Few-Shot Learning for Cold-Start Recommendation

Qiannan Zhu[1], **Mingming Li**[2,3,†], **Fuqing Zhu**[2], **Songlin Hu**[2,†]
[1]School of Artificial Intelligence,Beijing Normal University
[2]JD.com, Beijing, China
[3]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
zhuqiannan@bnu.edu.cn, limingming65@jd.com, {zhufuqing,husonglin}@iie.ac.cn

Abstract

Cold-start is a significant problem in recommender systems. Recently, advancements in few-shot learning and meta-learning have inspired many to integrate these techniques into recommender systems to tackle inherent issues like data scarcity and limited user interactions Nevertheless, we argue that recent work has a huge gap between few-shot learning and recommendations. In particular, users are locally dependent, not globally independent in recommendation. Recognizing the importance of these local user relationships, we present a novel Few-shot learning method for Cold-Start (FCS) recommendation that consists of three hierarchical structures. More concretely, this first hierarchy is the global-meta parameters for learning the global information of all users; the second hierarchy is the local-meta parameters whose goal is to learn the adaptive cluster of local users; the third hierarchy is the specific parameters of the target user. Both the global and local information are formulated, addressing the new user's problem in accordance with the few-shot records rapidly. Experimental results on two public real-world datasets show that the FCS method could produce stable improvements compared with the state-of-the-art.

Keywords: Cold-Start, Recommender Systems, Few-Shot Learning

1. Introduction

Recommender systems have become significant tools for discovering attractive information for users. Many prestigious approaches (Koren et al., 2009; Rendle et al., 2009; Mnih and Salakhutdinov, 2008; He et al., 2017; Cheng et al., 2016) have been proposed and made a great breakthrough in the recommendation community. Nevertheless, we argue that they are brittle to recommendation scenarios with few interactions. In other words, the cold-start problem is still an open challenge for practical recommendations.

To alleviate the cold-start problem, many researchers have presented lots of work (Vartak et al., 2017; Sedhain et al., 2017; Hu et al., 2018; Kang et al., 2019; Li and Tuzhilin, 2020; Lee et al., 2019; Luo et al., 2020; Pan et al., 2019). Existing methods can be roughly divided into three categories: 1) **data-driven** method, referring to the use of prior knowledge and sample augmentation, which is straightforward to understand. The representative works are the transfer learning (Hu et al., 2018; Kang et al., 2019; Man et al., 2017; Li and Tuzhilin, 2020; Wang et al., 2019b). 2) **model-driven** method, imposing the constraints to approximate the ground-truth hypothesis, such as multi-task learning. 3) **algorithm-driven** method, focusing on refining the existing parameters, such as meta-learning methods (Lee et al., 2019; Luo et al., 2020; Du et al., 2019; Pan et al., 2019; Vartak et al., 2017). We argue that the data-driven method faces the high cost of collecting annotated data. The model-driven method needs to explore the related task for end-to-end learning. The algorithm-driven method is a popular research field recently, which only relies on the few-shot samples to learn the task parameters, achieving a fast adaption for new tasks. Thus, the few-shot based method is naturally suitable for the recommendation where the data is also actually long-tailed, without sufficient user feedback. More importantly, the algorithm-driven method is devoted to achieving the goal of parameter selection for target users. In this paper, we continue this trajectory toward. The meta-learning-based method (Lee et al., 2019; Bharadhwaj, 2019; Luo et al., 2020) has been applied for estimating new users' preferences with a few consumed items. This type of method depends on the meta-learning technique Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017; Chen et al., 2019), which assumes that all tasks (corresponding to users in recommendation scenarios) are independent, and then each task/user could obtain the corresponding specific parameters by initializing the global parameters.

Despite the remarkable success of the above methods with few interactions, we argue that there is also a huge gap between meta-learning and recommendation. Specifically, there is a local clustering effect among users, differing from that the users/tasks are independent in the meta-learning model. The specific example is shown in Figure 1. There are 9 users and 10 items, respectively. We first exchange some rows of the left matrix. And

†Corresponding Author.

LREC-COLING 2024, pages 7185–7195

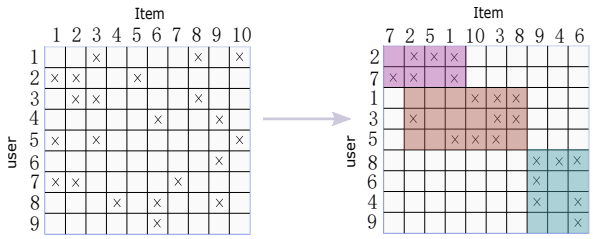

Figure 1: Illustration of user local dependency. $\times$ denotes the consumed/clicked/observed item. The left table is the original interaction matrix, and the right is the equivalent matrix after transformation.

then, some columns are further exchanged. Finally, the right matrix which is equivalent to the origin user-item interaction matrix could be obtained. From the right matrix, we can observe that u_2 and u_7 have similar preferences; u_1, u_3 and u_5 have similar preferences; and u_8, u_6, u_4, and u_9 have similar preferences. In this example, there are three clusters of all users, but there is no intersection between clusters with high probability (especially in a large sparsity matrix). In summary, users are locally dependent, not globally independent of each other. Therefore, if we adopt the meta-learning method into recommendation directly, the global parameters (or meta-parameters) can not be learned effectively. In light of these observations, we urgently need to consider the user's local dependent conditions, expecting more suitable for the recommendation scenarios.

Given the above considerations, we propose a novel Few-shot learning method for Cold-Start (FCS) recommendation. The core idea is the design of a three hierarchical structure for the user's local dependence problem. The FCS is also based on the meta-learning framework that contains the inner loop and outer loop for gradient updating. Specifically, the first hierarchy is global parameters, which aims to learn the meta knowledge from the different cross-users/tasks. The second hierarchy is the local-meta parameters to obtain the adaptive cluster of the target user in accordance with the whole record of the target user via a tree-like structure. After that, the originally similar user or local dependence users have more similar local parameters, addressing the problem of lack of globally shared information discussed above. The third hierarchy is the specific parameters for obtaining the optimal parameters via fine-tuning the corresponding local-meta parameters. Thus, when a new user arrives with few-shot interactions, it can obtain the local parameters quickly, either by taking advantage of the knowledge learned within the cluster it belongs to or initiating with the global parameters as the new cluster if it is wildly different from any existing clusters. Subsequently, we can use the given few-shot samples to obtain the user-specific parameters, aiming to predict unobserved items. It is worth noting that FCS is very suitable for online recommendation due to the rapid update of user parameters with meta parameters.

The main contributions are summarized as follows:

- The few-shot learning is integrated to address the cold-start problem in the recommendation, which can be achieved by the adaptation parameter selection, resulting in personalized modeling.
- A novel three-hierarchical structure is well designed to address the problem of the user's local dependent, which can be adaptive to the sparsity and few-shot scenarios well.
- On public explicit feedback datasets, extensive experimental results demonstrate that FCS produces competitive performances from multiple perspectives.

2. Related Work

This paper focuses on the cold-start recommendation via meta-learning (or few-shot learning) technology. Therefore, the cold-start problem is first discussed. Then, we introduce the background of meta-learning briefly, subsequence summarize the recent work of meta-learning for recommendation.

2.1. Cold-Start Recommendation

Cold-Start recommendation is a significant challenge over the decades, which has attracted huge attention. Generally, there are three directions for current work: data-driven methods (or data augment), model-driven methods, and algorithm-driven methods. The data-driven methods are devoted to integrating various related side-information to learn the priori knowledge. For example, transfer learning-based methods (Hu et al., 2018; Kang et al., 2019; Man et al., 2017; Wang et al., 2019b) assumes that users have similar or related preferences in both domains; content-based methods incorporate the item/user content information (McAuley and Leskovec, 2013; Ling et al., 2014; Amplayo et al., 2018a,b) or social information (Sedhain et al., 2017) into a unified model. However, they rely on the additional information which is unobtainable in some cases due to the privacy protection and business barrier. The model-driven methods aim at obtaining useful information from several similar tasks, e.g., multi-task learning (Wang et al., 2019a) constructs similar tasks and chooses an appropriate sharing network. In essence, both data-driven and model-driven methods can be seen

as a data-augment process. The algorithm-based method is a new direction, which cannot rely on additional data while only improving the optimization process to achieve the few-shot samples learning. The core of these methods is the meta-learning (Zhu et al., 2021; Lin et al., 2021; Wang et al., 2021; Yu et al., 2021), and more details will be described in the following subsection.

2.2. Meta-learning

Meta-learning, learning to learn, is to learn a model that can be fast adapted to previously unseen tasks with few-shot data. In principle, this can be achieved by leveraging knowledge obtained in other related tasks. The few-shot learning aims to learn a model only with a small number of data, which can be seen as a branch of meta-learning or as the validation of meta-learning. There are many representative works of meta-learning (Chen et al., 2019; Finn et al., 2017; Yao et al., 2019; Antoniou et al., 2018; Nichol et al., 2018; Oreshkin et al., 2018; Nichol and Schulman, 2018; Bao et al., 2019; Bertinetto et al., 2018).

The meta-learning method is naturally suitable for the recommendation case. Therefore, using the meta-learning technique to solve the adaptation problem, cold-start, or sparsity problem is attracting more attention in the recommendation community (Luo et al., 2020; Du et al., 2019; Pan et al., 2019; Vartak et al., 2017). Moreover, MAML (Finn et al., 2017) is model-agnostic meta-learning for fast adaptation of deep networks, which has been widely used in recommendations for generalization (Lee et al., 2019; Chen et al., 2018; Luo et al., 2020; Bharadhwaj, 2019; Manqing et al., 2020; Lu et al., 2020). In this paper, we also apply MAML as our base framework.

3. Proposed Method

In this section, we first describe a formal problem definition and specific notations, and then give a detail of the FCS structure. Finally, we give the complete algorithm of the meta-training process and meta-testing process. Furthermore, we discuss the main differences and connections between FCS and related frameworks from the perspective of optimization.

3.1. Problem Formulation and Notations

In this paper, the top-K recommendation problem can be formulated as follows. There is a set of users $\mathcal{U}$ and a set of items $\mathcal{V}$. $\mathcal{N}_u \subseteq \mathcal{V}$ denotes the set of items that user u has previously interacted with, and all user-item interactions are noted as $\mathcal{D} = \{(u, v, v^-) | u \in \mathcal{U} \wedge v \in \mathcal{N}_u \wedge v^- \notin \mathcal{N}_u\}$.

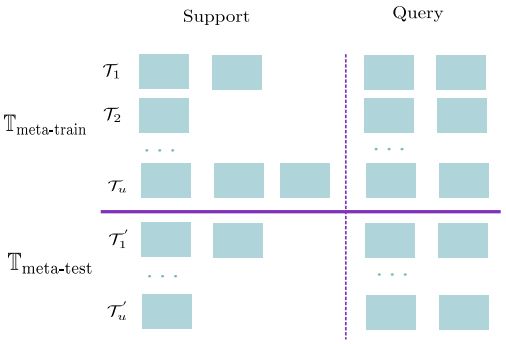

Figure 2: The specific description of dataset.

Given above interaction information, we aim to predict whether user u has potential interest in items $v \in \mathcal{V} \setminus \mathcal{N}_u$ with which he has not interacted before.

For a cold-start situation, each user only has a few historical items, such as $\mathcal{N}_u \subseteq \mathcal{V}$ for user u, where the number $|\mathcal{N}_u| = K$ is few, named few-shot user in this paper. The goal is to recommend the items for K-shot users via fast adaptation.

Similar to the standard few-shot learning setting, we first divide the original user-item interactions into two parts as shown in Figure 2: meta-training set, denoted as $T_{meta-train}$; and meta-testing set, denoted as $T_{meta-test}$, whose scenarios are unseen during meta-training process, i.e. $T_{meta-train} \cap T_{meta-test} =$. Each meta-training task $T_u \in T_{meta-train}$

corresponds to an user u and has the corresponding support D_u^S and query D_u^Q pairs: $\mathcal{T}_u = D_u^S \cup D_u^Q = \{(u, v, v^-) | v \in \mathcal{N}_u \wedge v^- \notin \mathcal{N}_u\}$; D_u^S and D_u^Q are the user u's support set and query set respectively, which are obtained by dividing the user-item interactions in each task/user randomly. The support sets D_u^S are used to optimize the base learner, while query sets D_u^Q are used to optimize the meta learner. Thus, the goal of cold-start recommendation is to learn a well-generalized meta-learner $\mathcal{M}(\cdot)$ from $T_{meta-train}$, and then to fast learn a base learner on new tasks $T_{meta-test}$ which only has a few user-item interaction records. What's more, for clear definition, bold lower-case letters represent vectors for clear definition throughout the rest of this paper.

3.2. Overview of Framework

Our method contains a base-learner and a meta-learner. The base-learner is a specific recommendation network for traditional representation and rank learning. The meta-learner is a three-hierarchical structure, aiming to integrate the tasks/users in a cluster-specific manner and init the base-learner parameters in gradient based on meta-training, so that the optimal parameters of

specific-user can be obtained as fast as possible. Therefore, we first describe the details of the base learner (recommendation network) and then introduce the intermediate layer of the meta-leaner (cluster integrated).

3.3. Recommendation Network

The classical recommendation network contains Logistical Regression, Factorization Machine (Rendle, 2010), Wide&Deep(Cheng et al., 2016), FFM (Juan et al., 2016), DeepFM (Guo et al., 2017). Because our framework is model-agnostic, we could choose an arbitrarily appropriate model to achieve the recommendation network depending on real requirements. Without loss of generality, we employ the classical neural network of MultiLayer Perceptron (MLP) in this paper.

3.3.1. Embedding

The embedding module aims at handling the input features and outputting a general representation for given (u, v). The uniform of the phase could be formulated as:

$$f_\Omega : (u, v) \rightarrow \mathbf{u}, \mathbf{v} \tag{1}$$

In general, features can be divided into two types: categorical features and continuous features. For the continuous feature, which needs to be turned into discrete fragments, e.g., the feature of age becomes four fragments in accordance with the general classification criteria: teenager, youth, middle age, and old age. For the categorical features, we represent them by using the one-hot or binary code, i.e., $\{0, 1\}^N$, where N is the total number. Thus, given an item v, the continnous representation can be denoted as $\mathbf{v_{id}} = \mathbf{e_v}\mathbf{V}$, where $\mathbf{e_v}$ is one-hot vector, and $\mathbf{V}$ is the embedding matrix.

According to the above operation, we can obtain the user representation by concatenating embeddings, denoted as $\mathbf{u} = [\mathbf{u_{type1}}; \mathbf{u_{type2}}; \dots;]$. Similarly, the item representation can be denoted as $\mathbf{v} = [\mathbf{v_{id}}; \mathbf{v_{type1}}; \dots;]$.

3.3.2. Hidden Layer Module

According to above representation of user $\mathbf{u}$ and item $\mathbf{v}$, we can obtain the concatenation vector as $\mathbf{h}_0 = [\mathbf{u}; \mathbf{v}]$. After that, the layers of MLP can be formulated as: $\mathbf{h}_n = g\left(\mathbf{W}_n^T \mathbf{h}_{n-1} + \mathbf{b}_n\right)$ where $\mathbf{W}_n$ denotes the weight matrix, $\mathbf{b}_n$ denotes the bias vector, $g(\cdot)$ is the activation function (such as Relu), and $\sigma(\cdot)$ is the activation function of the sigmoid. For simplicity, the hidden layer module can be described as:

$$h_o = \mathcal{M}\left(u, v; \theta, \Omega\right) = \sigma\left(\mathbf{W}_o^T \mathbf{h}_n + b_o\right) \tag{2}$$

where all parameters of $\mathbf{W}_n$ are denoted as θ.

3.3.3. Loss Function

Given the sample of $< u, v, v^- >$, we can obtain the score of $h_o = \mathcal{M}\left(u, v; \theta\right)$, and $h_{o^-} = \mathcal{M}\left(u, v^-; \theta\right)$, respectively. The loss function could be computed as:

$$\mathcal{L}\left(\mathcal{D}; \Omega, \theta\right) = \sum_{\{u,v,v^-\} \in \mathcal{D}} \log \frac{1}{1 + \exp - \left(h_o - h_{o^-}\right)} \tag{3}$$

3.4. Cluster Integrated

As mentioned above, the task/user is locally independent. Thus, we need to capture the relationships among different users to improve the performance of recommendations in few-shot scenarios. Firstly, we represent each task $\mathcal{T}_u$ by aggregated over representations of all items of $\mathcal{T}_u$, denoted as:

$$\mathbf{s}_{\mathcal{T}_u} = \frac{1}{|\mathcal{N}_u|} \sum_{v \in \mathcal{N}_u} \mathbf{v} \tag{4}$$

Given the representation of a task, we adopt a hierarchical task clustering structure to locate the cluster the task belongs. Specifically, there is a l-layer network, each layer contains several clusters, noted as $k^l, k \in \{1, \dots, K\}$. Note that, the input is the task's representation, and the output is the final representation. It is believed to be cluster-specific (Kim and Xing, 2010), encrypting the hierarchical clustering result. The softmax of two-layer is:

$$p_u^{k^l \rightarrow k^{l+1}} = \frac{\exp\left(-\|\mathbf{h}_u^{k^l} - \mathbf{c}_{k^{l+1}}\|_2^2/2\right)}{\sum_{k^{l+1}=1}^{K^{l+1}} \exp\left(-\|\mathbf{h}_u^{k^l} - \mathbf{c}_{k^{l+1}}\|_2^2/2\right)} \tag{5}$$

where $\mathbf{c}_{k^{l+1}}$ is the k-th central of $l+1$-layer, which is also learnable variable. According to the above softmax, we can compute the representation of the next layer by:

$$\mathbf{h}_u^{k^{l+1}} = \sum_{k^l}^{K^l} p^{k^l \rightarrow k^{l+1}} f\left(\mathbf{W}_u^{k^{l+1}} \mathbf{h}_u^{k^l} + \mathbf{b}_u^{k^{l+1}}\right) \tag{6}$$

where $f(*)$ denotes the activation function, i.e., tanh; $\mathbf{W}_u^{k^{l+1}}$ and $\mathbf{b}_u^{k^{l+1}}$ are the weight matrix and bias vector, respectively.

According to the above cluster integration, we could obtain the final representation of the given task $\mathcal{T}_u$, denoted as $\mathbf{h}_u^L$, whose representation reflects the similarity among tasks. Subsequently, for each task/user $\mathcal{T}_u$, the parameter gate o_k is obtained by fully-connected layer, donoted as:

$$o_k = \sigma\left(\mathbf{W}_f^T \left[\mathbf{h}_u^L; \mathbf{s}_{\mathcal{T}_u}\right] + \mathbf{b}_u^f\right) \tag{7}$$

where $\sigma(*)$ is the sigmoid function, and $\mathbf{W}_f$ and $\mathbf{b}_u^f$ are weight matrix and bias vector, respectively. In order to simplify, we denote all trainable parameters of this module as $\mathcal{W} = \{\mathbf{W}_u^*, \mathbf{W}_f, \mathbf{b}_u^*\}$.

Algorithm 1: Meta-Training

Input: Meta-training set $T_{meta-train}$, loss function $\mathcal{L}$

Output: $\theta, \Omega, \mathcal{W}$

Randomly initialize $\theta, \Omega, \mathcal{W}$

while *not converged* **do**

  Sample a batch of users $\mathcal{B} = \{u_1, u_2, \ldots, u_B\}$ from $\mathcal{U}$

  Construct a batch of training task from $T_{meta-train}$ according to users $\mathcal{B}$: $\mathcal{T} = \{D^S \bigcup D^Q\} = \{D_u^S \bigcup D_u^Q | u \in \mathcal{B}\}$

  for $u \in \mathcal{B}$ **do**

    Compute task representation in Eqn.4

    Compute the gate of the cluster o_k in Eqn.7

    Update parameters of the cluster: $\theta_k = o_k \circ \theta$

    Compute the loss of specific task $\mathcal{L}_u^{train} \leftarrow \mathcal{L}\left(\mathcal{D}_u^S; \theta_k, \Omega, \mathcal{W}\right)$

    Update parameters with gradient descent (taking one step as an example): $\theta_{k,u} \leftarrow \theta_k - \alpha \nabla_{\theta_k} \mathcal{L}_u^{train}$

  end

  Update the meta-parameters: $\theta \leftarrow \theta - \beta \sum_{u \in \mathcal{B}} \nabla_\theta \mathcal{L}\left(D_u^Q; \theta_{k,u}, \Omega, \mathcal{W}\right)$

  Update the embedding parameters: $\Omega \leftarrow \Omega - \beta \sum_{u \in \mathcal{B}} \nabla_\Omega \mathcal{L}\left(D_u^Q; \theta_{k,u}, \Omega, \mathcal{W}\right)$

  Update the cluster parameters: $\mathcal{W} \leftarrow \mathcal{W} - \beta \sum_{u \in \mathcal{B}} \nabla_\mathcal{W} \mathcal{L}\left(D_u^Q; \theta_{k,u}, \Omega, \mathcal{W}\right)$

end

3.5. Meta-training

The process of meta-training is shown in Algorithm 1. Given the training data $\mathcal{T}_{meta-train}$, our goal is to learn the meta-parameters. Concretely, we sample a batch of users from $\mathcal{U}$ and then construct support set D^S and a query set D^Q. The support set is used to inner-loop for specific tasks, and the query set is used to out-loop across different tasks.

In the inner loop, we first compute each target task-specific representation using the the record of user in equation 4. Then, according to the cluster integration tree network, we can obtain the gate of task θ_k. Finally, we use θ_k to update the user-specific parameters $\theta_{k,u}$ by gradient descent. It is worth noting that this step can be updated to multiple steps.

In the outer loop, we use the query set data and learned parameters $\theta_{k,u}$ to compute the loss and then update the meta-parameters θ. In this phase, other global parameters are also updated.

3.6. Meta-Testing

For the cold-start users with few-shot interactions, we also construct the support set and query set first.

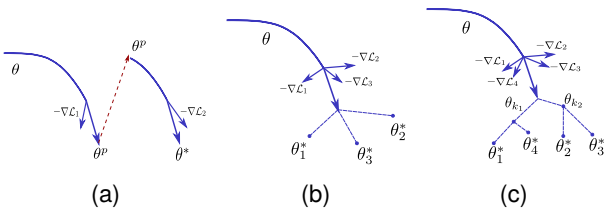

Figure 3: An intuitive illustration of recommendation framework.

And then, we use the global parameters as initialization, and then compute the user's cluster by the support dataset, to obtain the specific parameters θ_k; finally, we update the $\theta_{k,u}$ in accordance with gradient descent. For the query dataset, we compute their prediction score by $f\left(\mathcal{D}_u^Q; \theta_{k,u}, \Omega, \mathcal{W}\right)$ directly, subsequently rank the score and select the top items for recommendation.

Note that FCS can not only be trained with offline data set, but also be trained online with minor modifications by using the emerging cold-start user as the training samples.

3.7. Discussion of Related Framework

We re-examine two types of related frameworks, i.e., transfer learning-based framework and transfer learning-based framework, and highlight the significance of our proposed FCS through versatile comparisons.

1. Transfer learning framework is widely used in the cross-domain recommendation, which aims to learn the prior parameters from the source domain and then translate them into the target domain, where the user-item interactions are few-shot. As shown in Figure 3(a), its prior parameters θ^P are learned in the source domain via stochastic gradient descent with $-\nabla\mathcal{L}_1$; and then, the learned parameters will be translated to target domain for fine-tuning according to $-\nabla\mathcal{L}_2$. In particular, domain-adaptive is a special case of transfer learning. Although the prior knowledge is useful for initialization, the cross-domain task is not always available in the real world due to the problem of business barriers and user privacy protection. What's more, this type of method could not realize the model selection for different users.

2. MAML is a meta-learning method, aiming to learn the learning ability of the model, which is widely used in few-shot settings. Thus, MAML is appropriate for a cold-start scenario. As shown in Figure 3(b), it could learn the user/task-specific parameters θ_u^*, and obtain the new task parameters quickly in accordance with the meta-parameters. We know that it

doesn't request the auxiliary knowledge, and achieving the personalization learning. In the other perspective, the meta-learning framework is also a special case of transfer learning. The most important advantage is that MAML uses the support set to optimize the specific parameters, and uses the query set to optimize meta-parameters, making meta-parameters more generalized.

3. FCS is our method which is similar to MAML. As shown in Figure 3(c), each task will be first assigned to its cluster and obtain a local-meta parameter $\theta_{k,u}$. This step is tailored for recommender systems, considering the local dependence of users.

4. Experiments

In this section, we first describe the experimental settings, including datasets, evaluation protocols, baselines, and implementation details. Subsequently, we conduct extensive experiments to respond to the following research questions:

RQ1 How is the effectiveness of FCS? Can it provide a competitive performance compared with baselines?

RQ2 How does the local update step affect the performance?

RQ3 What is the performance of different ratio of meta-training datasets?

RQ4 What is the visualization of the user cluster?

4.1. Datasets

We perform experiments on two publicly accessible datasets: MovieLens (Harper and Konstan, 2016) and Bookcrossing (Ziegler et al., 2005), which have been brought into widely adopted in previous literature. For the MovieLens dataset, the user's features include gender, age, occupation, and zip-code, item's features contain itemID, and genre. For the Bookcrossing dataset, the user's features include province and country; and the item's features contain itemID, publisher, and publication year. Note that, the goal is to recommend the existing items to existing users. Thus, we only use the itemID, without userID as the new userID is unknown.

4.2. Evaluation protocols

To evaluate the recommendation performance in the cold-start scenario, we divide the original dataset into three parts randomly: meta-training, meta-validation, and meta-testing. The validation dataset is used to choose appropriate hyper-parameters. Without special instructions, the ratio of the three parts is set to 70%:10%:20%. The support set of each user contains 1 or 5 items that the target user has interacted with before, called the 1-shot or 5-shot setting.

Due to the enormous cost of ranking all items for each user, we sample 500 items that have no interaction with the target user, following previous work (Park et al., 2018; Tay et al., 2018; He et al., 2018). To evaluate the ranking accuracy and quality, we adopt two widely used metrics (Zhang et al., 2018; Wang et al., 2019a): Hit Ratio (HR@5 and HR@10), and Normalized Discounted Cumulative Gain (N@5 and N@10).

4.3. Baselines

There are many classical feature-based methods, in this paper, we only report the results on **Wide** and **Wide&Deep** (Cheng et al., 2016), because 1) there are closed with our recommendation network(MLP); 2) they represent the shallow method and deep method, respectively. Without loss of generality, we can also apply various existing models for further improvement due to the proposed FCS is model-agnostic. What's more, the existing meta-learning-based methods for recommendation are also various, we select the closed methods of **MAMO** (Manqing et al., 2020), **MetaEmb** (Pan et al., 2019), **MeLU** (Lee et al., 2019), and **AT-PAML** (Yu et al., 2021) as our strong baselines.

4.4. Implementation Details

We implement our model in TensorFlow. The implementation of the comparison methods is from the public codes that the authors provided in their papers or by modifying the part module in some open source project (Wide&Deep[1], MeLU[2], MAML[3], lstm-tree[4] and tutorial[5]).

We optimize the proposed FCS with the SGD optimizer and tune the learning rate of (α and β) in {0.01, 0.001, 0.0001} for different scenario settings. The embedding size is fixed to 30. The batch size is varied from 64 to 128. The number of local updates is varied from two to four. The maximum number of epochs is set to 100. The settings of Melu and AT-PAML are the same as FCS. For the Wide&Deep and Wide, the batch size is set to 128 and the negative sample is set to 30. All weight variables, i.e., θ, Ω, and $\mathcal{W}$, are initialized with uniform distributions of $[-0.01, 0.01]$ randomly.

4.5. Experimental Results

In this subsection, we will report the comparison results to answer the above questions in detail and

[1] https://github.com/cheungdaven/DeepRec
[2] http://github.com/hoyeoplee/MeLU
[3] https://github.com/cbfinn/maml
[4] https://github.com/stanfordnlp/treelstm
[5] https://github.com/AntreasAntoniou/HowToTrainYourMAML

Table 1: Experimental results on two public datasets in various scenarios (1-shot and 5-shot) using metrics such as HR and NDCG. The bolded values indicate the optimal values; the underlined values denote the suboptimal values.

Methods	MovieLens				Bookcrossing			
(**1-shot**)	HR@5	HR@10	N@5	N@10	HR@5	HR@10	N@5	N@10
Wide	0.3902	0.4584	0.4457	0.4839	0.5441	0.5733	0.6633	0.6803
Wide&Deep	0.4081	0.4800	0.5017	0.5419	0.5953	0.6228	0.7108	0.7262
MetaEmb	0.8524	0.8775	0.8846	0.9075	0.7872	0.8151	0.8517	0.8671
MAMO	0.8761	0.9090	0.9017	0.9449	0.7802	0.7931	0.8426	0.8561
AT-PAML	0.8890	0.9165	0.9183	0.9500	0.7666	0.7824	0.8386	0.8547
MeLU	0.8854	0.9169	0.9194	**0.9545**	0.7705	0.7852	0.8385	0.8464
FCS	**0.8944**	**0.9180**	**0.9384**	0.9402	**0.8068**	**0.8377**	**0.8657**	**0.8826**
Methods	MovieLens				Bookcrossing			
(**5-shot**)	HR@5	HR@10	N@5	N@10	HR@5	HR@10	N@5	N@10
Wide	0.4336	0.4993	0.5109	0.5479	0.5726	0.6169	0.6937	0.7188
Wide&Deep	0.4791	0.5546	0.5481	0.5904	0.6098	0.6499	0.7207	0.7437
MetaEmb	0.9237	**0.9535**	0.9478	0.9647	0.8347	0.8576	0.8828	0.8956
MAMO	0.9332	0.9469	0.9587	0.9689	0.7911	0.8209	0.8542	0.8701
AT-PAML	0.9345	0.9438	0.9576	0.9660	0.8054	0.8301	0.8704	0.8792
MeLU	**0.9411**	0.9454	**0.9607**	0.9631	0.8074	0.8283	0.8655	0.8770
FCS	0.9405	0.9502	0.9595	**0.9737**	**0.8589**	**0.8986**	**0.8901**	**0.9125**

summarize some insights.

4.5.1. Performance Comparison (RQ1)

The performance comparison results on two datasets concerning top-k metric are shown in Table 1.

We can observe that the proposed method FCS, achieves the best performance on two datasets with all evaluation metrics, which illustrates the superiority and effectiveness. More concretely, Compared to the Wide&Deep and MeLU, we can see that MeLU performs better on two datasets, which demonstrates the benefits of the appropriate optimization. Furthermore, compared to MeLU and FCS, we can observe that FCS makes a stable margin improvement, which further demonstrates the important rule of user local dependencies.

Note that the relative improvements of FCS vary among two datasets and different few-shot settings. For MovieLen dataset, there is a relatively small improvement to baselines, while for the Bookcrossing dataset, there is a larger margin gain than the traditional baselines and MeLU. It can be explained that the Bookcrossing dataset is more sparsity than MovieLens, containing more users and items than MovieLens. It also implies that the features of items are more widely distributed, resulting in less shared information across different tasks. There is an interesting phenomenon that FCS in a 1-shot setting still shows significant improvement results compared to MeLU. In the 1-shot setting, the representation of the task is equal to the single item. Thus, MeLU only uses a single item to calculate specific parameters, resulting in a worse performance. However, the cluster-integrated module of FCS helps to make full use of the related samples (including similar users and similar items), extending the samples for specific parameter learning. Therefore, FCS has the advantage of alleviating the cold-start problem.

4.5.2. Impact of local update step (RQ2)

To investigate the impact of the local update step on recommendation performance, we conducted several experiments on the Bookcrossing dataset with the 5-shot setting, and the results are shown in Figure 5. The step varies from {2,3,4}. We can observe that a small step in the inner loop is not sufficient to determine the optimal value. By increasing the step, the model has more capacity to obtain the more appropriate specific parameters. However, according to MAML(Antoniou et al., 2018; Finn et al., 2017; Nichol and Schulman, 2018), with the increase of step, it suffers from two issues:

- we need to store more parameters, which imposes a considerable computation and memory burden.
- The meta-parameters will shrink and vanish as the number of gradient steps grows, making

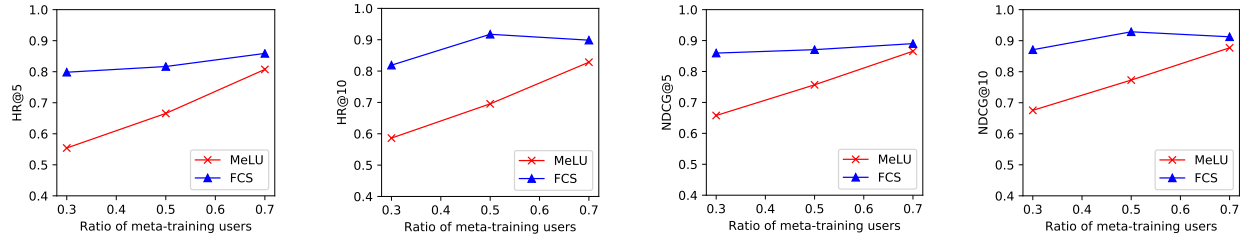

Figure 4: Impact of different ratios of users in meta-training.

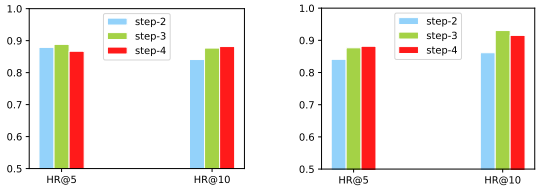

Figure 5: The performance comparisons of different local update steps.

meta-learning difficult. Therefore, we need to trade off the recommendation performance and computation efficiency.

4.5.3. Effect of cold-start users (RQ3)

To investigate the performance of FCS on different ratios of meta-training datasets, we extend several comparison experiments on the Bookcrossing dataset under 5-shot settings. We select $r\%$, 10%, and $(90-r)\%$ of all users to build the meta-training tasks, meta-validation tasks, and meta-testing tasks, respectively. r is varied from $\{30, 50, 70\}$. The results are shown in Figure 4.

We find the FCS achieves an excellent performance compared with MeLU on different ratios of users in terms of all top-n ranking metrics. In general, the performance of both MeLU and FCS improves as the proportion of meta-training tasks increases, which illustrates the importance of labeled data. The results are also consistent with meta-learning assumptions, i.e., the more tasks given in the meta-training phase, the more sufficient priori knowledge can be learned. Another interesting finding is that the lifting rate of the two methods is different. The lifting rate of MeLU is faster and the curve is steeper, while for FCS that is slower and its curve is more gentler. These results show that FCS is more robust and performs well under a few tasks.

4.5.4. Visualization of user cluster (RQ4)

In the section of the introduction, we discuss that there is a local clustering effect of users in original data distribution. In this subsection, we will visualize the clusters of users learned by FCS to further provide a more intuitive understanding. To this end, we first restore the intermediate results of the integrated target task's representation, i.e., $\left[\mathbf{h}_u^L; \mathbf{s}_{\mathcal{T}_u}\right]$, on MovieLens and Bookcrossing datasets under 1-shot setting. Then, we use the t-SNE (van der Maaten and Hinton, 2008) tools to visualize the representation vectors, results are shown in Figure 6. We can observe clearly that the representation vectors of users/tasks are generally grouped into several clusters on two datasets. This supports our assumption that users are locally dependent. Therefore, FCS captures the features of user-local dependent relationships, yielding a significant improvement in the few-shot scenarios.

5. Conclusion

In this paper, we propose a novel Few-shot learning method for Cold-Start (FCS) recommendation. Firstly, we argue that the cold-start recommendation is a typical few-shot learning problem, where each user has few records and there are also many new users arriving. Secondly, considering the effect of user local clusters, we design a novel three hierarchical structure, i.e., global-meta parameters, local-meta parameters, and specific parameters, which could alleviate the problem of user locally dependent well. It is worth noting that FCS is very suitable for online recommendation due to the rapid updating of user parameters with meta-parameters in meta-testing phases. What's more, FCS is model-agnostic, which implies that it is easy to extend by replacing the existing recommendation module. Extensive experiments conducted on two public real-world explicit feedback datasets demonstrate that the proposed method produces significant improvements compared with state-of-the-art methods from multiple perspectives. For the future work, this insight has extremely high scalability for other related tasks, such as representation learning.

References

(a) MovieLens (b) Bookcrossing

Figure 6: The visualization of the user/task representation on two datasets.

Reinald Kim Amplayo, Jihyeok Kim, Sua Sung, and Seung-won Hwang. 2018a. Cold-start aware user and product attention for sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2535–2544.

Reinald Kim Amplayo, Jihyeok Kim, Sua Sung, and Seung-won Hwang. 2018b. Cold-start aware user and product attention for sentiment classification. *arXiv preprint arXiv:1806.05507*.

Antreas Antoniou, Harrison Edwards, and Amos Storkey. 2018. How to train your maml. *arXiv preprint arXiv:1810.09502*.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2019. Few-shot text classification with distributional signatures. *arXiv preprint arXiv:1908.06039*.

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. 2018. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*.

H. Bharadhwaj. 2019. Meta-learning for user cold-start recommendation. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Fei Chen, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. 2018. Federated meta-learning for recommendation. *arXiv preprint arXiv:1802.07876*.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. 2019. A closer look at few-shot classification. *CoRR*, abs/1904.04232.

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*, pages 7–10. ACM.

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*, pages 2895–2904.

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-Volume 70*, pages 1126–1135. JMLR. org.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: A factorization-machine based neural network for ctr prediction. *arXiv: Information Retrieval*.

F. Maxwell Harper and Joseph A. Konstan. 2016. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19.

Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. Nais: Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 30(12):2354–2366.

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,*

WWW 2017, Perth, Australia, April 3-7, 2017, pages 173–182.

Guangneng Hu, Yu Zhang, and Qiang Yang. 2018. Conet: Collaborative cross networks for cross-domain recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 667–676. ACM.

Yuchin Juan, Yong Zhuang, Weisheng Chin, and Chihjen Lin. 2016. Field-aware factorization machines for ctr prediction. pages 43–50.

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, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 1563–1572. ACM.

Seyoung Kim and Eric P Xing. 2010. Tree-guided group lasso for multi-task regression with structured sparsity. *international conference on machine learning*, pages 543–550.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8).

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*, pages 1073–1082. ACM.

Pan Li and Alexander Tuzhilin. 2020. DDTCDR: deep dual transfer cross domain recommendation. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 331–339. ACM.

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*, pages 1306–1316.

Guang Ling, Michael R Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 105–112. ACM.

Yuanfu Lu, Yuan Fang, and Chuan Shi. 2020. Meta-learning on heterogeneous information networks for cold-start recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1563–1573. ACM.

Mi Luo, Fei Chen, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Jiashi Feng, and Zhenguo Li. 2020. Metaselector: Meta-learning for recommendation with user-level adaptive model selection. In *Proceedings of The Web Conference 2020*, pages 2507–2513.

Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-domain recommendation: An embedding and mapping approach. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2464–2470.

Dong Manqing, Yuan Feng, Yao Lina, Xu Xiwei, and Zhu Liming. 2020. Mamo: Memory-augmented meta-optimization for cold-start recommendation. In *26th SIGKDD Conference on Knowledge Discovery and Data Mining*.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM.

Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264.

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

Alex Nichol and John Schulman. 2018. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2.

Boris N Oreshkin, Pau Rodriguez Lopez, and Alexandre Lacoste. 2018. Tadam: Task dependent adaptive metric for improved few-shot learning. *neural information processing systems*, pages 721–731.

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*, pages 695–704.

Chanyoung Park, Donghyun Kim, Xing Xie, and Hwanjo Yu. 2018. Collaborative translational metric learning. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 367–376. IEEE.

Steffen Rendle. 2010. Factorization machines. pages 995–1000.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press.

Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. 2017. Low-rank linear cold-start recommendation from social data. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1502–1508. AAAI Press.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based a ention for collaborative ranking. In *Proceedings of the 27th International Conference on World Wide Web*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

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*, pages 6904–6914.

Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019a. Multi-task feature learning for knowledge graph enhanced recommendation. *arXiv preprint arXiv:1901.08907*.

Li Wang, Binbin Jin, Zhenya Huang, Hongke Zhao, Defu Lian, Qi Liu, and Enhong Chen. 2021. Preference-adaptive meta-learning for cold-start recommendation. In *IJCAI*, pages 1607–1614.

Yaqing Wang, Chunyan Feng, Caili Guo, Yunfei Chu, and Jenq-Neng Hwang. 2019b. Solving the sparsity problem in recommendations via cross-domain item embedding based on co-clustering. pages 717–725.

Huaxiu Yao, Ying Wei, Junzhou Huang, and Zhenhui Li. 2019. Hierarchically structured meta-learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7045–7054, Long Beach, California, USA. PMLR.

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*, volume 35, pages 10772–10780.

Shuai Zhang, Lina Yao, Chaoran Huang, Xiwei Xu, and Liming Zhu. 2018. Position and distance: Recommendation beyond matrix factorization. *CoRR*, abs/1802.04606.

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*, pages 1167–1176.

Cainicolas Ziegler, Sean M Mcnee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. *the web conference*, pages 22–32.