Aligning Distillation For Cold-start Item Recommendation

Feiran Huang
Jinan University, China
huangfr@jnu.edu.cn

Zefan Wang
Jinan University, China
wongzfn@gmail.com

Xiao Huang
The Hong Kong Polytechnic University, Hong Kong
xiaohuang@comp.polyu.edu.hk

Yufeng Qian
Georgia Institute of Technology, United States
yufengqian1@gmail.com

Zhetao Li
Jinan University, China
liztchina@hotmail.com

Hao Chen*
The Hong Kong Polytechnic University, Hong Kong
sundaychenhao@gmail.com

ABSTRACT

Recommending cold items in recommendation systems is a long-standing challenge due to the inherent differences between warm items, which are recommended based on user behavior, and cold items, which are recommended based on content features. To tackle this, generative models generate synthetic embeddings from content features, while dropout models enhance the robustness of the recommendation system by randomly dropping behavioral embeddings during training. However, these models primarily focus on handling the recommendation of cold items, but do not effectively address the differences between warm and cold recommendations. As a result, generative models may over-recommend either warm or cold items, neglecting the other type, and dropout models may negatively impact warm item recommendations. To address this, we propose the Aligning Distillation (ALDI) framework, which leverages warm items as "teachers" to transfer their behavioral information to cold items, referred to as "students". ALDI aligns the students with the teachers by comparing the differences in their recommendation characters, using tailored rating distribution aligning, ranking aligning, and identification aligning losses to narrow these differences. Furthermore, ALDI incorporates a teacher-qualifying weighting structure to prevent students from learning inaccurate information from unreliable teachers. Experiments on three datasets show that our approach outperforms state-of-the-art baselines in terms of overall, warm, and cold recommendation performance with three different recommendation backbones.

CCS CONCEPTS

• **Information systems → Social recommendation**; • **Human-centered computing → Social recommendation**.

KEYWORDS

cold-start recommendation, aligning distillation, content features

*Corresponding author.

SIGIR '23, June 03–05, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591732

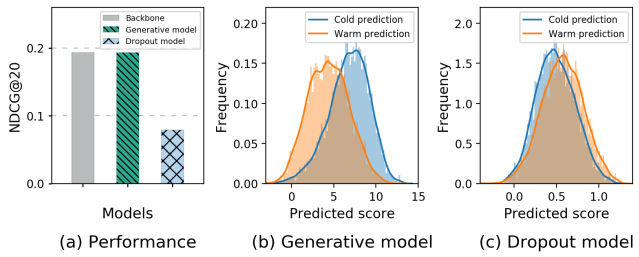
ACM Reference Format:
Feiran Huang, Zefan Wang, Xiao Huang, Yufeng Qian, Zhetao Li, and Hao Chen. 2023. Aligning Distillation For Cold-start Item Recommendation. In *Proceedings of Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23).* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3539618.3591732

1 INTRODUCTION

Embedding-based collaborative filtering (CF) models have achieved state-of-the-art performance in recommendations by learning meaningful and exact behavioral embedding vectors for each user and item from historical interactions [15, 16, 36, 47, 58, 59]. Specifically, traditional CF models learn embeddings by enforcing the dot product or MLP of historically interacted user-item embedding pairs to be greater than non-interacted pairs [16, 36]. Graph-based CF models incorporate neighbors' embeddings to better model users' and items' behavior and achieve the state-of-the-art recommendation performance [9, 15, 47, 56]. However, in addition to "warm" items with rich behaviors, there also exist thousands of "cold" items produced every second, such as live streams [30, 41] and short videos [13, 21, 28]. These cold items lack accurate historical embeddings, making it difficult for recommendation systems to accurately evaluate users' intent towards them, negatively impacting user experiences and the revenue of recommender systems.

To address this, modern recommender systems aim to generate accurate cold-start item embeddings and encourage cooperation between warm and cold items [33, 62, 63]. Specifically, generative models aim to generate powerful cold-start behavioral embeddings to increase the recommendation performance of the cold models. Examples include DeepMusic [43], which learns to generate the embeddings by minimizing the MSE loss between the generated embeddings and the trained warm embeddings, and meta-learning models [33, 62], which use few-shot learning theory to map cold content features to embeddings that can quickly converge to their warm embeddings. Dropout models, on the other hand, randomly drop the behavior embedding, which increases the robustness of the recommendation models. Examples include DropoutNet [44] and Heater [63], which both randomly drop the trained warm embeddings during the training process, and CLCRec [50], which reduces the discrepancy between the distribution of ratings between warm and cold recommendations in a contrastive learning solution.

Despite their ability to generate accurate cold-start embeddings, existing models have limitations in accommodating both warm and cold items. As shown in Figure 1(a), generative models handle



(a) Performance (b) Generative model (c) Dropout model

Figure 1: This is a brief comparison of typical generative and dropout models on CiteULike, showing (a) the warm recommendation performance, (b) the distribution of predictions for cold and warm items from DeepMusic (generative model), and (c) the distribution of predictions for cold and warm items from the DropoutNet (dropout model).

cold and warm recommendations independently, resulting in the same warm recommendation performance as the backbone model. However, their independent structure leads to a greater difference in the rating distribution of warm and cold items compared to dropout models (Figure 1(b)). This can result in over-recommendation of either warm or cold items, presenting undesired cold items instead of interesting warm items or neglecting the recommendation of cold items. On the other hand, dropout cold-start models train a hybrid model to accommodate both warm and cold items. As shown in Figure 1(c), dropout-based models have a nearly equal rating distribution of warm and cold items. But this comes at the cost of a significant decrease in warm recommendation performance compared to the backbone and generative-based models, negatively impacting the recommendation of existing warm items.

The observed phenomenon is due to the fundamental differences between recommendations based on user behavior (warm items) and those based on content features (cold items). Current generative and dropout models show promise in handling cold items but they do not effectively address the inherent differences. To improve these limitations, a recommendation distillation solution could be promising to bridge the gap by transferring the knowledge of warm items to cold items. Recently, distillation models such as Rank Distillation [40, 45] and Collaborative Distillation [25] have proven successful in transferring knowledge from teacher models to student models. However, applying this concept to cold-start recommendations presents several challenges:

(1) Joint Recommendation: Cold-start recommendations require cooperation between both cold students and warm teachers to make the best recommendation, while existing ranking distillation only utilizes the student model. This raises challenges in how to align the students with the teachers.
(2) Difference Modeling: The warm recommendation performance and cold recommendation performance can only be partially optimized by generative and dropout cold-start models, respectively. It is difficult to discover and model the core differences between teachers and students to cover both aspects.
(3) Unreliable Teachers: In cold-start recommendations, warm items with historical information serve as teachers. However, some warm items may not be well represented and their knowledge transfer may introduce noise and hinder distillation. Hence, accurately identifying unreliable teachers and reducing their distillation weights is a challenge.

To address the aforementioned challenges, we propose a novel Aligning Distillation (ALDI) framework for item cold-start recommendation, which utilizes warm items as "teachers" to distill behavioral information to cold items, referred to as "students". Specifically, during training, we simultaneously compute predictions for both teachers (incorporating behavioral embeddings) and students (without behavioral embeddings). Then we are able to compare the embeddings and prediction results between the teachers and the students to model the inherent differences caused by the absence of behavioral embeddings. After differences modeling, we design aligning distillation losses tailored to reduce these differences and guide the students toward alignment with teachers. In addition, we evaluate teacher reliability by analyzing their historical user interactions and we assign lower learning weights to unreliable teachers. The main contributions are as follows:

- We propose a novel distillation problem for cold-start recommendations, which aims to align the students with the teachers rather than substitute the teachers. Besides, the aligning distillation is generally powerful and can cold-start both traditional CF models and graph-based CF models.
- ALDI introduces three inherent differences between the recommendation of warm teachers and cold students due to lacking behavioral information. Three tailored aligning distillation losses are proposed to address these differences and align student models with teachers.
- To make the most of the teachers' information, we implement a teacher-qualifying weighting structure in ALDI to ensure that student models learn more from reliable teachers and less from unreliable ones.
- Comprehensive experiments demonstrate that ALDI achieves generally good recommendation performance on two public datasets using multiple backbone models. Ablation studies show that AlDI outperforms ranking distillation recommendation models in cold-start scenarios. The code is publicly available [1].

2 PROBLEM STATEMENT

2.1 Notations and Problem Definition

Notations. We utilize $\mathcal{U}$ and $\mathcal{I}$ to denote the user and item sets of a given recommendation dataset, and $\mathcal{H}$ to denote the historical interaction sets between users and items. Specifically, $\mathcal{I}_w$ refers to the warm item set that has at least one historical interaction record, and $\mathcal{I}_c$ refers to the cold item set that has no historical interaction records. With typical warm recommendation backbones such as Matrix Factorization, Neural Matrix Factorization, and Graph Neural Networks, we can learn behavioral embedding vectors for each user and warm item, namely $\boldsymbol{E}_{\mathcal{U}}$ and $\boldsymbol{E}_{\mathcal{I}_w}$. At the micro level, we employ $\boldsymbol{e}_u$ and $\boldsymbol{e}_i$ to denote the trained embedding vectors for user $u \in \mathcal{U}$ and warm item $i \in \mathcal{I}_w$. Since the cold items do not have behavioral embeddings, the recommender systems provide content features such as tags and descriptions for all items, including warm

[1]The source code is available at https://github.com/zfnWong/ALDI.

and cold items. Thus the cold items can be cold-started with their content features, which is denoted by $\boldsymbol{c}_i$ for each item $i \in \mathcal{I}_w \cup \mathcal{I}_c$.

Cold-start Recommendation. The recommender systems aim to recommend top-K items to users from a pool of items by computing relevance scores for warm and cold items. The performance of the cold-start recommendations and the impact on warm recommendations can be evaluated through the three tasks defined as follows: Overall recommendation (Rec_{all}): A ranking of both warm and cold items based on prediction scores, computed as

$$Rec_{all} = \text{rank}(\{\hat{y}_{u,i}^{(w)}, \forall i \in \mathcal{I}_w\} \cup \{\hat{y}_{u,i}^{(c)}, \forall i \in \mathcal{I}_c\}, K). \quad (1)$$

Warm recommendation (Rec_w): A ranking of warm items based on prediction scores, computed as

$$Rec_w = \text{rank}(\{\hat{y}_{ui}^{(w)}, \forall i \in \mathcal{I}_w\}, K). \quad (2)$$

Cold recommendation (Rec_c): A ranking of cold items based on prediction scores, computed as

$$Rec_c = \text{rank}(\{\hat{y}_{ui}^{(c)}, \forall i \in \mathcal{I}_c\}, K). \quad (3)$$

Here, $\text{rank}(\mathcal{S}, K)$ represents the ranking of prediction scores in the set $\mathcal{S}$ and returns the indices of the top-K ranked items.

2.2 Cold-start Recommendation

Generative Models. Generative models are designed to make use of the backbone recommendation model as the warm recommendation and generate approximate cold-start embeddings to coordinate with it. The aim is to learn a generator function $g(\cdot)$ that produces cold-start embeddings based on content features. The inference of the generative models can be expressed as follows:

$$\begin{aligned} \hat{y}_{ui}^{(w)} &= \boldsymbol{e}_u^\top \cdot \boldsymbol{e}_i, \quad i \in \mathcal{I}_w, \\ \hat{y}_{ui}^{(c)} &= \boldsymbol{e}_u^\top \cdot g(\boldsymbol{c}_i), \quad i \in \mathcal{I}_c. \end{aligned} \quad (4)$$

Though generative cold-start models can maintain the warm recommendation performance of the backbone recommendation model, there may still be considerable diversity between the warm recommendation and the cold recommendation in terms of rating distribution (Figure 1(b)).

Dropout Models. Dropout models, such as DropoutNet [44] and Heater [63], propose hybrid recommender systems that combine behavioral embeddings with zero vectors by dropout strategy to predict user-item relevance. They use an intermediary mapping function $f_{\mathcal{I}}(\cdot)$ to map warm recommendation vectors $\boldsymbol{e}_i, \boldsymbol{c}_i$ and cold recommendation vectors $\boldsymbol{0}, \boldsymbol{c}_i$ to a stable embedding vector. The prediction of warm items and cold items is done using this mapping function. The formal definition of the dropout cold-start recommendation is:

$$\begin{aligned} \hat{y}_{ui}^{(w)} &= f_{\mathcal{U}}(\boldsymbol{e}_u)^\top \cdot f_{\mathcal{I}}(\boldsymbol{e}_i, \boldsymbol{c}_i), \quad i \in \mathcal{I}_w, \\ \hat{y}_{ui}^{(c)} &= f_{\mathcal{U}}(\boldsymbol{e}_u)^\top \cdot f_{\mathcal{I}}(\boldsymbol{0}, \boldsymbol{c}_i), \quad i \in \mathcal{I}_c, \end{aligned} \quad (5)$$

where $f_{\mathcal{U}}(\cdot)$ and $f_{\mathcal{I}}(\cdot)$ are the intermediary mapping functions for users and items, respectively. Joint training of warm items and cold items using a dropout strategy ensures consistency between the warm and cold recommendations, but at the cost of reduced warm recommendation performance compared to backbone models (Figure 1(a)), as the behavioral embeddings are altered by the intermediary mapping function $f_{\mathcal{U}}(\cdot)$.

2.3 Ranking Distillation

Inspired by the success of knowledge distillation in computer vision, the Ranking Distillation approach [40] compresses a large backbone teacher model into a smaller yet powerful student model by minimizing the recommendation loss and a ranking distillation loss. The optimization problem is defined as:

$$\min_{\theta_{student}} \mathcal{L}_{rec} + \lambda \mathcal{L}_{RD}, \quad (6)$$

where $\theta_{student}$ are the parameters of the student model, $\mathcal{L}_{rec}$ is the basic recommendation loss, and $\mathcal{L}_{RD}$ is the ranking distillation loss. The ranking distillation loss $\mathcal{L}_{RD}$ evaluates the student's prediction scores with the scores given by the teacher [25, 40]. However, in a model compression scenario, only the student model is used for recommendations, while in a cold-start scenario, both the warm (teacher) and cold (student) items are ranked and recommended together. To ensure successful cold-start performance, the distillation model should align the student with the teacher, avoiding any mutual negative influence.

3 ALIGNING DISTILLATION–ALDI

In this section, we outline the proposed framework ALDI for item cold-start recommendation, including its overall structure and implementation details. We start by presenting our frameworks including defining the role of the "teachers" and "students", identifying the three key differences between warm teachers and cold students, and constructing the training batches. We then propose three tailored distillation strategies to align the teachers and students, considering these differences. Finally, we introduce a teacher-qualifying structure to prevent students from learning unreliable information from unqualified teachers.

3.1 Overall Framework

3.1.1 ***Definition of Teachers and Students.*** In ALDI (as shown in Figure 2), we assign the role of "teacher" to models that predict user-item relations based on behavioral information, while "student" models predict the same relationships using only content features. Given a user-item pair (u, i), the relevance score for a teacher can be calculated by taking the dot product of the behavior embeddings of user u and item i, as represented in Eq. (7).

$$\hat{y}_{ui}^{(t)} = \boldsymbol{e}_u^\top \cdot \boldsymbol{e}_i. \quad (7)$$

For students, which may be used to predict the relevance of true cold items with no historical data or simulated cold items created during training by removing behavioral embeddings from warm items, the framework generates cold-start embeddings from content features and maps the user's behavioral embedding to compute the relevance score, as described in Eq. (8).

$$\hat{y}_{ui}^{(s)} = f_{\mathcal{U}}(\boldsymbol{e}_u)^\top \cdot f_{\mathcal{I}}(\boldsymbol{c}_i). \quad (8)$$

The mapping functions $f_{\mathcal{U}}(\cdot)$ and $f_{\mathcal{I}}(\cdot)$ correspond to the transformations of the user behavioral embedding $\boldsymbol{e}_u$ and item content feature $\boldsymbol{c}_i$, respectively. We use a two-layer multi-layer perceptron (MLP) as the mapping functions $f_{\mathcal{U}}(\cdot)$ and $f_{\mathcal{I}}(\cdot)$, though more

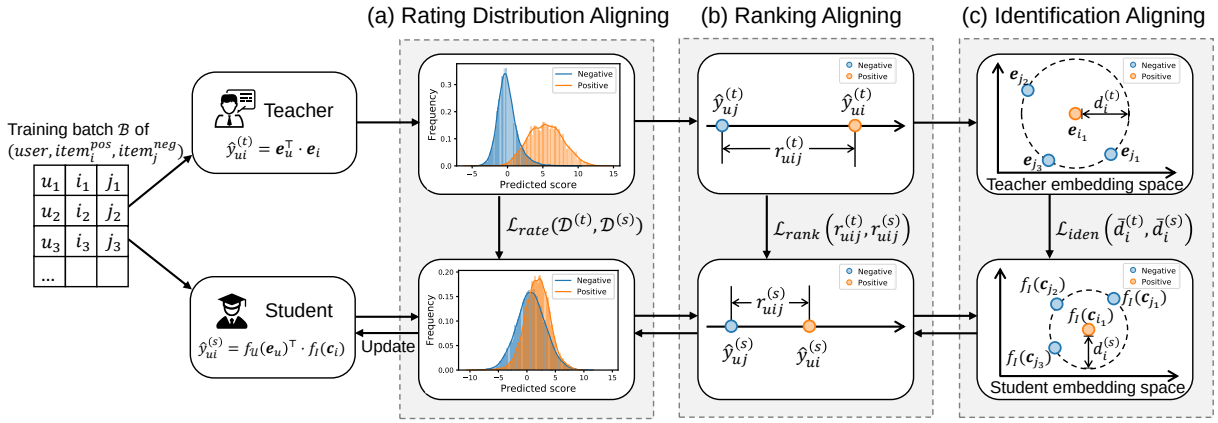


Figure 2: The sketch map of Aligning Distillation (ALDI). ALDI involves teachers predicting user-item relations using behavioral embeddings and students using mapped item content features. By computing both teacher and student predictions, we evaluate the differences between them and implement tailored rating distribution aligning, ranking aligning, and identification aligning strategies to narrow the gaps.

sophisticated structures such as the combination of experts [63] could also be employed. However, in order to demonstrate the framework's effectiveness, we choose to use a simple generator.

As demonstrated in Eq. (9), the recommendation process seamlessly integrates the strengths of Dropout and Generative models to result in a comprehensive recommendation system. Warm items are predicted using the teacher settings, whereas cold items are predicted using the student setting.

$$\hat{y}_{ui} = \begin{cases} \boldsymbol{e}_u^\top \cdot \boldsymbol{e}_i, & i \in \mathcal{I}_w \\ f_{\mathcal{U}}(\boldsymbol{e}_u)^\top \cdot f_{\mathcal{I}}(\boldsymbol{c}_i), & i \in \mathcal{I}_c \end{cases}. \tag{9}$$

This effective integration leverages the warm item recommendation capabilities of the backbone model, while also aligning user behavioral and cold-start item embeddings through mapping functions. This leads to improved recommendations for cold-start items and results in a more robust and effective recommendation system.

3.1.2 ***Introduction of the Differences.*** However, combining warm and cold item recommendations can present challenges in aligning the preferences of the teacher for warm items and the student for cold items. If the system places more importance on the relevance of cold items for the student than warm items for the teacher, it could result in the recommendation of unsuitable cold items. Conversely, if the system places less importance on the relevance of cold items for the student than warm items for the teacher, it may hinder the successful recommendation of desired cold items to the user. The recommendation of students, which is based on content features rather than directly reflecting users' behavioral intent as in teachers, has three inherent differences compared to teachers:

(1) **Rating Distribution Difference:** Figure 2(a) demonstrates that teachers with behavioral embeddings can precisely determine users' intentions towards their items, resulting in clear and distinguishable rating distributions for positive and negative user-item pairs. Conversely, students' behavior characters are inferred from content features, making it difficult to clearly differentiate the rating distributions for positive and negative pairs as compared to the teachers.
(2) **Ranking Difference:** Figure 2(b) shows that the difference in prediction scores between positive and negative user-item pairs may vary greatly between teachers and students. This difference may amplify the rating distribution difference and negatively impact warm recommendations.
(3) **Identification Difference:** Figure 2(c) highlights that with behavioral embeddings, teachers with similar behavioral patterns have similar behavioral interests, making it easy to identify their behavioral characteristics from other items. On the other hand, students' cold-start embeddings are derived from content features, making them less effective in identifying their behavioral characteristics as compared to teachers.

To reconcile the differences between the teacher and student item recommendations, we propose a training approach that enhances students' recommendation performance while aligning them with the teachers. This is done by augmenting the basic recommendation loss with a joint distillation loss. The optimization objective is expressed as follows:

$$\min_{\theta_f} \underbrace{\mathcal{L}_{basic}}_{Rec.\ loss} + \underbrace{\alpha\mathcal{L}_{rate} + \beta\mathcal{L}_{rank} + \gamma\mathcal{L}_{iden}}_{Distillation\ loss}, \tag{10}$$

where θ_f represents the parameters of the mapping functions $f_{\mathcal{U}}(\cdot)$ and $f_{\mathcal{I}}(\cdot)$. The first part of the objective, $\mathcal{L}_{basic}$, refers to the standard recommendation loss (such as Bayesian Personalized Ranking loss) that ensures the student model has the necessary capacity for recommendations. The second part, the joint distillation loss, aligns the students with the teachers by incorporating three different components: rating distribution alignment ($\mathcal{L}_{rate}$), ranking alignment ($\mathcal{L}_{rank}$), and identification alignment ($\mathcal{L}_{iden}$). The weights α, β, and γ control the importance of each alignment term.

3.1.3 ***Construction of Training Batches.*** In order to accurately calculate the differences in identification and rating distribution, we

propose a batch-wise loss calculation strategy (as shown in Figure 2). This strategy involves constructing triple sets by considering both positive user-item pairs (in $\mathcal{H}$) and negative sampled user-item pairs. The triple sets are defined as follows:

$$O = \{(u, i, j)|(u, i) \in \mathcal{H}, (u, j) \notin \mathcal{H}, i \in \mathcal{I}_w, j \in \mathcal{I}_w\}, \quad (11)$$

Here, $\mathcal{H}$ represents the observed user-item interactions. We will discuss the design of the distillation losses for the above-mentioned differences with O in the next sections.

3.2 Rating Distribution Aligning Distillation

In this subsection, we propose using rating distribution aligning distillation to synchronize the rating distributions of teacher and student models. This will ensure that when ranking items (in Eq. (1)), both warm and cold items receive similar basic ratings. The alignment prevents the recommender system from unduly prioritizing either warm or cold items, avoiding undesired recommendations.

To address the challenge of different relevance scores for positive and negative user-item pairs, we independently align the rating distributions of teachers and students for both positive and negative pairs. This approach ensures accurate recommendations by synchronizing the positive and negative rating distributions of teachers and students. Specifically, we use $\mathcal{D}^{(t)}$ to denote the rating distribution of the teachers and $\mathcal{D}^{(s)}$ to denote the rating distribution of the students. For a given training batch $\mathcal{B}$, we have the positive distributions $\mathcal{D}^{(t)}_{\mathcal{B}_I}$ and $\mathcal{D}^{(s)}_{\mathcal{B}_I}$, as well as the negative distributions $\mathcal{D}^{(t)}_{\mathcal{B}_J}$ and $\mathcal{D}^{(s)}_{\mathcal{B}_J}$. We define the rating distribution aligning distillation loss as the following formula:

$$\mathcal{L}_{rate} = \mathrm{Dis}(\mathcal{D}^{(t)}_{\mathcal{B}_I}, \mathcal{D}^{(s)}_{\mathcal{B}_I}) + \mathrm{Dis}(\mathcal{D}^{(t)}_{\mathcal{B}_J}, \mathcal{D}^{(s)}_{\mathcal{B}_J}), \quad (12)$$

where $\mathrm{Dis}(\cdot, \cdot)$ is a function measuring the distance between two distributions. In this study, we employ the pointwise implementation of the distance function, defined as:

$$\begin{aligned} \mathrm{Dis}(\mathcal{D}^{(t)}_{\mathcal{B}_I}, \mathcal{D}^{(s)}_{\mathcal{B}_I}) &= \frac{1}{|\mathcal{B}|} \sum_{(u,i,j)\in\mathcal{B}} \left|\hat{y}^{(t)}_{ui} - \hat{y}^{(s)}_{ui}\right|^2, \\ \mathrm{Dis}(\mathcal{D}^{(t)}_{\mathcal{B}_J}, \mathcal{D}^{(s)}_{\mathcal{B}_J}) &= \frac{1}{|\mathcal{B}|} \sum_{(u,i,j)\in\mathcal{B}} \left|\hat{y}^{(t)}_{uj} - \hat{y}^{(s)}_{uj}\right|^2, \end{aligned} \quad (13)$$

where $|\mathcal{B}|$ denotes the size of the sampled batch $\mathcal{B}$. Note that the distance evaluation can be further updated to more sophisticated metrics, such as KL-divergences [17] or Wasserstein distances [11]. However, in this study, we choose to use a simple metric to demonstrate the effectiveness of our approach.

3.3 Ranking Aligning Distillation

In this subsection, we aim to align the students with content features with behavior embeddings to the teachers by introducing a ranking aligning distillation loss. We use the Bayesian Personalized Ranking (BPR) discrepancy to measure the ranking capacity of the warm and cold models, as shown in Eq. (14):

$$\begin{aligned} r^{(t)}_{uij} &= \sigma\left(\hat{y}^{(t)}_{ui} - \hat{y}^{(t)}_{uj}\right), \\ r^{(s)}_{uij} &= \sigma\left(\hat{y}^{(s)}_{ui} - \hat{y}^{(s)}_{uj}\right), \end{aligned} \quad (14)$$

where $(u, i, j) \in O$ denotes a batch sampled from Eq. (11) and $\sigma(\cdot)$ is the Sigmoid function. $r^{(t)}_{uij}$ represents the extent to which teachers can score a positive item higher than a negative item, while $r^{(s)}_{uij}$ represents the extent to which students can do the same.

The BPR discrepancy maps the difference in ratings between positive and negative items to a range of $(0, 1)$ using the Sigmoid function. This discrepancy can be used to calculate a distillation loss by treating it as the logits for a binary classification prediction, as proposed in [18]. The ranking distillation loss, which is calculated using the formula in Eq. (15), is defined as follows:

$$\mathcal{L}_{rank} = -\sum_{(u,i,j)\in\mathcal{B}} \left(r^{(t)}_{uij} \ln r^{(s)}_{uij} + (1 - r^{(t)}_{uij}) \ln(1 - r^{(s)}_{uij})\right). \quad (15)$$

3.4 Identification Aligning Distillation

In this subsection, we aim to enhance the identification capability of the students by aligning it with the teachers. The behavioral embeddings of teachers effectively represent the behavioral characteristics of items and distinguish a teacher from other negatively sampled teachers. However, the content features of students do not effectively do the same. We compare the identification capacity of the teacher and student by measuring the distance between the embedding of a positive item and a randomly negative sampled item. The distance for a teacher or student can be computed as follows:

$$\begin{aligned} d^{(t)}_{ij} &= \sigma\left(\boldsymbol{e}_i^\top \cdot (\boldsymbol{e}_i - \boldsymbol{e}_j)\right), \\ d^{(s)}_{ij} &= \sigma\left(f_{\mathcal{I}}(\boldsymbol{c}_i)^\top \cdot (f_{\mathcal{I}}(\boldsymbol{c}_i) - f_{\mathcal{I}}(\boldsymbol{c}_j))\right), \end{aligned} \quad (16)$$

where $f_{\mathcal{I}}$ is the mapping function in Eq. (8) and $\sigma(\cdot)$ is the Sigmoid function to map the distance to $(0, 1)$ area.

To further improve the identification evaluation methods, we introduce a batch-wise approach. In this approach, we select a given positive item from the training batches as the query item and evaluate the distance between the teacher or student embeddings of the query item and the negative sampled items. Then, the identification distillation loss is defined as follows:

$$\mathcal{L}_{iden} = -\sum_{i\in\mathcal{B}_I} \sum_{j\in\mathcal{B}_J} \left(d^{(t)}_{ij} \ln d^{(s)}_{ij} + (1 - d^{(t)}_{ij}) \ln(1 - d^{(s)}_{ij})\right), \quad (17)$$

where $\mathcal{B}_I$ and $\mathcal{B}_J$ denote the sets of positive and negative items in the training batch $\mathcal{B}$. However, this approach has a complexity of $O(|\mathcal{B}|^2)$. To address this, we simplify the distillation loss by computing the average embedding of the negative sampling set, resulting in a complexity of $O(|\mathcal{B}|)$. The simplified distillation loss is defined as:

$$\mathcal{L}_{iden} = -\sum_{i\in\mathcal{B}_I} \left(\bar{d}^{(t)}_i \ln \bar{d}^{(s)}_i + (1 - \bar{d}^{(t)}_i) \ln(1 - \bar{d}^{(s)}_i)\right), \quad (18)$$

where $\bar{d}^{(t)}_i$ and $\bar{d}^{(s)}_i$ are the average identification distance for item i, which are computed as:

$$\bar{d}^{(t)}_{\mathcal{B}_J} = \frac{1}{|\mathcal{B}|} \sum_{j\in\mathcal{B}_J} \boldsymbol{e}_j, \quad \bar{d}^{(s)}_{\mathcal{B}_J} = \frac{1}{|\mathcal{B}|} \sum_{j\in\mathcal{B}_J} f_{\mathcal{I}}(\boldsymbol{c}_j). \quad (19)$$

$$\begin{aligned} \bar{d}^{(t)}_i &= \sigma\left(\boldsymbol{e}_i^\top \cdot (\boldsymbol{e}_i - \bar{d}^{(t)}_{\mathcal{B}_J})\right), \\ \bar{d}^{(s)}_i &= \sigma\left(f_{\mathcal{I}}(\boldsymbol{c}_i)^\top \cdot (f_{\mathcal{I}}(\boldsymbol{c}_i) - \bar{d}^{(s)}_{\mathcal{B}_J})\right). \end{aligned} \quad (20)$$

3.5 Teacher-qualifying Weighting Structure

In ALDI, we aim to align the students with the teachers by utilizing distillation techniques. Unlike traditional distillation methods that rely on the reliability and expertise of the teachers, our approach utilizes items with trained behavioral embeddings as teachers. However, due to potential inaccuracies in the learned embeddings, aligning the students to unqualified teachers may negatively impact the alignment. This is because the backbone recommendation model learns behavioral embeddings from historical behaviors, and there may be items without sufficient historical interactions to train accurate embeddings. Therefore, to mitigate this issue, we assign higher weights to qualified items with sufficient historical behaviors, while assigning lower weights to unqualified items. Specifically, we use a variant of the *tanh* function to control the distillation loss weight of each teacher:

$$w_i = \frac{2}{1 + e^{-\omega \cdot N_i / \bar{N}}} - 1, \tag{21}$$

where N_i is the number of historically interacted users for item i in the training set, $\bar{N}$ is the average number of historically interacted users for all warm items, and ω is a hyper-parameter that controls the intensity of weight gain as item frequency increases.

After incorporating the dynamic weighting of the teachers in the ranking distillation loss, the revised ranking aligning distillation loss can be updated with the following equation:

$$\mathcal{L}_{rank} = - \sum_{(u,i,j) \in \mathcal{B}} w_i \left(r_{uij}^{(t)} \ln r_{uij}^{(s)} + (1 - r_{uij}^{(t)}) \ln(1 - r_{uij}^{(s)}) \right). \tag{22}$$

Similarly, the revised identification aligning loss can be revised as:

$$\mathcal{L}_{iden} = - \sum_{i \in \mathcal{B}_I} w_i \left(\bar{d}_i^{(t)} \ln \bar{d}_i^{(s)} + (1 - \bar{d}_i^{(t)}) \ln(1 - \bar{d}_i^{(s)}) \right). \tag{23}$$

However, since students should have similar distributions as the teachers, regardless of the qualification of items, we do not adjust the weight of the rating distribution in the distillation loss.

4 EXPERIMENTS

We conduct comprehensive experiments on publicly available datasets to address the following three research questions:

- **Q1:** Can ALDI achieve superior overall, warm, and cold recommendation performance compared to state-of-the-art cold-start recommendation models?
- **Q2:** Can ALDI effectively reduce the differences in rating distribution, ranking, and identification between the teachers and the students?
- **Q3:** Is ALDI more effective in cold-start recommendation than compression-oriented distillation methods?

4.1 Experimental Setup

Datasets. We evaluate ALDI's performance on cold-start items using the CiteULike and XING datasets. CiteULike contains 5,551 users, 16,980 articles, and 204,986 interactions. The articles are represented by 300-dimensional vectors as item content features. XING is a subset of the ACM RecSys 2017 challenge dataset with 106,881 users, 20,519 jobs, and 4,306,183 interactions. The jobs' content is represented by a 2,738-dimensional vector that encodes attributes such as career level, tags, and other information. For each dataset, 20% of items are designated as cold-start items, with interactions split into a cold validation set and testing set (1:1 ratio). Records of the remaining 80% of items are divided into training, validation, and testing sets, using an 8:1:1 ratio.

Evaluation Metrics. We evaluate the overall, warm, and cold recommendation performance using a full-ranking evaluation approach [15, 47]. As defined in Section 2.1, we evaluate the overall, warm, and cold recommendation performance. We use precision@K, recall@K, and Normalized Discounted Cumulative Gain (NDCG@K) as metrics to evaluate the top-ranked articles. By default, K is set to 20 and the average values for all users in the testing set are reported.

Implementation Details. We implement the baselines using their officially provided implementations. In particular, for GAR, we use the updated version of the GAR implementation provided in the official repository, which is evaluated under the same CLCRec settings as we used in our papers[2]. The dimension of the embeddings is set to 200 for all models. We use the Adam optimizer with a learning rate of 0.001 and apply early stopping by observing NDCG@K on the validation set. The size of each training batch and the regularization weight are set to 1024 and 0.001, respectively. The hyperparameters α, β, and γ in Eq. (10) are tuned using a grid search. The ω in Eq. (21) is also tuned using a grid search. The best hyperparameters are found for each dataset. For fairness, we use the same options and follow the designs in their articles for all baselines.

Baselines. To evaluate ALDI's effectiveness and universality, we compare it to seven state-of-the-art cold-start recommendation models across three datasets:

- **DeepMusic**[43] uses deep neural networks to minimize the MSE difference between generated and warm embeddings.
- **MetaEmb**[33] trains a meta-learning-based generator for fast convergence.
- **GAR**[5] generates embeddings through a generative adversarial relationship with the warm recommendation model.
- **DropoutNet**[44] improves cold-start robustness by randomly discarding embeddings.
- **MTPR**[8] generates counterfactual cold embeddings considering dropout and BPR ranking.
- **Heater**[63] improves DropoutNet by using a mix-of-experts network and considering embedding similarity.
- **CLCRec**[50] models cold-start recommendation with contrastive learning from an information-theoretic viewpoint.
- **PGD**[46] is a graph-based cold-start recommendation model that learns the correlation between collaborative signal and attributed heterogeneous graph.

4.2 Main Results (Q1)

The comparison of overall (Eq. (1)), warm (Eq. (2)), and cold (Eq. (3)) recommendation performance between ALDI and other baselines on two datasets is presented in Table 1. To evaluate the effectiveness of ALDI, we conduct cold-start experiments using traditional Matrix Factorization (MF) [36], Neural Collaborative Filtering (NCF) [16],

[2] https://github.com/zfnWong/GAR

Table 1: Overall, cold and warm recommendation performance comparison over three backbone models (MF, NCF, LightGCN).

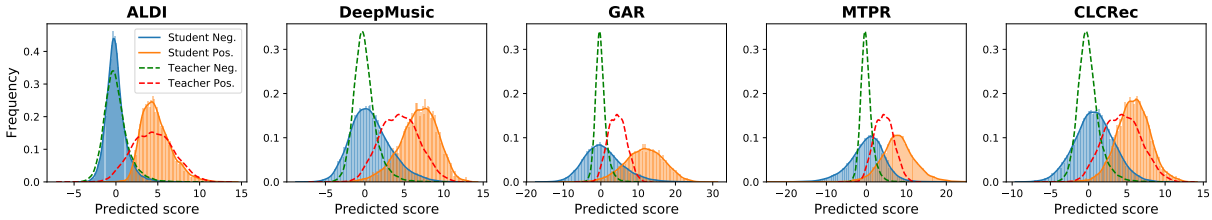
	Method	Overall Recommendation				Cold Recommendation				Warm Recommendation			
		CiteULike		XING		CiteULike		XING		CiteULike		XING	
		Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
MF	Backbone	.0776	.0647	.1444	.1276	.0056	.0031	.0037	.0021	**.2838**	**.1933**	**.4505**	**.2946**
	DeepMusic	.0956	.0789	.1876	.1467	.2141	.1262	.2691	.1606	**.2838**	**.1933**	**.4505**	**.2946**
	MetaEmb	.0972	.0804	.1569	.1350	.2232	.1306	.2698	.1554	**.2838**	**.1933**	**.4505**	**.2946**
	GAR	.1440	.1132	.2381	.2105	.2453	.1479	.2945	.2192	.2272	.1438	.4091	.2708
	DropoutNet	.0794	.0670	.1733	.1454	.2268	.1356	.2773	.1953	.1343	.0792	.3304	.2140
	MTPR	.1060	.0810	.2100	.1750	.2496	.1476	.3314	.2299	.1728	.0998	.3834	.2532
	Heater	.1134	.0913	.2059	.1675	.2372	.1419	.2885	.1978	.1871	.1153	.3622	.2334
	CLCRec	.1269	.0992	.2188	.1845	.2295	.1347	.3184	.2272	.1898	.1167	.3568	.2397
	ALDI	**.1618**	**.1204**	**.2727**	**.2223**	**.2684**	**.1550**	**.3595**	**.2521**	**.2838**	**.1933**	**.4505**	**.2946**
	%Improv.	12.36%	6.36%	14.53%	5.61%	7.53%	4.80%	8.48%	9.66%	-	-	-	-
NCF	Backbone	.0760	.0604	.1364	.1210	.0055	.0028	.0047	.0026	**.1920**	**.1142**	**.4294**	**.2882**
	DeepMusic	.0924	.0739	.1763	.1538	.1882	.1056	.2689	.1669	**.1920**	**.1142**	**.4294**	**.2882**
	MetaEmb	.0903	.0736	.1726	.1530	.2034	.1179	.2842	.1734	**.1920**	**.1142**	**.4294**	**.2882**
	GAR	.1224	.0947	.2231	.1888	.2149	.1260	.2985	.2067	.1875	.1113	.4034	.2731
	DropoutNet	.0837	.0677	.1832	.1547	.1986	.1180	.3089	.1852	.1582	.0925	.3520	.2318
	MTPR	.1041	.0810	.1846	.1538	.2102	.1204	.3332	.2262	.1730	.1048	.3669	.2337
	Heater	.0936	.0752	.2031	.1669	.2274	.1326	.2974	.2221	.1717	.1027	.3751	.2399
	CLCRec	.1047	.0820	.2117	.1750	.2197	.1257	.2989	.2155	.1665	.0986	.3537	.2346
	ALDI	**.1428**	**.1068**	**.2453**	**.2036**	**.2429**	**.1394**	**.3551**	**.2608**	**.1920**	**.1142**	**.4294**	**.2882**
	%Improv.	16.67%	12.78%	9.95%	7.84%	6.82%	5.13%	6.57%	15.30%	-	-	-	-
LightGCN	Backbone	.0812	.0622	.1311	.1161	.0041	.0019	.0045	.0026	**.2528**	**.1541**	**.4248**	**.2919**
	DeepMusic	.0985	.0745	.1651	.1460	.2239	.1259	.2870	.1702	**.2528**	**.1541**	**.4248**	**.2919**
	MetaEmb	.0924	.0714	.1696	.1494	.2252	.1295	.2823	.1764	**.2528**	**.1541**	**.4248**	**.2919**
	GAR	.1357	.1062	.2205	.1848	.2539	.1489	.3017	.2171	.2339	.1455	.4131	.2793
	DropoutNet	.0883	.0639	.1661	.1431	.2309	.1312	.2732	.1860	.1175	.0692	.3388	.2182
	MTPR	.1001	.0753	.1928	.1648	.2585	.1454	.3254	.1955	.1753	.0967	.3930	.2618
	Heater	.1118	.0894	.2095	.1830	.2438	.1407	.3271	.2224	.1946	.1193	.3982	.2708
	CLCRec	.1293	.0965	.2067	.1791	.2435	.1425	.3087	.2138	.2149	.1302	.3925	.2650
	PGD	-	-	.2099	.1725	-	-	.3245	.2284	-	-	.3491	.2215
	ALDI	**.1626**	**.1201**	**.2409**	**.2041**	**.2692**	**.1539**	**.3377**	**.2356**	**.2528**	**.1541**	**.4248**	**.2919**
	%Improv.	19.82%	13.09%	9.25%	10.44%	4.14%	3.36%	3.24%	3.15%	-	-	-	-

and graph-based LightGCN [15] models as representative recommendation models. For each backbone model, we present the cold-start performance of the backbone model, three generative models (DeepMusic, MetaEmb, and GAR), and four dropout models (DropoutNet, MTPR, Heater, and CLCRec), as well as the performance of the graph-based model PGD. The improvements are calculated by comparing ALDI to the best baseline for each backbone, which is highlighted by underlining.

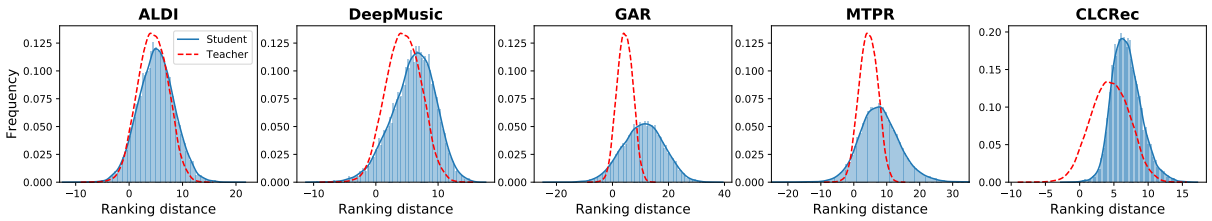
- ALDI outperforms all seven baselines in terms of overall and cold recommendation performance across all datasets and backbones. This success is attributed to the alignment of students and teachers in ALDI, which improves its recommendation performance.
- In the comparison of each backbone, GAR is the best baseline for overall recommendation performance on both CiteULike and XING. The generative models also consistently perform better in warm recommendation than dropout models.
- The use of randomly initialized embeddings as the cold-start embedding results in the worst recommendation performance across all datasets and backbones, emphasizing the importance of designing effective cold-start models.

4.3 Aligning Results (Q2)

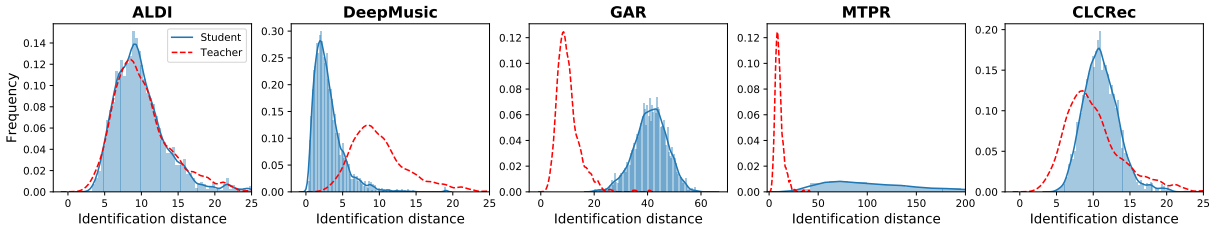
In Figure 3, we present the comparison of teachers and items in terms of our three proposed differences (§ 3.1.2) to determine whether ALDI aligns students with teachers best. Figure 3(a) shows the rating distribution of positive and negative user-item pairs



(a) Rating distribution difference comparison.



(b) Ranking difference comparison.



(c) Identification difference comparison.

Figure 3: Comparison of teacher (MF) and student (cold-start) models' rating distribution, ranking, and identification differences on CiteULike dataset.

as rated by teachers ($\hat{y}_{ui}^{(t)}$ and $\hat{y}_{uj}^{(t)}$) and students ($\hat{y}_{ui}^{(s)}$ and $\hat{y}_{uj}^{(s)}$). Figure 3(b) presents the distribution of BPR discrepancy for teachers ($r_{uij}^{(t)}$) and students ($r_{uij}^{(s)}$). Figure 3(c) shows the distribution of identification distance for teachers ($\bar{d}_{uij}^{(t)}$) and students ($\bar{d}_{uij}^{(s)}$). Comparing ALDI to four baselines, we conclude that it aligns students better with teachers than other methods. In Figure 3, we present the comparison of teachers and items in terms of our three proposed differences (§ 3.1.2) to determine whether ALDI aligns students with teachers best. Figure 3(a) shows the rating distribution of positive and negative user-item pairs as rated by teachers ($\hat{y}_{ui}^{(t)}$ and $\hat{y}_{uj}^{(t)}$) and students ($\hat{y}_{ui}^{(s)}$ and $\hat{y}_{uj}^{(s)}$). Figure 3(b) presents the distribution of ranking distance for teachers ($\hat{y}_{ui}^{(t)} - \hat{y}_{uj}^{(t)}$) and students ($\hat{y}_{ui}^{(s)} - \hat{y}_{uj}^{(s)}$). Figure 3(c) shows the distribution of identification distance without $\sigma(\cdot)$ for teachers ($\bar{d}_{uij}^{(t)}$) and students ($\bar{d}_{uij}^{(s)}$). Note that the Sigmoid function $\sigma(\cdot)$ has been omitted in Eq. (14) and Eq. (19) for improved visualization clarity. Reading Figure 3, we can observe that ALDI outperforms other baselines in approximating the teacher's distribution, and even matches it, demonstrating that ALDI effectively reduces the gap between students and teachers compared to baseline models.

4.4 Distillation Comparison(Q3)

In Table 2, we compare ALDI with two traditional ranking distillation models: Ranking Distillation (RD)[40] and Collaborative Distillation (CD)[25]. RD focuses on modeling the ranking order difference between the teacher and student models, while CD prioritizes the prediction scores of the student on top-ranked items from the teacher. As indicated by the results in the table, our ALDI consistently outperforms RD and CD in terms of Recall and NDCG across both the CiteULike and XING datasets with three different backbones. ALDI demonstrates superiority over CD by 17.83% on the CiteULike dataset and by 11.21% on the XING dataset. This supports the effectiveness of our tailored designed distillation loss in improving performance in cold-start recommendation tasks.

Ablation Study. The figure in Figure 4 demonstrates the impact of the hyperparameter ω on the recommendation performance on CiteULike and XING with the MF backbone. The best results were achieved when ω was set to 5 for CiteULike and 4 for XING. The results indicate that a moderate intensity of ω is crucial in avoiding the learning of unreliable teacher information.

5 RELATED WORK

5.1 Embedding-based Recommendation

Recommender systems are designed to recommend personalized items to billions of users out of millions of items [53, 60, 61]. One

Table 2: Comparing ALDI with ranking distillation recommendation methods for overall recommendation performance. The improvements are computed by comparing ALDI with the best baselines (underlined).

Backbone	Variant	CiteULike		XING	
		Recall	NDCG	Recall	NDCG
MF	RD	0.1269	0.0932	0.2089	0.1674
	CD	0.1336	0.0972	0.2385	0.1999
	ALDI	**0.1618**	**0.1204**	**0.2727**	**0.2223**
	%Improv.	21.11%	23.87%	14.34%	11.21%
NCF	RD	0.0920	0.0748	0.1854	0.1433
	CD	0.1032	0.0799	0.2152	0.1806
	ALDI	**0.1428**	**0.1068**	**0.2453**	**0.2036**
	%Improv.	38.37%	33.67%	13.99%	12.74%
LightGCN	RD	0.1234	0.0917	0.2093	0.1669
	CD	0.1380	0.1002	0.2132	0.1742
	ALDI	**0.1626**	**0.1201**	**0.2409**	**0.2041**
	%Improv.	17.83%	19.86%	12.99%	17.16%

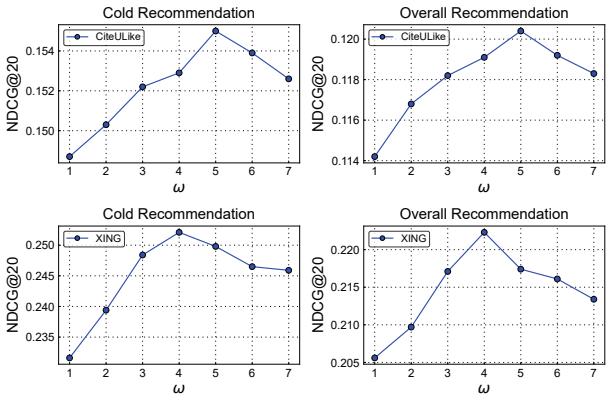


Figure 4: Performance of the proposal w.r.t. different ω in dynamic weight adjustment.

common technique is to represent each user or item as an embedding vector. To this end, MF-based models [16, 24, 54] learn the embeddings by factorizing the user-item interaction matrix into user embedding vectors and item embedding vectors.

Recently, Graph Neural Networks (GNNs)[3, 4, 6, 20] have been effective for multiple graph-based learning tasks, such as node classification and link prediction. Since the user-item interactions can be seen as the edges between users and items, the entire user-item interaction records can be reformulated as user-item graphs. Thus, motivated by the success of GNNs, NGCF[47] first adapts GNN-based collaborative filtering to learn the user/item embeddings. To accelerate the graph convolution process, LightGCN [15] skips the non-linear layers and achieves state-of-the-art recommendation performance.

5.2 Cold-start Recommendation

The cold-start problem, which refers to the difficulty in recommending items to new users with limited interaction history, has been addressed by several approaches in recommendation systems [10, 48]. Generative models, such as DeepMusic [43] and meta-learning-based methods [33], incorporate user and item contents and generate mappings from cold item embeddings to warm item embeddings. The use of Generative Adversarial Networks (GANs) such as RA-GAN [2], LARA [39], and GAR [5] has also proven effective in solving the cold-start problem.

Dropout models, such as DropoutNet [44], MTPR [8], CC-CC [37], Heater [63], VELF [52], and CLCRec [50], simulate the behavior of cold users and items by randomly dropping the trained embeddings during training. These methods differ in the manner in which the embeddings are dropped, ranging from replacing them with zero vectors to incorporating them into inference for CTR prediction.

5.3 Knowledge Distillation

In knowledge distillation, a simple student model is trained to mimic a complex teacher model to achieve similar or better performance [18, 42]. This technique is widely used in various fields of AI, including visual recognition, NLP, and recommender systems. For example, knowledge distillation has been applied to image classification [1, 27, 35], object detection [26, 38], face recognition [23, 31], and image/video segmentation [14, 32]. In NLP, knowledge distillation is used to create lightweight language models for various tasks [12, 19, 29, 57].

Recently, knowledge distillation has been applied to recommender systems, especially in ranking distillation [22, 25, 34, 40], adversarial distillation [7, 49], and privileged feature distillation [46, 51, 55]. These methods aim to improve the performance and efficiency of recommender systems by transferring knowledge from a complex teacher model to a simpler student model. However, they overlook the aligning distillation in cold-start recommendations, where the teachers have to work together with the students.

6 CONCLUSION

In conclusion, the Aligning Distillation (ALDI) framework is proposed to tackle the challenge of recommending cold items in recommendation systems by narrowing the inherent gap between warm items and cold items. ALDI effectively addresses the differences between warm and cold recommendations by transferring behavioral information from warm items to cold items and incorporating a teacher-qualifying weighting structure to ensure accuracy. Experiments on three datasets demonstrate the superiority of ALDI over state-of-the-art baselines in terms of overall, warm, and cold recommendation performance over three different but typical warm recommendation backbones. Besides, ablations studies present ALDI achieves superior cold-start recommendation performance than ranking distillation recommender systems.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China (No. U22A2095, 62032020, 62272200).

REFERENCES

[1] Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, and Ali Farhadi. 2018. Label refinery: Improving imagenet classification through label progression. *arXiv preprint arXiv:1805.02641* (2018).

[2] Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jaeho Choi. 2019. Rating Augmentation with Generative Adversarial Networks towards Accurate Collaborative Filtering. In *WWW*.

[3] Hao Chen, Wenbing Huang, Yue Xu, Fuchun Sun, and Zhoujun Li. 2020. Graph unfolding networks. In *CIKM*.

[4] Hao Chen, Zhong Huang, Yue Xu, Zengde Deng, Feiran Huang, Peng He, and Zhoujun Li. 2022. Neighbor enhanced graph convolutional networks for node classification and recommendation. *Knowledge-Based Systems* 246 (2022), 108594.

[5] Hao Chen, Zefan Wang, Feiran Huang, Xiao Huang, Yue Xu, Yishi Lin, Peng He, and Zhoujun Li. 2022. Generative Adversarial Framework for Cold-Start Item Recommendation. In *SIGIR*.

[6] Hao Chen, Yue Xu, Feiran Huang, Zengde Deng, Wenbing Huang, Senzhang Wang, Peng He, and Zhoujun Li. 2020. Label-aware graph convolutional networks. In *CIKM*.

[7] Xu Chen, Yongfeng Zhang, Hongteng Xu, Zheng Qin, and Hongyuan Zha. 2018. Adversarial Distillation for Efficient Recommendation with External Knowledge. *TOIS* 37, 1, Article 12 (dec 2018).

[8] Xiaoyu Du, Xiang Wang, Xiangnan He, Zechao Li, Jinhui Tang, and Tat-Seng Chua. 2020. How to Learn Item Representation for Cold-Start Multimedia Recommendation?. In *MM*.

[9] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-Guided Heterogeneous Graph Neural Network for Intent Recommendation. In *KDD*.

[10] Wenjing Fu, Zhaohui Peng, Senzhang Wang, Yang Xu, and Jin Li. 2019. Deeply Fusing Reviews and Contents for Cold Start Users in Cross-Domain Recommendation Systems. In *AAAI*.

[11] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. 2017. Improved Training of Wasserstein GANs. In *NeurIPS*.

[12] Md. Akmal Haidar and Mehdi Rezagholizadeh. 2019. TextKD-GAN: Text Generation Using Knowledge Distillation and Generative Adversarial Networks. In *ICAAI*.

[13] James Hale. 2019. More than 500 hours of content are now being uploaded to YouTube every minute. https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/

[14] Tong He, Chunhua Shen, Zhi Tian, Dong Gong, Changming Sun, and Youliang Yan. 2019. Knowledge Adaptation for Efficient Semantic Segmentation. In *CVPR*.

[15] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*.

[16] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*.

[17] John R Hershey and Peder A Olsen. 2007. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *ICASSP*.

[18] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[19] Minghao Hu, Yuxing Peng, Furu Wei, Zhen Huang, Dongsheng Li, Nan Yang, and Ming Zhou. 2018. Attention-Guided Answer Distillation for Machine Reading Comprehension. In *EMNLP*.

[20] Zhongyu Huang, Yingheng Wang, Chaozhuo Li, and Huiguang He. 2022. Going Deeper into Permutation-Sensitive Graph Neural Networks. In *International Conference on Machine Learning*. PMLR.

[21] Qing-Yuan Jiang, Yi He, Gen Li, Jian Lin, Lei Li, and Wu-Jun Li. 2019. SVD: A Large-Scale Short Video Dataset for Near-Duplicate Video Retrieval. In *ICCV*.

[22] SeongKu Kang, Junyoung Hwang, Wonbin Kweon, and Hwanjo Yu. 2020. DE-RRD: A knowledge distillation framework for recommender system. In *CIKM*.

[23] Hanyang Kong, Jian Zhao, Xiaoguang Tu, Junliang Xing, Shengmei Shen, and Jiashi Feng. 2019. Cross-resolution face recognition via prior-aided face hallucination and residual knowledge distillation. *arXiv preprint arXiv:1905.10777* (2019).

[24] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[25] Jae-woong Lee, Minjin Choi, Jongwuk Lee, and Hyunjung Shim. 2019. Collaborative distillation for top-N recommendation. In *ICDM*.

[26] Quanquan Li, Shengying Jin, and Junjie Yan. 2017. Mimicking Very Efficient Network for Object Detection. In *CVPR*.

[27] Zhizhong Li and Derek Hoiem. 2018. Learning without Forgetting. *TPAMI* 40, 12 (2018), 2935–2947.

[28] Hank Liao, Erik McDermott, and Andrew Senior. 2013. Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription. In *ASRU*.

[29] Jian Liu, Yubo Chen, and Kang Liu. 2019. Exploiting the Ground-Truth: An Adversarial Imitation Based Knowledge Distillation Approach for Event Detection. In *AAAI*.

[30] Zhicong Lu, Haijun Xia, Seongkook Heo, and Daniel Wigdor. 2018. You watch, you give, and you engage: a study of live streaming practices in China. In *CHI*.

[31] Ping Luo, Zhenyao Zhu, Ziwei Liu, Xiaogang Wang, and Xiaoou Tang. 2016. Face model compression by distilling knowledge from neurons. In *AAAI*.

[32] Ravi Teja Mullapudi, Steven Chen, Keyi Zhang, Deva Ramanan, and Kayvon Fatahalian. 2019. Online Model Distillation for Efficient Video Inference. In *ICCV*.

[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 *SIGIR*.

[34] Yiteng Pan, Fazhi He, and Haiping Yu. 2019. A novel enhanced collaborative autoencoder with knowledge distillation for top-N recommender systems. *Neurocomputing* 332 (2019), 137–148.

[35] Zhimao Peng, Zechao Li, Junge Zhang, Yan Li, Guo-Jun Qi, and Jinhui Tang. 2019. Few-Shot Image Recognition With Knowledge Transfer. In *ICCV*.

[36] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.

[37] Shaoyun Shi, Min Zhang, Xinxing Yu, Yongfeng Zhang, Bin Hao, Yiqun Liu, and Shaoping Ma. 2019. Adaptive Feature Sampling for Recommendation with Missing Content Feature Values. In *CIKM*.

[38] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. 2017. Incremental Learning of Object Detectors Without Catastrophic Forgetting. In *ICCV*.

[39] Changfeng Sun, Han Liu, Meng Liu, Zhaochun Ren, Tian Gan, and Liqiang Nie. 2020. LARA: Attribute-to-feature Adversarial Learning for New-item Recommendation. In *WSDM*.

[40] Jiaxi Tang and Ke Wang. 2018. Ranking Distillation: Learning Compact Ranking Models With High Performance for Recommender System. In *KDD*.

[41] John C. Tang, Gina Venolia, and Kori M. Inkpen. 2016. Meerkat and Periscope: I Stream, You Stream, Apps Stream for Live Streams. In *CHI*.

[42] Gregor Urban, Krzysztof J. Geras, Samira Ebrahimi Kahou, Ozlem Aslan, Shengjie Wang, Abdelrahman Mohamed, Matthai Philipose, Matt Richardson, and Rich Caruana. 2017. Do Deep Convolutional Nets Really Need to be Deep and Convolutional?. In *ICLR*.

[43] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *NeurIPS*.

[44] Maksims Volkovs, Guang Wei Yu, and Tomi Poutanen. 2017. DropoutNet: Addressing Cold Start in Recommender Systems. In *NeurIPS*.

[45] Haoyu Wang, Defu Lian, and Yong Ge. 2019. Binarized collaborative filtering with distilling graph convolutional networks. *arXiv preprint arXiv:1906.01829* (2019).

[46] Shuai Wang, Kun Zhang, Le Wu, Haiping Ma, Richang Hong, and Meng Wang. 2021. Privileged Graph Distillation for Cold Start Recommendation. In *SIGIR*.

[47] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*.

[48] Xinghua Wang, Zhaohui Peng, Senzhang Wang, Philip S Yu, Wenjing Fu, Xiaokang Xu, and Xiaoguang Hong. 2020. CDLFM: cross-domain recommendation for cold-start users via latent feature mapping. *Knowledge and Information Systems* 62 (2020).

[49] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2018. Kdgan: Knowledge distillation with generative adversarial networks. *NeurIPS* (2018).

[50] Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. 2021. Contrastive Learning for Cold-Start Recommendation. In *MM*.

[51] Chen Xu, Quan Li, Junfeng Ge, Jinyang Gao, Xiaoyong Yang, Changhua Pei, Fei Sun, Jian Wu, Hanxiao Sun, and Wenwu Ou. 2020. Privileged Features Distillation at Taobao Recommendations. In *KDD*.

[52] Xiaoxiao Xu, Chen Yang, Qian Yu, Zhiwei Fang, Jiaxing Wang, Chaosheng Fan, Yang He, Changping Peng, Zhangang Lin, and Jingping Shao. 2022. Alleviating Cold-Start Problem in CTR Prediction with A Variational Embedding Learning Framework. In *WWW*.

[53] Peiyan Zhang, Jiayan Guo, Chaozhuo Li, Yueqi Xie, Jae Boum Kim, Yan Zhang, Xing Xie, Haohan Wang, and Sunghun Kim. 2023. Efficiently Leveraging Multi-level User Intent for Session-based Recommendation via Atten-Mixer Network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*.

[54] Yiding Zhang, Chaozhuo Li, Xing Xie, Xiao Wang, Chuan Shi, Yuming Liu, Hao Sun, Liangjie Zhang, Weiwei Deng, and Qi Zhang. 2022. Geometric Disentangled Collaborative Filtering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[55] Yuan Zhang, Xiaoran Xu, Hanning Zhou, and Yan Zhang. 2020. Distilling structured knowledge into embeddings for explainable and accurate recommendation. In *Proceedings of the 13th international conference on web search and data mining*.

[56] Jun Zhao, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. 2019. IntentGC: A Scalable Graph Convolution Framework Fusing Heterogeneous Information for Recommendation. In *KDD*.

[57] Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding Knowledge Distillation in Non-autoregressive Machine Translation. In *ICLR*.

[58] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep Interest Evolution Network for Click-Through Rate

Prediction. In *AAAI*.
[59] Guorui Zhou, Xiaoqiang Zhu, Chengru 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 *KDD*.
[60] Huachi Zhou, Jiaqi Fan, Xiao Huang, Ka Ho Li, Zhenyu Tang, and Dahai Yu. 2022. Multi-Interest Refinement by Collaborative Attributes Modeling for Click-Through Rate Prediction. In *CIKM*.
[61] Huachi Zhou, Qiaoyu Tan, Xiao Huang, Kaixiong Zhou, and Xiaoling Wang. 2021. Temporal augmented graph neural networks for session-based recommendations. In *SIGIR*.
[62] 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 *SIGIR*.
[63] Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. 2020. Recommendation for New Users and New Items via Randomized Training and Mixture-of-Experts Transformation. In *SIGIR*.