On Softmax Direct Preference Optimization for Recommendation

Yuxin Chen[1*] **Junfei Tan**[2*] **An Zhang**[1†]
Zhengyi Yang[2] **Leheng Sheng**[1] **Enzhi Zhang**[3]
Xiang Wang[2] **Tat-Seng Chua**[1]
[1]National University of Singapore
[2]University of Science and Technology of China
[3]Hokkaido University
yuxin.chen@u.nus.edu, sober_clever@mail.ustc.edu.cn, anzhang@u.nus.edu
leheng.sheng@u.nus.edu, enzhi.zhang.n6@elms.hokudai.ac.jp
yangzhy1998@gmail.com, xiangwang1223@gmail.com, dcscts@nus.edu.sg

Abstract

Recommender systems aim to predict personalized rankings based on user preference data. With the rise of Language Models (LMs), LM-based recommenders have been widely explored due to their extensive world knowledge and powerful reasoning abilities. Most of the LM-based recommenders convert historical interactions into language prompts, pairing with a positive item as the target response and fine-tuning LM with a language modeling loss. However, the current objective fails to fully leverage preference data and is not optimized for personalized ranking tasks, which hinders the performance of LM-based recommenders. Inspired by the current advancement of Direct Preference Optimization (DPO) in human preference alignment and the success of softmax loss in recommendations, we propose Softmax-DPO (**S-DPO**) to instill ranking information into the LM to help LM-based recommenders distinguish preferred items from negatives, rather than solely focusing on positives. Specifically, we incorporate multiple negatives in user preference data and devise an alternative version of DPO loss tailored for LM-based recommenders, which is extended from the traditional full-ranking Plackett-Luce (PL) model to partial rankings and connected to softmax sampling strategies. Theoretically, we bridge S-DPO with the softmax loss over negative sampling and find that it has an inherent benefit of mining hard negatives, which assures its exceptional capabilities in recommendation tasks. Empirically, extensive experiments conducted on three real-world datasets demonstrate the superiority of S-DPO to effectively model user preference and further boost recommendation performance while providing better rewards for preferred items. Our codes are available at https://github.com/chenyuxin1999/S-DPO.

1 Introduction

Recommender systems aim to predict personalized rankings based on user preference data, *i.e.,* historical interactions such as purchases, clicks, and ratings [1, 2]. Recently, leveraging the extensive world knowledge and powerful reasoning abilities of language models (LMs) [3–6], LM-based recommenders have been broadly explored [7–9]. These recommenders convert historical interaction data into language prompts and either perform in-context learning or fine-tune LMs, demonstrating notable

[*]These authors contributed equally to this work.
[†]An Zhang is the corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-0863

advantages, including zero-shot and few-shot reasoning [10–13], enhanced generalization abilities [14, 15], and rich semantic understanding [16–19]. However, current LM-based recommenders typically utilize language modeling loss for personalized ranking objectives—predicting the next token—which significantly differs from the objective of modeling user preferences in recommendation tasks [20, 21].

We argue that the current objective of LM-based recommenders does not fully utilize preference data and is not optimized for personalized ranking tasks, thereby hindering recommendation performance. Most LM-based recommenders address recommendation tasks by leveraging specialized language prompts [14, 17, 22–24], incorporating collaborative signals as a new modality [18, 19, 25], or extending the vocabulary of LMs with item tokens [16, 26–30]. Typically, these recommenders pair each language prompt, including the user's historical interaction item lists, with a single positive item and then update LM parameters using language modeling loss [14, 19]. Despite being designed for recommendation tasks, these LM-based recommenders do not consider negative items and are not directly optimized for personalized rankings. Such a training paradigm fails to fully leverage user preference data and overlooks the role of negative items in recommendations, thereby impeding the alignment of LMs with user preferences.

Inspired by the success of using human-labeled data to align LMs with human preferences [31–33] and advancements in direct preference optimization (DPO) [34–36], we make progress on aligning LMs with recommendations by fine-tuning them to predict the next item in accordance with the user's preference. This preference alignment stage aims to instill ranking information into the LMs and help recommenders distinguish preferred items from negatives, rather than solely focus on positives.

Towards this end, we incorporate multiple negatives in user preference data and devise an alternative version of DPO loss tailored for recommendation, connected to softmax sampling strategies [20, 37–39], which we call **S-DPO**. Specifically, we first devise supervised fine-tuning to inject domain knowledge and improve LM's ability to follow the instructions before preference alignment phase, following [14, 33]. In the preference alignment stage, instead of constructing solely positive pairs, we initially pair each language prompt with both positive and randomly sampled multiple negatives to build text-based preference data. Building upon these preference data, we extend conventional DPO with the Bradley-Terry preference model [34, 40] on pairwise data to the Plackett-Luce preference model [41, 42], which handles relative rankings among multiple samples. Furthermore, we generalize the traditional Plackett-Luce preference model, which is designed for full relative rankings, to accommodate partial rankings, a more natural fit for recommendation tasks.

Benefiting from the multiple negatives in preference data, S-DPO offers three appealing properties. On the one hand, S-DPO serves as the first specialized personalized ranking loss for LM-based recommenders, effectively utilizing multiple negatives and acknowledging the importance of preference data. Empirically, we demonstrate that it provides more effective ranking gradients and better rewards for preferred items compared with DPO (*cf.* Section 4.2). On the other hand, we theoretically bridge the DPO loss with the pairwise BPR loss [43] over pairwise data and connect S-DPO with the softmax loss over negative sampling (also known as contrastive loss in self-supervised recommendations, which achieves state-of-the-art performance [37, 44, 45]). This connection naturally underscores the ranking performance of S-DPO and highlights the critical role of multiple negatives. Furthermore, gradient analysis demonstrates that S-DPO has an inherent benefit of mining hard negative examples similar to contrastive learning paradigm [38], which not only boosts the performance but also accelerates the training process (*cf.* Section 3.1), assuring its exceptional capabilities in recommendation tasks.

Overall, our contributions can be concluded as follows:

- We are among the first to point out that the widely used language modeling loss in LM-based recommendation is not designed for ranking tasks and fails to fully utilize user preference data, thereby hindering recommendation performance.
- We propose S-DPO, an alternative version of DPO loss extended from the traditional Plackett-Luce preference model, incorporating multiple negatives to instill ranking information into LM and tailoring for LM-based recommenders.
- We theoretically bridge S-DPO with the softmax loss over negative sampling to highlight the critical role of multiple negatives and find its inherent benefit of mining hard negatives, assuring its capabilities.

https://doi.org/10.52202/079017-0863

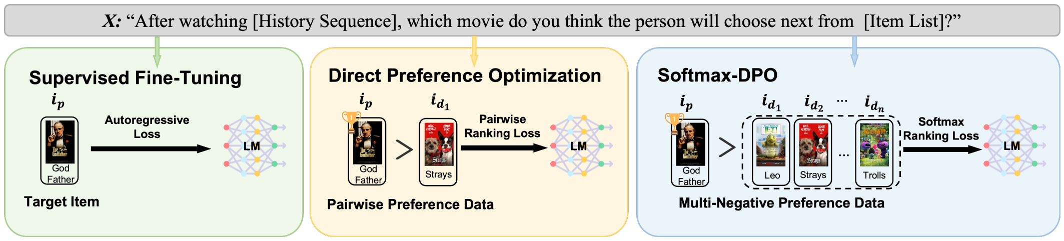


Figure 1: Framework of S-DPO. Different from existing methods which fine-tune LMs with a language modeling loss without tailoring for recommendations, S-DPO proposes to explicitly instill ranking information into LMs. To take one step further, S-DPO incorporates multiple negatives in user preference data and generalizes pairwise DPO loss to softmax ranking loss.

2 Preliminary

In this section, we first formalize sequential recommendation as the task of aligning language models (LMs) with user preferences. Then, we discuss the general framework of current LM-based recommenders that utilizes language modeling loss to fine-tune LMs. Finally, we outline the training process widely used to align LMs with human preferences, including reinforcement learning from human feedback (RLHF) and direct preference optimization (DPO).

Task Formulation. Given the historical interactions $\mathcal{H}_u$ of one user u in chronological order, the goal of LM-based sequential recommender $\mathcal{M}_\theta$, where θ represents trainable parameters, is to select the item i_p preferred by user u from candidate set $C = \{i_j\}_{j=1}^N$, where N is the number of candidates. This task requires that item i_p be preferred over the other candidate items, denoted by $\mathcal{I}_d = C \backslash \{i_p\}$. This requirement explicitly defines a multi-negative preference understanding for LM-based recommenders, which can be formulated as follows:

$$\forall i_d \in \mathcal{I}_d, \quad i_p \succ_u i_d, \tag{1}$$

wherein $\succ_u$ stands for the preference of user u.

Fine-tuning LM-based recommenders. Current LM-based recommenders widely adopt supervised fine-tuning (SFT) [33] on recommendation-specific data to enhance their performance [7, 9]. Generally, this involves two steps: structuring recommendation data as text-based pairs and then fine-tuning LMs based on these pairs. In the first step, for user u, a recommendation task prompt x_u encompasses the user's historical interactions $\mathcal{H}_u$, the candidate item set C, and a description of the sequential recommendation task. This prompt x_u is paired with the title of the preferred item i_p in the candidate set C, denoted as e_p, to form the pair data (x_u, e_p). In the second step, the (x_u, e_p) pairs are utilized to fine-tune the LM-based recommender $\mathcal{M}_\theta$ through language modeling loss. This loss, commonly used in SFT in language modeling tasks, implicitly treats the recommendation task as predicting the next token based on preceding tokens. Formally, the objective of optimizing the LM-based recommender $\mathcal{M}_\theta$ with pair data (x_u, e_p) can be formulated as:

$$\max_\theta \sum_{(x_u,e_p)} \sum_{t=1}^{|e_p|} \log(P_\theta((e_p)_t | x_u, (e_p)_{<t}), \tag{2}$$

where $|e_p|$ is the number of tokens in e_p, $(e_p)_t$ is the t-th token of e_p and $(e_p)_{<t}$ is the tokens preceding $(e_p)_t$.

However, recommendation tasks are essentially user preference alignment tasks, as formalized in the above task formulation, and differ from language modeling tasks that consider only positive responses. Such a gap necessitates further exploration into aligning LM-based recommenders with user preference, an area that has been underexplored.

RLHF pipeline and DPO. Recent studies in natural language processing (NLP) explore the use of human-labeled pairwise data as a reward signal to align LMs with human preferences, such as RLHF [33] and DPO [34]. Specifically, the RLHF [33] pipeline adds two additional phases after the SFT phase: reward model training and reinforcement learning (RL) optimization. After obtaining the SFT model π^{SFT}, RLHF further optimizes it with pairwise preference data.

https://doi.org/10.52202/079017-0863

Inspired by the success of RLHF in NLP, we leverage RLHF to inject recommendation-specific user pairwise preference into LM-based recommenders. Let $\mathcal{E} = \{e_j\}_{j=1}^{N}$ denote the title set of candidate items, where e_j denotes the title of item i_j. Given two items $i_j, i_k \in \mathcal{C}$, the user preference $i_j >_u i_k$ can be seamlessly transformed into a response preference, stipulating that e_j is preferred over e_k when presented with prompt x_u, denoted as $e_j \succ e_k|x_u$. By sampling one dispreferred item i_d from dispreferred candidate set $\mathcal{I}_d$, we can curate a preference dataset $\{(e_p, e_d, x_u)\}$.

After that, RLHF utilizes a preference model for preference distribution modeling, such as Bradley-Terry (BT) model [40]. This preference model assumes there is a latent function $r(x_u, e_j)$ representing the reward of prompt-response pair (x_u, e_j). The bigger reward $r(x_u, e_j)$ means the more user u prefers item i. From this perspective, reward function $r(x_u, e_j)$ serves as a scoring function that quantifies the preference of user u to item i. Besides, the preference model defines a mapping from the reward function $r(x_u, e_j)$ to a preference distribution $p_r(e_j \succ e_k|x_u)$. Based on preference distribution, an optimal reward function is trained by maximizing the likelihood of preference data. The training objective of this phase is as follows:

$$\mathcal{L}_{\text{RM}} = -\mathbb{E}_{(x_u,e_p,e_d)}[\log p_r(e_p \succ e_d|x_u)]. \tag{3}$$

Let $\pi_\theta(e|x_u)$ be the probability that LM-based recommender $\mathcal{M}_\theta$ output title e given prompt x_u. The final reinforcement learning phase aims to maximize the expected reward of policy while not deviate too far from the reference model, formulating the following objective for optimal policy:

$$\max_{\pi_\theta} \mathbb{E}_{x_u\sim\mathcal{D}, e\sim\pi_\theta(e|x_u)}[r(x_u, e)] - \beta\mathbb{D}_{\text{KL}}[\pi_\theta(e|x_u)||\pi_{\text{ref}}(e|x_u)], \tag{4}$$

where $\mathcal{D}$ denotes the distribution of x_u and $\pi_{\text{ref}} = \pi^{\text{SFT}}$.

A recent study, DPO [34], theoretically proves the optimal policy in a closed form to Eq.(4) is

$$\pi^*(e|x_u) = \frac{1}{Z(x_u)}\pi_{\text{ref}}(e|x_u)\exp\left(\frac{1}{\beta}r(x_u, e)\right), \tag{5}$$

which is equivalent to

$$r(x_u, e) = \beta \log \frac{\pi(e|x_u)}{\pi_{\text{ref}}(e|x_u)} + \beta \log Z(x_u), \tag{6}$$

where $Z(x_u) = \sum_e \pi_{\text{ref}}(e|x_u)\exp\left(\frac{1}{\beta}r(x_u, e)\right)$ is the partition function.

By defining $p_r(e_p \succ e_d|x_u)$ as $\sigma(r(x_u, e_p) - r(x_u, e_d))$ in Eq.(3) according to the BT model used in RLHF and substituting term $r(x_u, e)$ in Eq.(3) with Eq.(6), the last two phases of RLHF pipeline can be equivalently transformed into optimizing DPO loss below:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(x_u,e_p,e_d)}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(e_p|x_u)}{\pi_{\text{ref}}(e_p|x_u)} - \beta \log \frac{\pi_\theta(e_d|x_u)}{\pi_{\text{ref}}(e_d|x_u)}\right)\right], \tag{7}$$

wherein $\sigma(x)$ is the sigmoid function.

DPO is able to directly extract the optimal policy from pairwise preference data, making it more practical for preference alignment than RLHF. Nevertheless, DPO and RLHF are usually designed for pairwise preference. The oversight of other negative items impedes the performance of the LM-based recommenders. To bridge the gap, we expand DPO to S-DPO in recommendation tasks, in consideration of multiple negative items.

3 Methodology

3.1 Derivation of S-DPO loss

To align LM-based recommender $\mathcal{M}_\theta$ with multi-negative preference, we first derive the preference distribution and then propose a new loss function called S-DPO (depicted in Figure 1).

Multi-negative Preference Distribution. As mentioned in Section 2, for user u, there is a partial ranking stipulating $i_p \succ_u i_d, \forall\, i_d \in \mathcal{I}_d$ in sequential recommendation tasks. Let $\mathcal{E}_d$ be the titles of dispreferred items $\mathcal{I}_d$. The aforementioned partial ranking is equivalent to $e_p \succ e_d|x_u, \forall e_d \in \mathcal{E}_d$, from which a multi-negative preference dataset $\{x_u, e_p, \mathcal{E}_d\}$ can be curated in an analogous way to RLHF.

For the dataset pairing one preferred item with multiple dispreferred items, we leverage the Plackett-Luce (PL) model [41, 42] to build preference distribution. Given prompt x_u, K titles $e_1, e_2, \cdots, e_K$ and a permutation $\tau : [K] \to [K]$ reflecting the user preference, with $\tau(j)$ denoting the j-th element of permutation τ, the PL model estimates that the ranking $e_{\tau(1)}, e_{\tau(2)}, \cdots, e_{\tau(K)}$ turns out true, as:

$$p(\tau|e_1, e_2, \cdots, e_K, x_u) = \prod_{j=1}^{K} \frac{\exp\left(r(x_u, e_{\tau(j)})\right)}{\Sigma_{l=j}^{K} \exp(r\left(x_u, e_{\tau(l)}\right))}. \tag{8}$$

By enumerating all the permutations starting with p and calculating sum of their probability given by the PL model, the final multi-negative preference distribution p^* can be derived as:

$$p^*(e_p \succ e_d, \forall e_d \in \mathcal{E}_d|x_u) = \frac{\exp(r(x_u, e_p))}{\sum_{j=1}^{K} \exp(r(x_u, e_j))}. \tag{9}$$

For brevity, the complete derivation is delegated to Appendix A.1.

Deriving S-DPO. By substituting reward function $r(x_u, e)$ in Eq.(9) with Eq.(6), the multi-negative preference distribution can be rewritten as:

$$p^*(e_p \succ e_d, \forall\, e_d \in \mathcal{E}_d|x_u) = \frac{1}{1 + \sum_{e_d \in \mathcal{E}_d} \exp\left(\beta \log \frac{\pi(e_d|x_u)}{\pi_{\text{ref}}(e_d|x_u)} - \beta \log \frac{\pi(e_p|x_u)}{\pi_{\text{ref}}(e_p|x_u)}\right)}. \tag{10}$$

Through plugging distribution given by Eq.(10) in the reward learning objective in Eq.(3), our S-DPO loss can be formulated for policy π_θ as:

$$\mathcal{L}_{\text{S}-\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)\sim\mathcal{D}} \left[\log \sigma \left(-\log \sum_{e_d \in \mathcal{E}_d} \exp\left(\beta \log \frac{\pi_\theta(e_d|x_u)}{\pi_{\text{ref}}(e_d|x_u)} - \beta \log \frac{\pi_\theta(e_p|x_u)}{\pi_{\text{ref}}(e_p|x_u)}\right)\right)\right]. \tag{11}$$

Notably, when the number of candidates N is 2, which means there is only one dispreferred item, S-DPO reduces to DPO. The proof is provided in Appendix A.2.

Gradient Analysis. We conduct gradient analysis on S-DPO. The gradient of $\mathcal{L}_{\text{S}-\text{DPO}}$ with respect to parameters θ takes the following formulation:

$$\nabla_\theta \mathcal{L}_{\text{S}-\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$
$$-\beta \mathbb{E}_{(x_u, e_p, \mathcal{E}_d)} \Bigg[\underbrace{\sigma\left(\log \sum_{e_d \in \mathcal{E}_d} \exp(g(e_d, e_p, x_u))\right)}_{\text{higher weight when reward deviates from preference}} \cdot \Bigg[\nabla_\theta \log \pi_\theta(e_p|x_u) - \sum_{e_d \in \mathcal{E}_d} \underbrace{\frac{\nabla_\theta \log \pi_\theta(e_d|x_u)}{\sum_{e'_d \in \mathcal{E}_d} \exp(g(e'_d, e_d, x_u))}}_{\text{higher weight when reward is larger}}\Bigg]\Bigg],$$

wherein $g(e_j, e_k, x_u) = r_\theta(x_u, e_j) - r_\theta(x_u, e_k)$ and similar to DPO, $r_\theta(x_u, e) = \beta \log \frac{\pi_\theta(e|x_u)}{\pi_{\text{ref}}(e|x_u)}$ is the implicit reward function defined by π_θ. See Appendix A.3 for a complete derivation.

Recap the DPO gradient below:

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\beta \mathbb{E}_{(x_u, e_p, e_d)} \Bigg[\underbrace{\sigma(g(e_d, e_p, x_u))}_{\text{higher weight when reward is wrong}} \cdot [\nabla_\theta \log \pi_\theta(e_p|x_u) - \nabla_\theta \log \pi_\theta(e_d|x_u)] \Bigg].$$

Similar to DPO, the gradient of S-DPO loss increases the likelihood of the preferred item and decreases the likelihood of all the dispreferred items. Each example is also weighed by how much the

https://doi.org/10.52202/079017-0863

implicit reward $r(x_u, e)$ deviates from the preference data. However, compared with DPO, S-DPO harnesses information of multiple dispreferred items in this weight.

Moreover, S-DPO treats gradients of different negative (dispreferred) items differently by assigning the gradient of each negative item with an extra weight $\frac{1}{\sum_{e'_d \in \mathcal{E}_d} \exp(g(e'_d, e_d, x_u))} = \frac{\exp(r_\theta(x_u, e_d))}{\sum_{e'_d \in \mathcal{E}_d} \exp(r_\theta(x_u, e'_d))}$. This term reflects the relative reward of each negative item compared with other negative items. Similar to [38], we can categorize negative items into two categories: (1) Hard negative items, whose reward $r_\theta(x_u, e_d) = \beta \frac{\pi_\theta(e_d|x_u)}{\pi_{\text{ref}}(e_d|x_u)}$ is relatively high, making it more probable to be chosen by LM-based recommenders; (2) Easy negative items, whose reward $r_\theta(x_u, e_d)$ is relatively low, making it less likely to be output. For hard negative items, the extra weight term $\frac{\exp(r_\theta(x_u, e_d))}{\sum_{e'_d \in \mathcal{E}_d} \exp(r_\theta(x_u, e'_d))}$ tends to be larger, leading to more decline for likelihood. This mechanism makes LM-based recommenders more discriminative and endows S-DPO with more effectiveness and stability than DPO.

3.2 Properties of S-DPO

In this section, we will discuss the structural correlation between DPO and BPR [43], together with S-DPO and softmax loss [39], which demonstrates the advantage of S-DPO over DPO and language modeling loss.

For user u, preferred item i_p and one dispreferred $i_d \in \mathcal{I}_d$, BPR loss takes the form:

$$\mathcal{L}_{\text{BPR}} = -\mathbb{E}_{(u, i_p, i_d)} \left[\log \sigma \left(f(u, i_p) - f(u, i_d) \right) \right], \tag{12}$$

wherein $f(u, i)$ represents preference score of user u for item i.

Similarly, given dispreferred item set $\mathcal{I}_d$, the softmax loss takes the form:

$$\mathcal{L}_{\text{softmax}} = -\mathbb{E}_{(u, i_p, \mathcal{I}_d)} \left[\log \sigma \left(-\log \sum_{i_d \in \mathcal{I}_d} \exp \left(f(u, i_d) - f(u, i_p) \right) \right) \right]. \tag{13}$$

Review the DPO loss in Eq.(7) and S-DPO loss in Eq.(11). Notably, term $\beta \log \frac{\pi_\theta(e|x_u)}{\pi_{\text{ref}}(e|x_u)}$ is the implicit reward function, denoted by $r_\theta(x_u, e)$ in Section 3.1. According to Section 2, $r_\theta(e, x_u)$ reflects the preference of user u to item i corresponding to title e. When the reference model has no knowledge about recommendation, *i.e.,* when $\pi_{\text{ref}}(e|x_u)$ is approximately a uniform distribution, term $r_\theta(x_u, e) = \beta \log \frac{\pi_\theta(e|x_u)}{\pi_{\text{ref}}(e|x_u)}$ exactly reveals absolute preference. Hence, $r_\theta(x_u, e)$ possesses a similar function to $f(u, i)$.

From this perspective, DPO and S-DPO can be seen as special patterns of BPR and softmax loss, respectively. Given the effectiveness of BPR and InfoNCE loss in recommendation, we argue that sampled-based loss which explicitly compares preferred and dispreferred items such as DPO and S-DPO is more suitable for training LM-based recommenders than only utilizing language modeling loss. Moreover, as softmax loss works better than BPR loss in multi-negative scenarios [20], it can be inferred that S-DPO will be more tailored for multi-negative user preference alignment than DPO.

4 Experiments

In this section, we aim to answer the following research questions:

- **RQ1:** How does S-DPO compare with traditional and LM-based sequential recommendation models on performance?
- **RQ2:** How does the LM-based recommender benefit from the multiple negatives in S-DPO?
- **RQ3:** What are the impacts of the essential parameters (β) on S-DPO?

Baselines. We thoroughly compare S-DPO with three categories of recommenders in sequential recommendations: traditional recommenders (GRU4Rec [46], Caser [47], SASRec [48]), LM-enhanced recommenders (MoRec [49]) and LM-based recommenders (LLaMA2 [32], Chat-REC

Table 1: The performance comparison on three real-world datasets. "Rel.Ipv" denotes the relative improvement of S-DPO compared with baselines.

		Goodreads			LastFM			MovieLens		
		HR@1	ValidRatio	Rel.Ipv	HR@1	ValidRatio	Rel.Ipv	HR@1	ValidRatio	Rel.Ipv
Traditional	GRU4Rec	0.3867	1.0000	*70.91%*	0.2616	1.0000	*153.36%*	0.3750	1.0000	*40.35%*
	Caser	0.4174	1.0000	*58.34%*	0.2233	1.0000	*196.82%*	0.3861	1.0000	*36.31%*
	SASRec	0.3581	1.0000	*84.56%*	0.2233	1.0000	*196.82%*	0.3444	1.0000	*52.82%*
LM-based	LLaMA2	0.0233	0.3845	*2736.48%*	0.0246	0.3443	*2594.31%*	0.0421	0.4421	*1150.12%*
	ChatRec	0.3306	1.0000	*99.91%*	0.3770	1.0000	*75.81%*	0.2000	0.9895	*163.15%*
	MoRec	0.2877	1.0000	*129.72%*	0.1652	1.0000	*301.21%*	0.2822	1.0000	*86.50%*
	TALLRec	0.4983	0.9573	*32.63%*	0.4180	0.9836	*58.56%*	0.3895	0.9263	*35.12%*
	LLaRA	<u>0.5292</u>	0.9950	*24.89%*	<u>0.4508</u>	0.9918	*47.03%*	<u>0.4737</u>	0.9684	*11.10%*
Ours	S-DPO	**0.6609**	0.9900	-	**0.6628**	0.9992	-	**0.5263**	0.9895	-

[15], TALLRec [14], LLaRA [19]). See detailed introduction and comparison of baselines in Appendix B.

Datasets. We conduct extensive experiments on three real-world benchmark datasets which differ in size and domain (Movielens [50], Goodreads[3], and LastFM [51]). Following standard settings of [15, 19], we employ a commonly used metric Hit Ratio@1 (HR@1) for performance evaluation and an additional metric Valid Ratio to evaluate the LM-based methods' ability to generate appropriate responses. See detailed introductions of datasets and evaluation metrics in Appendix B.

Implementation. We implement all LM-based recommenders on 4 NVIDIA A100 GPUs. For all LM-based recommenders, we conduct a supervised fine-tuning stage for a maximum of 5 epochs. For S-DPO and its variants, we conduct a preference alignment stage for a further 3 epochs. Different from existing methods, we only optimize loss on item titles and find it effective in recommendation tasks. Refer to Appendix B for more implementation details.

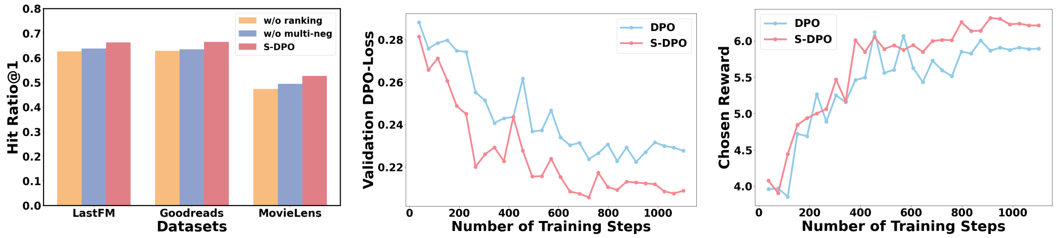
4.1 Overall Performance Comparison (RQ1)

Table 1 presents a comparative analysis of the performance of our proposed S-DPO and baselines. Bold and underlined indicate the best and the second-best performance, respectively. We observe that:

- **LM-based recommenders have driven impressive performance breakthroughs compared with traditional recommenders.** Our results reveal that traditional recommenders outperform untuned LM-based recommenders (LLaMA, ChatRec) but fall short compared to LM-based recommenders fine-tuned on historical interactions (TALLRec and LLaRA). It is noted that untuned LM-based recommenders are limited by inadequate instruction-following capabilities (indicated by a low valid ratio) or a lack of domain-specific knowledge (indicated by a suboptimal performance), which highlights the necessity of the supervised fine-tuning stage to further ground the inherent ability of language models down to sequential recommendation tasks. Moreover, MoRec also exhibits suboptimal performance compared to its traditional variant because it leaves the reasoning ability of LM untouched. The superior performance of recent LM-based recommenders indicates the significant roles of knowledge and reasoning ability in language models for recommendation tasks in semantically informative datasets, which highlights the potential of LM-based recommenders.
- **Tailoring language models for recommendation task further boosts the performance of LM-based recommenders.** For LM-based recommenders, the substantial performance gap between fine-tuned and untuned approaches emphasizes the importance of tailoring models for recommendations. TALLRec adapts LM for recommendation by supervised fine-tuning LM on historical interactions, surpassing traditional recommenders. Additionally, LLaRA consistently outperformed TALLRec across all datasets, suggesting that introducing collaborative signals through appropriate item representations is a viable direction for further adapting LM. However, existing LM-based methods adapt LM from either item representation methods or corpus construction, leaving the adaptation of optimization objectives unexplored. Instead, S-DPO aligns the language model with multi-negative

[3] https://www.goodreads.com

https://doi.org/10.52202/079017-0863



(a) Ablation study. (b) Study of validation loss. (c) Study of preferred item reward.

Figure 2: Study on S-DPO. (2a) Ablation study of S-DPO compared with SFT and DPO on three datasets. (2b) Comparison of the trend of validation loss between DPO and S-DPO on LastFM. (2c) Comparison of the reward of preferred items between DPO and S-DPO on LastFM.

user preference data by extending DPO to include a softmax ranking loss, making it a more appropriate loss function for recommendation tasks.

- **S-DPO consistently outperforms all traditional recommenders and the state-of-the-art LM-based recommenders on all datasets.** S-DPO shows an improvement ranging from 11.10% to 47.03% on Hit Ratio@1 compared to the second-best baseline. Building on a supervised fine-tuning stage, we observe a further improvement to the preference alignment stage, which explicitly instills ranking information into LM and utilizes preference data with multiple negative samples. Such superior performance suggests that explicitly tailoring LM for recommendation using user preference data at the training objective level is more effective than other LM-based recommenders. By leveraging the inherent abilities of the LM and incorporating ranking information from user preference data, S-DPO effectively differentiates between preferred and less preferred items. Notably, the preference alignment stage hardly harms the inherent ability of LM, illustrated by a high valid ratio.

4.2 Study on S-DPO

Ablation Study. To investigate the effect of explicit ranking optimization and multiple negative samples of S-DPO, we compare it with the vanilla supervised fine-tuned model (w/o ranking), and a variant of S-DPO with only a single negative sample (w/o multi-neg), downgrading to pairwise DPO loss. The experimental results are reported in Figure 2a. We can observe that DPO can achieve an overall better performance compared to SFT, which underscores the effectiveness of instilling ranking relationships into existing LM-based recommenders. With a more effective ranking gradient provided by multiple negative samples, S-DPO can further boost performance and achieve the best among all baseline methods and variants.

Study on the number of negative samples (RQ2). Benefiting from the utilization of multiple negative pairs in preference data, our S-DPO offers two empirically appealing properties compared to DPO: 1) S-DPO has more effective gradients facilitating the optimization; 2) S-DPO provides a better boost for rewards of preferred items compared to DPO. Figure 2b provides the comparison of validation loss between S-DPO and DPO, illustrating that the loss of S-DPO decreases faster and more significantly. This observation demonstrates that multiple negative pairs provide larger and more meaningful gradients for model optimization, which is attributed to the inherent benefit of S-DPO to mine negative samples [38] (*cf.* Section 3.1).

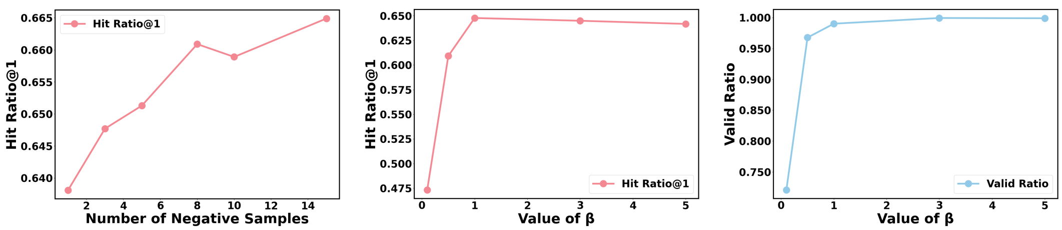
On the other hand, we study the behavior of S-DPO which is illustrated in Figure 2c. We surprisingly find that S-DPO exhibits continually increasing rewards of preferred items that are more significant and stable than DPO, which shows better effectiveness in distinguishing preferred items and a potential of mitigating data likelihood decline issues[52, 53].

To further verify the superiority of the multiple negative samples of S-DPO compared with DPO, we select the number of negative samples from {1, 3, 5, 8, 10, 15} to conduct experiments to explore the potential of the number of negative samples, with the results depicted in Figure 3a. It can be observed that utilizing multiple negative samples allows the model to achieve better performance than with a single one. Furthermore, as the number of negative samples increases, the model's performance exhibits continual improvements. We attribute this success of S-DPO to more effective ranking

https://doi.org/10.52202/079017-0863

Table 2: Effectiveness comparison between DPO with single negative, a variant of DPO with multiple negatives and S-DPO with the same number of negatives (we set K as 3 to get the performance in this table).

Datasets	LastFM		MovieLens		Goodreads		Complexity
Measure	HitRatio@1	ValidRatio	HitRatio@1	ValidRatio	HitRatio@1	ValidRatio	
DPO-1negative	0.6342	0.9972	0.4947	0.9684	0.6381	0.9900	$\Theta(2C_{\mathcal{M}}S_t)$
DPO-Knegative	0.6413	0.9964	0.4947	0.9474	0.6628	0.9900	$\Theta(2KC_{\mathcal{M}}S_t)$
S-DPO-Knegative	**0.6477**	0.9980	**0.5263**	0.9895	**0.6661**	0.9950	$\Theta((K+1)(C_{\mathcal{M}}+1)S_t)$



(a) Study of negative samples. (b) Study of β on Hit Ratio@1. (c) Study of β on Valid Ratio.

Figure 3: Studies on values of β and negative samples numbers of S-DPO on LastFM. (3a) Performance comparisons with varying numbers of negative samples (β = 1). (3b) Performance comparisons with varying values of β setting negative samples number as 3. (3c) Validity comparisons with varying values of β setting negative samples number as 3.

gradients provided by multiple negatives which can be connected to the superior performance of contrastive loss in self-supervised recommendations [38, 39, 54].

To validate the superiority of S-DPO over the DPO variant with multi-negatives, we conduct effectiveness and efficiency comparisons. Table 2 demonstrates that introducing more negative samples benefits both DPO and S-DPO, and S-DPO achieves comparable performance with fewer training steps. We further analyze how S-DPO outperforms DPO in terms of computational efficiency. While the complexity of DPO for K negative samples is $\Theta(2KC_{\mathcal{M}}S_t)$, where $C_{\mathcal{M}}+1$ denotes the base LLM's computational complexity and S_t represents the size of the training data, S-DPO's complexity for the same number of negatives is reduced to $\Theta((K+1)(C_{\mathcal{M}}+1)S_t)$. This efficiency gain can be expressed by scaling the complexity of DPO by the factor $\frac{1}{2}+\frac{1}{2K}+\frac{1}{2C_{\mathcal{M}}}+\frac{1}{2KC_{\mathcal{M}}}$, highlighting that S-DPO offers significant advantages, especially when working with a larger number of negative samples.

Study on values of β (RQ3). In S-DPO, β is a hyperparameter controlling the deviation of LM from the base reference policy [34]. Typically, a smaller value of β implies that the language model is more heavily influenced by the preference signals and vice versa. In this section, we select the value of β from {0.1, 0.5, 1, 3, 5} to explore the effect of β on S-DPO. As indicated in Figure 3b through 3c, a higher β can achieve overall better performance in our task, while a lower β may overwhelm the model's learned knowledge from the supervised fine-tuning stage, as evidenced by both low valid ratio and hit ratio. On the other hand, an excessively large β prevents the model from effectively learning ranking relationships, leading to suboptimal performance. In all our main experiments and studies, we set β as 1 to achieve a balance between ranking signals and inherent knowledge of language models.

5 Related Work

5.1 LM for Recommendation

Recent advancements in recommendation systems have increasingly incorporated Language Models (LMs) due to their extensive knowledge and robust reasoning abilities. This integration occurs primarily in two forms: LM-enhanced recommenders and LM-based recommenders. LM-enhanced recommenders utilize LM embedding as semantic representations to provide contrastive signals [55–58] or utilize LM as advanced feature extractors improving the representation of user and

 https://doi.org/10.52202/079017-0863

item features [59–61]. However, these systems still rely on traditional recommenders for the final recommendation task, which leaves the reasoning ability of LM largely untouched.

On the other hand, LM-based recommenders directly employ LMs for making recommendations. Early works leverage LMs' in-context learning capabilities for zero-shot or few-shot recommendations, demonstrating significant potential [10–12, 15]. However, untuned LM-based recommenders are limited by inadequate instruction-following capabilities and a lack of domain-specific knowledge. To bridge this gap, recent efforts in this category include supervised fine-tuning of LMs on the historical interactions to enhance their performance in recommendation tasks [14, 17, 23, 24, 27]. More recently, researchers have discovered that exploring item representation methods in the finetuning phase may further boost LM's ability for recommendation [30]. This branch of works includes integrating collaborative signals [18, 19, 25, 62–64], adjusting numeric representations [22, 65, 66] or introducing additional item tokens [16, 26, 28, 29].

However, existing finetuned methods follow the training objective of language generation without any specific adjustments for personalized ranking. Different from them, S-DPO proposes to explicitly optimize item ranking information on preference data.

5.2 Preference Alignment of Language Models

Reinforcement Learning from Human Feedback (RLHF) [31–33] is a prevalent method of LMs to learn from human preferences. The RLHF pipeline comprises reward model training and reinforcement learning (RL) optimization, the latter of which suffers instability and inefficiency. Direct Preference Optimization (DPO) [34] bypasses the brittle RL phase via a particular reward model parameterization and is thus simpler to implement while still keeping the performance of RLHF.

DPO proves to be effective in many scopes, like NLP [34, 67] and multimodal LMs [35, 68–70]. Besides, several variants have been proposed for further improvement of DPO. ΨPO [71] is a generalization of DPO loss and its representative IPO can better overcome the problem of overfitting. ODPO [36] treats preference pairs differently by stipulating that the likelihood gap of two responses should be greater than a corresponding offset value. KTO [72] utilizes prospect theory for preference alignment tasks. Other variants including GPO [73], f-DPO [74], RSO [75] also enhance or expand DPO in various aspects. Despite these contributions, the possibilities for leveraging and further adapting DPO for recommendation are still largely unexplored and few studies discuss extending DPO to handle multi-negative scenarios.

6 Limitation

Despite effectiveness, there are several limitations not addressed in this paper. On the one hand, the number of negative samples is capped at 15 in our experiments. The potential of multiple negative samples hasn't been fully explored due to the limited time and computation resources. On the other hand, increasing the number of negative examples inevitably results in higher training costs, a phenomenon that becomes more pronounced as the number of negative examples grows in the context of language models.

7 Conclusion

In this work, we devised a principled Softmax-DPO (S-DPO) loss specially tailored for LM-based recommenders, utilizing multiple negatives in preference data to explicitly instill ranking information into LM. Empirically, S-DPO surpasses all baseline models including traditional and LM-based methods on three datasets in sequential recommendation tasks while successfully providing better rewards for preferred items compared to DPO. Grounded by theoretical proof, we bridge S-DPO with the softmax loss in self-supervised recommendations, underscoring the ranking performance of S-DPO and highlighting the critical roles of multiple negatives. Also, we theoretically find that S-DPO has an inherent benefit to mine hard negatives which provide larger and more effective gradients to model optimization, assuring its exceptional capabilities in recommendation tasks. We believe that S-DPO, as a generalization of DPO, provides valuable insights for future LM-based recommenders and has the potential to benefit research fields other than recommender systems[4].

[4]Thc extension to broader impact will be detailed in Appendix D

https://doi.org/10.52202/079017-0863

Acknowledgments and Disclosure of Funding

This research is supported by the National Science and Technology Major Project (2023ZD0121102), the NExT Research Center, National Natural Science Foundation of China (92270114). This research is also supported by the advanced computing resources provided by the Supercomputing Center of the USTC.

References

[1] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *ACM Trans. Inf. Syst.*, 2020.

[2] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. Sequential recommender systems: Challenges, progress and prospects. In *IJCAI*, 2019.

[3] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models. *CoRR*, abs/2303.18223, 2023.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

[5] Jingnan Zheng, Han Wang, An Zhang, Tai D. Nguyen, Jun Sun, and Tat-Seng Chua. Ali-agent: Assessing llms' alignment with human values via agent-based evaluation. In *NeurIPS*, 2024.

[6] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L. Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karén Simonyan. Flamingo: a visual language model for few-shot learning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *NeurIPS*, 2022.

[7] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. Recommender systems in the era of large language models (llms). *CoRR*, abs/2307.02046, 2023.

[8] Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. How can recommender systems benefit from large language models: A survey. *CoRR*, abs/2306.05817, 2023.

[9] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. A survey on large language models for recommendation. *CoRR*, abs/2305.19860, 2023.

[10] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. Is chatgpt a good recommender? A preliminary study. *CoRR*, abs/2304.10149, 2023.

[11] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. Uncovering chatgpt's capabilities in recommender systems. In *RecSys*, 2023.

[12] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian J. McAuley, and Wayne Xin Zhao. Large language models are zero-shot rankers for recommender systems. In *ECIR*, 2024.

 https://doi.org/10.52202/079017-0863

[13] An Zhang, Leheng Sheng, Yuxin Chen, Hao Li, Yang Deng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. *CoRR*, abs/2310.10108, 2023.

[14] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *RecSys*, 2023.

[15] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *CoRR*, abs/2303.14524, 2023.

[16] Yaochen Zhu, Liang Wu, Qi Guo, Liangjie Hong, and Jundong Li. Collaborative large language model for recommender systems. *CoRR*, abs/2311.01343, 2023.

[17] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. Recommendation as instruction following: A large language model empowered recommendation approach. *CoRR*, abs/2305.07001, 2023.

[18] Zhengyi Yang, Jiancan Wu, Yanchen Luo, Jizhi Zhang, Yancheng Yuan, An Zhang, Xiang Wang, and Xiangnan He. Large language model can interpret latent space of sequential recommender. *CoRR*, abs/2310.20487, 2023.

[19] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, and Xiang Wang. Llara: Aligning large language models with sequential recommenders. *CoRR*, abs/2312.02445, 2023.

[20] Steffen Rendle. Item recommendation from implicit feedback. In *Recommender Systems Handbook*, pages 143–171. Springer US, 2022.

[21] Lanling Xu, Junjie Zhang, Bingqian Li, Jinpeng Wang, Mingchen Cai, Wayne Xin Zhao, and Ji-Rong Wen. Prompting large language models for recommender systems: A comprehensive framework and empirical analysis. *CoRR*, abs/2401.04997, 2024.

[22] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (RLP): A unified pretrain, personalized prompt & predict paradigm (P5). In *RecSys*, 2022.

[23] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. M6-rec: Generative pretrained language models are open-ended recommender systems. *CoRR*, abs/2205.08084, 2022.

[24] Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. *CoRR*, abs/2308.11131, 2023.

[25] Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. Collm: Integrating collaborative embeddings into large language models for recommendation. *CoRR*, abs/2310.19488, 2023.

[26] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Mahesh Sathiamoorthy. Recommender systems with generative retrieval. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *NeurIPS*, 2023.

[27] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. ONCE: boosting content-based recommendation with both open- and closed-source large language models. In Luz Angelica Caudillo-Mata, Silvio Lattanzi, Andrés Muñoz Medina, Leman Akoglu, Aristides Gionis, and Sergei Vassilvitskii, editors, *WSDM*, 2024.

[28] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, Yinghai Lu, and Yu Shi. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. *CoRR*, abs/2402.17152, 2024.

[29] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. *CoRR*, abs/2311.09049, 2023.

[30] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. How to index item ids for recommendation foundation models. In Qingyao Ai, Yiqin Liu, Alistair Moffat, Xuanjing Huang, Tetsuya Sakai, and Justin Zobel, editors, *SIGIR-AP*, 2023.

[31] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize with human feedback. In *NeurIPS*, 2020.

[32] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

[33] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

[34] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

[35] Ruohong Zhang, Liangke Gui, Zhiqing Sun, Yihao Feng, Keyang Xu, Yuanhan Zhang, Di Fu, Chunyuan Li, Alexander Hauptmann, Yonatan Bisk, and Yiming Yang. Direct preference optimization of video large multimodal models from language model reward. *CoRR*, abs/2404.01258, 2024.

[36] Afra Amini, Tim Vieira, and Ryan Cotterell. Direct preference optimization with an offset. *CoRR*, abs/2402.10571, 2024.

[37] An Zhang, Leheng Sheng, Zhibo Cai, Xiang Wang, and Tat-Seng Chua. Empowering collaborative filtering with principled adversarial contrastive loss. In *NeurIPS*, 2023.

[38] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. Self-supervised graph learning for recommendation. In *SIGIR*, 2021.

[39] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[40] Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 1952.

[41] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 1975.

[42] R. Duncan Luce. *Individual Choice Behavior: A Theoretical analysis*. 1959.

[43] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. *CoRR*, abs/1205.2618, 2012.

https://doi.org/10.52202/079017-0863

[44] Junliang Yu, Xin Xia, Tong Chen, Lizhen Cui, Nguyen Quoc Viet Hung, and Hongzhi Yin. Xsimgcl: Towards extremely simple graph contrastive learning for recommendation. *CoRR*, abs/2209.02544, 2022.

[45] An Zhang, Wenchang Ma, Xiang Wang, and Tat-Seng Chua. Incorporating bias-aware margins into contrastive loss for collaborative filtering. In *NeurIPS*, 2022.

[46] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

[47] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, 2018.

[48] Wang-Cheng Kang and Julian J. McAuley. Self-attentive sequential recommendation. In *ICDM*, 2018.

[49] Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. Where to go next for recommender systems? ID- vs. modality-based recommender models revisited. In *SIGIR*, 2023.

[50] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 2016.

[51] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik, editors. *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems, HetRec '11, Chicago, Illinois, USA, October 27, 2011*, 2011. ACM.

[52] Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *CoRR*, abs/2402.13228, 2024.

[53] Huayu Chen, Guande He, Hang Su, and Jun Zhu. Noise contrastive alignment of language models with explicit rewards. *CoRR*, abs/2402.05369, 2024.

[54] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. Are graph augmentations necessary?: Simple graph contrastive learning for recommendation. In *SIGIR*, 2022.

[55] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Llmrec: Large language models with graph augmentation for recommendation. In Luz Angelica Caudillo-Mata, Silvio Lattanzi, Andrés Muñoz Medina, Leman Akoglu, Aristides Gionis, and Sergei Vassilvitskii, editors, *WSDM*, 2024.

[56] Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Representation learning with large language models for recommendation. *CoRR*, abs/2310.15950, 2023.

[57] Yupeng Hou, Zhankui He, Julian J. McAuley, and Wayne Xin Zhao. Learning vector-quantized item representation for transferable sequential recommenders. In *WWW*, pages 1162–1171. ACM, 2023.

[58] Leheng Sheng, An Zhang, Yi Zhang, Yuxin Chen, Xiang Wang, and Tat-Seng Chua. Language models encode collaborative signals in recommendation. *arXiv preprint arXiv:2407.05441*, 2024.

[59] Yunjia Xi, Weiwen Liu, Jianghao Lin, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, Rui Zhang, and Yong Yu. Towards open-world recommendation with knowledge augmentation from large language models. *CoRR*, abs/2306.10933, 2023.

[60] Binzong Geng, Zhaoxin Huan, Xiaolu Zhang, Yong He, Liang Zhang, Fajie Yuan, Jun Zhou, and Linjian Mo. Breaking the length barrier: Llm-enhanced CTR prediction in long textual user behaviors. *CoRR*, abs/2403.19347, 2024.

[61] Zheng Chen. PALR: personalization aware llms for recommendation. *CoRR*, abs/2305.07622, 2023.

https://doi.org/10.52202/079017-0863

[62] Xiangyang Li, Bo Chen, Lu Hou, and Ruiming Tang. Ctrl: Connect tabular and language model for ctr prediction. *arXiv preprint arXiv:2306.02841*, 2023.

[63] Xiaoyu Kong, Jiancan Wu, An Zhang, Leheng Sheng, Hui Lin, Xiang Wang, and Xiangnan He. Customizing language models with instance-wise lora for sequential recommendation. In *NeurIPS*, 2024.

[64] Xiaohao Liu, Jie Wu, Zhulin Tao, Yunshan Ma, Yinwei Wei, and Tat-seng Chua. Harnessing large language models for multimodal product bundling. *arXiv preprint arXiv:2407.11712*, 2024.

[65] Shijie Geng, Juntao Tan, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. VIP5: towards multimodal foundation models for recommendation. In *EMNLP (Findings)*, 2023.

[66] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. A multi-facet paradigm to bridge large language model and recommendation. *CoRR*, abs/2310.06491, 2023.

[67] Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *CoRR*, abs/2401.10020, 2024.

[68] Zhiyuan Zhao, Bin Wang, Linke Ouyang, Xiaoyi Dong, Jiaqi Wang, and Conghui He. Beyond hallucinations: Enhancing lvlms through hallucination-aware direct preference optimization. *CoRR*, abs/2311.16839, 2023.

[69] Anisha Gunjal, Jihan Yin, and Erhan Bas. Detecting and preventing hallucinations in large vision language models. In *AAAI*, 2024.

[70] Lei Li, Zhihui Xie, Mukai Li, Shunian Chen, Peiyi Wang, Liang Chen, Yazheng Yang, Benyou Wang, and Lingpeng Kong. Silkie: Preference distillation for large visual language models. *CoRR*, abs/2312.10665, 2023.

[71] Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Rémi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *AISTATS*, 2024.

[72] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. KTO: model alignment as prospect theoretic optimization. *CoRR*, abs/2402.01306, 2024.

[73] Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. Generalized preference optimization: A unified approach to offline alignment. *CoRR*, abs/2402.05749, 2024.

[74] Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse KL: generalizing direct preference optimization with diverse divergence constraints. *CoRR*, abs/2309.16240, 2023.

[75] Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J. Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. *CoRR*, abs/2309.06657, 2023.

[76] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[77] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[78] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *ICML*, 2023.

 https://doi.org/10.52202/079017-0863

[79] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *CoRR*, abs/2310.06825, 2023.

A Mathematical Derivations

A.1 Deriving Preference Distribution

The PL model takes the form:

$$p^*(\tau|e_e, i_2, \cdots, e_K, x_u) = \prod_{j=1}^{K} \frac{\exp\left(r(x_u, e_{\tau(j)})\right)}{\Sigma_{l=j}^{K} \exp(r\left(x_u, e_{\tau(l)}\right))}. \tag{14}$$

.

The ranking in multi-negative preference data is $e_p \succ e_d | x_u, \forall e_d \in \mathcal{E}_d$. Our new preference distribution that estimates the probability of the ranking can be derived:

$$\begin{aligned}
p^*(e_p \succ e_d, \forall\, e_d \in \mathcal{I}_d | x_u, e_p, \mathcal{E}_d) &= \sum_{\tau \in \{\tau' | \tau'(1) = p\}} p^*(\tau | x_u, e_p, \mathcal{E}_d) \\
&= \sum_{\tau \in \{\tau' | \tau'(1) = p\}} \prod_{j=1}^{K} \frac{\exp\left(r(x_u, e_{\tau(j)})\right)}{\sum_{l=j}^{K} \exp\left(r(x_u, e_{\tau(l)})\right)} \\
&= \frac{\exp(r(x_u, e_p))}{\sum_{j=1}^{K} \exp(r(x_u, e_j))} \times \sum_{\tau' \in \mathrm{Per}(\mathrm{ind}(\mathcal{I}_d))} \prod_{j=1}^{K-1} \frac{\exp\left(r(x_u, e_{\tau'(j)})\right)}{\sum_{l=j}^{K-1} \exp(r(x_u, e_{\tau'(l)}))} \\
&= \frac{\exp(r(x_u, e_p))}{\sum_{j=1}^{K} \exp(r(x_u, e_j))} \times \sum_{\tau' \in \mathrm{Per}(\mathrm{ind}(\mathcal{I}_d))} p^*(\tau' | x_u, \mathcal{E}_d) \\
&= \frac{\exp(r(x_u, e_p))}{\sum_{j=1}^{K} \exp\left(r(x_u, e_j)\right)},
\end{aligned}$$

wherein $\mathrm{ind}(\mathcal{E}_d)$ denotes the indices of titles in $\mathcal{E}_d$ and $\mathrm{Per}(\mathrm{ind}(\mathcal{E}_d))$ denotes the set of permutations of index set $\mathrm{ind}(\mathcal{E}_d)$. The third equation is because a permutation of $\{1, 2 \cdots, K\}$ starting with p can be divided into the prefix p and a subsequent permutation of the rest indices, which is exactly $\mathrm{Per}(\mathrm{ind}(\mathcal{E}_d))$.

A.2 Connection Between DPO and S-DPO

When $N = 2$, the following equations hold:

$$\begin{aligned}
&\mathcal{L}_{\mathrm{S-DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) && \text{(Eq.(11))} \\
&= -\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)} \left[\log \sigma \left(-\log \sum_{e_d \in \mathcal{E}_d} \exp \left(\beta \log \frac{\pi_\theta(e_d|x_u)}{\pi_{\mathrm{ref}}(e_d|x_u)} - \beta \log \frac{\pi_\theta(e_p|x_u)}{\pi_{\mathrm{ref}}(e_p|x_u)} \right) \right) \right] \\
&= -\mathbb{E}_{(x_u, e_p, e_d)} \left[\log \sigma \left(-\log \exp \left(\beta \log \frac{\pi_\theta(e_d|x_u)}{\pi_{\mathrm{ref}}(e_d|x_u)} - \beta \log \frac{\pi_\theta(e_p|x_u)}{\pi_{\mathrm{ref}}(e_p|x_u)} \right) \right) \right] && (N = 2) \\
&= -\mathbb{E}_{(x_u, e_p, e_d)} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(e_p|x_u)}{\pi_{\mathrm{ref}}(e_p|x_u)} - \beta \log \frac{\pi_\theta(e_d|x_u)}{\pi_{\mathrm{ref}}(e_d|x_u)} \right) \right] \\
&= \mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}).
\end{aligned}$$

Therefore, DPO is a special case of S-DPO.

A.3 Deriving the Gradient of S-DPO Loss

Let $V(\theta; e_d) = g(e_d, e_p, x_u) = \beta \log \frac{\pi_\theta(e_d|x_u)}{\pi_{\mathrm{ref}}(e_d|x_u)} - \beta \log \frac{\pi_\theta(e_p|x_u)}{\pi_{\mathrm{ref}}(e_p|x_u)}$ and the S-DPO loss takes the following form:

$$\begin{aligned}
&\mathcal{L}_{\mathrm{S-DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) \\
&= -\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)} \left[\log \sigma \left(-\log \sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d)) \right) \right]
\end{aligned} \tag{15}$$

https://doi.org/10.52202/079017-0863

The gradient of $V(\theta; e_d)$ can be formulated as:

$$\nabla_\theta V(\theta; e_d) = \beta(\nabla_\theta \log \pi_\theta(e_d|x_u) - \nabla_\theta \log \pi_\theta(e_p|x_u)) \tag{16}$$

Using properties of the sigmoid function that $\sigma'(x) = \sigma(x)(1-\sigma(x)) = \sigma(x)\sigma(-x)$ and thus $((\log\sigma(x))' = \frac{1}{\sigma(x)} \times \sigma(x)\sigma(-x) = \sigma(-x)$, we have:

$$\begin{aligned}
&\nabla_\theta \mathcal{L}_{\mathrm{S-DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) \\
&= -\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)}\left[\nabla_\theta \log \sigma\left(-\log \sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d))\right)\right] \\
&= \mathbb{E}_{(x_u, e_p, \mathcal{E}_d)}\left[\sigma\left(\log \sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d))\right) \cdot \nabla_\theta \log \sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d))\right] \\
&\hspace{22em} ((\log\sigma(x))' = \sigma(-x)) \\
&= \mathbb{E}_{(x_u, e_p, \mathcal{E}_d)}\left[\sigma\left(\log \sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d))\right) \cdot \frac{\sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d)) \cdot \nabla_\theta V(\theta; e_d)}{\sum_{e'_d \in \mathcal{E}_d} \exp(V(\theta; e'_d))}\right] \\
&= -\beta\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)}\left[\sigma\left(\log \sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d))\right) \cdot \sum_{e_d \in \mathcal{E}_d} \frac{\nabla_\theta \log \pi_\theta(e_p|x_u) - \nabla_\theta \log \pi_\theta(e_d|x_u)}{\sum_{e'_d \in \mathcal{E}_d} \exp(V(\theta; e'_d) - V(\theta; e_d))}\right] \\
&\hspace{28em} (\text{Eq. } (16)) \\
&= -\beta\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)}\left[\sigma\left(\log \sum_{e_d \in \mathcal{E}_d} \exp(g(e_d, e_p, x_u))\right) \cdot \sum_{e_d \in \mathcal{E}_d} \frac{\nabla_\theta \log \pi_\theta(e_p|x_u) - \nabla_\theta \log \pi_\theta(e_d|x_u)}{\sum_{e'_d \in \mathcal{E}_d} \exp(g(e'_d, e_d, x_u))}\right] \\
&\hspace{24em} (\text{Definition of } V(\theta; e_d)) \\
&= -\beta\mathbb{E}_{(x_u, e_p, \mathcal{E}_d)}\left[\sigma\left(\log \sum_{e_d \in \mathcal{E}_d} \exp(g(e_d, e_p, x_u))\right) \cdot \left[\nabla_\theta \log \pi_\theta(e_p|x_u) - \sum_{e_d \in \mathcal{E}_d} \frac{\nabla_\theta \log \pi_\theta(e_d|x_u)}{\sum_{e'_d \in \mathcal{E}_d} \exp(g(e'_d, e_d, x_u))}\right]\right]
\end{aligned}$$

The last equation is because:

$$\sum_{e_d \in \mathcal{E}_d} \frac{1}{\sum_{e'_d \in \mathcal{E}_d} \exp(g(e'_d, e_d, x_u))} = \frac{\sum_{e_d \in \mathcal{E}_d} \exp(V(\theta; e_d))}{\sum_{e'_d \in \mathcal{E}_d} \exp(V(\theta; e'_d))} = 1$$

B Experimental Settings

B.1 Baselines

We compare the performance of S-DPO, against both traditional and LM-based baselines to showcase the effectiveness of our method. Specifically, for traditional methods, we have:

- **GRU4Rec** [46] utilizes the GRU (Gated Recurrent Unit) architecture to model sequences, enabling effective prediction in recommendation tasks.
- **Caser** [47] employs both horizontal and vertical convolutional operations to enhance the capture of high-order interactions within item sequences, improving recommendation accuracy.
- **SASRec** [48] incorporates a multi-head self-attention mechanism in its self-attentive sequential recommendation model, facilitating the modeling of intricate sequential data patterns.

For LM-enhanced method, we have:

- **MoRec** [49] advances traditional recommendation systems by incorporating the modality features of items instead of the id feature. we employ BERT for the text encoder and SASRec for the recommendation architecture.

For LM-based methods, we have:

- **LLaMA2** [32] utilized vanilla LLaMA2-7B to directly generate recommendation results through direct prompting.

https://doi.org/10.52202/079017-0863

Table 3: Statistics of datasets.

Dataset	MovieLens	Goodreads	LastFM
#Sequence	943	6,031	1,220
#Items	1,682	4,500	4,606
#Interactions	100,000	220,100	73,510

- **Chat-REC** [15] is implemented based on the framework discussed in [15], we retain user interaction sequences consisting of item titles as use profiles for a fair comparison. We use GPT4 [76] as its primary large language model.
- **TallRec** [14] first propose to transform interaction sequences into textual prompts and then fine-tunes large language models using domain-specific corpus.
- **LLaRA** [19] combines collaborative signals from traditional recommendation systems into the fine-tuning of large language models for improved recommendation performance.

B.2 Datasets

To evaluate the effectiveness of S-DPO, we conduct experiments on three widely used real-world datasets: Movielens [50], Goodreads[5], and LastFM [51]. The statistics of datasets are illustrated in Table 3. The MovieLens dataset is widely used for movie recommendation tasks and includes user ratings and movie titles, we select the MovieLens100K dataset in our experiment. Similarly, Goodreads is sourced from a social book cataloging website, where users can explore, rate, and review a variety of books. LastFM dataset comprises users' listening history and artists' names from the Last.fm online music platform. Following [19], we maintain their titles as textual descriptions for each dataset. For Goodreads, we remove users and books with less than 20 interactions, which keeps the same as the processing of MovieLens. For all datasets, we organize sequences chronologically before dividing the data into training, validation, and testing sets in an 8:1:1 ratio to prevent any potential information leakage.

B.3 Implementation Details

We implement all approaches with Python 3.9.7, PyTorch 2.2.2, and transformers 4.38.2 on 4 NVIDIA A100s. We select Llama2-7B [32] as the LM backbone for S-DPO. Following [19], we randomly select prompts from several prompt formats during training and evaluation to ensure flexibility and generality. For optimization of all the traditional methods, the Adam optimizer is employed with a learning rate adjusted to 0.001, and a batch size configured at 256. All models undergo L2 regularization, with coefficients experimentally determined from [1e-3, 1e-4, 1e-5, 1e-6, 1e-7]. In all experiments involving large language models, we train each method for a maximum of 5 epochs using a batch size of 128 and select the checkpoint with the lowest loss on the validation set as the final checkpoint. A warm-up strategy is applied to the learning rate, starting at 5% of its maximum value, and gradually adjusting it through a cosine scheduler throughout the training process. For S-DPO and all of its ablation studies, we further conduct preference training for further 3 epochs with a batch size of 128 and a learning rate of 1e-5. Setting the value of β as 1, we search the number of negative samples in [3,5] for the main results. The effects of both factors are further explored in 4.2.

B.4 Evaluation Metrics

Given that LMs primarily produce textual responses rather than comprehensive item rankings, we utilize a re-ranking metric in line with previous research [19] to assess recommendation performance. For each sequence, a candidate set is constructed by randomly selecting 20 non-interacted items and always includes the correct item. We assess all models based on their ability to pinpoint the correct item within this candidate set, employing the HitRatio@1 (HR@1) metric for performance evaluation. Following [19], we also introduce an additional metric called the Valid Ratio to evaluate the LM-based methods' adherence to instructions and their ability to generate appropriate responses. Due to the difficulty LMs face in producing ranked results for candidate items, position-aware metrics like NDCG are deemed unsuitable for this evaluation.

[5] https://www.goodreads.com

 https://doi.org/10.52202/079017-0863

Table 4: The performance comparison among three different backbone language models on LastFM and MovieLens.

		LLaMA1-7B		Mistral-7B		Pythia-2.8B	
		HR@1	ValidRatio	HR@1	ValidRatio	HR@1	ValidRatio
LastFM	Vanilla	0.0465	0.5872	0.0633	0.3648	0.0265	0.3648
	Language Modeling	0.5980	0.9980	**0.7828**	0.9992	0.1611	0.4281
	DPO	0.6084	0.9976	0.7415	0.9964	0.1896	0.4220
	S-DPO (3 negatives)	<u>0.6285</u>	0.9976	0.7679	0.9972	<u>0.1948</u>	0.4689
	S-DPO (8 negatives)	**0.6365**	0.9988	<u>0.7820</u>	0.9972	**0.2200**	0.4685
MovieLens	Vanilla	0.0316	0.5158	0.0842	0.6737	0.0421	0.4421
	Language Modeling	0.3895	0.9684	0.4211	0.9895	0.1053	0.5684
	DPO	0.3789	0.9684	<u>0.4421</u>	0.9684	<u>0.1271</u>	0.8449
	S-DPO (3 negatives)	**0.4526**	0.9474	<u>0.4421</u>	0.9895	<u>0.1271</u>	0.8737
	S-DPO (8 negatives)	**0.4526**	0.9579	**0.4947**	0.9895	**0.1474**	0.8737

C Study on Backbone Language Models

In order to examine whether the superiority of S-DPO loss over traditional language modeling loss can be generalized across different backbone language models, we conducted experiments using models with varying architectures and sizes, including LLAMA1-7b [77], Pythia-2.8b [78], and Mistral-7b [79]. These models were tested on two distinct datasets: LastFM and MovieLens. We evaluated three training approaches: language models fine-tuned with standard language modeling loss, and models further trained using DPO and S-DPO losses. Due to limitations in computational resources and time, we experimented with two variants of S-DPO: S-DPO (3 negatives) and S-DPO (8 negatives). As shown in Table 4, S-DPO consistently outperformed the language modeling loss, enhancing model performance while maintaining or even improving the validity of the generated outputs. Additionally, the performance continued to improve as the number of negative samples increased from 3 to 8.

D Broader Impact

We left further exploration of softmax ranking loss in LM including more negative samples and validation on various settings as future works. We believe that S-DPO, a generalization of DPO loss has the potential to benefit other research areas other than recommender systems. This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

https://doi.org/10.52202/079017-0863

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the claim including both empirical results and theoretical findings. Refer to Section 1 for more details.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We explicitly discuss the limitations of our work in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

https://doi.org/10.52202/079017-0863

Justification: We provide the full set of assumptions, together with complete and correct proofs of our theory in Section 2, Section ?? and Appendix A.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide extensive implementation details in Section B.3 including package version, device, datasets and hyperparameter setting.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide our anonymized versions of data and codes with the link in the abstract.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All the details about the experimental setting and datasets can be found in Section 4.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We validate the p-value of experiment in Table 1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

https://doi.org/10.52202/079017-0863

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the details of computation resource needed to reproduce the experiment and also the number of epochs we conduct in Section B.3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We strictly follow the NeurIPS Code of Ethics to conduct our research. Specifically, we have taken steps to ensure that our data collection methods are ethical, our experiments are conducted with responsibility, and all potential biases are thoroughly addressed.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The work presented in our paper aims to instill ranking information into LMs to improve the performance of LM-based recommenders. We do not foresee any negative societal impacts stemming from the outcomes of our research. We further claim our broader impact in Appendix D.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks. Our language model only serves as a recommender from a given item list, which makes it impossible to be misused or dual-use.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We incorporate three datasets in our experiments including MovieLens [50], Goodreads[6] and LastFM [51], all of which are open-source. Also, the backend language model used in our research is LLAMA2-7B [32], which is also an open-source model. The details of our used assets can be found in Appendix B.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

[6] https://www.goodreads.com

https://doi.org/10.52202/079017-0863

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We won't release any assets in our research except necessary checkpoint and processed data in order to reproduce our experiments.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing or research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-0863