# Few-Shot Representation Learning for Cold-Start Users and Items

Bowen Hao[1,2], Jing Zhang[1,2](✉), Cuiping Li[1,2], and Hong Chen[1,2]

[1] Key Laboratory of Data Engineering and Knowledge Engineering
of Ministry of Education, Renmin University of China, Beijing, China
{jeremyhao,zhang-jing,licuiping,chong}@ruc.edu.cn
[2] School of Information, Renmin University of China, Beijing, China

**Abstract.** Existing recommendation algorithms suffer from cold-start issues as it is challenging to learn accurate representations of cold-start users and items. In this paper, we formulate learning the representations of cold-start users and items as a few-shot learning task, and address it by training a representation function to predict the target user (item) embeddings based on limited training instances. Specifically, we propose a novel attention-based encoder serving as the neural function, with which the $K$ training instances of a user (item) are viewed as the interactive context information to be further encoded and aggregated. Experiments show that our proposed method significantly outperforms existing baselines in predicting the representations of the cold-start users and items, and improves several downstream tasks where the embeddings of users and items are used.

**Keywords:** Cold-start representation learning · Few-shot learning · Attention-based encoder

## 1 Introduction

Existing recommendation systems (RS) such as Matrix Factorization [14] and Neural Collaborative Filtering [11] are facing serious challenges when making cold-start recommendations, i.e., when dealing with a new user or item with few interactions for which the representation of the user or the item can not be learned well.

To deal with such cold-start challenges, some researches are conducted which can be roughly classified into two categories. The first category incorporates side information such as knowledge graph (KG) to alleviate the cold-start issues [3,25,26,28]. Specifically, these methods first pre-process a KG by some knowledge graph embedding methods such as TransE [1], TransH [27] and so on, and then use the entities' embeddings from KG to enhance the corresponding items' representations. For instance, Zhang et al. [28] learn item representations by combining their embeddings in the user-item graph and the KG. Cao et al. [3] and Wang et al. [25] jointly optimize the recommendation and KG embedding

X. Wang et al. (Eds.): APWeb-WAIM 2020, LNCS 12317, pp. 363–377, 2020.
https://doi.org/10.1007/978-3-030-60259-8_27

tasks in a multi-task learning setting via sharing item representations. However, existing KGs are far from complete and it is not easy to link some items to the existing entities in KG due to the missing entities in KG or the ambiguous issues.

The second category uses meta learning [2] to solve the cold-start issues. The goal of meta learning is to design a meta-learner that can efficiently learn the meta information and can rapidly adapt to new instances. For example, Vartak et al. [21] propose to learn a neural network to solve the user cold-start problem in the Tweet recommendation. Specifically, the neural network takes items from user's history and outputs a score function to apply to new items. Du et al. [4] propose a scenario-specific meta learner framework, which first trains a basic recommender, and then tunes the recommendation system according to different scenarios. Pan et al. [16] propose to learn an embedding generator for new ads by making use of previously learned ads' features (e.g., the attributes of ads, the user profiles and the contextual information) through gradient-based meta-learning.

All these KG-based and meta learning based methods aim to directly learn a powerful recommendation model. Different from these methods, in this paper, we focus on how to learn the representations of the cold-start users and items. We argue that the high-quality representations can not only improve the recommendation task, but also benefit several classification tasks such as user profiling classification, item classification and so on (which is justified in our experiments). Motivated by the recently proposed inductive learning technique [7,23], which learns node representations by performing an aggregator function over each node and its fixed-size neighbours, in this paper, we aim to learn the high-quality representations of the cold-start users and items in an inductive manner. Specifically, we view the items that a target user interacts with as his/her contextual information and view the users that a target item interacts with as its contextual information. We then propose an attention-based context encoder (AE), which adopts either soft-attention or multi-head self-attention to integrate the contextual information to estimate the target user (item) embeddings.

In order to obtain a AE model to effectively predict the cold-start user and item embeddings from just a few interactions, we formulate the cold-start representation learning as a few-shot learning task. In each episode, we suppose a user (item) which has enough interactions with items (users) as the target object to predict. Then AE is asked to predict this target object using only $K$ contextual information, i.e., for each target user, AE is asked to use $K$ interacted items to predict his/her representation, while for each target item, AE is asked to use $K$ interacted users to predict the representation of the target item. This training scheme can simulate the real scenarios where there are cold-start users or cold-start items which only have a few interactions.

We conduct several experiments based on both intrinsic and extrinsic embedding evaluation. The intrinsic experiment is to evaluate the quality of the learned embeddings of the cold-start users and items, while the extrinsic experiments are three downstream tasks that the learned embeddings are used as inputs. Experiments results show that our proposed AE can not only outperform the baselines in the intrinsic evaluation task, but also benefit several extrinsic evaluation tasks such as personalized recommendation, user classification and item classification.

Our contributions can be summarized as: (1) we formulate the cold-start representation learning task as a $K$-shot learning problem and propose a simulated episode-based training schema to predict the target user or item embeddings. (2) We propose an attention-based context encoder which can encode the contextual information of each user or each item. (3) Experiments on both intrinsic and extrinsic embedding evaluation tasks demonstrate that our proposed method is capable of learning the representations of cold-start users and items, and can benefit the downstream tasks compared with the state-of-the-art baselines.

## 2 Approach

In this section, we first formalize learning the representations of cold-start users and cold-start items as two separated few-shot learning tasks. We then present our proposed attention-based encoder (AE) in solving both these two tasks.

### 2.1 Few-Shot Learning Framework

**Problem Formulation.** Let $U = \{u_1, \cdots, u_{|U|}\}$ be a set of users and $I = \{i_1, \cdots, i_{|I|}\}$ be a set of items. $I_u$ denotes the item set that the user $u$ has selected. $U_i$ denotes the user set in which each user $u \in U_i$ selects the item $i$. Let $M$ be the whole dataset that consists of all the $(u, i)$ pairs.

**Problem 1: Cold-Start User Embedding Inference.** Let $D_T^{(u)} = \{(u_k, i_k)_{k=1}^{|T^u|}\}$ be a meta-training set, where $i_k \in I_{u_k}$, $|T^u|$ denotes the number of users in $D_T^{(u)}$. Given $D_T^{(u)}$ and a recommendation algorithm[1] (e.g., Matrix factorization) that yields a pre-trained embedding for each user and item, denoted as $e_u \in \mathbf{R}^d$ and $e_i \in \mathbf{R}^d$. Our goal is to infer embeddings for cold-start users that are not observed in the meta-training set $D_T^{(u)}$ based on a new meta-test set $D_N^{(u)} = \{(u'_k, i'_k)_{k=1}^{|N^u|}\}$, where $i'_k \in I_{u'_k}$, $|N^u|$ denotes the number of users in the meta-test set $D_N^{(u)}$.

**Problem 2: Cold-Start Item Embedding Inference.** Let $D_T^{(i)} = \{(i_k, u_k)_{k=1}^{|T^i|}\}$ be a meta-training set, where $u_k \in U_{i_k}$, $|T^i|$ denotes the number of items in $D_T^{(i)}$. Given $D_T^{(i)}$ and a recommendation algorithm that yields a pre-trained embedding for each user and item, denoted as $e_u \in \mathbf{R}^d$ and $e_i \in \mathbf{R}^d$. Our goal is to infer embeddings for cold-start items that are not observed in the meta-training set $D_T^{(i)}$ based on a new meta-test set $D_N^{(i)} = \{(i'_k, u'_k)_{k=1}^{|N^i|}\}$, where $u'_k \in U_{i'_k}$, $|N^i|$ denotes the number of items in $D_N^{(i)}$.

[1] We also select some node embedding methods (e.g., DeepWalk [17], LINE [20]) which accept user-item bipartite graph as input and output a pre-trained embedding for each user and item.

Note that these two tasks are symmetrical and the difference between these two tasks is that the roles of users and items are swapped. For simplicity, we present the cold-start user embedding inference scenario, and the cold-start item embedding inference scenario is similar to the cold-start user embedding inference scenario if we simply change the role of the users and items. In the following parts, we omit the subscript and simply use $D_T$ and $D_N$ to denote the meta-training set and meta-test set in both two tasks.

For the cold-start user embedding inference task, $D_N$ is usually much smaller than $D_T$, and the cold-start users in $D_N$ only have selected a few items, i.e., there are few $(u'_k, i'_k)$ pairs in $D_N$. Thus it is difficult to directly learn the user embedding from $D_N$. Our solution is to learn a neural model $f_\theta$ parameterized with $\theta$ on $D_T$. The function $f_\theta$ takes the item set $I_u$ of user $u$ as input, and outputs the predictive user embedding $\hat{e}_u$. The predictive user embedding is expected to be close to its target embedding. Note that the user in $D_T$ has enough interactions, thus the pre-trained embedding $e_u$ is convincing and we view it as the target embedding.

In order to mimic the real scenarios that the cold-start users only have interacted with few items, we formalize the training of the neural model as a few-shot learning framework, where the model is asked to predict cold-start user embedding with just a few interacted items. To train the neural function $f_\theta$, inspired by [24], we form episodes of few-shot learning tasks. In the cold-start user inference task, in each episode, for each user $u_j$, we randomly sample $K$ items from $I_{u_j}$ and construct a positive support set $\mathbf{S}^K_{u_j^+} = \{i_{u_j^+,k}\}_{k=1}^K$, where $i_{u_j^+,k}$ is sampled from $I_{u_j}$ and denotes the $k$-th sampled item for the target user $u_j$. We also randomly sample $K$ negative items and construct a negative support set $\mathbf{S}^K_{u_j^-} = \{i_{u_j^-,k}\}_{k=1}^K$, where each item $i_{u_j^-,k}$ is not in $I_{u_j}$. Based on the sampled items, the model $f_\theta$ is expected to predict more similar embedding to the target user embedding when given $\mathbf{S}^K_{u_j^+}$ and more dissimilar embedding when given $\mathbf{S}^K_{u_j^-}$. We use cosine similarity to indicate whether the predicted embedding is similar to the target embedding. To further optimize the neural model $f_\theta$, we minimize the regularized log loss defined as follows [10]:

$$L = -\frac{1}{|T_u|}\sum_{j=1}^{|T_u|}(\log(\sigma(\hat{y}_{u_j^+})) + \log(1-\sigma(\hat{y}_{u_j^-}))) + \lambda||\theta||^2, \tag{1}$$

where $\hat{y}_{u_j^+} = \cos(f_\theta(\mathbf{S}^K_{u_j^+}), u_j)$, $\hat{y}_{u_j^-} = \cos(f_\theta(\mathbf{S}^K_{u_j^-}), u_j)$, $\theta$ denotes the parameters of the proposed model $f_\theta$, $\sigma$ is a sigmoid function, the hyper-parameter $\lambda$ controls the strength of $L_2$ regularization to prevent overfitting. Once the model $f_\theta$ is trained based on $D_T$, it can be used to predict the embedding of each cold-start user $u'$ in $D_N$ by taking the item set $I'_u$ as input. Similarly, we can also design another neural model $g_\phi$ to learn the representations of cold-start items. Specifically, $g_\phi$ can be trained on $D_T$, and can be used to predict the embedding of each cold-start item $i'$ in $D_N$ by taking the user set $U'_i$ as input.

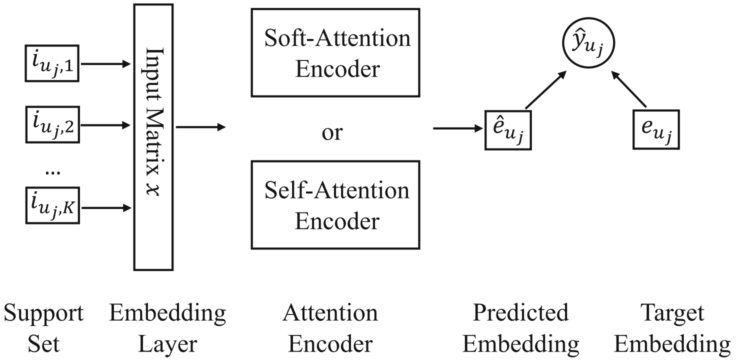


**Fig. 1.** The proposed attention-based encoder $f_\theta$ framework. $g_\phi$ is similar to $f_\theta$ if we simply swap the role of the users and items.

### 2.2 Attention-Based Representation Encoder

In this section, we detail the architecture of the proposed neural model $f_\theta$ ($g_\phi$ is similar if we simply swap the role of the users and items). For the cold-start user embedding inference task, the key idea is to view the items that a user has selected as his/her contextual information, and we expect $f_\theta$ to be able to analyze the semantics of the contextual information, to aggregate these items for predicting the target user embedding. Using AE as $f_\theta$, a more sophisticated model to process and aggregate contextual information can be learned to infer target user embedding.

**Embedding Layer.** As mentioned before, we first train a recommendation (node embedding) algorithm on the whole dataset $M$ to obtain the pre-trained embeddings $e_u$ and $e_i$. Note that we view $e_i$ as contextual information, and $e_u$ in $D_T$ as target user embedding. Both $e_u$ and $e_i$ are fixed. Given each target user $u_j$ and the support set $\mathbf{S}_{u_j}^K = \{\mathbf{S}_{u_j^+}^K \cup \mathbf{S}_{u_j^-}^K\}$, we map the support set $\mathbf{S}_{u_j}^K$ to the input matrix $x^{K \times d} = [e_{i_1}, \cdots, e_{i_K}]$ using the pre-trained embeddings, where $K$ is the number of interacted items, $d$ is the dimension of pre-trained embeddings. The input matrix is further fed into the aggregation encoder.

**Aggregation Encoder.** We present two types of aggregation encoder, namely soft-attention encoder and self-attention encoder.

**(1) Soft-attention Encoder.** Inspired by [10] that uses soft-attention mechanism to distinguish which historical items in a user profile are more important to a target user, in this paper, we first calculate the attention score between the target user embedding $e_{u_j}$ and each item embedding $e_{i_k}$ that he/she has selected, then we use weighted average items' embeddings to represent the predicted user embedding $\hat{e}_{u_j}$:

$$a_{u_j i_k} = \frac{\exp(r(e_{u_j}, e_{i_k}))}{\sum_{k'=1}^{K} \exp(r(e_{u_j}, e_{i_{k'}}))}, \tag{2}$$

$$r(e_{u_j}, e_{i_k}) = W_1^T \text{RELU}(W_2(e_{u_j} \odot e_{i_k})), \tag{3}$$

$$\hat{e}_{u_j} = \frac{1}{K} \sum_{k=1}^{K} a_{u_j i_k} e_{i_k}, \tag{4}$$

where $r$ is soft-attention neural function that has the element-wise operation $\odot$ between the two vectors $e_{u_j}$ and $e_{i_k}$, $W_1 \in \mathbf{R}^{d \times 1}$, $W_2 \in \mathbf{R}^{d \times d}$ are two weight matrices, RELU is an activate function, $K$ is the number of interacted items.

**(2) Self-attention Encoder.** Same as [22], our self-attention encoder consists of several encoding blocks. Each encoding block consists of a self-attention layer and a fully connected layer. Using such encoding blocks can enrich the interactions of the input items to better predict the target user embedding.

Self-attention layer consists of several multi-head attention units. For each head unit $h$, we view the input matrix $x$ into query, key and value matrices. Then linear projections are performed to map the query, key, value matrices to a common space by three parameters matrices $W_h^Q$, $W_h^K$, $W_h^V$. Next we calculate the matrix product $xW_h^Q(xW_h^K)^T$ and scale it by the square root of the dimension of the input matrix $\frac{1}{\sqrt{d_x}}$ to get mutual attention matrix. We further multiply the attention matrix by the value matrix $xW_h^V$ to get the self attention vector $a_{self,h}$ for head $h$:

$$a_{self,h} = \text{softmax}(\frac{xW_h^Q(xW_h^K)^T}{\frac{1}{\sqrt{d_x}}})xW_h^V. \tag{5}$$

We concatenate all the self attention vectors $\{a_{self,h}\}_{h=1}^{H}$ and use a linear projection $W^O$ to get the self-attention output vector $SA(x)$, where $H$ is the number of heads. Note that $SA(x)$ can represent fruitful relationships of the input matrix $x$, which has more powerful representations:

$$SA(x) = \text{Concat}(a_{self,1}, \cdots, a_{self,H})W^O. \tag{6}$$

A fully connected feed-forward network (FFN) is performed to accept $SA(x)$ as input and applies a non-linear transformation to each position of the input matrix $x$. In order to get higher convergence and better generalization, we apply residual connection [9] and layer normalization [13] in both self-attention layer and fully connected layer. Besides, we do not incorporate any position information as the items in the support set $\mathbf{S}_{u_j}^K$ have no sequential dependency. After averaging the encoded embeddings in the final FFN layer, we can obtain the predicted user embedding $\hat{e}_{u_j}$.

Given the target user embedding $e_{u_j}$ and the predicted user embedding $\hat{e}_{u_j}$, the regularized log loss are performed to train AE (Eq. 1). For the self-attention model, the parameters $\theta = [\{(W_h^Q, W_h^K, W_h^V)\}_{h=1}^H, \{(w_l, b_l)\}_{l=1}^H, W^O]$, where $w_l$, $b_l$ are the weights matrix and bias in the $l$-th FFN layer, for the soft-attention model, the parameters $\theta = [W_1, W_2]$. Figure 1 illustrates the proposed model $f_\theta$.

# 3 Experiment

In this section, we present two types of experiments to evaluate the quality of embeddings resulted by the proposed AE model. One is an intrinsic evaluation which involves two tasks: cold-start user inference task and cold-start item inference task. The other one is an extrinsic evaluation on three downstream tasks: (1) Personalized recommendation, (2) Item classification and (3) User classification.

**Table 1.** Statistics of the datasets.

| Dataset | #Users | #Items | #Interactions | #Sparse Ratio |
|---|---|---|---|---|
| MovieLens-1M | 6,040 | 3,706 | 1,000,209 | 4.47% |
| Pinterest | 55,187 | 9,916 | 1,500,809 | 0.27% |

## 3.1 Settings

We select two public datasets, namely MovieLens-1M[2] [8] and Pinterest[3] [6]. Table 1 illustrates the statistics of the two datasets. For simplicity, we detail the settings of training $f_\theta$ (the settings of training $g_\phi$ is similar if we simply swap the roles of users and items). For each dataset, we first train the baseline on the whole dataset $M$ to get the pre-trained user embedding $e_u$ and item embedding $e_i$. We then split the dataset into meta-training set $D_T$ and meta-test set $D_N$ according to the number of interactions for each user. In MovieLens-1M, the users in $D_T$ interact with more than 40 items, and this splitting setting results 4689 users in $D_T$ and 1351 users in $D_N$. In Pinterest, the users in $D_T$ interact with more than 30 items, and this results 13397 users in $D_T$ and 41790 users in $D_N$[4]. We use $D_T$ to train $f_\theta$, and use $D_N$ to do downstream tasks. The pre-trained $e_u$ in $D_T$ is viewed as target user embedding and $e_i$ is viewed as contextual information.

[2] https://grouplens.org/datasets/movielens/.

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

[4] When training $g_\phi$, in MovieLens-1M, the items in $D_T$ interact with more than 30 users, and this results 2819 items in $D_T$ and 887 items in $D_N$. In Pinterest, the items in $D_T$ interact with more than 30 users, and this results 8544 items in $D_T$ and 1372 items in $D_N$.

**Table 2.** Performance on cold-start user and item embedding evaluation. We use averaged cosine similarity as the evaluation metric.

| Methods | MovieLens (user) | | Pinterest (user) | | MovieLens (item) | | Pinterest (item) | |
|---|---|---|---|---|---|---|---|---|
| | 3-shot | 8-shot | 3-shot | 8-shot | 3-shot | 8-shot | 3-shot | 8-shot |
| LINE | 0.623 | 0.709 | 0.499 | 0.599 | 0.423 | 0.593 | 0.516 | 0.578 |
| AEw-LINE | 0.680 | 0.749 | 0.502 | 0.644 | 0.460 | 0.602 | 0.534 | 0.585 |
| **AEo-LINE** | **0.926** | **0.962** | **0.926** | **0.928** | **0.726** | **0.802** | **0.726** | **0.804** |
| **AEe-LINE** | **0.964** | **0.990** | **0.984** | **0.987** | **0.797** | **0.849** | **0.783** | **0.845** |
| DW | 0.413 | 0.535 | 0.504 | 0.596 | 0.489 | 0.528 | 0.526 | 0.563 |
| AEw-DW | 0.445 | 0.568 | 0.518 | 0.630 | 0.496 | 0.521 | 0.564 | 0.596 |
| **AEo-DW** | **0.828** | **0.835** | **0.847** | **0.892** | **0.603** | **0.784** | **0.664** | **0.736** |
| **AEe-DW** | **0.866** | **0.887** | **0.950** | **0.988** | **0.767** | **0.834** | **0.739** | **0.820** |
| MF | 0.399 | 0.503 | 0.444 | 0.524 | 0.579 | 0.729 | 0.503 | 0.569 |
| AEw-MF | 0.424 | 0.512 | 0.492 | 0.556 | 0.592 | 0.743 | 0.524 | 0.589 |
| **AEo-MF** | **0.836** | **0.945** | **0.646** | **0.813** | **0.713** | **0.809** | **0.698** | **0.823** |
| **AEe-MF** | **0.949** | **0.971** | **0.799** | **0.857** | **0.849** | **0.932** | **0.837** | **0.908** |
| FM | 0.542 | 0.528 | 0.539 | 0.564 | 0.535 | 0.543 | 0.474 | 0.495 |
| AEw-FM | 0.568 | 0.559 | 0.583 | 0.584 | 0.553 | 0.573 | 0.495 | 0.513 |
| **AEo-FM** | **0.702** | **0.803** | **0.809** | **0.826** | **0.723** | **0.809** | **0.694** | **0.804** |
| **AEe-FM** | **0.810** | **0.866** | **0.948** | **0.968** | **0.817** | **0.870** | **0.794** | **0.867** |
| GS | 0.693 | 0.735 | 0.584 | 0.664 | 0.593 | 0.678 | 0.642 | 0.682 |
| AEw-GS | 0.704 | 0.744 | 0.624 | 0.642 | 0.624 | 0.686 | 0.654 | 0.694 |
| **AEo-GS** | **0.806** | **0.896** | **0.825** | **0.906** | **0.747** | **0.828** | **0.712** | **0.812** |
| **AEe-GS** | **0.951** | **0.972** | **0.912** | **0.984** | **0.869** | **0.942** | **0.816** | **0.903** |
| GAT | 0.723 | 0.769 | 0.604 | 0.684 | 0.613 | 0.698 | 0.684 | 0.702 |
| AEw-GAT | 0.724 | 0.784 | 0.664 | 0.682 | 0.664 | 0.726 | 0.694 | 0.712 |
| **AEo-GAT** | **0.846** | **0.935** | **0.886** | **0.916** | **0.757** | **0.868** | **0.725** | **0.821** |
| **AEe-GAT** | **0.969** | **0.981** | **0.952** | **0.991** | **0.869** | **0.950** | **0.846** | **0.912** |

### 3.2 Baseline Methods

We select the following baseline models for learning the user and item embeddings, and compare our method with the corresponding baseline methods.

**Matrix Factorization (MF)** [12]: Learns user and item representations by decomposing the rating matrix.

**Factorization Machine (FM)** [18]: Learns user and item representations through considering the first-order and high-order interactions between features. For fair comparison, we only use the users and items as features.

**LINE** [20]: Learns node embeddings through maximizing the first-order proximity and the second-order proximity between a user and an item in the user-item bipartite graph.

**DeepWalk (DW)** [17]: Learns node embeddings through first performing random walk to sample sequences of nodes from the user-item bipartite graph, and then using Skip-Gram algorithm to learn user and item embeddings.

**GraphSAGE (GS)** [7]: Learns node embeddings through aggregating node information from a node's local neighbors. We first formalize the user-item interaction ratings as a user-item bipartite graph, and then aggregate at most third-order neighbours of each user (item) to update the user (item) representation. We find using second-order neighbours can lead to the best performance.

**GAT** [23]: Learns node embeddings through adding attention mechanism upon the GraphSAGE method. We also find using second-order neighbours can lead to the best performance.

**AE-Baseline:** Is our proposed method which accepts the pre-trained embeddings of items (users), and predicts the final embeddings of the corresponding users (items) by the trained $f_\theta$ or $g_\phi$. We use the name AE-baseline to denote the pre-trained embeddings are produced by the corresponding baseline method. We compare our model AE with these baselines one by one. To verify the effectiveness of the attention part, we have three variant models: **(1) AEo-baseline** which uses soft-attention as attention encoder. **(2) AEe-baseline** which uses self-attention as attention encoder. **(3) AEw-baseline** which discards the attention part and use multilayer perceptron (MLP) to replace it.

### 3.3 Intrinsic Evaluation: Evaluate Cold-Start Embeddings

Here we illustrate the settings in the cold-start user inference task. We select both MovieLens-1M and Pinterest datasets to do evaluation. As mentioned before, we train our model $f_\theta$ on $D_T$. However, in order to make effective evaluation of the predicted user embeddings, the target users should be obtained from the users with sufficient interactions. Thus in this task, we drop out $D_N$ and split the meta-training set $D_T$ into training set $T_r$ and test set $T_e$ with ratio 7:3. We first use each baseline method to train the meta-training set $D_T$ to get the target user embedding. Then for each user in $T_e$, we randomly drop out other items and only maintain $K$ items to predict the user embedding. This simulates the scenario that the users in the test set $T_e$ are cold-start users. We train $f_\theta$ on $T_r$ and do the evaluation on $T_e$. After trained on $T_r$, $f_\theta$ outputs the predicted user embeddings in $T_e$ based on the $K$ interacted items. For each user, we calculate the cosine similarity between the predicted user embedding and the target user embedding, and average them to get the final cosine similarity to denote the quality of the predicted embeddings. For all the baseline methods, we use $T_r$ and the $T_e$ (each user in $T_e$ only has $K$ items) to obtain the predicted user embeddings and calculate the average cosine similarity. In our experiments, $K$ is set as 3 and 8, the number of encoding blocks is 4, the number of heads $H$ is 2, the parameter $\lambda$ is 1e$-$6, the batch size is 256, the embedding dimension $d$ is 16 and the learning rate is 0.01.

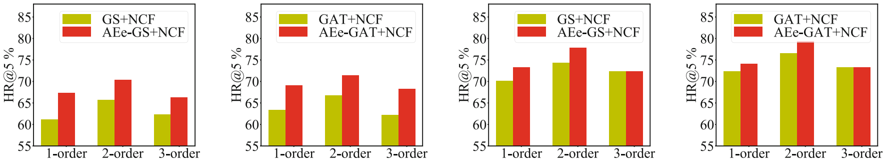
**Experimental Results.** Table 2 lists the performance of the proposed model AEo-baseline, AEe-baseline and other baselines under $K$-shot training settings. The results show that our proposed AEo-baseline and AEe-baseline significantly improve the quality of the learned embeddings comparing with each baseline. Besides, we have four findings: (1) Compared with AEw-baseline, both AEo-baseline and AEe-baseline have better performance, which demonstrates adding attention mechanism is useful. (2) The performance of AEe-baseline is better than AEo-baseline, which implies that using self-attention is better than using soft-attention. The reason is that multi-head self-attention has a more powerful representation ability than soft-attention. (3) When $K$ is relative small (i.e., $K = 3$), the performance of all the baselines gets lower, while the proposed method AEo-baseline and AEe-baseline still have a good performance. (4) Some competitive baselines such as GraphSAGE and GAT can alleviate the cold-start problem by aggregating user's (item's) information from user's (item's) first-order or high-order neighbours, however, their performance is lower than our proposed method. The reason is that for the cold-start users and the cold-start items, there are still few high-order neighbours. Both (3) and (4) demonstrates all the baselines are difficult to deal with the cold-start issues, while our model is capable of generating good representations for cold-start users and items.

### 3.4 Extrinsic Evaluation: Evaluate Cold-Start Embeddings on Downstream Tasks

To illustrate the effectiveness of our proposed method in dealing with learning the representations of the cold-start users and items, we evaluate the resulted embeddings on three downstream tasks: (1) Personalized recommendation (2) User classification and (3) Item classification. For each task, for the proposed method, we use $f_\theta$ and $g_\phi$ to generate the user and item embeddings in $D_N$ to do evaluation; for the baseline methods, we directly train the baseline on $M$ and use the resulted user and item embeddings to do evaluation.

**Personalized Recommendation Task.** Personalized recommendation task aims at recommending proper items to users. Recent approaches for recommendation tasks use randomly initialized user and item embeddings as their inputs, which often get suboptimal recommendation performance. We claim that a high-quality pre-trained embeddings can benefit the recommendation task.

We use MovieLens-1M and Pinterest datasets and select Neural Collaborative Filtering (NCF) [11] as the recommender. We first randomly split $D_N$ into training set and test set with ratio 7:3, and then feed the user and item embeddings generated by our model or the baselines into the GMF and MLP unit in NCF as pre-trained embeddings, which are further fine-tuned during training process. During the training process, for each positive pairs $(u, i)$, we randomly sample one negative pairs. During the test process, for each positive instance, we randomly sample 99 negative instance [11]. We use Hit Ratio of top $m$ items (HR@$m$), Normalized Discounted Cumulative Gain of top $m$ items (NDCG@$m$) and Mean Reciprocal Rank (MRR) as the evaluation indicator.



(a) MovieLens-1M (b) MovieLens-1M (c) Pinterest (d) Pinterest

**Fig. 2.** Recommendation performance of GraphSAGE, GAT and our proposed method when using first-order and high-order neighbours.

The hyperparameters we used are the same as [11]. Table 3 illustrates the recommendation performance. Note that the method NCF represents using the randomly initialized embeddings. The results show that: (1) Using pre-trained embeddings can improve the recommendation performance. (2) Our model beats all the baselines. (3) Compared with AEw-baseline+NCF method which uses MLP layer to replace the attention encoder, using soft-attention and self-attention can improve the performance. (4) Due to the strong representation ability of multi-layer self-attention mechanism, the performance of using self-attention encoder is better than using soft-attention encoder. All the above analysis shows that our proposed method has the ability of learning high-quality representations of cold-start users and items. We further show the recommendation performance of GraphSAGE (GS), GAT and our proposed method AEe-GS, AEe-GAT when using first-order, second-order and third-order neighbours of target users and target items. Figure 2 illustrates the recommendation performance. The results show that all the methods have better performance when using second-order neighbours. Besides, our proposed method significantly beats GS and GAT due to the strong representation ability.

**Item Classification Task.** We evaluate the encoded item embeddings in AE through a multi-label classification task. The goal is to predict multi-labels of items given the user-item interactive ratings. Intuitively, similar items have a higher probability belonging to the same genre, thus this task needs high-quality item embeddings as input features. We select MovieLens-1M dataset, in which the movies are divided into 18 categories (e.g., Comedy, Action, War). Note that each movie belongs to multi genres, for example, the movie ‘Toy Story (1995)’ belongs to there genres, namely animation, children’s, and comedy. We use logistic regression classifier which accepts the item embeddings as input features to do evaluation. Specifically, we first randomly split $D_N$ into training set and test set with ratio 7:3, and then use item embeddings generated by our model or the baselines as input features. Next we train the logistic regression classifier in the training set and finally evaluate the performance in the test set. Micro-averaged F1-score is used as an evaluation metric. Table 4 illustrates the item classification performance. The result shows that our proposed model beats all the baselines, which verifies our model can produce high-quality item representations. Besides, the performance of AEw-baseline is lower than AEo-baseline and AEe-baseline;

**Table 3.** Performance on recommendation performances.

| Methods | MovieLens | | | Pinterest | | |
|---|---|---|---|---|---|---|
| | HR@5 | NDCG@5 | MRR | HR@5 | NDCG@5 | MRR |
| NCF | 0.392 | 0.263 | 0.260 | 0.627 | 0.441 | 0.414 |
| LINE + NCF | 0.633 | 0.631 | 0.648 | 0.642 | 0.543 | 0.587 |
| AEw-LINE + NCF | 0.641 | 0.637 | 0.652 | 0.644 | 0.549 | 0.582 |
| **AEo-LINE + NCF** | **0.659** | **0.640** | **0.664** | **0.651** | **0.568** | **0.590** |
| **AEe-LINE + NCF** | **0.666** | **0.646** | **0.679** | **0.659** | **0.585** | **0.593** |
| DW+NCF | 0.621 | 0.620 | 0.634 | 0.587 | 0.392 | 0.367 |
| AEw-DW + NCF | 0.628 | 0.624 | 0.643 | 0.593 | 0.403 | 0.369 |
| **AEo-DW + NCF** | **0.646** | **0.640** | **0.663** | **0.624** | **0.483** | **0.402** |
| **AEe-DW+NCF** | **0.673** | **0.643** | **0.684** | **0.652** | **0.462** | **0.433** |
| MF + NCF | 0.558 | 0.577 | 0.579 | 0.711 | 0.660 | 0.666 |
| AEw-MF + NCF | 0.562 | 0.564 | 0.581 | 0.718 | 0.672 | 0.678 |
| **AEo-MF + NCF** | **0.573** | **0.583** | **0.589** | **0.726** | **0.702** | **0.693** |
| **AEe-MF + NCF** | **0.597** | **0.591** | **0.595** | **0.748** | **0.725** | **0.736** |
| FM + NCF | 0.448 | 0.286 | 0.265 | 0.641 | 0.453 | 0.424 |
| AEw-FM + NCF | 0.451 | 0.291 | 0.271 | 0.652 | 0.482 | 0.482 |
| **AEo-FM + NCF** | **0.482** | **0.334** | **0.326** | **0.723** | **0.702** | **0.672** |
| **AEe-FM + NCF** | **0.495** | **0.357** | **0.346** | **0.756** | **0.721** | **0.729** |
| GS + NCF | 0.657 | 0.664 | 0.657 | 0.743 | 0.642 | 0.681 |
| AEw-GS + NCF | 0.668 | 0.672 | 0.675 | 0.753 | 0.652 | 0.693 |
| **AEo-GS + NCF** | **0.683** | **0.693** | **0.684** | **0.778** | **0.683** | **0.723** |
| **AEe-GS + NCF** | **0.703** | **0.724** | **0.704** | **0.782** | **0.693** | **0.735** |
| GAT + NCF | 0.667 | 0.672 | 0.664 | 0.765 | 0.664 | 0.702 |
| AEw-GAT + NCF | 0.684 | 0.681 | 0.682 | 0.771 | 0.674 | 0.719 |
| **AEo-GAT + NCF** | **0.694** | **0.702** | **0.704** | **0.782** | **0.693** | **0.723** |
| **AEe-GAT + NCF** | **0.713** | **0.724** | **0.735** | **0.793** | **0.713** | **0.746** |

AEe-baseline has the best performance, which verifies adding attention encoder can improve the performance; due to the strong representation ability, using self-attention is a better choice than using soft-attention.

**User Classification Task.** We further evaluate the encoded user embeddings in AE through a classification task. The goal is to predict the age bracket of users given the user-item interactions. Intuitively, similar users have same tastes, thus they have a higher probability belonging to the same age bracket. We select MovieLens-1M dataset, and the users are divided into 7 age brackets, (i.e., Under 18, 18–24, 25–34, 35–44, 44–49, 50–55, 56+). We use logistic regression classifier which accepts user embeddings as input features to do evaluation. Specifically, we first randomly split $D_N$ into training set and test set with ratio 7:3, and

**Table 4.** Performance on item classification and user classification task.

| Methods | Movielens-1M | |
|---|---|---|
| | Items classification (micro-averaged F1 score) | Users classification (averaged F1 score) |
| LINE | 0.6052 | 0.3031 |
| AEw-LINE + NCF | 0.6111 | 0.3067 |
| **AEo-LINE + NCF** | **0.6478** | **0.3294** |
| **AEe-LINE** | **0.6620** | **0.3309** |
| DW | 0.5335 | 0.2605 |
| AEw-DW+ NCF | 0.5435 | 0.2685 |
| **AEo-DW + NCF** | **0.5687** | **0.2799** |
| **AEe-DW** | **0.5707** | **0.2894** |
| MF | 0.4791 | 0.2273 |
| AEw-MF + NCF | 0.4852 | 0.2291 |
| **AEo-MF + NCF** | **0.5364** | **0.2368** |
| **AEe-MF** | **0.5496** | **0.2477** |
| FM | 0.4809 | 0.2803 |
| AEw-FM + NCF | 0.4883 | 0.2894 |
| **AEo-FM + NCF** | **0.4912** | **0.3194** |
| **AEe-FM** | **0.5062** | **0.3286** |
| GS | 0.5931 | 0.2941 |
| AEw-GS + NCF | 0.6012 | 0.3011 |
| **AEo-GS + NCF** | **0.6342** | **0.3134** |
| **AEe-GS** | **0.6546** | **0.3295** |
| GAT | 0.6135 | 0.3147 |
| AEw-GAT + NCF | 0.6243 | 0.3256 |
| **AEo-GAT + NCF** | **0.6464** | **0.3456** |
| **AEe-GAT** | **0.6646** | **0.3673** |

then use user embeddings generated by our model or the baselines as input features. Next we train the logistic regression classifier in the training set and finally evaluate the performance in the test set. Averaged F1-score is used as an evaluation metric. Table 4 shows the user classification performance. The result shows that our method beats all baselines, which further demonstrates our model is capable of learning the high-quality representations.

## 4 Related Work

Our work is highly related to the meta learning method, which aims to design a meta-learner that can efficiently learn the meta information and can rapidly adapt to new instances. It has been successfully applied in Computer Vision (CV) area and can be classified into two groups. One is the metric-based method which learns a similarity metric between new instances and instances in the training set. Examples include Matching Network [24] and Prototypical Network [19]. The other one is model-based method which designs a meta learning model to directly predict or update the parameters of the classifier according to the training data. Examples include MAML [5] and Meta Network [15]. Recently, some works attempt to use meta learning to solve the cold-start issue in the recommendation systems. Pan et al. [16] propose to learn a embedding generator for new ads by making use of previously learned ads' features through gradient-based meta-learning. Vartak et al. [21] propose to learn a neural network which takes items from user's history and outputs a score function to apply to new items. Du et al. [4] propose a scenario-specific meta learner, which adjust the parameters of the recommendation system when a new scenario comes. Different from these methods that aim to directly learn a powerful recommendation model, we focus on how to learn the representations of the cold-start users and items, and we design a novel attention-based encoder that encode the contextual information to predict the target embeddings.

## 5 Conclusion

We present the first attempt to solve the problem of learning accurate representations of cold-start users and cold-start items. We formulate the problem as a few-shot learning task and propose a novel attention-based encoder AE which learns to predict the target users (items) embeddings by aggregating only $K$ instances corresponding to the users (items). Different from recent state-of-the-art meta learning methods which aim to directly learn a powerful recommendation model, we focus on how to learn the representations of cold-start users and items. Experiments on both intrinsic evaluation task and three extrinsic evaluation tasks demonstrate the effectiveness of our proposed model.

**Acknowledgments.** This work is supported by National Key R&D Program of China (No. 2018YFB1004401) and NSFC (No. 61532021, 61772537, 61772536, 61702522).

## References

1. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NeurlPS 2013 (2013)
2. Brazdil, P., Giraud-Carrier, C.G., Soares, C., Vilalta, R.: Metalearning - Applications to Data Mining. Cognitive Technologies. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-73263-1
3. Cao, Y., Wang, X., He, X., Hu, Z., Chua, T.S.: Unifying knowledge graph learning and recommendation: towards a better understanding of user preferences. In: WWW 2019. ACM (2019)
4. Du, Z., Wang, X., Yang, H., Zhou, J., Tang, J.: Sequential scenario-specific meta learner for online recommendation. In: SIGKDD 2019 (2019)

5. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML 2017, pp. 1126–1135. JMLR.org (2017)
6. Geng, X., Zhang, H., Bian, J., Chua, T.S.: Learning image and user features for recommendation in social networks. In: ICCV 2015, pp. 4274–4282 (2015)
7. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurlPS 2017, pp. 1024–1034 (2017)
8. Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. TIIS **5**(4), 1–9 (2016)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR 2016, pp. 770–778 (2016)
10. He, X., He, Z., Song, J., Liu, Z., Jiang, Y.G., Chua, T.S.: NAIS: neural attentive item similarity model for recommendation. IEEE Trans. Knowl. Data Eng. **30**(12), 2354–2366 (2018)
11. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: WWW 2017, pp. 173–182 (2017)
12. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
13. Lei Ba, J., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
14. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. **7**(1), 76–80 (2003)
15. Munkhdalai, T., Yu, H.: Meta networks. In: ICML 2017, pp. 2554–2563. JMLR.org (2017)
16. Pan, F., Li, S., Ao, X., Tang, P., He, Q.: Warm up cold-start advertisements: improving CTR predictions via learning to learn id embeddings. In: SIGIR 2019 (2019)
17. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: SIGKDD 2014, pp. 701–710. ACM (2014)
18. Rendle, S.: Factorization machines with libFM. TIST **3**(3), 57 (2012)
19. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NeurlPS 2017, pp. 4077–4087 (2017)
20. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: WWW 2015, pp. 1067–1077 (2015)
21. Vartak, M., Thiagarajan, A., Miranda, C., Bratman, J., Larochelle, H.: A meta-learning perspective on cold-start recommendations for items. In: NeurlPS 2017, pp. 6904–6914 (2017)
22. Vaswani, A., et al.: Attention is all you need. In: NeurlPS 2017, pp. 5998–6008 (2017)
23. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR 2018 (2018)
24. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: NeurlPS 2016, pp. 3630–3638 (2016)
25. Wang, H., Zhang, F., Zhao, M., Li, W., Xie, X., Guo, M.: Multi-task feature learning for knowledge graph enhanced recommendation. In: WWW 2019 (2019)
26. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.: KGAT: knowledge graph attention network for recommendation. In: SIGKDD 2019, pp. 950–958 (2019)
27. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI 2014 (2014)
28. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: Collaborative knowledge base embedding for recommender systems. In: SIGKDD 2016. ACM (2016)