POSO: Personalized Cold Start Modules for Large-scale Recommender Systems

Shangfeng Dai*, Haobin Lin*, Zhichen Zhao*, Jianying Lin, Honghuan Wu, Zhe Wang, Sen Yang, Ji Liu
{daishangfeng,linhaobin,zhaozhichen,linjianying,wuhonghuan,wangzhe,senyang,jiliu}@kuaishou.com
Kuaishou Technology
Beijing, China

ABSTRACT

Recommendation for new users, also called user cold start, has been a well-recognized challenge for online recommender systems. Most existing methods view the crux as the lack of initial data. However, in this paper, we argue that there are neglected problems: 1) New users' behaviour follows different distributions from regular users. 2) Although personalized features are used, heavily imbalanced samples prevent the model from balancing new/regular user distributions, as if the personalized features are overwhelmed. We name this problem as the "submergence" of personalization. To tackle this problem, we propose a novel method: **P**ersonalized C**O**ld **S**tart M**O**dules (POSO). From a model architecture perspective, POSO introduces user-group-specialized sub-modules to reinforce the personalization of existing modules. Then, it fuses their outputs by personalized gates, resulting in more comprehensive representations. In this way, POSO equalizes imbalanced features by assigning individual combinations of modules. POSO can be flexibly integrated into many existing modules such as Multi-layer Perceptron, Multi-head Attention and Multi-gated Mixture of Experts, and effectively improves their performance with negligible computational overheads. The proposed POSO shows remarkable advantage in large-scale industrial recommendation scenario. It has been deployed on Kwai and improves new user Watch Time by a large margin (+7.75%). Moreover, POSO can be further generalized to regular users, inactive users and returning users (+2%-3% on Watch Time), as well as item cold start (+3.8% on Watch Time). Its effectiveness has also been verified on public dataset (MovieLens 20M). We believe the proposed POSO can be well generalized to other scenarios.

KEYWORDS

cold start problem, personalized modules

1 INTRODUCTION

Large-scale Recommender Systems (RS) confronts numerous new visitors everyday. One challenging but important problem is how to make accurate recommendation for these unseen users. On one hand, these users hardly have historical description or initial data. On the other hand, they are more sensitive and impatient than regular users. Inaccurate recommendation fails to draw their attention, prompting them not to return the platform. Then we probably lose the potential value of new users.

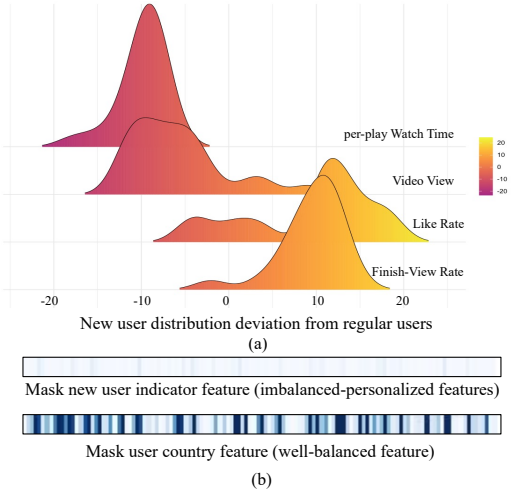

Figure 1: (a) Visualization of new user posterior behavior (based on the relative difference of action count/rate from regular users). It shows that new users follow very different distributions from regular users. (b) Sensitivity of imbalanced and well-balanced features, visualized by two vectors of size 128. Bins in each vector present activation difference when masking imbalanced/balanced features. Deeper color visualizes more significant difference.

The problem is known as "user cold start problem" [7, 15]. Unlike item cold start problem [25] where we can exploit content features [6, 21, 22], user cold start hardly provides alternative description and requires the system to swiftly capture user interests. Meta-learning based methods [10, 12] mitigate the problem by producing well generalized initialization. Besides, other work [14, 26] tries to generate ID embedding by the rest features, thus supplying missing cues.

However, we argue that there exists another neglected problem: the submergence of personalization. The problem describes a phenomenon that even though personalized features are used to balance various user groups (whose distributions are very different), these features are overwhelmed because of heavily imbalanced samples.

As shown in Fig.1 (a), we average regular users' posterior behaviour (Watch Time/Video View Count/Like Rate/Finish-View Rate) as original points, and show distributions of new users. It is

*equal contribution

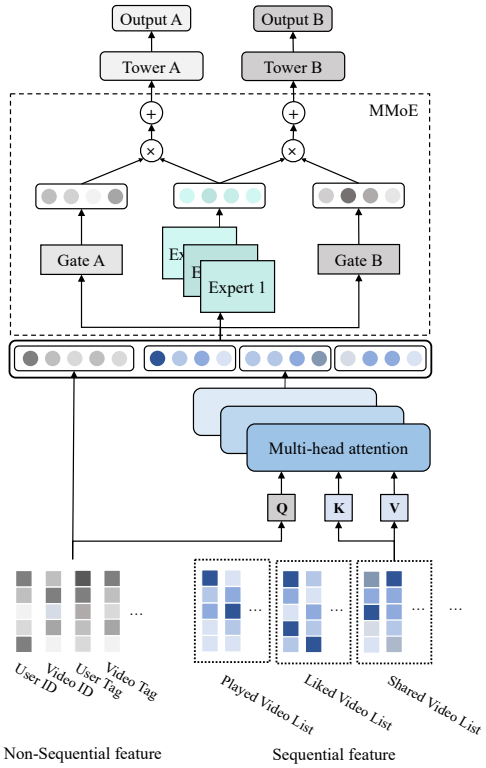

Figure 2: Existing model on the large-scale recommender system of Kwai. There are 3 stages: embedding generation, sequential feature modeling (by MHA) and multi-task optimizing (by MMoE).

shown that new users follow very different distributions. Theoretically, we expect personalized features to distinguish user groups. Do such features help the model balance various distributions in practice? The answer is **NO**. We find that the personalized inputs are overwhelmed, as shown in Fig.1 (b). In both cases we use the same well-trained model, and visualize the activation difference (near the end of the network, averaged across multiple batches) when some features are masked by 0. In the former one we mask new user indicator (0 for regular users and 1 for new users). Surprisingly, the activations almost keep unchanged. The reason is that such features are heavily imbalanced: new users' samples only occupy less than 5% over all samples. In the training procedure the indicator barely changes most of the time, so this feature becomes dispensable. In contrast, in the latter one a well-balanced feature (user country) is masked. Unlike the former case, activations significantly change. The above observation suggests that a plain model architecture is insufficient to maintain personalization.

直接加用户活跃度特征不行

样本不均衡，使得用户活跃度特征训练不足，在整个模型中重要程度不够

In this paper, we propose an effective module to solve the above problems: **P**ersonalized C**O**ld **S**tart M**O**dules (POSO). First, POSO equalizes imbalanced samples by assigning individual combinations of modules, each of which only focuses on its assigned user groups. Then, POSO generates personalized gates varying from raw personalized features. At last gate and module outputs are combined to form comprehensive representations. Its effectiveness is two-fold: 1) Samples are evenly assigned to specialized sub-modules, regardless of majority or minority. 2) Gating network is fully determined by the chosen personalized features (called "Personalization Code"), which avoids their "submergence". POSO reinforces personalization, balancing various distributions and mitigating the cold start problem. POSO does not serve as a standalone method. It can be integrated into many existing modules, such as Multi-layer Perceptron (MLP), Multi-head Attention (MHA) and Multi-gated Mixture of Experts (MMoE). By proper approximation and detailed analysis, we derive their personalized versions, which brings compelling gains consistently but with negligible computational overheads.

对不同用户类型分别建模

One of the advantages of POSO is that it excellently benefits large-scale systems: 1) It follows the standard training procedure, unlike meta-learning based methods which manually split training data into support/query set and probably slow down training speed. 2) The computational overheads are negligible. 3) It can be adopted to other data imbalance problems, which widely exists in users/items/countries/regions.

We conduct extensive experiments on large-scale recommender system of Kwai as well as public dataset. In real-world scenarios, POSO (MLP)/POSO (MHA)/POSO (MMoE) consistently improve the performance, and outperform existing methods. When deployed to our online system, it brings +7.75% Watch Time and +1.52% Retention Rate for new users. Meanwhile, it benefits regular/inactive/returning users (+2-3% Watch Time) as well. Besides user cold start scenario, the proposed architecture improves item cold start (+3.8% Watch Time for new videos) and outperforms existing methods on MovieLens 20M dataset [8].

The contributions of this paper are summarized as follows:

(1) We reveal the submergence of personalization problem. Without customized architectures, personalized features can be overwhelmed. That eventually hurts the performance.
(2) We propose a novel method named POSO, which reinforces personalization under imbalance data, and significantly mitigates the cold start problem.
(3) We present detailed derivation and show that POSO can be integrated into many existing modules with negligible computational overheads. The personalized modules advance the large-scale recommender system by a large margin.

2 RELATED WORK

Related research for user cold start problem can be summarized into two genres: meta-learning and embedding generation. Meta-learning refers to a series of methods aiming at training generalized networks to produce well predictions for brand-new tasks [9, 19]. MAML [5] shows promising results on few-shot learning, but mainly focuses on classification tasks. Following its idea, meta-learning based methods are introduced to the recommender system: MeLU [10] treats recommendation for each user as an individual task. In local update steps, embedding receives no gradients to ensure the stability of the network. Similarly, Du and Wang et al. [3] use meta-learning to transfer knowledge between scenarios, e.g. from traveling task to babysitting task. The work of [17] successfully implements meta-learning strategy on production data. It has

two architectures to adjust weights in Matrix Factorization methods. DropoutNet [20] can be viewed as a similar try on improving generation. It randomly masks user inputs to imitate new users.

Another genre tries to generate meaningful IDs embedding by other features. Meta-E [12] learns to generate user IDs embedding from other embedding. The learning procedure is supervised by cold-start phase and warm-up phase respectively. MAMO [2] proposes multiple memory: profile memory, user memory and task-specific memory. Such memory can be viewed as user feature bases, which are used to decompose cold users into warm features. MWUF [26] believes that there exists scaling difference between regular/new items for item embedding and shifting difference for users. The final embedding is formed by scaling and shifting networks.

3 EXISTING PRODUCTION MODEL

In this section, we briefly describe the structure of the existing production model on the large-scale recommender system of Kwai. As illustrated in Fig.2, the model follows the classic Embedding&MLP paradigm [23]. In addition, some advanced modules (e.g. MHA, MMoE) are introduced to achieve better performance.

The inputs are composed of non-sequential features (e.g. user ID) and sequential features (e.g. user's past watched videos). In the embedding generation stage, all features are firstly mapped into low dimensional vectors, by an embedding look-up table. For each sequential feature, a Multi-Head Attention (MHA) module [18] is applied to fuse the sequence of embedding into a single one, which has been introduced to recommender system by [1]. In existing implementation, keys (K) and values (V) are produced by linear projection of sequential embedding X^{seq}. Namely, $K = W^K X^{\text{seq}}$, and $V = W^V X^{\text{seq}}$ (W^Q, W^K and W^V are trainable matrices)[1]. Differently, query Q receives concatenated non-sequential embedding x^{non} as inputs: $Q = W^Q\, \mathrm{x}^{\text{non}}$. For a single head, $\text{head}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d^h}}\right)V$, where d^h denotes the dimension of projected features. The result of MHA is simply the concatenation of each head output.

In the following stage, all non-sequential embedding and transformed sequential embedding is concatenated as intermediate activation x. The production model needs to predict T targets simultaneously, such as Long-View Rate and Like Rate (see the definition in Sec.5.1). In order to model the task relationships, the Multi-gate Mixture of Experts (MMoE) module [11] is introduced, with N^e MLPs $\{e^{(i)}\}_{i=1}^{N^e}$ as experts. A gating network g^t is trained for task t, which ensembles the expert outputs into $\hat{\mathrm{x}}^t$. Finally, a task-specific MLP h^t takes $\hat{\mathrm{x}}^t$ and gives the prediction y^t. The formulation of the MMoE module is given by:

$$g^t(\mathrm{x}) = \text{softmax}\left(W^t \mathrm{x}\right), \tag{1}$$

$$\hat{\mathrm{x}}^t = \sum_{i=1}^{N^e} \left[g^t(\mathrm{x})\right]_i e^{(i)}(\mathrm{x}), \tag{2}$$

$$y^t = h^t\left(\hat{\mathrm{x}}^t\right), \tag{3}$$

for $t = 1, 2, ..., T$, where W^t is the trainable matrix for gating network.

[1]In this paper, superscript names a specific modules. Signs like (i) indexes modules. While $[\cdot]_i$ index the i-th element of a vector

Method	#Params	#FLOPs
MLP	2.48M	1.50M
POSO (MLP)	2.59M	1.85M
MHA	2.91M	7.82M
POSO (MHA)	3.15M	8.55M
MMoE	13.20M	40.10M
POSO (MMoE)	13.20M	40.10M
Baseline	23.15M	69.46M
Overall (All Combined)	23.51M	70.54M

Table 1: The comparison on params and FLOPs of the production model.

4 PERSONALIZED COLD START MODULES

It is well-known that the system suffers from lack of initial data for new users. However, we argue that one problem has been neglected: the "submergence" of personalization, which means the system fails to balance various distributions because of data imbalance, even though personalized features are provided.

First, we show that new users' behaviour distributes differently with regular users'. In Fig.1 (a) we visualize posterior behaviour of new/regular users. Metrics of regular users are averaged as original points. We show the relative difference of new user metrics. We observe that 1) New users produce lower Video View (VV), indicating that the system hardly captures their interests. 2) New users have higher Finish-View Rate but lower per-play Watch Time. They may enjoy short videos, but have little patience on long videos. 3) New users tend to "like" more frequently, seeming to feel fresh for a wide range of videos. All the observations imply that new users' behaviour follows very different distributions from regular users.

One may believe that existing model implicitly balances various distributions by utilizing personalized features for granted, such as an indicator that distinguishes new/regular users. However, because of data imbalance, such features are overwhelmed. In Fig.1 (b) we utilize a well-trained model, mask personalized features and visualize the activation difference. Surprisingly, masking heavily imbalanced new user indicator almost has no impact on activation. On the contrary, when masking well-balanced user country feature, the activation significantly changes. Since new users only occupy 5% samples. Most of the time, the indicator keeps unchanged. The model easily concentrates on other features to search for solutions, and "forget" the new user indicator, which is critical for cold start problem. We call such problem as the "submergence" of personalization.

In this paper, we reinforce personalized features from a model architecture perspective. We equalize imbalanced-personalized features by assigning individual combinations of models to solve the submergence problem. Ideally, one can construct an exclusive model for a specific user:

$$y^u = f^u\left(\mathrm{x}^u\right), \tag{4}$$

where x, y, f denote inputs, outputs and the model respectively. Subscript "u" refers to a specific user. In such scheme personalization is fully preserved in the corresponding models. Unfortunately, due to the large amount of users, the proposal above is infeasible. One possible solution is establishing several individual models for

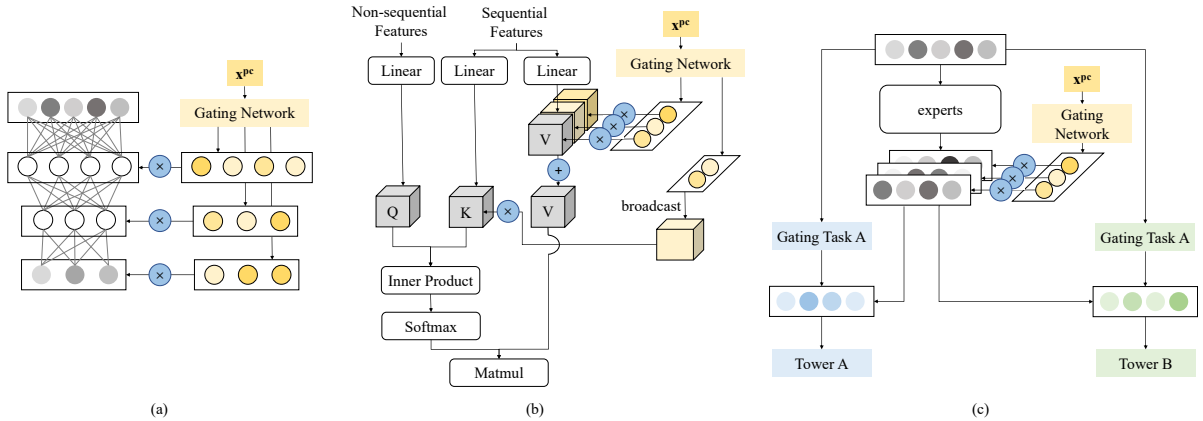

Figure 3: Personalized modules by POSO: (a) POSO (MLP) masks each activation in each layer respectively. (b) In POSO (MHA), Q is not personalized, K is lightly personalized and V is totally personalized. (c) In POSO (MMoE), personalization is firstly adopted, then the output fed the specific tasks. All modules of POSO are colored by yellow.

each kind of user groups (such as new users, returning users and so on). One specific user can be viewed as combination of various user groups (e.g. one can be half inactive user and half regular user). Subsequently, we can decompose the prediction for the specific user as combination of predictions for user groups:

$$y^u = \sum_{i=1}^{N} w_i f^{(i)}(\mathrm{x}), \tag{5}$$

where i denotes model index and we have N models. In practice, it is hard to generate w_i. Instead, we use gating networks to produce w_i from personalized features: $w_i = [g(\mathrm{x}^{\mathrm{pc}})]_i$ where pc refers to Personalization Code, i.e. critical features that identify user group. So far, we still have to prepare N individual models for capturing user group interests, which is computationally intensive. One key point of our method is that we apply decomposition on intermediate modules, and maintain the rest modules unchanged:

$$\hat{\mathrm{x}} = C \sum_{i}^{N} \left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_i f^{(i)}(\mathrm{x}), \tag{6}$$

where f denotes modules from now, $\hat{\mathrm{x}}$ and x are activations of two adjacent layers. Note there is no constraints on the sum of $g(\mathrm{x})$, to avoid overall scale drifting, a rectified factor C is applied.

Eq.6 shows the prototype of the proposed method. As it introduces personalization into intermediate modules, we name it "**P**ersonalized C**O**ld **S**tart M**O**dules (POSO)".

The design of POSO is marked by the following principles:

Personalization. The POSO solves the submergence in two aspects: 1) Features are equalized by assigning multiple modules and gates. Even though regular user data dominates, new user samples wouldn't be neglected because POSO exploits another set of modules and gates to make predictions. 2) Whichever layer to be applied, POSO highlights personalization through raw features instead of second-hand activations, which is hardly achieved by self-learning techniques (such as MoE, see Sec.5.4).

Flexibility. Note that POSO is not a standalone module, but a general approach to personalize existing modules. POSO can be integrated into many existing methods, and equips them with personalization. In the following, we derive personalized versions of MLP, MHA and MMoE. We also believe it has favorable prospect when applied on other unexplored modules.

Non-aftereffect. Sub-modules of POSO share the same input and their outputs are finally fused into single comprehensive result. This ensures the structural independency. No dependency is introduced between upstream and downstream modules.

4.1 POSO of Linear Transformation

We begin with the most basic module: linear transformation, which is formulated as $f(\mathrm{x}) = W\mathrm{x}$, where $\mathrm{x} \in R^{d^{\text{in}}}$ and $\hat{\mathrm{x}} \in R^{d^{\text{out}}}$. Substituting its formulation into Eq.6 gives

$$\hat{\mathrm{x}} = C \sum_{i=1}^{N} \left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_i W^{(i)}\mathrm{x}. \tag{7}$$

Specifically, the p-th entry of $\hat{\mathrm{x}}$ is given by

$$\hat{\mathrm{x}}_p = C \sum_{i=1}^{N} \sum_{q=1}^{d^{\text{in}}} \left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_i W_{p,q}^{(i)} \mathrm{x}_q, \tag{8}$$

where $W_{p.q}^{(i)}$ refers to the element of $W^{(i)}$ at location (p, q). Though Eq.8 introduces N times of complexity, sufficient free parameters allow us to apply simplification in flexible ways. Here we present a simple but effective case. Let $N = d^{\text{out}}$, $W_{p,q}^{(i)} = W_{p,q}\ \forall p, q$ when $i = p$, and $W_{p,q}^{(i)} \equiv 0$ for any $i \neq p$. We have:

$$\hat{\mathrm{x}}_p = C \cdot \left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_p \sum_{q=1}^{d^{\text{in}}} W_{p,q}\mathrm{x}_q, \tag{9}$$

or equivalently,

$$\hat{\mathrm{x}} = C \cdot g\left(\mathrm{x}^{\mathrm{pc}}\right) \odot W\mathrm{x}, \tag{10}$$

where $\odot$ denotes element-wise multiplication. This simplification leads to a computationally efficient operation: just applying element-wise multiplication on the original output by the personalized gates.

4.2 POSO of Multi-Layer Perceptron

Following the similar derivation in Sec.4.1, the personalized version of Fully-Connected layer (FC) with activation function is designed as:

$$\hat{\mathrm{x}} = C \cdot g\left(\mathrm{x}^{\mathrm{pc}}\right) \odot \sigma\left(W\mathrm{x}\right), \tag{11}$$

where σ denotes activation function. It presents a similar form with LHUC [16], whose hidden unit contributions are here replaced by personalized gates.

Naturally, the personalized version of MLPs, called POSO (MLP), is formed by stacking the personalized FCs. Its framework is shown in Fig.3 (a). In Table.1 we detail params and FLOPs of each module, and figure that the proposed modules are computationally efficient.

4.3 POSO of Multi-Head Attention

In this part, we derive POSO version of the Multi-Head Attention (MHA) module. For clarity, let's first consider the formulation of a single head:

$$\hat{\mathrm{x}} = \mathrm{softmax}\left(\frac{QK^{\top}}{\sqrt{d^h}}\right)V. \tag{12}$$

By substituting Eq.12 into Eq.6 as $f^{(i)}$ we have:

$$\hat{\mathrm{x}} = C\sum_{i=1}^{N}\left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_i\left(\mathrm{softmax}\left(\frac{Q^{(i)}\left(K^{(i)}\right)^{\top}}{\sqrt{d^h}}\right)V^{(i)}\right). \tag{13}$$

This naive implementation introduces multi-fold Q, K and V. Despite improving the performance (see Sec.5.5), it is computationally expensive. To reduce overheads, we rethink the role of Q, K and V.

Firstly, Q contains all user features except historical behaviour, so it's already highly personalized. Therefore, we just set $Q^{(i)} = Q, \forall i$. On the contrary, $V^{(i)}$ involves little user information. Considering that V directly determines output, we take no simplification on multi-fold $V^{(i)}$. We notice that using multi-fold K introduces redundant free parameters, since the attention weight produced by K and Q has much lower dimension than K itself. Alternatively, a personalized gate G^k for element-wise multiplication as introduced in Sec.4.1 is sufficient to adjust attention weight, i.e. $K^{(i)} = G^k(\mathrm{x}^{\mathrm{pc}}) \odot K$. [2].

By now, both Q and K become irrelevant to i and thus can be moved out from the summation. Eq.13 is then simplified as:

$$\hat{\mathrm{x}} = C \cdot \mathrm{softmax}\left(\frac{Q \cdot \left(G^k\left(\mathrm{x}^{\mathrm{pc}}\right) \odot K\right)^{\top}}{\sqrt{d^h}}\right)\sum_{i=1}^{N}\left[g(\mathrm{x}^{\mathrm{pc}})\right]_i V^{(i)}. \tag{14}$$

In summary, we respectively personalize the components at 3 levels: no personalization for Q, light-weight personalization for K and full personalization for V. The extent of personalization on these three tensors also agrees with their roles in MHA, as mentioned above. Finally, for multi-head cases, outputs of each head are concatenated to form the representations.

The proposed module is named as "POSO (MHA)", whose framework is shown in Fig.3 (b). In our scenario, compared with the original version of MHA, POSO (MHA) has comparable complexity (see Table.1) but significantly better performance (see Sec.5.5).

[2]In fact $G^k(\mathrm{x}^{\mathrm{pc}})$ is a 2-dimensional tensor while K is a 3-dimensional (including the batch dimension) tensor, so the element-wise operation broadcasts on the last dimension of K.

4.4 POSO of Multi-gated Mixture of Experts

In this part, we present the POSO version of MMoE. Substituting Eq.2 into Eq.6 as $f^{(i)}$ gives:

$$\hat{\mathrm{x}}^t = C\sum_{i=1}^{N}\left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_i\left(\sum_{j}^{N^e}\left[g^t\left(\mathrm{x}\right)\right]_j e^{(j)}\left(\mathrm{x}\right)\right), \tag{15}$$

where i, j, t index personalized gates, experts and tasks. In Eq.15 there are two implicit constraints: each group of experts shares the same personalized gate $g^{(i)}$, each group of g^t is normalized by Softmax. We relax the constraints to simplify the implementation. First, we allow each expert to have its own personalized gate. Then we implement normalization over all task gates. Thereby we have:

$$\hat{\mathrm{x}}^t = C\sum_{i=1}^{N}\sum_{j=1}^{N^e}\left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_{ij}\left[g^t\left(\mathrm{x}\right)\right]_{ij} e^{(ij)}\left(\mathrm{x}\right), \tag{16}$$

where g^t is normalized over all pairs of (i, j). Note that in Eq.16 the indices i and j jointly index experts. Let $\hat{N} = NN^e$, we can re-index the modules and rewrite the above equation as:

$$\hat{\mathrm{x}}^t = C\sum_{i=1}^{\hat{N}}\left[g\left(\mathrm{x}^{\mathrm{pc}}\right)\right]_i\left[g^t\left(\mathrm{x}\right)\right]_i e^{(i)}(\mathrm{x}), \tag{17}$$

$$g^t\left(\mathrm{x}\right) = \mathrm{softmax}\left(W^t\mathrm{x}\right). \tag{18}$$

The overall unit count $\hat{N}$ is actually a hyper-parameter that can be manually adjusted. In our implementation we just set $\hat{N} = N$ to save computation complexity.

In Eq.17 we obtain the finalized version of personalized MMoE, namely, POSO (MMoE). The implementation is extremely light-weighted (also see Table.1): one can keep all the structure of MMoE, and just mask each expert by its personalized gate, as shown in Fig.3 (c).

How the POSO (MMoE) improves experts performance? In MMoE, experts are only task-aware, but have ambiguous knowledge on samples. In POSO (MMoE), experts are personalized activated: if samples belonging to new users produce higher weight in $g\left[\cdot\right]_i$, the corresponding i-th expert obtains higher learning weight, and becomes more sensitive for new users, vice versa. In such way experts become specialized. We can say the experts are not only task-aware, but also field-aware on user groups. In sec.5.6 we visualize gating network outputs of value matrices in MHA. They are similarly specialized.

4.5 POSO for Cold Start

Now, we demonstrate how to mitigate the cold start problem, with the knowledge of POSO.

User Cold Start. New users are defined as users whose first launch happens within T_{du} hours. For user cold start, we exploit a fine-grained feature to reveal how many items have been impressed for this user, i.e. bucketized Accumulated View Count (AVC). This feature is fed into the gating network g as the PC. In each module, we keep the same input for gating network and intensify personalization.

Item Cold Start. The definition of new item (video) is two-fold: 1) It is uploaded within T_{dv} days and 2) Its overall impression count

Methods	New Users				Regular Users			
	Long-View	Finish-View	Like	Follow	Long-View	Finish-View	Like	Follow
MeLU [10]	-0.135	-0.099	+0.124	+0.160	-0.068	-0.038	+0.028	-0.093
Meta-E [12]	-0.024	+0.024	-0.260	-0.338	-0.038	+0.005	-0.283	-0.345
MWUF [26]	+0.023	-0.002	+0.392	+0.123	+0.072	+0.032	+0.212	+0.136
POSO (MLP)	+0.188	+0.121	+0.314	**0.761**	+0.277	**+0.173**	+0.095	+0.775
POSO (MHA)	+0.279	+0.190	+0.089	+0.529	+0.251	+0.150	-0.008	**+0.892**
POSO (MMoE)	+0.223	+0.132	**+0.428**	+0.388	+0.295	+0.162	+0.184	+0.726
POSO (All Combined)	**+0.442**	**+0.248**	+0.344	+0.211	**+0.339**	+0.171	**+0.329**	+0.492

Table 2: Results (percent point) of offline experiments, compared with baseline. Colored tasks are more important.

is less than T_s. Similarly, we exploit video age to distinguish regular/new video. It still produces personalization, but from the view of videos.

In this paper, the gating network is composed of a two-layer MLP, whose outputs are activated by Sigmoid functions.

5 EXPERIMENTS

In this section, we present the performance of POSO on large-scale recommendation scenario of Kwai. We conduct both offline and online experiments. We also validate the generalization of POSO on public dataset. Besides, we demonstrate how to select PC and show visualization inspiration for personalized modules.

5.1 Offline Experiments

Dataset Setup. For offline experiments, samples come from our large-scale recommender system. We build training dataset from 7-consecutive-day record and test dataset from the day next.

Tasks. In video recommender system, users may have two kinds of feedbacks. Explicit feedbacks can be giving a like (in the following we use "Like" to denote) and deciding to follow an author (denoted as "Follow"). Implicit feedbacks mainly refer to whether the user watches the video long enough (Long-View) or completely finishes the video (Finish-View). We follow a multi-task framework to optimize Long-View/Finish-View/Like/Follow Rate simultaneously. Empirically, Long-View and Finish-View are more authoritative metrics that determine online performance, In this paper, a watching event is defined as "Long-View" when its duration exceeds the T_{lv} percent of the video length and thus it is modeled as a CTR-like task.

Metrics. In our experiments, we adopt GAUC [24] to measure the performance of a model, where the AUC is first calculated within samples of each user, and averaged w.r.t sample count.

To validate the effectiveness of the proposed method, we compare various POSO with the existing approaches that are also focusing on cold start problem. MELU [10] utilizes Meta Learning [5] in recommender system, and formulates cold start problem as few-shot learning. Meta-E [12] and MWUF [26] considers generate ID embedding to supply missing cues. These methods cover various aspects from optimization to embedding initialization.

The results are shown in Table.2, where "Rate" is omitted. Because of privacy policy, we only show the absolute difference between baseline and other methods, denoted as percent point (pp). MeLU moderately improves Like and Follow Rate. However, it fails on watching tasks. It seems that MeLU performs well when the task has sparse positive samples. In Meta-E, the interaction tasks drop in turn. MWUF provides improvement on regular users and interaction tasks for new users. Potentially its embedding can be supplementary for IDs embedding (such as user ID embedding, user Tag embedding). However, in industrial scenario, numerous features supply ID embedding, and the improvement is mostly covered.

The comparison of POSO (MLP), POSO (MHA) and POSO (MMoE) is interesting: All of them improve watching tasks. POSO (MHA) performs better on new users while POSO (MMoE) prefers regular users. Meanwhile, POSO (MLP) also focuses more on regular users. It implies that primary heads of MHA redundantly focus on regular users while experts in MMoE are concentrating on new users. That actually causes redundancy. On the contrary, POSO solves the problem by assigning personalized modules. With our POSO, activations, heads and experts are specialized in various users, they become field-aware (see Sec.5.6). Combined POSO provide significant improvement on both user groups and all the tasks, such improvement further empowers online gains by a large margin.

Moreover, we verify whether the proposed method can be adopted to other tasks, such as item cold start (The definition of new video is given in Sec.4.5). To this end we replace the PC by video ages (time from the video uploaded). In Table.3 we show the results compared with the baseline. There are two interesting results: 1) New video POSO performs better results on regular evaluation. 2) It produces larger improvement on regular video samples instead of new video samples. We analyze the results, and figure that the reason is two-fold: on one hand, the system has been struggling for ensuring that new videos can obtain impressions, which essentially sacrifices performance of other videos. On the other hand, existing modules tend new videos excessively. The proposed POSO decouples new/regular videos with their specific modules, thus improves both groups consistently. POSO (MLP) improves Like/Follow Rate, but provides competitive results on watching tasks. It implies value matrices/experts are more capable than activations to balance various user groups.

5.2 Online Experiments

In this section, we conduct online A/B experiments and show the results of large-scale industrial recommender system. We focus on the following metrics (from high importance to low): Watch Time, Retention Rate and Like/Follow Rate. Watch Time reflects how users are attracted by the recommended videos, and Retention

Methods	New Videos				Regular Videos			
	Long View	Finish View	Like	Follow	Long View	Finish View	Like	Follow
POSO (MLP)	-0.042	+0.003	+0.292	+0.388	+0.022	-0.001	+0.019	+0.003
POSO (MHA)	+0.256	**+0.211**	+0.520	+0.428	+0.460	+0.288	+0.046	+0.442
POSO (MMoE)	+0.163	+0.164	+0.518	**+1.456**	-0.059	-0.009	+0.218	+0.053
POSO (All Combined)	**+0.269**	**+0.211**	**+0.727**	+0.430	**+0.466**	**+0.298**	**+0.399**	**+0.503**

Table 3: Offline experiment results on video cold start, compared with baseline.

Metrics		Retention Rate	Watch Time	Like Rate	Follow Rate	Maturing Rate
New User		**+1.52%**	**+7.75%**	+2.09%	**+11.56%**	-
Inactive User	active 2 days in past 7 days	-	**+1.50%**	-	-	-
	active 1 days in past 7 days	-	**+1.98%**	-	-	-
Returning User		-	**+3.01%**	-	-	-
Regular User		+0.15%	**+1.99%**	+0.05%	+1.42%	
New Video		-	+3.81%	-	-	**+0.58%**

Table 4: Online A/B results of various user groups: New User, Inactive user, Returning User, Regular User and New Video. We omit some results because they are undefined or too sparse to give statistical advice. Red results mean they are statistically significant (see text for details). All user groups except new video share the same PC (AVC).

measures whether a user will keep using the application on the following days.

We show the online results on various user groups in Table.4. Inactive users refer to the users who only keep active few days during a week, and returning users' last launch happens at least 7 days ago. Red results mean they are statistically significant, identified by the platform. We view baseline and the experiment as individual distributions. The performance accumulated over a natural day forms a single sample. Then we apply the Student's t-test on the samples. If the experiment samples cannot be generalized by the baseline distribution with more than 95% probability, the experiment is marked as significant.

First, we discuss the performance on new users. Their Watch Time is significantly improved by 7.75%. Such improvement not only verifies the effectiveness of the proposed method, it also brings further positive feedbacks to the whole system: new users play more videos and meanwhile their features and training samples are enriched. The Follow Rate is improved by 11.56%, which means the model makes more accurate predictions on user-author relationships. All of the improvement leads to positive results on Retention Rate. We can confirm that new users are more interested in recommended videos than before, and we are more likely to increase DAU (Daily Active Users). For regular users, our method also obtains consistent improvement on Watch Time (+1.99%), while keeping competitive Retention Rate and interaction metrics.

One interesting observation is that inactive/returning users get significantly improved even that they have been deeply silent. With the silence increases, our model produces larger improvement: from +1.50%, +1.98% to +3.01%. Combining these results with the ones on regular users and new users, we can conclude that when users tend to be inactive, their distributions deviate and personalization becomes critical.

Methods	Favorite	Satisfied
Baseline	76.08	74.57
MeLU [10]	76.16	74.51
Meta-E [12]	76.13	74.53
MWUF [26]	76.20	74.51
POSO	**76.89**	**75.29**

Table 5: Results on the MovieLens 20M dataset.

We also show the results for video cold start. The metrics are "Maturing Rate" and per-play Watch Time. The former one describes the speed that new videos become regular ones as defined in Sec.4.5, and the latter averages Watch Time on videos. POSO significantly improves the primary metric: Maturing Rate.

In summary, POSO is verified to be effective, and generalized for the large-scale industrial recommender system. It reinforces personalization and improves cold start problem significantly.

5.3 Public Dataset

We verify our method on the public dataset: MovieLens 20M [8], which collects user rating scores on movies. It contain more than $130k$ users and more than 20 millions of samples. Since there is no off-the-shelf setup on new user tasks, we split the dataset based on user ID. $100k$ users are divided as training set and the rest is test set. We setup two tasks: 1) Whether the user rates the movie by score=5 (Favorite), and 2) Whether the rating score is not less than 4 (Satisfied). We use two kinds of list features: user's past rating movie ID list and user's past rating tag list, whose length is limited by 30. The PC is AVC. In both task we use GAUC as the metric.

The results are shown in Table.5, existing approaches mostly trade off the performance. Favorite can be improved, however, Satisfied drops. It seems meta-learning methods suit the task with denser positive samples. The proposed POSO reaches the best results on

Personalization Code	Long-View	Finish-View
all features (MoE)	-0.274	-0.501
is-new-user	+0.240	+0.145
user ID embedding	+0.154	+0.007
user ID + video ID embedding	+0.235	+0.117
Accumulated View Count	**+0.442**	**+0.248**

Table 6: Comparison on various Personalization Codes. All results show difference compared with baseline.

Settings	Long-View	Finish-View
$Q^{(i)}, K^{(i)}, V^{(i)}, N = 4$	+0.130	+0.131
$Q^{(i)}$=Q, $K^{(i)}, V^{(i)}, N = 4$	+0.157	**+0.296**
$Q^{(i)}$=Q, $K^{(i)} = G^k \odot K$,$V^{(i)}, N = 4$	**+0.279**	+0.190
$Q^{(i)}$=Q, $K^{(i)} = G^k \odot K$,$V^{(i)} = G^v \odot V$	+0.044	+0.119

Table 7: POSO (MHA) under various settings from full personalization to most light-weighted implementation. For simplification we only show new user metrics.

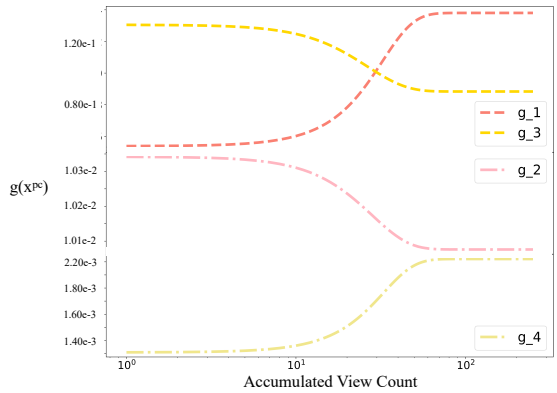

Figure 4: The gating network outputs with bucketized Accumulated View Count increases. #2 and #3 are specialized in new users while the others manage regular users. #1 and #3 dominate in the combination while #2 and #4 work on fine-tuning.

both tasks. Interestingly, on a more difficult task (Favorite), it brings larger improvement (0.81pp vs 0.72pp).

5.4 Evolution of Personalization Code

In POSO, personalization derives from the usage of Personalization Code (i.e. the input feature of gating network). There could be various choices for the specific designing and formulation of PC. In this section, we study the evolution of PC in the user cold start scenario.

The comparison is shown in Table.6 (measured on new user). The first knowledge is that highlighting personalized feature achieves the effectiveness of POSO. When using all features as input, which degrades into MoE [4, 13], we obtain inferior results. That is to say, involving overall features in gating network even worsens submergence. As for personalized features, the most trivial choice is an indicator $\mathbb{1}_{is-new-user}$ whose value equals to 1 when the user is a new visitor and 0 otherwise. Such PC has provided large-margin improvement. Since ID embedding implicitly encodes personalization cues, we exploit it as PC. User ID provides moderate improvement. However, its personalization is heterogeneous for cold start task so the results are inferior to the previous PC. Similarly, Adding video ID embedding further draws back the performance. The best results are produced by the bucketized Accumulated View Count, which counts each impression from the user's first launch, and finely describes user activity and the period of lifecycle. Its improvement even exceeds the difference between with/without PC.

5.5 Up to What Extent of Personalization?

In the derivation of POSO we actually have many choices to simplify or maintain the original formulation. Here we take MHA for example, detail performance of each version and explain why we choose the formulation as stated in Sec.4.

As shown in Table.7, the original version of POSO (Eq.6) can already bring better results. However, it cost huge overheads since no simplification is considered. Interestingly, fixing Q provides larger improvement, which also verifies that Q has been highly personalized. Redundantly personalizing it in contrast draws back the performance. Masking K by an element-wise multiplication trades off Long-View Rate and Finish-View Rate. Considering this setting significantly saves computational overheads and meanwhile provides promising results, we use it as the standard POSO (MHA). Further simplifying $V^{(i)}$ reduces the improvement, which degrades personalized $V^{(i)}$ into personalized activations. As discussed above, $V^{(i)}$ and experts are more capable than activations.

5.6 Specialization of Modules

As a whole, POSO reinforces personalization. Specifically, sub-modules are specialized. In Fig.4 we visualize the gating network outputs for $V^{(i)}$ in our POSO (MHA), which is determined by input (bucketized Accumulated View Count). For new users (lower AVC), gate #3 is decisive. With AVC increases, gate #3 gradually becomes underprivileged and gate #1 dominates. It implies that #3 has been specialized in managing new users and #1 focuses on regular users. #2 and #4 performs similarly, however, they work differently and finely tune the final results.

6 CONCLUSION

Personalization is critical for rank model in recommender system. In this paper, we figure that in existing model architecture, imbalanced-personalized features can be easily overwhelmed. To balance various user groups, we propose the Personalized Cold Start Modules method, which flexibly adopts existing methods and derives their personalized versions with negligible computational overheads. The method is verified to effectively improves the performance for new users, new items and returning/inactive users by a large margin. We also discuss the choice of Personalization Code and how to efficiently personalize a specific module. We believe the practical experience can be well generalized to many other scenarios.

REFERENCES

[1] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior Sequence Transformer for E-Commerce Recommendation in Alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data* (Anchorage, Alaska) *(DLP-KDD '19)*. Association for Computing Machinery, New York, NY, USA, Article 12, 4 pages. https://doi.org/10.1145/3326937.3341261

[2] Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. 2020. MAMO: Memory-Augmented Meta-Optimization for Cold-Start Recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 688–697. https://doi.org/10.1145/3394486.3403113

[3] Zhengxiao Du, Xiaowei Wang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Sequential Scenario-Specific Meta Learner for Online Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2895–2904. https://doi.org/10.1145/3292500.3330726

[4] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2014. Learning Factored Representations in a Deep Mixture of Experts. arXiv:1312.4314 [cs.LG]

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1126–1135. http://proceedings.mlr.press/v70/finn17a.html

[6] Tiezheng Ge, Liqin Zhao, Guorui Zhou, Keyu Chen, Shuying Liu, Huimin Yi, Zelin Hu, Bochao Liu, Peng Sun, Haoyu Liu, Pengtao Yi, Sui Huang, Zhiqiang Zhang, Xiaoqiang Zhu, Yu Zhang, and Kun Gai. 2018. Image Matters: Visually Modeling User Behaviors Using Advanced Model Server. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 2087–2095. https://doi.org/10.1145/3269206.3272007

[7] Sugandha Gupta and Shivani Goel. 2018. Handling User Cold Start Problem in Recommender Systems Using Fuzzy Clustering. In *Information and Communication Technology for Sustainable Development*, Durgesh Kumar Mishra, Malaya Kumar Nayak, and Amit Joshi (Eds.). Springer Singapore, Singapore, 143–151.

[8] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19 pages. https://doi.org/10.1145/2827872

[9] T. M. Hospedales, A. Antoniou, P. Micaelli, and A. J. Storkey. 5555. Meta-Learning in Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 01 (may 5555), 1–1. https://doi.org/10.1109/TPAMI.2021.3079209

[10] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1073–1082.

[11] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 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* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1930–1939. https://doi.org/10.1145/3219819.3220007

[12] Feiyang Pan, Shuokai Li, Xiang Ao, Pingzhong Tang, and Qing He. 2019. Warm Up Cold-Start Advertisements: Improving CTR Predictions via Learning to Learn ID Embeddings. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 695–704. https://doi.org/10.1145/3331184.3331268

[13] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. *CoRR* abs/1701.06538 (2017). arXiv:1701.06538 http://arxiv.org/abs/1701.06538

[14] Shaoyun Shi, Weizhi Ma, Min Zhang, Yongfeng Zhang, Xinxing Yu, Houzhi Shan, Yiqun Liu, and Shaoping Ma. 2020. Beyond User Embedding Matrix: Learning to Hash for Modeling Large-Scale Users in Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 319–328. https://doi.org/10.1145/3397271.3401119

[15] Le Hoang Son. 2016. Dealing with the new user cold-start problem in recommender systems: A comparative review. *Information Systems* 58 (2016), 87–104. https://doi.org/10.1016/j.is.2014.10.001

[16] Pawel Swietojanski and Steve Renals. 2014. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *2014 IEEE Spoken Language Technology Workshop (SLT)*. 171–176. https://doi.org/10.1109/SLT.2014.7078569

[17] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. 2017. A Meta-Learning Perspective on Cold-Start Recommendations for Items. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/51e6d6e679953c6311757004d8cbbba9-Paper.pdf

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[19] Ricardo Vilalta and Youssef Drissi. 2002. A Perspective View and Survey of Meta-Learning. *Artif. Intell. Rev.* 18, 2 (Oct. 2002), 77–95. https://doi.org/10.1023/A:1019956318069

[20] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. 2017. DropoutNet: Addressing Cold Start in Recommender Systems. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/dbd22ba3bd0df8f385bdac3e9f8be207-Paper.pdf

[21] Yu Wang, Jixing Xu, Aohan Wu, Mantian Li, Yang He, Jinghe Hu, and Weipeng Yan. 2018. Telepath: Understanding Users from a Human Vision Perspective in Large-Scale Recommender Systems. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16066

[22] Zhichen Zhao, Lei Li, Bowen Zhang, Meng Wang, Yuning Jiang, Li Xu, Fengkun Wang, and Weiying Ma. 2019. What You Look Matters? Offline Evaluation of Advertising Creatives for Cold-Start Problem. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 2605–2613. https://doi.org/10.1145/3357384.3357813

[23] Guorui Zhou, Kailun Wu, Weijie Bian, Zhao Yang, Xiaoqiang Zhu, and Kun Gai. 2019. Res-embedding for Deep Learning Based Click-Through Rate Prediction Modeling. arXiv:1906.10304 [stat.ML]

[24] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized Cost per Click in Taobao Display Advertising. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 2191–2200. https://doi.org/10.1145/3097983.3098134

[25] Yu Zhu, Jinhao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2018. Addressing the Item Cold-start Problem by Attribute-driven Active Learning. arXiv:1805.09023 [cs.IR]

[26] Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. 2021. Learning to Warm Up Cold Item Embeddings for Cold-Start Recommendation with Meta Scaling and Shifting Networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 1167–1176. https://doi.org/10.1145/3404835.3462843