Both Supply and Precision: Sample Debias and Ranking Consistency Joint Learning for Large Scale Pre-Ranking System

Feng Gao[1], Xin Zhou[1], Yinning Shao[2,1*], Yue Wu[1], Jiahua Gao[1], Yujian Ren[1], Fengyang Qi[1], Ruochen Deng[1], Jie Liu[1]

[1]Sina Weibo Inc.
[2] Tongji University, China
{gaofeng15, zhouxin6, wuyue7, yujian7, fengyang7, ruochen, liujie20}@staff.weibo.com,
yinningshao@tongji.edu.cn, jiahua3@staff.sina.com

Abstract

Cascade ranking architecture, composed of matching, pre-ranking, ranking and re-ranking stages, is usually adopted to balance the efficiency and effectiveness in real-world recommendation system (RS). As the middle stage of RS, pre-ranking aims to quickly filter out the low-quality items selected at the matching stage and then forwarding high-quality items to the ranking stage. Existing pre-ranking approaches mainly endure two problems 1) Sample Selection Bias (SSB) problem, which heavily limits the performance improvement of filtering out low-quality items owing to ignoring the data flow between stages; and 2) Ranking Consistency (RC) problem, which may cause the ranked lists of the ranking stage and previous pre-ranking stage to be inconsistent. As a result, the competitive items with high scores at the ranking stage may not be selected because of low scores at the pre-ranking stage. These both two problems may cause sub-optimal performances, but previous works usually only focus on the one of them. In this paper, we propose a novel Sample Debias and Ranking Consistency Joint Learning Framework (SDCL) to jointly alleviate SSB and RC problems. SDCL consists of two main modules including 1) Multi-Task Distillation Module (MTD), which enhances the ability of identifying high-quality items by distilling knowledge across all tasks simultaneously from the more complex ranking model which jointly trained with the pre-ranking model; and 2) Adaptive Negative Sample Learning Module (ANSL), which improves the performance of filtering out low-quality items by adaptively adjusting negative samples learning weights based on the current performance of model. SDCL seamlessly integrates two modules in an end-to-end multi-task learning framework. Evaluations on both real-world large-scale traffic logs and online A/B test demonstrate the efficacy and superiority of SDCL.

Introduction

Weibo homepage recommendation, the largest feed recommendation system in China, is a typical multi-stage cascade ranking architecture, as illustrated in Figure 1. Each stage takes different magnitude of items, ranks these items and then feeds top ones to the next stage. The pre-ranking stage needs to select thousands of items from the matching output set, these selected items subsequently ranked by following ranking and re-ranking stages, and finally best tens of items will be displayed to the user. The items ranked at pre-ranking stage is on ten orders of magnitude greater than the ranking stage. To ensure system efficiency, the pre-ranking stage usually adopts vector-product based model (Huang et al. 2013) or interaction based model (Wang et al. 2020) with lighter network architecture and fewer features. In recent years, most researchers mainly focus on improving model expressive power via feature interaction enhancement, feature selection, network compression, distillation (Li et al. 2022a; Ma et al. 2021; Li et al. 2022b; Xu et al. 2020) based on these two model paradigms. However, little research has been done for Sample Selection Bias (SSB) and Ranking Consistency (RC) problems at pre-ranking stage.

The SSB problem is caused by the inconsistency between the training and inference data space. As shown in Figure 1, matching and pre-ranking serve as the initial stage in the cascade ranking architecture, where their inference data spaces, D_0 and D_1, are significantly larger in magnitude compared to the data space of training, D_4, which is the data exposed to users. This means the data distribution during training is far different from that during inference. Consequently, the online ranking score of the data unseen during training will be lack of reliability. Specifically, some low-quality items may get high ranking score and then forwarded to the ranking stage, which will result in sub-optimal performances. A practical recipe for resolving the SSB problem is negative sampling. (Covington, Adams, and Sargin 2016) ensures the consistency between training and inference phases by randomly sampling negatives from the whole item pool (Item Pool in Figure 1), (Gillick et al. 2019) employs a sampling method in training batches where for each user, positive samples from other users within the same batch are used as negative samples for that user. Moreover, (Huang et al. 2020) proposes to mix up randomly negative sampling and hard negative mining for sample optimization. (Song et al. 2022) randomly sampling negatives from non-exposure of pre-ranking $D_2 \cap \bar{D}_4$ and non-pre-ranking of matching $D_1 \cap \bar{D}_2$. Those works ignore data flow among stages and treat all negative samples equally in the final loss, although proving effective, this strategy is not optimal.

Apart from SSB problem, the pre-ranking stage also suffers from RC problem owing to the differences in ranking quality between the pre-ranking and ranking stages. Com-

*Corresponding author

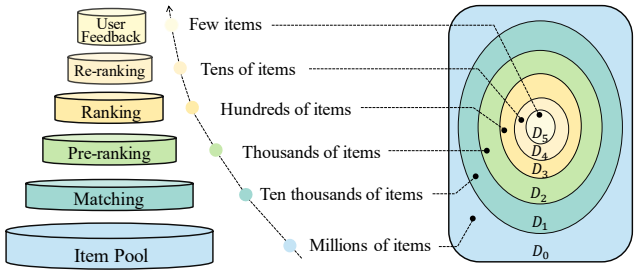

Figure 1: The cascade ranking architecture of weibo online recommendation system.

pared with the model at the pre-ranking stage, the model at the ranking stage with more complicated network architecture and features is naturally of higher ranking quality. However some high-quality items with high ranking scores at the ranking stage may be filter out at the pre-ranking stage in advance. Especially, the RC problem is worse when the Multi-Task Learning (MTL) (Caruana 1997) technology is adopted at both pre-ranking and ranking stages, as the differences in ranking quality of every single task may lead final ranking inconsistency. In recent years, some knowledge distillation based methods (Xu et al. 2020; Gu and Sheng 2022; Cao et al. 2022) that the teacher model (ranking model) outputs are used as proxy labels to supervise the training of the student model (pre-ranking model), are adopted to alleviate RC problem. These methods train the model of the pre-ranking and ranking independently, inherently leading to inconsistencies, and the improvement achieved solely through distillation is limited.

To address the above problems, we propose a novel method named Sample Debias and Ranking Consistency Joint Learning Framework (SDCL). It is built upon the Multi-Task Learning (MTL) framework and consists of two key modules including a Multi-Task Distillation Module (MTD) and a Adaptive Negative Sample Learning Module (ANSL). MTD aims to improve ranking quality of the pre-ranking model by distilling knowledge from ranking model and sharing common input feature embedding space, achieved by training the models of pre-ranking and ranking synchronously. ANSL improves the performance of distinguishing negative samples by devising a novel weighting strategy for different negative samples according to the prediction confidence of current model and the similarity of item content between the positive and negative samples. Our proposed method simultaneously considers the impact of the SSB and RC problems at the pre-ranking stage, and then optimize them concurrently in a unified learning framework to achieve a balance between supply and precision.

Our contributions can be summarized as follows:

- We propose a novel method named Sample Debias and Ranking Consistency Joint Learning Framework for the pre-ranking stage in RS, which alleviates SSB and RC problem concurrently in an end-to-end multi-task learning framework.
- SDCL can effectively mitigate the SSB problem by adaptively adjusting the learning weights assigned to negative samples which sampling from the non-matching of item pool $D_0 \cap \bar{D}_1$, non-pre-ranking of matching $D_1 \cap \bar{D}_2$, non-exposure of pre-ranking $D_2 \cap \bar{D}_4$ based on the data flow and the possible interactions between stages. Moreover, SDCL can improve the ranking consistency between pre-ranking and ranking models by multi-task distillation.
- Besides extensive experiments on real-world datasets and online A/B test , we also provide a detailed analysis on the weights of negative samples learned by our SDCL, which offers additional support for the superiority of our proposed method.

Related Work

Multi-Task Learning

Multi-task Learning can learn commonalities and differences across different tasks based on shared-bottom network and task specific network learning to achieve overall improvement. MOE (Jacobs et al. 1991) combines experts through a gating network which control the weight of each expert for different tasks. MMOE (Ma et al. 2018) extends MOE, which assigns unique gating network for each task. SNR (Ma et al. 2019) provides a more flexible experts combination method through sub-Network Routing. PLE (Tang et al. 2020) proposes a personalized integration through task specific experts and common shared experts. In this paper, we employ the idea of SNR in SDCL to achieve multi-task learning.

Knowledge Distillation

Knowledge distillation is a way to achieve the improvement of the lighter model (student model) by learning knowledge from a complex model (teacher model). Knowledge distillation mainly includes two ways: logit based distillation and feature based distillation. Logit based distillation uses the logit of pre-trained complex network as the proxy label to assist the training of student model (Hinton, Vinyals, and Dean 2015; Furlanello et al. 2018; Tang et al. 2019). Feature based distillation make the intermediate layers parameter of student model fit the intermediate layers parameter of teacher model (Romero et al. 2014; Yim et al. 2017; Chen et al. 2017). In this paper, we employ the idea of logit based distillation, which improve the performance of all tasks in the pre-ranking model through fitting the logit of corresponding tasks in the ranking model.

Preliminaries

Data Space in Cascade Ranking Architecture

As shown in Figure 1, the cascade ranking architecture consisting matching, pre-ranking, ranking and re-ranking stages is usually adopted by industrial recommendation system. The goal of matching stage is to retrieve ten thousands of candidate items D_1 from the item pool with millions of items D_0. The pre-ranking stage is responsible for filtering out low-quality items from retrieved candidate items D_1 and then returning thousands of high-quality items D_2 to the

Stage	Data description	Sample rate
D_4	exposure	100%
$D_2 \cap \bar{D}_4$	non-exposure of the pre-ranking	0.8%
$D_1 \cap \bar{D}_2$	non-pre-ranking of the matching	0.5%
$D_0 \cap \bar{D}_1$	non-matching of the item pool	0.1%

Table 1: Components of training samples.

ranking stage. At the ranking stage, hundreds of items D_3 are selected and then forwarded to the re-ranking stage. Finally tens of win items D_4 will be displayed to the users, which is usually chosen to train ranking models of each stage. The few items D_5 with user feedback information like click, interaction (commenting, sharing, or other behaviors) and watch time are usually regarded as model learning objectives which defined as positive samples (PS), and the rest of D_4 is defined as negative samples (NS).

Obviously, the inference data space D_1 of the pre-ranking contains a large number of items which are not in D_4. The model trained only based on D_4 inevitably suffers from SSB problem due to the inconsistency between the data space of inference and training. Under the consideration of the data flow and interactions between stages in the cascade ranking architecture, the items filtered out by each stage can be regarded as less competitive items which users have no or weak interest in, so we randomly sampling negative samples from $D_0 \cap \overline{D}_1$, $D_1 \cap \overline{D}_2$, and $D_2 \cap \overline{D}_4$ as supply of the training samples to alleviate SSB problem of the pre-ranking model, as shown in Table 1. Three kinds of supplied negative samples can be denoted as Matching NS, Pre-ranking NS, Ranking NS, respectively.

SSB Problem at Pre-ranking Stage

The modeling of RS is to estimate the probabilities of user behaviors such as click, interaction and watch time, denoted as $P(y|\boldsymbol{x})$, where $\boldsymbol{x}$ represent the feature vectors of samples and y represent the corresponding labels. Generally, pre-ranking models are trained on the data space D_4 to approximate $P(y|\boldsymbol{x})$. D_4 only includes exposed samples, i.e., for any sample i, it satisfies $P(\mathcal{O}=1|\boldsymbol{x_i})=1$, where $\mathcal{O}$ denotes whether or not it is exposed. During inference, predictions for all samples in the D_1 are based on the assumption $i \in D_4$, where the exposure based conditional distribution, $P(y|\boldsymbol{x}, \mathcal{O}=1)$, is used instead of $P(y|\boldsymbol{x})$. However, this assumption is violated because the training domain D_4 is only a small part of the inference domain D_1. In fact, the model needs to make predictions over D_1, and $\mathcal{O} = 1$ does not hold for most of the samples in $D_1 \cap \overline{D}_4$. And the difference between D_4 and D_1 can lead to a shift between distribution of training samples and the true distribution, therefore generating biases, i.e., violating the principle of Independent Identically Distribution for training and inference.

RC Problem at Pre-ranking Stage

To satisfy both the diverse demand of users and the commercial intent of platform, each stage in industrial cascade ranking architecture usually needs to consider multiple learning tasks together, such as CTR (Click-Through Rate), ITR (Interaction Rate), TIME (Watch Time), etc. The scores of different tasks are fused to obtain the final score for ranking. Assume that there is no Ranking Consistency problem at each task between pre-ranking and ranking stages, which can be defined as :

$$\text{Rank}(\mathcal{R}^t_{pre}(\boldsymbol{x}_i)) = \text{Rank}(\mathcal{R}^t_{rank}(\widetilde{\boldsymbol{x}}_i)),\ \forall i \in D_{tr},\ \forall t \in T, \tag{1}$$

where $\mathcal{R}^t_{pre}$ and $\mathcal{R}^t_{rank}$ represent the pre-ranking and ranking models for the t-th task, respectively. $\boldsymbol{x}_i$ and $\widetilde{\boldsymbol{x}}_i$ are the feature vectors for sample i input into the pre-ranking and ranking models, which belongs to training dataset D_{tr}. T is the set of all tasks, which in this paper includes CTR, ITR and TIME. Eq.1 indicates that the ranking orders of the pre-ranking and ranking models are the same for all samples on all tasks. However, Ranking Consistency problem still may occur after score fusion of all tasks. We provide a simple example in Table 2 to further explain, and assume that each objective is equally important in the fusion score calculations. CTR, ITR, and TIME in Table 2 are the scores of three tasks, and the final fused ranking score is given by: $y_i = y_i^{ctr} + y_i^{itr} + y_i^{time}$.

Sample	Pre-ranking					Ranking				
ID	CTR	ITR	TIME	Fuse	Order	CTR	ITR	TIME	Fuse	Order
1	0.4	0.01	6.2	6.61	**1**	0.2	0.01	7.7	7.91	**3**
2	0.5	0.02	5.8	6.32	**2**	0.5	0.02	7.5	8.02	**2**
3	0.6	0.03	5.5	6.13	**3**	0.8	0.03	7.3	8.13	**1**

Table 2: Although the pre-ranking and ranking are consistent for each task, ranking inconsistency still occurs after the fusion of multiple tasks.

Methodology

The goal of pre-ranking is to select high-quality items that meets the users interest from the matching candidate set for ranking stage, and optimizing Sample Selection Bias or Ranking Consistency problems alone cannot achieve this goal. In this section, we mainly discuss how our proposed optimization techniques including data construction, distillation, and negative sample learning can achieve this goal. We first introduce the overall framework of SDCL, and then we analyze the role of each part.

Overall Framework for Sample Debias and Ranking Consistency Joint Learning

As shown in Figure 2, we design a Sample Debias and Ranking Consistency Joint Learning Framework to improve the performance of pre-ranking model. SDCL is trained based on positive and negative items which contain multiple objectives. The core network of multi-task learning employs the idea of SNR (Ma et al. 2019). On the one hand, SDCL adaptively learns the negative samples sampled in the entire-chain of the cascade ranking architecture, which meets the personalized needs of different tasks and users for negative samples at different training stages. The SSB prob-

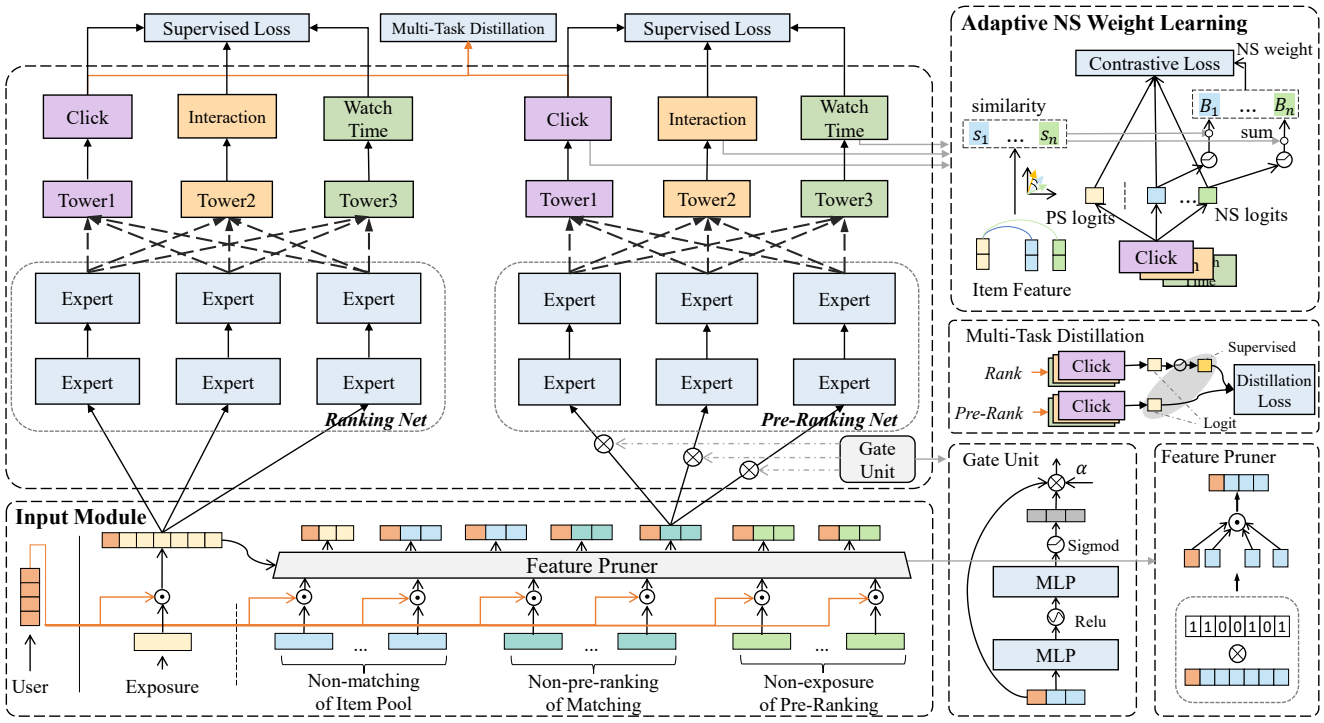

Figure 2: The overall framework of our proposed Sample Debias and Ranking Consistency Joint Learning Framework.

lem can be effectively alleviated through personalized negative samples learning, that strengthening the ability to filter out low-quality items. On the other hand, the precision of pre-ranking learning tasks can be simultaneously improved based on the feature embedding space sharing between the pre-ranking and ranking nets and task specific logits distillation from ranking net. The ranking consistency of multiple tasks in pre-ranking and those in ranking is improved, thereby improving the accuracy of identifying high-quality items in pre-ranking. As a result, SDCL can boost model efficiency of filtering out low-quality items and the accuracy of identifying high-quality items concurrently without incurring extra model inference time consumption as Ranking Net removed during inference.

To achieve the above goals, we design the loss for SDCL as follows:

$$L = \alpha_r L_{rank} + \alpha_d L_{distill} + \alpha_w L_{wcl}, \tag{2}$$

where α_r, α_d, α_w are used to control the proportion of different losses in the total loss. L_{rank} is used to fit the distribution of the prediction of each task to corresponding ground-truth labels, which can be achieved as follow:

$$L_{rank} = -\sum_{t \in T} \sum_{i \in D_{tr}} \beta_r^t (y_i^t log \hat{y}_i^t + (1 - y_i^t) log(1 - \hat{y}_i^t)), \tag{3}$$

where $\hat{y}_i^t$ denotes the ground-truth of sample i for task t and β_r^t denotes the weight of the ranking loss for task t. $L_{distill}$ is used to fit the distribution of each pre-ranking task logit to corresponding ranking task logit. The logit is referred to as the model output before the last activation layer. L_{wcl} is the contrastive loss (Becker and Hinton 1992; Oord, Li, and Vinyals 2018; Cao et al. 2022), which we innovatively introduce adaptive weights to achieve personalized negative samples learning. We will provide a detailed explanation in the following section.

Input Module

The Input Module is responsible for the processing of input samples and features to satisfy the various needs of Pre-ranking Net and Ranking Net. As shown in Figure 2, each sample includes the exposure items and items sampled from each stage of the cascade ranking architecture in the current request, and contains all features of user and item.

For the Ranking Net, it only needs to be trained based on exposure items. $\widetilde{\boldsymbol{x}}_i$ denotes the input feature vector for Ranking Net of sample i, which contains all features of user and item. The Pre-ranking Net, on the other hand, incorporates entire-chain sampled items on top of exposure items for training. The Feature Pruner is responsible to filter out the pre-selected features for Pre-ranking Net based on FSCD (Ma et al. 2021) as the strict limit on inference time consumption for pre-ranking stage. Specifically, the process of feature pruner can be formulated as follows:

$$\boldsymbol{x}_i = \text{Pruner}(\widetilde{\boldsymbol{x}}_i) = \text{Concat}(\ \widetilde{\boldsymbol{x}}_{i,j} | \widetilde{\boldsymbol{x}}_{i,j} \odot m_j \neq \boldsymbol{0}), \tag{4}$$

where $\odot$ represents the Hadamard product, m_j is j-th value in the one-hot encoded list $\boldsymbol{m}$ for pre-ranking feature filtering. $\widetilde{\boldsymbol{x}}_{i,j}$ is the vector corresponding to the j-th feature in $\widetilde{\boldsymbol{x}}_i$, and $\boldsymbol{x}_i$ is the input feature vector for Pre-ranking Net.

Multi-Task Distillation

To improve the ranking consistency of all tasks simultaneously between the Pre-ranking Net and Ranking Net, we employ multi-task distillation which let the prediction of each task in the Pre-ranking Net as close as that of the corresponding prediction of tasks in the Ranking Net. Specifically, the logit of tasks in the Ranking Net is regarded as proxy label, and the Pre-ranking net is trained to approximate the proxy label. The cross-entropy loss is employed as the loss function:

$$L_{distill} = -\sum_{t \in T} \sum_{i \in D_4} \beta_d^t (\mathcal{R}_{rank}^t(\widetilde{\boldsymbol{x}}_i) log(\mathcal{R}_{pre}^t(\boldsymbol{x}_i)) + (1 - \mathcal{R}_{rank}^t(\widetilde{\boldsymbol{x}}_i)) log(1 - \mathcal{R}_{pre}^t(\boldsymbol{x}_i))), \tag{5}$$

where β_d^t represents the distillation loss weight for task t, $\mathcal{R}_{rank}^t(\widetilde{\boldsymbol{x}}_i)$ and $\mathcal{R}_{pre}^t(\boldsymbol{x}_i)$ is the prediction of the Ranking Net and the Pre-ranking Net respectively for task t. The precision of all tasks in the Pre-ranking Net can be improved under the instruction of the Ranking Net which trained with more complex networks and features.

Adaptive Negative Sample Learning

The main goal of this module is to improve the performance of identifying low-quality items based on adaptive negative sample learning which assigns personalized learning weights to negative samples for different users and learning tasks at different training stages.

Negative Sample Gate Unit Inspired by the pepnet (Chang et al. 2023), we add a gate network unit named Negative Sample Gate Unit after the Input Module to learn the personalized input feature information of different negative samples. Specifically, the input of Gate Unit is the input feature vector of Pre-ranking Net filtered by Feature Pruner, and then Negative Sample Gate Unit can learn sample difference in the bottom feature space, which can further personalized input feature information for different users and learning tasks. The Gate Unit contains two layers of neural networks with the following structure:

$$\begin{aligned} \text{Gate}(\boldsymbol{x}_i) &= \alpha \cdot \text{Sigmoid}(\text{Relu}(\boldsymbol{x}_i W_{g1} + \boldsymbol{b}_{g1}) W_{g2} + \boldsymbol{b}_{g2}), \\ \boldsymbol{g}_i &= \boldsymbol{x}_i \odot \text{Gate}(\boldsymbol{x}_i), \end{aligned} \tag{6}$$

where W_g and $\boldsymbol{b}_g$ are learnable weights and bias terms, ReLU and Sigmoid are non-linear activation functions, and $\boldsymbol{g}_i$ is the weighted vector input to the Pre-ranking Net. $\text{Gate}(\boldsymbol{x}_i) \in [0, \alpha]$ represents the weight for $\boldsymbol{x}_i$, and α is a scaling factor that is set as 2.

Negative Sample Learning with Contrastive Loss To achieve personalized negative sample learning, the learning loss is formulated as follows:

$$L_{wcl} = -\sum_{t \in T} \sum_{i \in D_{tr}^+} \beta_w^t log(\frac{\exp(y_i^t/\tau)}{\exp(y_i^t/\tau) + \sum_{z \in D_{tr}^-} w_z^{t,-} \exp(y_z^t/\tau)}), \tag{7}$$

where τ represents the temperature coefficient, which is used to control the extent of learning from negative samples. β_w^t is the contrastive loss weight for task t. D_{tr}^+ denotes the positive samples and D_{tr}^- denotes corresponding negative samples sampled from $D_0 \cap \overline{D}_1$, $D_1 \cap \overline{D}_2$, and $D_2 \cap \overline{D}_4$. $w_z^{t,-}$ is the adaptive weight for negative sample z during each training iteration for task t, defined as:

$$w_z^{t,-} = \delta(\frac{\exp(\text{sim}(\boldsymbol{x}_i^t, \boldsymbol{x}_z^t))}{\sum_{j \in D_{tr}^-} \exp(\text{sim}(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t))} + \text{Sigmod}(y_z^t)), \tag{8}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_z$ are the input feature vectors of sample i and z, with $i \in D_{tr}^+$ and $z \in D_{tr}^-$, and sim represents the cosine similarity. δ is a hyper-parameter. $\text{Sigmoid}(y_z^t)$ represents the probability that sample z belongs to the positive instance.The higher value of $\text{Sigmoid}(y_z^t)$ means the lower discrimination ability of the current model for negative sample z. The purpose of Eq. (8) is to give more emphasis to the hard negative samples, i.e., the negatives that the input feature vectors are most similar to the positive sample, and the negatives with higher discrimination difficulty at current training stage will be paid more attention to.

From Eq. (7), we can observe that L_{wcl} attempts to maximize the distance between positive and negative samples, especially the hard negatives with higher similarity to the positive sample and discrimination difficulty.

Experiments

Experiment Setting

Dataset To our knowledge, there is no available benchmark for pre-ranking models. Similar to (Wang et al. 2020; Ma et al. 2021; Cao et al. 2022), we conduct experiments using data obtained from our own recommendation system. The daily training samples are at the billion level, and the logged-in users reach the ten million level.

Evaluation Metrics Precision@K and Recall@K are used to measure the performance of model (Qin et al. 2022; Song et al. 2022). Apart from that, we introduce an additional evaluation metric called Reverse Rate to further evaluate the Ranking Consistency between pre-ranking and ranking stages. Reverse Rate can be calculated as follows:

$$\text{Reverse Rate} = \frac{2N_{re}}{|D_2|(|D_2| - 1)}, \tag{9}$$

where $|D_2|(|D_2| - 1)/2$ represents the number of all item pairs in D_2. N_{re} represents the number of item pairs which the ranking orders are inverse between pre-ranking and ranking stages.

Ablation Study of SDCL and Main Results

The following models are introduced to validate the performance of SDCL, and the effectiveness of different components in SDCL also can be investigated.

- *SNR* (Ma et al. 2019): It represents our baseline model currently deployed online.
- *SNR w/ MTD*: It employs Multi-Task Distillation Module based on SNR.

Baseline	Reverse	Recall@100	Precision@100
SNR	0.4393	0.29458	0.01222
SNR w/ MTD	0.4254	0.33074	0.01200
SNR w/ MTD & fixed weights	0.3634	0.33963	0.01543
SDCL	**0.3631**	**0.35023**	**0.01719**

Table 3: Experiment results for different model.

User Group	Reverse	Recall@100	Precision@100
Low	0.4537	0.2846	0.0132
Mid-Low	0.4210	0.3333	0.0157
Mid-High	0.4107	0.4049	0.0132
High	0.4048	0.3472	0.0136
Full	0.4027	0.4004	0.0155

Table 4: Comparison of model capability for different users.

- *SNR w/ MTD & fixed weights*: It further introduces negative samples with fixed weights which are equally treated during training.
- *SDCL*: The method we proposed in this paper which further introduces Adaptive Negative Sample Learning Module.

As shown in Table 3, when we add MTD, the results of evaluation metrics show that the Ranking Consistency between pre-ranking and ranking stages has a significant improvement. Furthermore, after adding negative samples with fixed weights, the Recall@100 and Precision@100 increase significantly, the reverse rate decrease, which shows the effectiveness of negative sample learning to enhance the performance of pre-ranking models, although the weights are fixed. *SDCL* achieves optimal results across all evaluation metrics, which demonstrates the effectiveness of *SDCL* in improving the ability of filtering out low-quality items and identifying high-quality items.

A strict online A/B testing is also conducted to verify the effectiveness and efficiency of *SDCL*. The testing lasts for a week, with 1% of traffic is distributed for each model respectively. Compared with the *SNR*, *SNR w/ MTD* achieves 0.28% improvement on Average Watch Time(AWT) which shows the effectiveness of multi-task distillation. Compared with *SNR w/ MTD*, *SNR w/ MTD & fixed weights* shows significant improvement in AWT, Average Number of Requests (ANR), Average Number of Valid Reads (ANVR), which are +2.12%, +1.43%, and +2.53%, respectively. After adding the Adaptive Negative Sample Learning Module, *SDCL* achieve further improvement in performance metrics, including a 0.17% increase in AWT, a 0.37% increase in ANR, and a 0.33% increase in ANVR. Overall, compared with baseline *SNR*, *SDCL* achieves **2.57%** improvement in AWT, **1.38%** improvement in ANVR, **1.41%** improvement in ANR, which validates the effectiveness of *SDCL* and can bring significant benefits to our system.

NS Numbers	AUC		
	CTR	TIME	ITR
8	0.838	0.837	0.959
18	0.852(+1.7%)	0.854(+2.0%)	0.946(-1.4%)
28	0.889(**+6.1%**)	0.892(**+6.6%**)	0.962(**+0.3%**)

Table 5: Comparison of AUC for each task with different negative sample numbers.

NS Kinds	AUC		
	CTR	TIME	ITR
Matching NS	0.913	0.907	0.966
Pre-ranking NS	0.816	0.800	0.949
Ranking NS	0.804	0.801	0.950

Table 6: Comparison of AUC for each task with different kinds of negative samples.

Negative Sample Weight Analysis

In this part, we firstly present the variation in model capability to different users, the demand of negative samples for different tasks and the learning difficulty of negative samples sampling from different stages. We then provide a detailed analysis of the negative sample weights learned during model training, and the result demonstrates that the learning of negative samples in SDCL for different users, learning tasks and negative samples sampling from different stages is personalized.

The Model Capability to Different Users The classification of user groups (such as Low, Mid-Low, Mid-High, High, Full) is according to the login frequency in the past 30 days in our recommendation system. As shown in Table 4, the metrics indicate that the degree of the SSB and RC problems is different among user groups.

The Demand of Negative Samples for Different Tasks We train three groups of models with same network architecture (SNR) and features simultaneously, and then we separately assign different number of negative samples to the models in each group. As shown in Table 5, AUC (Zhou et al. 2018) of CTR and TIME shows a significant improvement with a rise in the number of of negative samples, but AUC of ITR shows minimal change.

The Learning Difficulty of Negative Samples Sampling from Different Stages We train three groups of models with same network architecture (SNR) and features simultaneously, and then we assign same number of negative samples which separately comes from Matching NS, Pre-ranking NS and Ranking NS to the model in each group. As shown in Table 6, AUC of each task with negative samples from Matching NS is higher than that of Pre-ranking NS and Ranking NS, and the model with Ranking NS has the lowest AUC, which indicates higher learning difficulty.

Effectiveness Analysis of Adaptive Negative Sample Learning The negative sample weights produced by

User Group	CTR			ITR			TIME		
	Matching NS	Pre-ranking NS	Ranking NS	Matching NS	Pre-ranking NS	Ranking NS	Matching NS	Pre-ranking NS	Ranking NS
Low	0.0674	0.2526	0.2974	0.0437	0.0488	0.0491	0.1837	0.2854	0.2954
Middle-Low	0.0670	0.2206	0.2715	0.0477	0.0544	0.0547	0.1713	0.2563	0.2675
Middle-High	0.0668	0.2241	0.2772	0.0449	0.0502	0.0504	0.1690	0.2561	0.2674
High	0.0671	0.2261	0.2831	0.0419	0.0456	0.0458	0.1685	0.2587	0.2707
Full	0.0683	0.2233	0.2751	0.0399	0.0426	0.0427	0.1637	0.2511	0.2622

Table 7: Weights of negative samples for different user groups, tasks, and sampling stages.

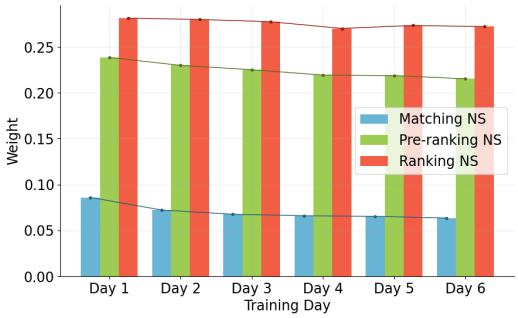

Figure 3: The negative sample learning weight changing over days.

Adaptive Negative Sample Learning Module are reported in Table 7. The following findings can demonstrate that the learning of negative samples for different users, learning tasks and negative samples sampling from different stages is personalized.

- **Finding 1**: The variation in model capabilities across different users can be learned during training. As shown in Table 7, the weights to three kinds of negative samples are different at all tasks for different user group, and a continuous decrease in weights with the increase of user frequency is noted. As the high login frequency means that the more user feedback information can be learned by model which makes the model capability better, so the demand to the negative samples is relatively low.
- **Finding 2**: The variation in the demand for negative samples of different learning task can be identified during training. As shown in Table 7, the learning weights of negative samples to the CTR and TIME are significantly higher than that of ITR. Compared to click and watch behaviors, interaction (such as commenting, sharing, following, or other behaviors) is strongly explicit feedback which are simpler for model to distinguish from negative samples. As a result, the inclusion of additional negative samples has a minimal impact on the model performance of ITR to distinguish between positive and negative instances.
- **Finding 3**: The variation in the learning difficulty of negative samples sampling from different stages can be identified during training. As shown in Table 7, the learning weights of negative samples to Matching NS, Pre-ranking NS and Ranking NS are different, and a gradual increase from Matching NS to Ranking NS can be noted. Compared to the Matching NS and Pre-ranking NS, the samples from Ranking NS are filtered through the stages of matching, pre-ranking and ranking, so the interest level of users is relatively high to these negative samples from Matching NS and Pre-ranking NS, which indicates that its learning difficulty is higher.
- **Finding 4**: The varying demand of negative samples at different training stages can be identified. As shown in Figure 3, the average weights of different negative samples continuously reduce as training time increases, which means that the need for negative samples is gradually decreasing as model performance improves. And compared with Pre-ranking NS and Ranking NS, the learning weights of Matching NS quickly reaches stability, which also indicates the lower learning difficulty of the Matching NS.

In summary, SDCL could perceive the varying demands of negative samples and then dynamically adjust the learning weight during training. This approach satisfies the diverse needs for negative samples at different training stages, for different tasks and users.

Conclusion

In this paper, we propose a novel learning framework for pre-ranking to alleviate the problems of Sample Selection Bias and Ranking Consistency, which mainly consists of Multi-Task Distillation Module (MTD) and Adaptive Negative Sample Learning Module (ANSL). MTD allows for a simultaneous enhancement of all learning tasks by multi-task joint distillation. ANSL satisfies the diverse needs of negative samples for different users, learning tasks and training stages by reweighing the negative samples in an adaptive way during training. Extensive experiments in the real-world industrial datasets demonstrate the effectiveness of SDCL. As for future work, we plan to add the model of matching stage into our joint learning framework, which can alleviate the problems of Sample Selection Bias and Ranking Consistency occurs at the matching stage.

References

Becker, S.; and Hinton, G. E. 1992. Self-organizing neural network that discovers surfaces in random-dot stereograms.

Nature, 355(6356): 161–163.

Cao, Y.; Zhou, X.; Huang, P.; Xiao, Y.; Chen, D.; and Chen, S. 2022. Contrastive Information Transfer for Pre-Ranking Systems. *arXiv preprint arXiv:2207.03073*.

Caruana, R. 1997. Multitask learning. *Machine learning*, 28: 41–75.

Chang, J.; Zhang, C.; Hui, Y.; Leng, D.; Niu, Y.; Song, Y.; and Gai, K. 2023. Pepnet: Parameter and embedding personalized network for infusing with personalized prior information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3795–3804.

Chen, G.; Choi, W.; Yu, X.; Han, T.; and Chandraker, M. 2017. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems*, 30.

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, 191–198.

Furlanello, T.; Lipton, Z.; Tschannen, M.; Itti, L.; and Anandkumar, A. 2018. Born again neural networks. In *International conference on machine learning*, 1607–1616. PMLR.

Gillick, D.; Kulkarni, S.; Lansing, L.; Presta, A.; Baldridge, J.; Ie, E.; and Garcia-Olano, D. 2019. Learning dense representations for entity retrieval. *arXiv preprint arXiv:1909.10506*.

Gu, S.; and Sheng, X. 2022. On Ranking Consistency of Pre-ranking Stage. *arXiv preprint arXiv:2205.01289*.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Huang, J.-T.; Sharma, A.; Sun, S.; Xia, L.; Zhang, D.; Pronin, P.; Padmanabhan, J.; Ottaviano, G.; and Yang, L. 2020. Embedding-based retrieval in facebook search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2553–2561.

Huang, P.-S.; He, X.; Gao, J.; Deng, L.; Acero, A.; and Heck, L. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2333–2338.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; and Hinton, G. E. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1): 79–87.

Li, X.; Chen, B.; Guo, H.; Li, J.; Zhu, C.; Long, X.; Li, S.; Wang, Y.; Guo, W.; Mao, L.; et al. 2022a. Inttower: the next generation of two-tower model for pre-ranking system. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 3292–3301.

Li, X.; Zhou, X.; Xiao, Y.; Huang, P.; Chen, D.; Chen, S.; and Xian, Y. 2022b. AutoFAS: Automatic Feature and Architecture Selection for Pre-Ranking System. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3241–3249.

Ma, J.; Zhao, Z.; Chen, J.; Li, A.; Hong, L.; and Chi, E. H. 2019. Snr: Sub-network routing for flexible parameter sharing in multi-task learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 216–223.

Ma, J.; Zhao, Z.; Yi, X.; Chen, J.; Hong, L.; and Chi, E. H. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 1930–1939.

Ma, X.; Wang, P.; Zhao, H.; Liu, S.; Zhao, C.; Lin, W.; Lee, K.-C.; Xu, J.; and Zheng, B. 2021. Towards a better tradeoff between effectiveness and efficiency in pre-ranking: A learnable feature selection based approach. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2036–2040.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Qin, J.; Zhu, J.; Chen, B.; Liu, Z.; Liu, W.; Tang, R.; Zhang, R.; Yu, Y.; and Zhang, W. 2022. Rankflow: Joint optimization of multi-stage cascade ranking systems as flows. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 814–824.

Romero, A.; Ballas, N.; Kahou, S. E.; Chassang, A.; Gatta, C.; and Bengio, Y. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*.

Song, J.; Huang, R.; Wang, X.; Huang, W.; Yu, Q.; Chen, M.; Yao, Y.; Fan, C.; Peng, C.; Lin, Z.; et al. 2022. Rethinking Large-scale Pre-ranking System: Entire-chain Cross-domain Models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 4495–4499.

Tang, H.; Liu, J.; Zhao, M.; and Gong, X. 2020. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Proceedings of the 14th ACM Conference on Recommender Systems*, 269–278.

Tang, R.; Lu, Y.; Liu, L.; Mou, L.; Vechtomova, O.; and Lin, J. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Wang, Z.; Zhao, L.; Jiang, B.; Zhou, G.; Zhu, X.; and Gai, K. 2020. Cold: Towards the next generation of pre-ranking system. *arXiv preprint arXiv:2007.16122*.

Xu, C.; Li, Q.; Ge, J.; Gao, J.; Yang, X.; Pei, C.; Sun, F.; Wu, J.; Sun, H.; and Ou, W. 2020. Privileged features distillation at taobao recommendations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2590–2598.

Yim, J.; Joo, D.; Bae, J.; and Kim, J. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4133–4141.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 2018. Deep interest network

for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 1059–1068.