

PNMTA: A Pretrained Network Modulation and Task Adaptation Approach for User Cold-Start Recommendation

Haoyu Pang
College of Computer Science and Technology, Jilin University
Changchun, China
panghy20@mails.jlu.edu.cn

Fausto Giunchiglia
DISI, University of Trento
Trento, Italy
fausto.giunchiglia@unitn.it

Ximing Li, Renchu Guan*, Xiaoyue Feng*
College of Computer Science and Technology, Jilin University
Changchun, China
{ximingli,guanrenchu,fengxy}@jlu.edu.cn

ABSTRACT

User cold-start recommendation is a serious problem that limits the performance of recommender systems (RSs). Recent studies have focused on treating this issue as a few-shot problem and seeking solutions with model-agnostic meta-learning (MAML). Such methods regard making recommendations for one user as a task and adapt to new users with a few steps of gradient updates on the meta-model. However, none of those methods consider the limitation of user representation learning imposed by the special task setting of MAML-based RSs. And they learn a common meta-model for all users while ignoring the implicit grouping distribution induced by the correlation differences among users. In response to the above problems, we propose a pretrained network modulation and task adaptation approach (PNMTA) for user cold-start recommendation. In the pretraining stage, a pretrained model is obtained with non-meta-learning methods to achieve better user representation and generalization, which can also transfer the learned knowledge to the meta-learning stage for modulation. During the meta-learning stage, an encoder modulator is utilized to realize the memorization and correction of prior parameters for the meta-learning task, and a predictor modulator is introduced to condition the model initialization on the task identity for adaptation steps. In addition, PNMTA can also make use of the existing non-cold-start users for pretraining. Comprehensive experiments on two benchmark datasets demonstrate that our model can achieve significant and consistent improvements against other state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; Data mining; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Recommender systems, Cold-start problem, Meta learning, Transfer learning

* Corresponding Author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
WWW '22, April 25–29, 2022, Virtual Event, Lyon, France.
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3511963

ACM Reference Format:
Haoyu Pang, Fausto Giunchiglia, and Ximing Li, Renchu Guan*, Xiaoyue Feng*. 2022. PNMTA: A Pretrained Network Modulation and Task Adaptation Approach for User Cold-Start Recommendation. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3485447.3511963

1 INTRODUCTION

To alleviate the information overload problem caused by the explosive growth of data and provide convenience for users, recommender systems (RSs) predict items that may be of interest to users by discovering their personalized preferences [8, 9, 18]. To achieve better performance, traditional recommender models always require a large amount of user interaction data for training. However, the interactions of new users are often insufficient and sparse, which leads to the user cold-start problem [17, 37].

Since the advent of RSs, the user cold-start has been one of the most important factors influencing its effectiveness [3]. Traditional solutions to this problem include adding the side information of users or items to the model [6, 7, 63], introducing additional knowledge (such as cross-domain RSs [20, 41, 64]), and developing approaches based on reinforcement learning (such as multi-armed bandit algorithms [13, 56]). Inspired by few-shot learning [57], many promising works have attempted to solve this problem from the perspective of meta-learning [10, 23, 26, 62]. Owing to its excellent performance and flexibility with respect to model selection, model-agnostic meta-learning (MAML) [14, 59] is one of the most popular meta-learning-based frameworks for the user cold-start problem. MAML-based RSs can learn a common initialization for all users and adapt to new users rapidly through gradient updates.

However, most of the existing MAML-based RSs regard each user as a task and train the model task by task (detailed in Section 3.1.1), which makes the input user feature vectors lack randomness and face the risk of overfitting [2, 10, 23, 24, 60]. Figure 1 demonstrates this problem. Specifically, in other machine learning domains [14, 59], meta-learning-based methods do not affect the randomness of data streams. Taking image classification as an example, under the setting of N-way K-shot, each meta-learning task contains N classes of images, each class includes K images, and the selection of these classes and images is performed randomly. However, in MAML-based RSs, for the model regards user u_k as a task τ_k, and the form of the input data stream can be expressed as

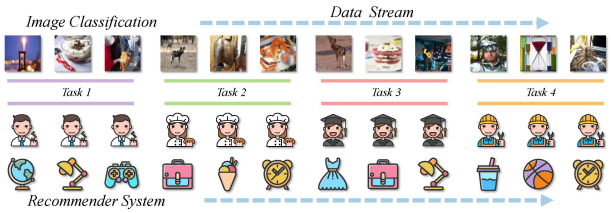


Figure 1: Applications of the MAML-based approach in image classification and RSs. The blue arrows indicate the order in which data flow into the model for training. Notably, in image classification, data flow into the model with a random label order. However, in RSs, because the interactive data are divided into different tasks according to different users, the information and interactions of only one user are available in one task.

$[(u_k; i_1), (u_k; i_2), (u_k; i_3)...(u_k; i_n)]$, where each tuple is an interactive representation vector. It is obvious that the model only receives the feature vector of user u_k in task τ_k. From the perspective of the training process, the data stream received by the recommendation model is grouped according to the given users, which greatly weakens the representation and generalization abilities of the model.

In addition, the existing MAML-based RSs usually generate a common meta-model for all users [10, 23, 26, 55, 60]. However, different from other domains, according to the research of collectivism in social psychology [4, 45], personalized human preferences present obvious implicit grouping distributions (proved in Section 4.7). For users with substantially different preference distributions, it is unwise to generate a common initial meta-model that is close to all of them. These methods not only make the learning process unsmooth and increase the difficulty of convergence but also reduce the quality of the knowledge learned by the resulting model which makes it difficult to adapt the obtained model to new tasks [53, 58]. We show the details of this issue in Figure 2.

To this end, we design a **p**retrained **n**etwork **m**odulation and **t**ask **a**daptation approach called PNMTA for the above two problems. For the first problem, inspired by the schema theory in cognitive psychology [43], we propose a novel transfer learning method for MAML-based RSs with pretraining and encoder modulation. Through pretraining with randomly ordered data and parameter modulating with memory, the representation and generalization abilities of the model can be significantly improved. For the second problem, inspired by two main meta-learning techniques, model-based and model-agnostic meta-learning, we propose a task-adaptive predictor and a predictor modulator that can discover and utilize the implicit tasks distribution. The task encoder is first built and estimate the input user's identity, then the predictor modulator generates modulation parameters based on this identity to condition the initialized predictor for more accurate adaptation. In addition, different from the previous methods that only train models with cold-start users, PNMTA can utilize all users' interactions (including non-cold-start users) during pretraining to enrich the output representations of users and items.

To summarize, the main contributions of our work are as follows:

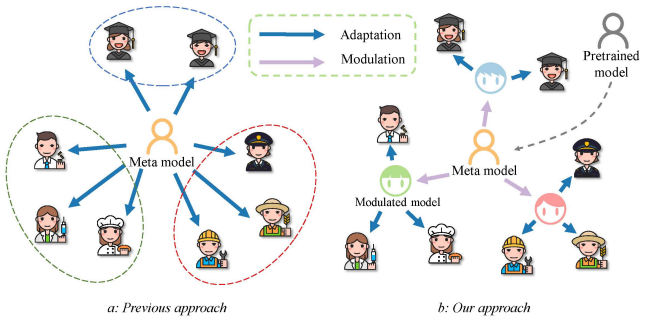


Figure 2: The previous MAML-based approaches (a) and our approach (b). In the previous approaches, each user's model is adapted from a common meta-model (the purple arrows). These approaches assume that users have a uniform distribution and ignore the implicit grouping effect of users. Our approach includes two steps, network modulation (the blue arrows) and task adaptation (the purple arrows), to obtain a model for each user. The network can be premodulated according to the identity of the given task to activate the implicit user groups and then adapted to specific users through a few gradient descent steps. (This figure is an intuitive toy display of this distribution phenomenon. In real situations, each user's preference belongs to different groups.)

- We propose a novel meta-learning-based RS framework called PNMTA for the user cold-start problem. Our model includes an encoder, a predictor, and two modulators (an encoder modulator and a predictor modulator).
- To achieve superior representation learning performance, we propose a meta-transfer learning approach based on pretraining and modulation. For the user group distribution problem commonly faced by RSs, we design a two-step meta-learning method based on network modulation and task adaptation.
- Extensive experimental results on two benchmark datasets show that PNMTA outperforms several state-of-the-art meta-learning-based recommenders.

2 RELATED WORK

2.1 Meta-Learning

Meta-learning is an emerging machine learning framework that gains general experience from several related tasks and applies this experience to improve its future learning performance on novel tasks [19]. This "learning to learn" optimization strategy is better aligned with human learning and achieves good results in many domains, such as few-shot learning [42], reinforcement learning [1], unsupervised learning [28], neural architecture search [12, 25, 35], and hyper-parameter optimization [15].

The existing meta-learning models can be divided into three categories. 1) Metric-based meta-learning frameworks [22, 40, 48, 51]. These methods obtain metric functions to measure the similarity levels among different tasks. Although excellent results have been produced in many domains, it is nontrivial to apply such approaches to complex tasks. 2) Model-based meta-learning frameworks [16, 29, 30, 34, 38]. These frameworks learn to adjust the

employed neural network or external memory according to the input training data to generate parameters that modulate the base models. These methods can accurately obtain the representations of tasks to optimize models, but they exhibit limited generalization on new tasks. 3) Model-agnostic meta-learning frameworks [14, 32, 36, 46]. The goals of these frameworks are to obtain task-specific initializations through gradient updates. Benefiting from their flexibility in terms of model selection, these methods are applied in many domains. However, the fact that such models adapt to all tasks from a common initialization limits their performance in cases with unbalanced task distributions [53, 58].

2.2 User Cold-Start Recommendation

User cold-start recommendation is an unavoidable problem in RSs [17] that seriously affects the experiences of new users and reduces the user retention rate. Early methods focused on incorporating side-information into collaborative filtering to alleviate this problem [52, 54]. With the deepening of research, a variety of methods have been introduced, such as active learning [65], curriculum learning [61], transfer learning [27], interview-based approaches [13], and social-based approaches [39].

Recent research on meta-learning has provided another perspective to solve the user cold-start problem by regarding it as a few-shot problem [5, 11, 47, 50, 62, 66]. Among the developed methods, MAML is the most popular framework and has shown outstanding results. MeLU [23] is a MAML-based recommendation model that regards making recommendations for a user as one task and divides the model parameters into two groups for optimization, and most of the follow-up MAML-based methods are based on it. MetaCS [2] follows a method similar to that of MeLU and it performs gradient descent on all parameters during the local update process. MetaHIN [26] introduces heterogeneous information networks (HINs) into MAML-based RSs to enrich user and item information. Two recent works follow similar goals to the predictor modulation in our model: MAMO [10] enhances the accuracy of task adaptation with external memory modules, and PAML [60] utilizes an task-adaptive learning rate to improve the resulting model's performance on major and minor users simultaneously.

3 FRAMEWORK OF PNMTA

3.1 Overview

3.1.1 Problem Definition. We consider the user cold-start recommendation to be a few-shot problem and solve it with meta-learning. Therefore, we first define this problem. Suppose that there are a user set U and an item set I in an RS. The interaction $x_{u,i}$ between user $u \in U$ and item $i \in I$ is represented as a tuple (u, i), and $y_{u,i}$ is the corresponding rating. We regard making recommendations for user u as a task τ_u, which is denoted as $[x_{u,i_1}, x_{u,i_2}, x_{u,i_3}, ...x_{u,i_{\mathcal{N}_u}}]$, where $\mathcal{N}_u$ is the number of interactions in τ_u. According to the actual RS, task τ_u is divided into a variable-length support set $\tau_u^{sup} = \{x_{u,i_n}\}_{n=1}^{\mathcal{N}_s}$ and a fixed-length query set $\tau_u^{qry} = \{x_{u,i_n}\}_{n=\mathcal{N}_s+1}^{\mathcal{N}_s+\mathcal{N}_q}$, where $\mathcal{N}_s$ and $\mathcal{N}_q$ are the numbers of interactions in τ_u^{sup} and τ_u^{qry}, respectively, and $\mathcal{N}_s + \mathcal{N}_q = \mathcal{N}_u$. Finally, our goal is to acquire a meta-model from existing users (tasks) and adapt it to novel users (tasks) through gradient updates. The notations in this paper are summarized in Table 1.

Table 1: Notations.

Notation	Explanation
U, I	user set and item set
$\mathcal{D}_x, \mathcal{D}_\tau$	interaction data and task data
$\mathcal{D}^{train}, \mathcal{D}^{test}$	training data and test data
$\mathcal{R}, \mathcal{R}'$	meta-model and task-specific model
$\mathcal{E}, \mathcal{P}$	encoder and predictor
$\mathcal{M}_e, \mathcal{M}_p$	modulators of the encoder and predictor
$x_{u,i} = (u, i)$	interaction between u and i
$\hat{y}_{u,i}, y_{u,i}$	predicted rating and real rating of i by u
τ_u	task of u
$\tau_u^{sup}, \tau_u^{qry}$	support and query sets of τ_u
$(*;*)$	concatenation operation
$d(*)$	dimensionality of vector $*$
$p(MLP_h^l)$	number of parameters in the l-th layer
θ_e, θ_e^*	parameters of the encoder and pretrained encoder
$\theta_p, \theta_p{}'$	predictor parameters of $\mathcal{R}$ and $\mathcal{R}'$
ϕ_e, ϕ_p	parameters of $\mathcal{M}_e$ and $\mathcal{M}_p$
ω	modulation parameters of $\mathcal{P}$
$v_{x_{u,i}}, v_u, v_h$	vectors of $x_{u,i}$, τ_u and output of hidden layers
ρ, α, β	learning rates of the pretraining, meta-adaptation and meta-update processes

3.1.2 Model Structure. PNMTA consists of four parts: an encoder $\mathcal{E}$ with parameters θ_e for generating embedding vectors $v_{x_{u,i}}$ and v_u (detailed in Section 3.2.1), a predictor $\mathcal{P}$ with parameters θ_p for outputting ratings $\hat{y}$ (detailed in Section3.2.2), and two modulators $\mathcal{M}_e$ and $\mathcal{M}_p$ with parameters ϕ_e and ϕ_p for conditioning the encoder and predictor networks, respectively (detailed in Section 3.3.1 and 3.3.2). The two modulators principles are different: the encoder modulator indirectly adjusts the pretrained encoder by updating its own parameters, and the predictor modulator receives the output v_u from the task encoder to compute the user-specific modulation parameters ω for the predictor with neural networks.

We divide the training process of PNMTA into three steps: pretraining (detailed in Section 3.4.1), meta-adaptation (detailed in Section 3.4.2), and meta-updating (detailed in Section 3.4.3). During pretraining, PNMTA obtains θ_e through an interaction data stream in random order. In meta-adaptation, θ_e and θ_p are modulated, and a few steps of gradient descent are performed on the prior parameters θ_p of the meta-model for user u. During meta-updating, ϕ_e, ϕ_p and θ_p are updated and then the meta-adaptation of the next task starts. We show the framework of PNMTA in Figure 3.

3.2 Recommender

3.2.1 Encoder: $(x_{u,i}, \theta_e) \rightarrow v_{x_{u,i}}, v_{x_{u,i}} \rightarrow v_u$. We introduce an encoder $\mathcal{E}$ to convert interaction $x_{u,i}$ (with $\mathcal{E}_i$) or task τ_u (with $\mathcal{E}_\tau$) into an embedding vectors $v_{x_{u,i}}$ or v_u. First, we represent user demographic information and the contents of items as one-hot vectors (or integers) $\{c_u^p\}_{p=1}^P$ and $\{c_i^q\}_{q=1}^Q$, respectively, where

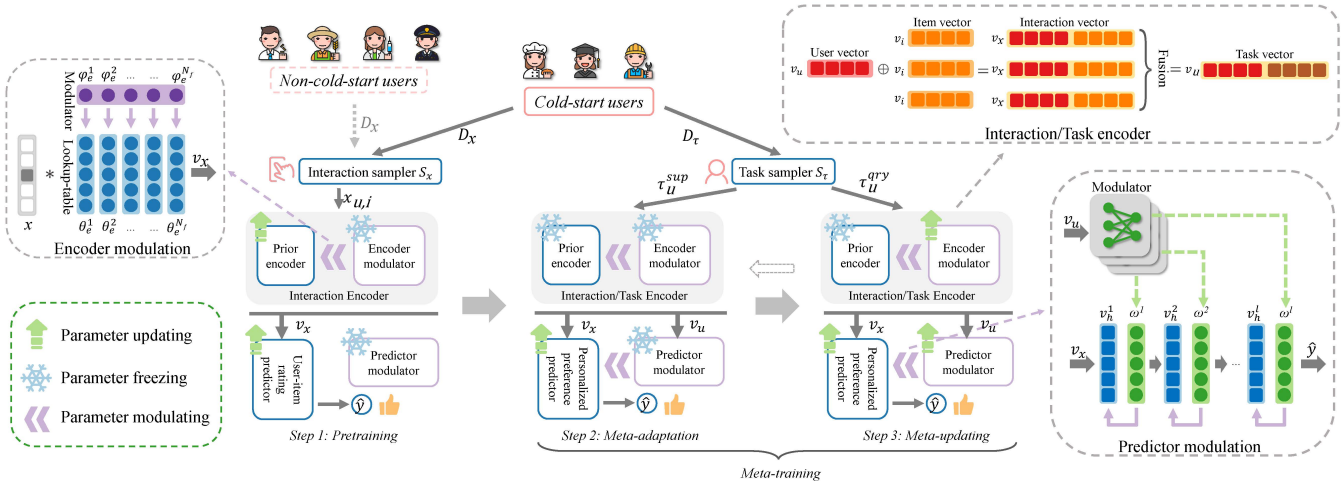


Figure 3: The overall framework of PNMTA. The model optimization process is divided into three steps: pretraining (step 1), meta-adaptation (step 2), and meta-updating (step 3). The pretraining step takes the interactions as inputs, and the other two steps take the tasks as inputs. The model structure consists of an encoder, a predictor, and two modulators.

P and Q are the numbers of user and item features. Then, these vectors are encoded into feature vectors by their corresponding embedding matrices. Finally, all the feature vectors are concatenated to constitute the interaction vector. The formula for doing so is described as follows:

$$v_{x_{u,i}} = \mathcal{E}_i(c_u, c_i) = [\theta^1_{e_u} c^1_u; ...; \theta^P_{e_u} c^P_u; \theta^1_{e_i} c^1_i; ...; \theta^Q_{e_i} c^Q_i] \quad (1)$$

In addition, the task vectors are obtained from the interaction vectors contained in each task with a fusion function $\mathcal{E}_\tau$:

$$v_u = \mathcal{E}_\tau(\{v_{x_{u,i}}\}_{i=1}^{N_u}) \quad (2)$$

The function $\mathcal{E}_\tau$ can be a neural network, an attention-based layer, or an average pooling layer. In this paper, we choose an average pooling layer as the default fusion operator.

3.2.2 Predictor: $(v_{x_{u,i}}, \theta_p) \rightarrow \hat{y}$. Given the interaction vector $v_{x_{u,i}}$, we obtain the corresponding prediction $\hat{y}$ of the real rating y through the predictor $\mathcal{P}$:

$$\hat{y} = \mathcal{P}(v_{x_{u,i}}) = MLP_{\theta_p}(v_{x_{u,i}}) \quad (3)$$

The predictor $\mathcal{P}$ in this paper denotes a multilayer perceptron (MLP) with parameters θ_p.

3.3 Modulators

3.3.1 Encoder Modulator: $(\theta_e, \phi_e) \rightarrow \theta_e'$. The pretrained parameters θ^*_e learned from the interactions contain the generalized representations of users and items. We need to develop an encoder modulator $\mathcal{M}_e$ to further modulate this knowledge to meta-learning tasks. Compared with the traditional pretraining approach, PNMTA has different optimization goals in the pretraining and meta-training stages. Therefore, $\mathcal{M}_e$ needs to eliminate the gap between different optimization goals while retaining the ability to accurately represent users and items. Notably, simply fixing θ_e during the meta-training phase does not meet this goal, but not fixing it will lead to the memory forgetting problem.

Schema theory was proposed in *Gestalt psychology*, the predecessor of cognitive psychology. A schema is a pattern for humans to understand the world or a mental structure of preconceived ideas that constantly evolves under the influence of environment and personal experience. The evolution of schemata is divided into *assimilation* (means absorption) and *accommodation* (means distortion).

Inspired by this, we consider the pretrained parameters θ^*_e from the perspective of schema theory, and then modulate them with operations similar to *assimilation* and *accommodation*:

$$\theta_e = \mathcal{M}_e(\theta^*_e) = \phi^w_e \cdot \theta^*_e + \phi^b_e \quad (4)$$

where $\phi^w_e \in \mathbb{R}^{N_f}$, $\phi^b_e \in \mathbb{R}^{d_{emb}}$, and $\theta^*_e \in \mathbb{R}^{d_{emb} * N_f}$, where N_f denotes the size of each feature vector. Each feature dimension of the encoder parameter matrix θ_e is considered an entire concept, that is, a schema. Modulation is performed on each dimension without disassembly, and the weight and bias simulate *assimilation* and *accommodation*, respectively. This method is named channel-wise modulation. In addition, we also propose several other modulators, such as channel-wise (w/o bias) (-Cb), element-wise (-E), element-wise (w/o bias) (-Eb), scale (-S), and scale&shift (-SS) modulators, and their detailed descriptions are included in Appendix A.

3.3.2 Predictor Modulator: $(v_u, \phi_p) \rightarrow \omega, (\theta_p, \omega) \rightarrow \theta_p'$. For the initialization of predictor $\mathcal{P}$, we introduce a predictor modulator $\mathcal{M}_p$ to pre-activate the implicit grouping distribution of user u according to its embedding vector v_u given by the task encoder $\mathcal{E}_\tau$. First, the task-specific modulation parameters ω for each block of $\mathcal{P}$ are generated by additional task-aware modules $\mathcal{M}^g_p$, and the vectors ω contain the task distribution feature of each user. The generation of a modulation vector ω^l for the l-th layer in the predictor network is denoted as:

$$\omega^l = \mathcal{M}^{g(l)}_p(v_u) = MLP_{\phi^l_p}(v_u) \quad (5)$$

Then, ω is divided into two components ω_w and ω_b. The output activation v_h of each hidden layer in the predictor network are modulated by their corresponding ω^l by modulator module $\mathcal{M}_p^h$:

$$v_h^l = \mathcal{M}_p^h(v_h^l) = \omega_w^l \odot v_h^l + \omega_b^l \tag{6}$$

The above modulation process is named feature-wise linear modulation (-FiLM), which can be viewed as a more generic form of attention mechanism. We set the FiLM layer between the connection layer and active layer. If batch normalization layers are included, the FiLM layer is set behind them. We also investigate another modulator called softmax attention (-SA) for comparison. More details are included in Appendix A.

3.4 Optimization

3.4.1 Pretraining. During pretraining, we optimize the parameters θ_e and θ_p through a traditional training process to minimize the loss of interaction data $\mathcal{D}_x$ and obtain a pretrained initialization of the encoder $\mathcal{E}$. First, an interaction sampler $\mathcal{S}_x$ randomly samples $x_{u,i}$ from $\mathcal{D}_x$. Then, the predicted rating $\hat{y}_{u,i}$ of $x_{u,i}$ is inferred by the recommender:

$$\hat{y}_{u,i} = \mathcal{P}(\mathcal{E}(x_{u,i})) \tag{7}$$

We use the mean square error (MSE) as the loss function of pretraining:

$$\mathcal{L}^{pt} = \frac{1}{\mathcal{N}} \sum_{x_{u,i}} (\hat{y}_{u,i} - y_{u,i})^2 \tag{8}$$

where $\mathcal{N}$ is the number of interactions in the training set.

Finally, the recommender parameters θ_e and θ_p are updated through gradient descent:

$$\theta_* = \theta_* - \rho \nabla_{\theta_*} \mathcal{L}^{pt}(\theta_e, \theta_p), (* \in (e, p)) \tag{9}$$

In addition, unlike meta-training, which takes cold-start users as tasks, pretraining is not restricted by the given task settings. Therefore, we can use interaction data obtained from not only cold-start users but also non-cold-start users to further improve the encoder's representation capabilities. The model pretrained on all available data is named PNMTA (Full-PT).

However, to be fair, we still use the PNMTA model, which is pretrained on the same data as those used by the meta-learning-based baseline methods, as the basic model in experiments to explore the improvement derived from model structure instead of data. PNMTA (Full-PT) is used as an ablation model in comparative experiments and we will discuss more about it in Section 4.5. The pseudocode of the pretraining procedure is given in Appendix B.

3.4.2 Meta-Adaptation. The aim of meta-adaptation is to obtain a task-specific model for each user from the meta-model through the two-step optimization of network modulation and task adaptation. After pretraining, the encoder parameters θ_e are kept fixed, the parameters θ_p and ϕ_p of the predictor $\mathcal{P}$ and its modulator $\mathcal{M}_p$ are initialized randomly, and the parameters ϕ_e^w and ϕ_e^b of the encoder modulator $\mathcal{M}_e$ are initialized to 1 and 0, respectively. This initial model is denoted as the meta-model $\mathcal{R}$.

During meta-adaptation, we introduce a task sampler $\mathcal{S}_\tau$ to randomly sample a task from $\mathcal{D}_\tau$. First, the encoder $\mathcal{E}$ is modulated by $\theta_e = \mathcal{M}_e(\theta_e)$, and the interactions $x_{u,i}$ in task τ_u are encoded into interaction vectors $v_{x_{u,i}}$ by $v_{x_{u,i}} = \mathcal{E}_i(x_{u,i})$.

After that, the task encoder $\mathcal{E}_\tau$ estimates the user identity according to its interactions $\tau_u = (\tau_u^{sup}, \tau_u^{qry})$ and generates the corresponding user vector v_u by $v_u = \mathcal{E}(\tau_u)$. The predictor modulator $\mathcal{M}_p$ receives this vector and generates corresponding modulation parameters $\{\omega^l\}_{l=1}^L$ for each layer of the predictor network by $\omega^l = \mathcal{M}_p^{g(l)}(v_u)$ to activate the user's implicit grouping distribution for the following task adaptation step. The modulated predictor provides rating predictions $\hat{y}_{u,i}$ for the interactions in the support set τ_u^{sup} according to the interaction vectors v_x by $\hat{y}_{u,i} = \mathcal{P}(v_{x_{u,i}})$. Finally, we calculate the MSE loss $\mathcal{L}_u^{sup}$ of the modulated model on the support set τ_u^{sup} and update the predictor parameter θ_p to adapt to user u:

$$\mathcal{L}_u^{sup} = \frac{1}{\mathcal{N}_s} \sum_{x_{u,i} \in \tau_u^{sup}} (\mathcal{R}(x_{u,i}) - y_{u,i})^2 \tag{10}$$

$$\theta_p{}' = \theta_p - \alpha \nabla_{\theta_p} \mathcal{L}_u^{sup}(\theta_e, \theta_p, \phi_e, \phi_p) \tag{11}$$

At this point, we obtain the task-specific model $\mathcal{R}'$ with predictor parameters $\theta_p{}'$.

During serving, the same procedure of meta-adaptation is used to obtain novel task-specific models $\mathcal{R}'$ for new users.

3.4.3 Meta-Update. To evaluate the performance of the task-specific model $\mathcal{R}'$ and optimize the meta-model $\mathcal{R}$, we calculate the prediction loss $\mathcal{L}_u^{qry}$ of $\mathcal{R}'$ on the query set τ_u^{qry} and minimize it by updating $\mathcal{R}$:

$$\mathcal{L}_u^{qry} = \frac{1}{\mathcal{N}_q} \sum_{x_{u,i} \in \tau_u^{qry}} (\mathcal{R}'(x_{u,i}) - y_{u,i})^2 \tag{12}$$

$$\theta_p = \theta_p - \beta \nabla_{\theta_p} \mathcal{L}_u^{qry}(\theta_e, \theta_p{}', \phi_e, \phi_p) \tag{13}$$

$$\phi_* = \phi_* - \beta \nabla_{\phi_*} \mathcal{L}_u^{qry}(\theta_e, \theta_p{}', \phi_e, \phi_p), (* \in (e, p)) \tag{14}$$

After the meta updating, model $\mathcal{R}$ completes a round of meta-training, and then it starts the meta-adaptation process for the next user. We describe the concrete procedure in the form of pseudocode in Appendix B.

3.5 Time Complexity

Due to the calculation of the Hessian matrix, MAML-based methods usually suffer from high time complexity [44]. We analyze the time complexity of PNMTA to evaluate the efficiency of its optimization and inference. For simplicity, we assume that the predictor $\mathcal{P}$ contains l layers, the number of neurons in each layer is n, and the encoder contains m matrices with the sizes of $a \cdot b$. Since only $\mathcal{P}$ is updated during meta-adaptation (i.e., the local update in MAML), we only need to estimate its second derivatives. Other modules perform gradient updates during meta-updating (i.e., the global update in MAML), which does not bring much additional burden to the model. Overall, the back propagation time complexity of pretraining can be approximated as $O(l \cdot n^2 + m \cdot a \cdot b)$, and the time complexity of back propagation during meta-training can be approximated as $O(l \cdot n^3)$. In addition, benefitting from the representation learning ability brought by pretraining, PNMTA can converge through a few epochs of meta-training, which also alleviates the impact of the Hessian matrix by reducing the overall time complexity of meta-training.

Table 2: Experimental results of different models on MovieLens-1M and Bookcrossing. For the MAE and RMSE, smaller values are better, and vice versa for NDCG@3 and NDCG@5. The best results are marked in bold, and the second-best results are marked by underline.

Model	MovieLens-1M				Bookcrossing			
	MAE	RMSE	NDCG@3	NDCG@5	MAE	RMSE	NDCG@3	NDCG@5
MeLU	0.7561	0.9547	0.8829	0.8894	1.6789	1.9626	0.8346	0.8391
MetaCS-DNN	0.7609	0.9655	0.8810	0.8867	1.6645	1.9570	0.8269	0.8274
s^2Meta	0.7594	0.9589	0.8775	0.8835	1.6442	1.9688	0.8283	0.8388
MetaEmb	0.7983	0.9963	0.8671	0.8694	1.7131	2.1242	0.8177	0.8255
MetaHIN	0.7344	0.9278	0.8896	0.8932	1.6855	2.0328	0.8091	0.8111
MAMO	0.7458	0.9317	0.8808	0.8914	1.6576	1.9674	0.8359	0.8386
TaNP	0.7260	0.9088	0.8835	0.8991	1.6396	1.9631	0.8427	0.8435
PAML	0.7199	0.9222	0.8913	0.8948	1.6223	1.9598	0.8394	0.8466
MAML-PT	0.7233	0.9119	0.8875	0.9021	1.6460	1.9833	0.8316	0.8333
PNMTA (w/o PT)	0.7364	0.9239	0.8866	0.8889	1.6544	1.9669	0.8374	0.8389
PNMTA	**0.6921**	**0.8797**	0.8993	**0.9042**	1.6058	**1.8722**	**0.8568**	**0.8596**
PNMTA (Full-PT)	0.6963	0.8825	**0.9007**	0.9016	**1.6005**	1.8947	0.8411	0.8432

4 EXPERIMENTS

In this section, we conduct experiments to evaluate the performance of PNMTA and answer the following research questions (RQs):

- **RQ1** : How well does PNMTA perform compared to its ablation models and the state-of-the-art methods?
- **RQ2** : How do different modulators affect the performance of PNMTA?
- **RQ3** : Does full pretraining make PNMTA remain stable even with few cold-start users? How does the number of interactions in the support set affect PNMTA?
- **RQ4** : How is PNMTA impacted by its hyperparameters?
- **RQ5** : In addition to the improvement in recommendation performance, does PNMTA actually learn better user representations? Can this be seen visually?

4.1 Datasets

We evaluate PNMTA on two public benchmark datasets: MovieLens-1M [1] and Bookcrossing [2]. MovieLens-1M is a widely used dataset with user ratings on movies, and Bookcrossing contains user ratings on books obtained from the Book-Crossing community.

We regard users with rating counts between 13 and 100 as cold-start users. The last 10 rating records of each user are used as the query set, and the rest are used as the support set.

4.2 Setup

4.2.1 Evaluation Metrics. We utilize the following metrics to evaluate the compared recommendation models: the mean average error (MAE), root mean squard Error (RMSE), and normalized discounted cumulative gain (NDCG)@K (K = 3, 5). For each metric, consistent with previous works [10, 23, 24, 26], we calculate the average results for all users from the query sets in the test set. More details regarding these metrics are included in Appendix C.1.

[1] https://grouplens.org/datasets/movielens/
[2] http://www2.informatik.uni-freiburg.de/~cziegler/BX/

4.2.2 Compared Models. We compare our approach with the following meta-learning-based state-of-the-art methods: **MeLU** [23], **MetaCS-DNN** [2], **s^2Meta** [11], **MetaEmb** [33], **MetaHIN** [26], **MAMO** [10], **TaNP** [24], and **PAML** [60]. More details about these baseline methods are described in Appendix C.2.

4.2.3 Hyperparameters and Implementation Details. The hyperparameters and implementation details of PNMTA and the baseline models can be found in Appendices C.3 and C.4.

4.3 Performance Comparison (RQ1)

We compare the performance of PNMTA with that of its ablation models and the baseline models. **MAML-PT** denotes the pretrained PNMTA model without modulators and the pretrained parameters of its encoder are not fixed during meta-training, i.e., the pretrained MAML. **PNMTA (w/o PT)** denotes PNMTA without pretraining. **PNMTA (Full-PT)** denotes PNMTA pretrained with cold-start and non-cold-start users. The experimental results are shown in Table 2. More ablation experiments about the modulators are introduced in Section 4.4.

From the experimental results, we draw the following conclusions: 1) PNMTA outperforms other state-of-the-art methods with significant improvements on both datasets in terms of four metrics. For example, PNMTA achieves 4.01-13.30% improvements on MovieLens-1M and 1.62-5.71% improvements on Bookcrossing with respect to the MAE. 2) Among all the compared baseline models, MeLU and MetaCS-DNN are traditional MAML-based methods, and they generally suffer from the problems we raised earlier. MetaEmb is also based on MAML, but its solution is different from those of other models. Its poor performance (ranked last) may be because its meta-embedding generator is not fit for the prediction of users' rating of items.MetaHIN incorporates HINs into the MAML-based RS, but this also brings additional construction costs. Its performance on Bookcrossing is not excellent. One possible reason for this is that this dataset is sparse, which leads to less information

Table 3: Ablation study of different modulators in terms of the MAE metric on MovieLens-1M and Bookcrossing, where PNMTA-C-FiLM is our basic PNMTA model. The best result is marked in bold, and the second-best results are marked by underline.

MovieLens-1M	-E	-Eb	-C	-Cb	-S	-SS	w/o EM	-Fix
-SA	0.7178	0.7344	0.7126	0.7082	0.7268	0.7233	0.7391	0.7417
-FiLM	0.7125	0.7287	**0.6921**	<u>0.6998</u>	0.7249	0.7076	0.7284	0.7441
w/o PM	0.7274	0.7296	0.7315	0.7108	0.7160	0.7134	0.7233	0.7478
Bookcrossing	**-E**	**-Eb**	**-C**	**-Cb**	**-S**	**-SS**	**w/o EM**	**-Fix**
-SA	1.6355	1.6481	1.6158	<u>1.6065</u>	1.6221	1.6305	1.6366	1.6665
-FiLM	1.6322	1.6270	**1.6058**	1.6196	1.6359	1.6351	1.6458	1.6534
w/o PM	1.6569	1.6432	1.6377	1.6364	1.6399	1.6508	1.6460	1.6724

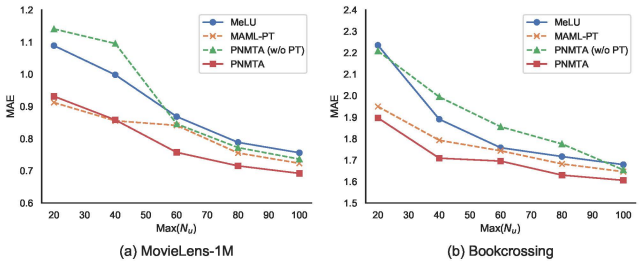


Figure 4: Impact of N_s.

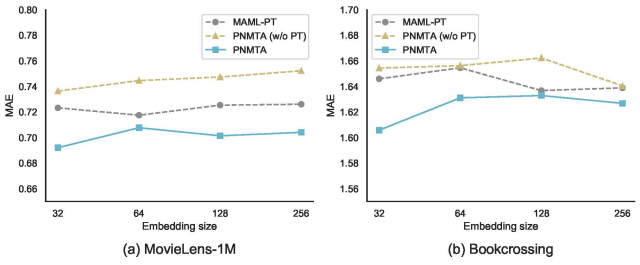


Figure 6: Sensitivity to the embedding size.

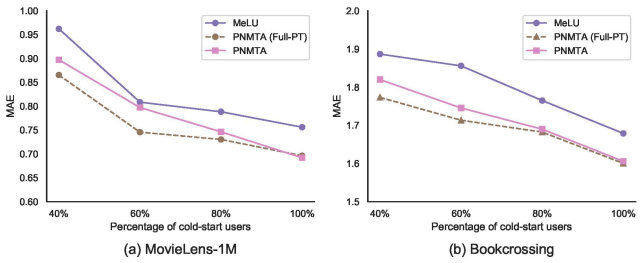


Figure 5: Impact of cold-start users.

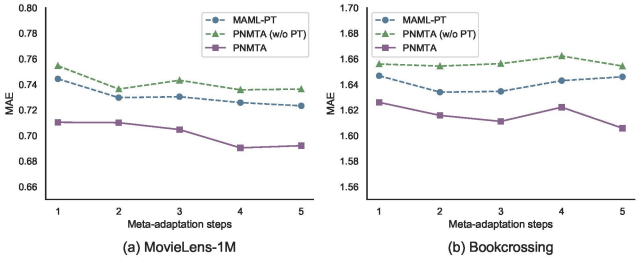


Figure 7: Sensitivity to the number of meta-adaptation steps.

contained in its meta paths. MAMO introduces additional memory modules to accurately adapt to different users; this is similar to the goal of PNMTA's predictor modulator. However, MAMO requires more hyperparameters to adjust its feature-specific and task-specific memory, while PNMTA can learn to modulate predictors without additional hyperparameters. s^2Meta and TaNP are not model-agnostic methods, so their model structures are more restricted. PAML achieves improved recommendation effects for both major and minor users, but it does not consider highly fine-grained distinctions among users. 3) The performance of MAML-PT and PNMTA (w/o PT) is worse than that of PNMTA, which proves that modulators and pretraining are both necessary. 4) Compared with PNMTA, PNMTA (Full-PT) does not obtain a significant improvement from additional pretraining data.

We also plot the training curves of the models during the meta-training process. See Appendix D for more details.

4.4 Modulator Analyses (RQ2)

The MAE results of PNMTA with different modulators are detailed in Table 3, where -Fix and w/o PM denote the model without an encoder modulator or predictor modulator, respectively, and w/o EM denotes that the pretrained encoder parameters are not fixed in meta-training and without modulation.

From the experimental results, we observe that PNMTA-C-FiLM (i.e. PNMTA) achieves the best result, and PNMTA-Cb-FiLM and PNMTA-Cb-SA are the second-best models on MovieLens-1M and Bookcrossing, respectively. Regarding the encoder modulators, PNMTA-E and PNMTA-Eb perform worse than PNMTA-C and PNMTA-Cb since element-wise modulators result in too many parameters and damage the integrity of the schema. PNMTA-S and PNMTA-SS do not achieve significantly improved performance because feature-wise modulation is not applicable to encoded vectors. The performance of PNMTA (w/o) is better than that of PNMTA-Fix, which shows that due to the gap between the pretraining and

meta-training tasks, only fixing the pretraining parameters without modulation can damage the representation learning ability of the model. In terms of the predictor modulators, although PNMTA-SA achieves improvements over PNMTA (w/o PM), it is still not as good as PNMTA-FiLM.

4.5 Impact of Data (RQ3)

We change the maximum value of N_s to investigate the impact of the size of the support set on the models. We select MeLU, MAML-PT, PNMTA (w/o PT), and PNMTA for the experiment and report the MAE performances achieved on the two datasets in Figure 4. We observe that with a larger support set, all models achieve better performance. With the reduction in the number of items in support set, compared to MeLU and PNMTA (w/o PT), PNMTA and MAML-PT exhibit more stable results. Overall, PNMTA is better than other models.

We also set different percentages of cold-start users for the training data and report the MAE performance of MeLU, PNMTA (Full-PT), and PNMTA on both datasets in Figure 5. According to the experimental results, the performances of all models improve with the increase in the number of cold-start users, and PNMTA and PNMTA (Full-PT) always perform better than MeLU. It is worth noting that when there are fewer available cold-start users, the improvement of PNMTA (Full-PT) over PNMTA is more obvious, which proves that full pretraining can produce more advantages when the meta-training data are less.

4.6 Hyperparameter Analysis (RQ4)

Here we consider the sensitivity of PNMTA and its ablation models to two main hyperparameters: the embedding size of each feature and the number of meta-adaptation steps. We plot the experimental results in Figures 6 and 7 in terms of the MAE metric. We observe that PNMTA performs better than the ablation models in most cases and is not very sensitive to these parameters. Therefore, we can appropriately reduce the embedding size and the number of meta-adaptation steps to improve the efficiency of PNMTA.

4.7 Visualization (RQ5)

To study the improvement achieved by PNMTA in terms of user representation, we visualize the user vectors generated by the following four models: (a) MeLU with 1 training epoch, (b)MeLU with 30 training epochs, (c) The pretrained PNMTA without meta-training, and (d) The pretrained PNMTA with 10 meta-training epochs. These vectors are processed into 2-dimensions through dimensionality reduction, and the user ages are highlighted by different colors and styles. We show the results in Figure 8.

First, according to (a) and (b), we observe that the user representation vectors generated by MeLU have a low degree of discrimination and a high degree of overlap in the embedding space, and the users are not differentiated by age. After model training, this situation is not changed. Second, from (b) and (c), we find that PNMTA obtains an excellent user representation capability through pretraining. In the meta-training stage, it can maintain this ability without over-fitting the meta-training data or encountering the memory forgetting problem. This indicates that PNMTA acquires better user representations than those of existing methods through pretraining, and the encoder modulator makes it perform well on meta-learning tasks while maintaining this advantage. Overall, it is certain that PNMTA does indeed achieve better user representation performance than other methods.

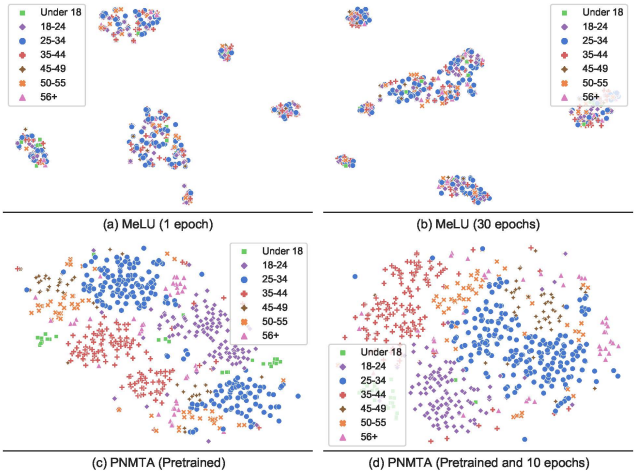


Figure 8: The 2-D visualizations with t-distributed stochastic neighbor embedding (t-SNE) [49] of the user vectors obtained on MovieLens-1M. The colors and styles of points represent user ages.

5 CONCLUSION

In this work, we propose a novel approach called PNMTA for user cold-start recommendation. PNMTA can obtain excellent representation learning capabilities from pretraining, and then incorporate the pretrained knowledge into meta-training through encoder modulation. By combining model-agnostic and model-based meta-learning methods, PNMTA utilizes an additional predictor modulator to activate the implicit grouping distribution characteristics of users according to their identities, thereby conditioning the network parameters adapt to novel users more accurately and rapidly. Extensive experiments on two benchmark datasets show that PNMTA consistently outperforms the state-of-the-art methods.

ACKNOWLEDGMENTS

The authors are grateful for the support of the Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, National Key R&D Program of China [No. 2021ZD0112501 and No. 2021ZD0112502], National Natural Science Foundation of China [No. 61972174 and No.62172187], the Science-Technology Development Plan Project of Jilin Province [No. 20190303006SF and No. 20190302107GX], the Science and Technology Planning Project of Guangdong Province [No. 2020A0505100018], Guangdong Key-Project for Applied Fundamental Research [No. 2018KZDXM076] and the Tencent Rhino-Bird Research Program. Fausto Giunchiglia's work is funded by National Project PRIN 2017 "Delphi".

REFERENCES

[1] Ferran Alet, Martin F Schneider, Tomas Lozano-Perez, and Leslie Pack Kaelbling. 2019. Meta-learning curiosity algorithms. In *International Conference on Learning Representations*.

[2] Homanga Bharadhwaj. 2019. Meta-learning for user cold-start recommendation. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[3] JesúS Bobadilla, Fernando Ortega, Antonio Hernando, and Jesús Bernal. 2012. A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-based systems* 26 (2012), 225–238.

[4] Dorwin Cartwright. 1979. Contemporary social psychology in historical perspective. *Social Psychology Quarterly* (1979), 82–93.

[5] Yudong Chen, Xin Wang, Miao Fan, Jizhou Huang, Shengwen Yang, and Wenwu Zhu. 2021. Curriculum Meta-Learning for Next POI Recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2692–2702.

[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[8] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 101–109.

[9] Debashis Das, Laxman Sahoo, and Sujoy Datta. 2017. A survey on recommendation system. *International Journal of Computer Applications* 160, 7 (2017).

[10] 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*. 688–697.

[11] 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*. 2895–2904.

[12] Thomas Elsken, Benedikt Staffler, Jan Hendrik Metzen, and Frank Hutter. 2020. Meta-learning of neural architectures for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12365–12375.

[13] Crícia Z Felício, Klérisson VR Paixão, Celia AZ Barcelos, and Philippe Preux. 2017. A multi-armed bandit model selection for cold-start user recommendation. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. 32–40.

[14] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*. PMLR, 1126–1135.

[15] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*. PMLR, 1568–1577.

[16] Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. 2018. Conditional neural processes. In *International Conference on Machine Learning*. PMLR, 1704–1713.

[17] Jyotirmoy Gope and Sanjay Kumar Jain. 2017. A survey on solving cold start problem in recommender systems. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 133–138.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[19] Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. [n.d.]. Meta-Learning in Neural Networks: A Survey. *IEEE transactions on pattern analysis and machine intelligence* ([n. d.]).

[20] SeongKu Kang, Junyoung Hwang, Dongha Lee, and Hwanjo Yu. 2019. Semi-supervised learning for cross-domain recommendation to cold-start users. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1563–1572.

[21] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.

[22] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, Vol. 2. Lille.

[23] 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*. 1073–1082.

[24] Xixun Lin, Jia Wu, Chuan Zhou, Shirui Pan, Yanan Cao, and Bin Wang. 2021. Task-adaptive Neural Process for User Cold-Start Recommendation. In *Proceedings of the Web Conference 2021*. 1306–1316.

[25] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. DARTS: Differentiable Architecture Search. In *International Conference on Learning Representations*.

[26] Yuanfu Lu, Yuan Fang, and Chuan Shi. 2020. Meta-learning on heterogeneous information networks for cold-start recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1563–1573.

[27] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-Domain Recommendation: An Embedding and Mapping Approach.. In *IJCAI*, Vol. 17. 2464–2470.

[28] Luke Metz, Niru Maheswaranathan, Brian Cheung, and Jascha Sohl-Dickstein. 2018. Meta-Learning Update Rules for Unsupervised Representation Learning. In *International Conference on Learning Representations*.

[29] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2017. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141* (2017).

[30] Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *International Conference on Machine Learning*. PMLR, 2554–2563.

[31] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Icml*.

[32] Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999* (2018).

[33] 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*. 695–704.

[34] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[35] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 33. 4780–4789.

[36] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. 2018. Meta-Learning with Latent Embedding Optimization. In *International Conference on Learning Representations*.

[37] Guillaume Salha-Galvan, Romain Hennequin, Benjamin Chapus, Viet-Anh Tran, and Michalis Vazirgiannis. 2021. Cold Start Similar Artists Ranking with Gravity-Inspired Graph Autoencoders. *arXiv preprint arXiv:2108.01053* (2021).

[38] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*. PMLR, 1842–1850.

[39] Suvash Sedhain, Aditya Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. 2017. Low-rank linear cold-start recommendation from social data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[40] Pranav Shyam, Shubham Gupta, and Ambedkar Dukkipati. 2017. Attentive recurrent comparators. In *International Conference on Machine Learning*. PMLR, 3173–3181.

[41] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 650–658.

[42] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 4080–4090.

[43] Robert L Solso, M Kimberly MacLin, and Otto H MacLin. 2005. *Cognitive psychology*. Pearson Education New Zealand.

[44] Xingyou Song, Wenbo Gao, Yuxiang Yang, Krzysztof Choromanski, Aldo Pacchiano, and Yunhao Tang. 2020. ES-MAML: Simple Hessian-Free Meta Learning. In *ICLR*.

[45] Ivan D Steiner. 1974. Whatever happened to the group in social psychology? *Journal of Experimental Social Psychology* 10, 1 (1974), 94–108.

[46] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. 2019. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 403–412.

[47] Xuehan Sun, Tianyao Shi, Xiaofeng Gao, Yanrong Kang, and Guihai Chen. 2021. FORM: Follow the Online Regularized Meta-Leader for Cold-Start Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1177–1186.

[48] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1199–1208.

[49] Laurens Van Der Maaten. 2014. Accelerating t-SNE using tree-based algorithms. *The Journal of Machine Learning Research* 15, 1 (2014), 3221–3245.

[50] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. 2017. A meta-learning perspective on cold-start recommendations for items. (2017).

[51] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems* 29 (2016), 3630–3638.

[52] Maksims Volkovs, Guang Wei Yu, and Tomi Poutanen. 2017. Content-based neighbor models for cold start in recommender systems. In *Proceedings of the Recommender Systems Challenge 2017*. 1–6.

[53] Risto Vuorio, Shao-Hua Sun, Hexiang Hu, and Joseph J Lim. 2019. Multimodal model-agnostic meta-learning via task-aware modulation. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 1–12.

[54] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1235–1244.

[55] Li Wang, Binbin Jin, Zhenya Huang, Hongke Zhao, Defu Lian, Qi Liu, and Enhong Chen. [n.d.]. Preference-Adaptive Meta-Learning for Cold-Start Recommendation. ([n. d.]).

[56] Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S Sitharama Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. 2018. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *IEEE Transactions on Knowledge and Data Engineering* 31, 8 (2018), 1569–1580.

[57] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)* 53, 3 (2020), 1–34.

[58] Huaxiu Yao, Ying Wei, Junzhou Huang, and Zhenhui Li. 2019. Hierarchically structured meta-learning. In *International Conference on Machine Learning*. PMLR, 7045–7054.

[59] Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. 2018. Bayesian model-agnostic meta-learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 7343–7353.

[60] Runsheng Yu, Yu Gong, Xu He, Yu Zhu, Qingwen Liu, Wenwu Ou, and Bo An. 2021. Personalized Adaptive Meta Learning for Cold-start User Preference Prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 10772–10780.

[61] Yin Zhang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Lichan Hong, and Ed H Chi. 2021. A Model of Two Tales: Dual Transfer Learning Framework for Improved Long-tail Item Recommendation. In *Proceedings of the Web Conference 2021*. 2220–2231.

[62] Yujia Zheng, Siyi Liu, Zekun Li, and Shu Wu. 2021. Cold-start Sequential Recommendation via Meta Learner. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4706–4713.

[63] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 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.

[64] Yongchun Zhu, Kaikai Ge, Fuzhen Zhuang, Ruobing Xie, Dongbo Xi, Xu Zhang, Leyu Lin, and Qing He. 2021. Transfer-Meta Framework for Cross-domain Recommendation to Cold-Start Users. *arXiv preprint arXiv:2105.04785* (2021).

[65] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2019. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Transactions on Knowledge and Data Engineering* 32, 4 (2019), 631–644.

[66] 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. *arXiv preprint arXiv:2105.04790* (2021).

A MODULATOR DETAILS

We describe the details of other encoder and predictor modulators in Tables 4 and 5, respectively.

B PSEUDOCODE

The pseudo codes of pretraining and meta-training are shown in Algorithm 1 and 2, respectively.

Algorithm 1: Pretraining procedure of PNMTA.

Input: Interaction data D_x, Hyper-parameters ρ.
Output: Pretrained encoder parameters θ_e^*.

Randomly initialize θ_e and θ_p.
while *not convergence* **do**
  for $x_{u,i} \in D_x$ **do**
    PRETRAINING:
    Encode interaction $x_{u,i}$ into $v_{x_{u,i}}$ with $\mathcal{E}_i$ by Eq. 1;
    Get prediction rating $\hat{y}_{u,i}$ of $y_{u,i}$ with $\mathcal{P}$ by Eq. 3;
    Calculate loss $\mathcal{L}_{u,i}^{pt}$ by Eq. 8;
    Update parameters θ_e and θ_p of $\mathcal{E}$ and $\mathcal{P}$ by Eq. 9;
  end
end
$\theta_e^* \leftarrow \theta_e$.

Algorithm 2: Meta-training procedure of PNMTA.

Input: Pretrained encoder parameters θ_e^*,
Hyper-parameters α and β.
Output: Meta-model $\mathcal{R}$ with parameters θ_e, θ_p, ϕ_e, and ϕ_p.

Randomly initialize θ_p and ϕ_p, $\theta_e \leftarrow \theta_e^*$, $\phi_e^w \leftarrow 1$, $\phi_e^b \leftarrow 0$.
while *note convergence* **do**
  for $\tau_u \in \mathcal{D}_\tau$ **do**
    META-ADAPTATION: $\mathcal{R} \rightarrow \mathcal{R}'$
    Modulate $\mathcal{E}$ with $\mathcal{M}_e$ by Eq. 4;
    Encoder interactions $x_{u,i}$ in τ_u into $v_{x_{u,i}}$ with $\mathcal{E}_i$ by Eq.1;
    Generate task vector v_u for user u with $\mathcal{E}_\tau$ by Eq. 2;
    Calculate modulation vectors ω with $\mathcal{M}_p^g$ by Eq. 5;
    Modulate $\mathcal{P}$ with $\mathcal{M}_p^h$ and ω by Eq. 6;
    for $x_{u,i} \in \tau_u^{sup}$ **do**
      Predict rating $\hat{y}_{u,i}$ of $x_{u,i}$ from τ_u^{sup} with $\mathcal{P}$ by Eq. 3;
      Calculate loss $\mathcal{L}_{u,i}^{sup}$ by Eq. 10;
      Update θ_p to $\theta_p{}'$ by Eq. 11;
    end
    META-UPDATE: $\mathcal{R} \rightarrow \mathcal{R}$
    Predict rating $\hat{y}_{u,i}$ of $x_{u,i}$ from τ_u^{qry} with $\mathcal{P}$ by Eq. 3;
    Calculate loss $\mathcal{L}_{u,i}^{qry}$ by Eq. 12;
    Update θ_p, ϕ_e, ϕ_p by Eq. 13,14;
  end
end

C EXPERIMENTAL SETUP

C.1 Evaluation Metrics

The formulas of the utilized evaluation metrics are as follows:

$$MAE = \frac{1}{\left|\mathcal{D}_\tau^{test}\right|} \sum_{\tau_u \in \mathcal{D}_\tau^{test}} \frac{1}{\mathcal{N}_q} \sum_{i \in \tau_u^{qry}} \left|y_{u,i} - \hat{y}_{u,i}\right| \tag{15}$$

$$RMSE = \sqrt{\frac{1}{\left|\mathcal{D}_\tau^{test}\right|} \frac{1}{\mathcal{N}_q} \sum_{\tau_u \in \mathcal{D}_\tau^{test}} \sum_{i \in \tau_u^{qry}} (y_{u,i} - \hat{y}_{u,i})^2} \tag{16}$$

$$nDCG@K = \frac{1}{\left|\mathcal{D}_\tau^{test}\right|} \sum_{\tau_u^{qry} \in \mathcal{D}_\tau^{test}} \frac{DCG@K}{IDCG@K} \tag{17}$$

$$DCG@K = \sum_{k=1}^{K} \frac{2^{r_{u,k}} - 1}{log_2(k+1)} \tag{18}$$

C.2 Baseline Methods

The detailed description of the baseline models is as follows:

- **MeLU** [23] is a MAML-based recommendation model that regards predicting the preferences of each user as one task and performs updates through meta-optimization. Its embedding layer parameters are only changed during global updates and are fixed during local updates.
- **MetaCS-DNN** [2] follows an idea similar to that of MeLU, which is also a MAML-based method for the cold-start problem. The difference is that its embedding layer parameters are also updated during local updates.
- **s^{2}Meta** [11] is a meta-learning-based approach for sequential recommendation. It uses update and stop controllers to generate scenario-specific learning settings for different scenarios.
- **MetaEmb** [33] is a MAML-based approach that improves click-through-rate predictions by generating desirable initial embeddings for new ad IDs with their contents and attributes. We employ this method to generate new user embeddings for the user cold-start problem.
- **MetaHIN** [26] integrates HINs into MAML to alleviate the cold-start problem at the data and model levels. It proposes a semantic-enhanced task constructor to capture HIN-based semantics under the meta-learning setting and a co-adaptation meta-learner to learn the general knowledge to adapt to multifaceted semantics.
- **MAMO** [10] is a memory-augmented MAML-based approach with two memory matrices. To accurately adapt to users who exhibit different gradient directions from those of the majority of training users, this method uses feature-specific memories to guide the model with personalized parameter initializations and task-specific memories to guide the model quickly predicting user preference.
- **TaNP** [24] is a meta-learning recommender based on the neural process that associates making recommendations for each user with a corresponding stochastic process. TaNP directly maps the observed interactions of each user to a predictive distribution, and it contains a task-adaptive mechanism to balance the capacity and adaptation reliability of the model.

Table 4: Encoder modulators.

Modulator $\mathcal{M}_e$	Abbr	Equation	Parameter count
-Element-wise	-E	$\theta_e = \phi_e^w \odot \theta_e + \phi_e^b$	$2 * d(c_u; c_i) * d(v_x)$
-Element-wise (w/o bias)	-Eb	$\theta_e = \phi_w \odot \theta_e$	$d(c_u; c_i) * d(v_x)$
-Channel-wise	-C	$\theta_e = \phi_e^w \cdot \theta_e + \phi_e^b$	$2 * d(c_u; c_i)$
-Channel-wise (w/o bias)	-Cb	$\theta_e = \phi_e^w \cdot \theta_e$	$d(c_u; c_i)$
-Scale	-S	$v_x = v_x \odot \phi_e^s$	$d(v_x)$
-Scale&Shift	-SS	$v_x = v_x \odot \phi_e^w + \phi_e^b$	$2 * d(v_x)$

Table 5: Predictor modulators.

Modulator $\mathcal{M}_p$	Abbr	Modulation network	Equation	Parameter count (for one block in predictor)
-SoftMax attention	-SA	$v_\tau \rightarrow MLP \rightarrow SoftMax \rightarrow \omega_a$	$v_h^l = \omega_a^l \odot v_h^l$	$p(MLP_h^{[:-1]}) + d(MLP_h^{-1}) * d(v_h^l) + d(v_h^l)$
-Feature-wise linear modulation	-FiLM	$v_\tau \rightarrow MLP \rightarrow [\omega_w; \omega_b]$	$v_h^l = \omega_w^l \odot v_h^l + \omega_b^l$	$p(MLP_h^{[:-1]}) + d(MLP_h^{-1}) * d(v_h^l) * 2 + d(v_h^l) * 2$

- **PAML** [60] is a MAML-based approach with personalized adaptive learning rates that improves recommendation performance by focusing on both the major and minor users. It introduces a similarity-based method to find similar users as references and a tree-based method to store user features for fast searching.

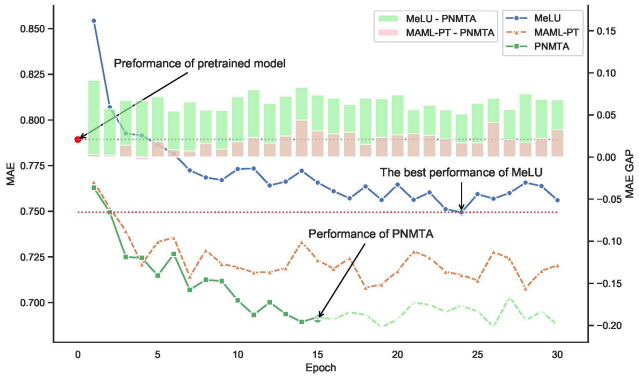


Figure 9: Training curves of MeLU, MAML-PT, and PNMTA on MovieLens-1M. The curves indicate the MAE results obtained on the test data after gradient-based updates and modulation, and the bars represent the MAE gaps between different models.

C.3 Hyperparameters

For fairness, the embedding size of each feature vector is fixed to 32, and the predictors of all models are set to the same $[d(v_x) \rightarrow 64 \rightarrow 64 \rightarrow 1]$ MLP. For all MAML-based methods, the global and local learning rates are determined by grid searches in the range of $[1, 0.1, 0.01, 0.001, 0.0001]$, each batch contains 32 tasks, and the meta-adaptation steps are searched in the range of $[1, 2, 3, 4, 5]$. We set the other model-specific hyperparameters according to the authors' suggestions. In PNMTA, ρ, α and β are set to 0.01, 0.01 and 0.001, the number of meta-adaptation steps is 5, and the numbers of pretraining and meta-training epochs are 20 and 15, respectively. The activation function is Rectified Linear Unit (ReLU) [31], and the model is optimized by Adaptive Moment Estimation (Adam) [21].

C.4 Implementation Details

The codes of MeLU [3], s^{2}Meta [4], MetaEmb [5], MetaHIN [6], MAMO [7], and TaNP [8] are released. We modify their inputs and outputs to fit our experimental settings. MetaCS-DNN is implemented based on the code of MeLU, which has a similar idea of it. We implement the version of PAML called Regularizer PAML (REG-PAML), which preforms best in terms of the rating prediction according to the reported results in the original paper. Our code is implemented with PyTorch[9] 1.4.1 and Python[10] 3.6.8. All experiments are conducted on a Linux server with a GPU (NVIDIA TITAN X) and CPU (Intel Core i7-5930K), and the operating system is Ubuntu[11] 16.04.5.

D TRAINING PROCESS

We show the performance of MeLU, MAML-PT and PNMTA on test data during the training process in Figure 9. 1) PNMTA always exhibits a significant improvement over MeLU. 2) Benefiting from the user representation ability obtained via pretraining, MAML-PT performs better than MeLU but is unstable. 3) In the early stage of meta-training, PNMTA and MAML-PT have similar results, but as the training process progresses, the gap between them increases and PNMTA shows better potential and stability.

[3] https://github.com/hoyeoplee/MeLU
[4] https://github.com/THUDM/ScenarioMeta
[5] https://github.com/Feiyang/MetaEmbedding
[6] https://github.com/rootlu/MetaHIN
[7] https://github.com/dongmanqing/Code-for-MAMO
[8] https://github.com/IIEdm/TaNP
[9] https://pytorch.org/
[10] https://www.python.org/
[11] https://ubuntu.com/