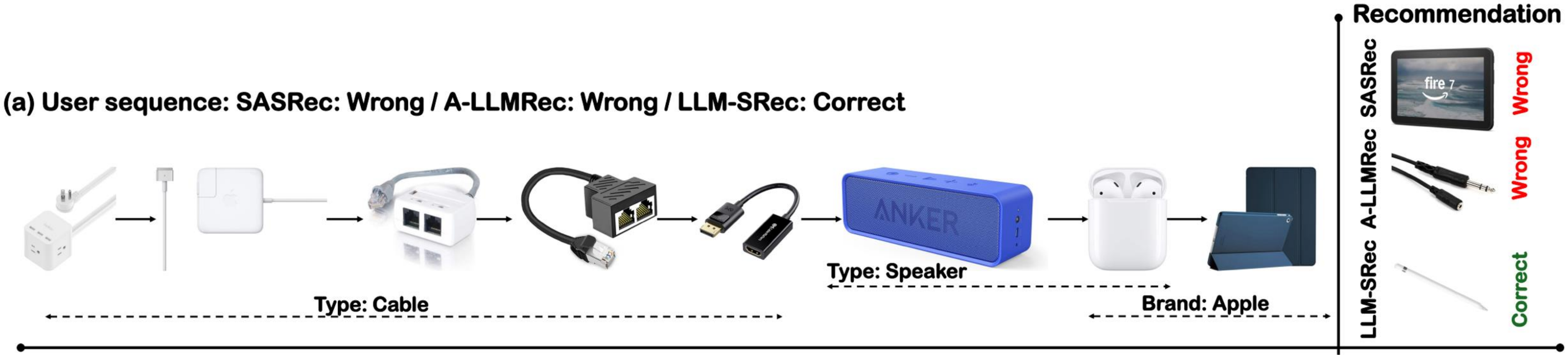


<Backbone LLM sizes>

- Train/Inference Time
  - LLM-SRec achieves the fastest training and inference time since it does not fine-tune the LLMs
  - Compared to A-LLMRec, which also does not fine-tune the LLMs, LLM-SRec is more efficient as it consists of a single learning stage and does not use user representations as prompt
- Backbone LLM sizes
  - LLM-SRec consistently achieves the highest performance across different LLM sizes
  - Notably, LLM-SRec with a small 3B model outperforms LLM4Rec baselines using a larger 8B model

    → This highlights that <u>effectively injecting sequential information is more crucial for recommendation performance than simply increasing LLM size</u>

Experiments: Case Study



- Case: Only LLM-Srec provides the correct recommendation
  - Sequence Info: The user's preference shifts from cables to Apple products
  - Textual Info: Preference for the Apple brand
    - SASRec: Capture the preference shift and stop recommending cable-related items, but fails to identify the brand preference (Apple), recommending items from other brands (Amazon) instead
    - A-LLMRec: Fail to capture the change in preference and continues to recommend only cable-related items
    - LLM-SRec: Successfully captures both the shift in interest and the emerging brand preference, recommending relevant Apple-brand items such as the Apple Pencil

Summary

- Demonstrates that existing LLM-based recommenders struggle to handle sequential information

- Proposes a method for **distilling sequential knowledge** from a pre-trained Collaborative filtering-based sequential recommender model

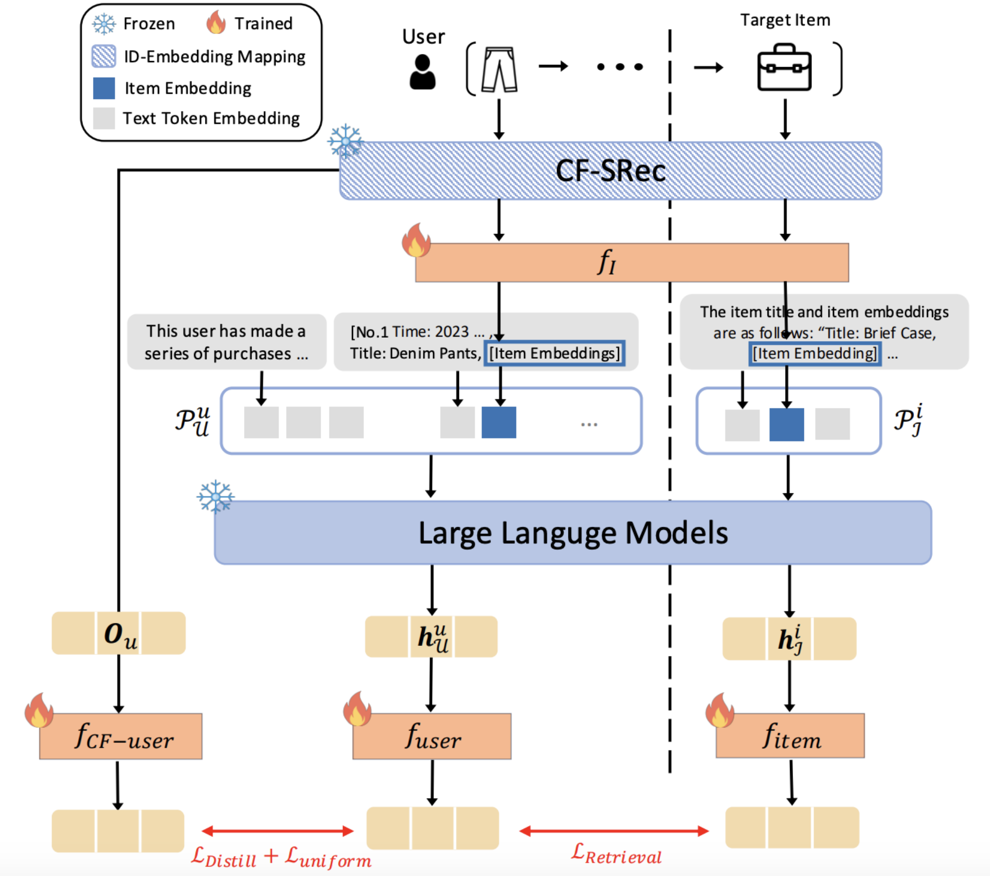
2-layer MLP

$$\mathcal{L}_{\text{Distill}} = \mathop{\mathbb{E}}_{u\in\mathcal{U}}\left[\text{MSE}\left(f_{CF-user}(\mathbf{O}_u), f_{user}(\mathbf{h}^u_{\mathcal{U}})\right)\right]$$

User representation from CF-SRec (e.g., SASRec)

User representation from LLM

→ We proposed LLM-based sequential recommender, called **LLM-SRec**



- Through extensive experiments, shows the importance of LLMs understanding sequential information
  - Overall experiments, Transition/Non-Transition, Warm/Cold, Cross-domain, LLM-size and Case study

Thank you!

[KDD' 25] Lost in Sequence: Do Large Language Models Understand Sequential Recommendation ?
[Full Paper] https://arxiv.org/abs/2502.13909
[Source Code] https://github.com/Sein-Kim/LLM-SRec

Paper **Code**

[Lab Homepage] http://dsail.kaist.ac.kr

[Email] rlatpdlsgns@kaist.ac.kr
ghdtjr0311@kaist.ac.kr

DSAIL Data Science & Artificial Intelligence