Research on Improving Online Recommendations Involving Sparse Interaction Data

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2024

Zhi LI

List of Publications

1. Journal Paper

1. Z., Li, D., Amagata, Y., Zhang, T., Maekawa, T., Hara, K., Yonekawa, and M., Kurokawa, HML4Rec: Hierarchical Meta-learning for Cold-start Recommendations in Flash Sale E-commerce, Knowledge-Based Systems (KBS), vol.255, pp.109674, 2022.

2. International Conference Paper

1. Z., Li, D., Amagata, T., Maekawa, K., Yonekawa, M., Kurokawa, and T., Hara, Trends-enhanced Attention & Memory Networks for E-commerce Recommendations, *Proc. Int'l ACM SIGIR Workshop On eCommerce (SIGIReCom workshop)*, 2022.

2. Z., Li, D., Amagata, Y., Zhang, T., Hara, S., Haruta, K., Yonekawa, and M., Kurokawa, Debiasing Graph Transfer Learning via Item Semantic Clustering for Cross-Domain Recommendations, *Proc. IEEE Int'l Conf. on Big Data (BigData)*, pp.762-769, 2022.

3. Z., Li, D., Amagata, Y., Zhang, T., Hara, S., Haruta, K., Yonekawa, and M., Kurokawa, Semantic Relation Transfer for Non-overlapped Cross-domain Recommendations, *Proc. The Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD)*, vol.13937, pp.271-283, 2023.

4. Z., Li, R., Takeda, and T., Hara, Meta-domain Adversarial Contrastive Learning for Alleviating Individual Bias in Self-sentiment Predictions, *Proc. Annu. Conf. of the Int. Speech Communication Association (INTERSPEECH)*, pp.2428-2432, 2023.

3. Domestic Conference Paper

1. Z., Li, B., Gao, D., Amagata, T., Maekawa, T., Hara, H., Niu, K., Yonekawa, and M., Kurokawa, Trends Tracking Memory Recommender Networks for Product Recommendations in E-commerce, *Proc. DEIM Forum*, 2020.

Abstract

With the explosive growth of the number of online services and the items (i.e., products) they provide, it becomes extremely time-consuming for users to explore their interested products from the countless available ones. Recommender systems (RSs) are developed to generate personal recommendations for users by modeling their preferences towards items, where the recommended items are empirically shown to be superior in satisfying users' needs, resulting in substantial value for service providers as well as users. The tremendous commercial profits of RSs motivate flourishing research that strives to improve the performances of RSs in the past few decades.

Thanks to the rapid development of deep learning models, deep learning-based RSs have demonstrated their effectiveness in accurately modeling users' preferences and also practically boosting commercial profits over the past few years. Such RSs, however, require a large amount of users' item-interacted data to learn their parameters due to the data-hungry nature of deep learning. This requirement significantly limits the practical implementation of deep learning-based RSs, especially in cases where there is only a limited number of interactions for each user or each item, a scenario often referred to as sparse interaction data (i.e., the sparse scenario). As the mainstream solution to tackle the issue of sparse interaction data, transfer learning-based RSs are widely studied. Transfer learning can be applied inner one service or across two services. The former transfers knowledge from users or items that have relatively sufficient interactions to the ones with sparse interactions. The latter transfers knowledge from an external service with sufficient data to facilitate the learning of user preferences in the sparse local service.

However, user preferences vary from service to service and even differ in different periods of the same service. Current transfer learning-based approaches ignore these changing user preferences, leading to a significantly limited performance on recommendations. Since sparse scenarios and changing user preferences are ubiquitous in real-world services, handling changing user preferences in sparse scenarios becomes an essential and critical demand when developing RSs for practical services. By capturing such preferences accurately, RSs are expected to further boost business revenue and better satisfy users' needs by providing high-preference recommendations.

Based on the above observations, our goal in this thesis is to improve recommendation performances by developing RSs that can handle changing user preferences and work well in sparse scenarios. Recall that user preferences change in different periods of the same service and vary from service to service, this thesis studies both types of changes in user preferences. For the first type of changes, we study the RSs for flash sale e-commerce services, which is a typical scenario that user preferences change very frequently (i.e., every few weeks). The challenge of handling changing user preferences in such scenario have not been discussed in the literature yet. If our RSs can effectively handle the frequently changed user preferences in flash sale scenario, it is reasonable to expect that our RSs can work well in other sparse scenarios with changing user preferences. For the scenarios that user preferences are different between two services, we studied RSs that improve recommendations in the local sparse service by effectively leveraging data from external services to boost the poor recommendation performance in previous RSs. External data include public data and private commercial data, this thesis, hence, studies how to effectively leverage both types of external data. We elaborate on the details of the three practical scenarios we studied and introduce the challenges still required to be tackled in these scenarios below.

i) In a flash sale e-commerce scenario, such as Amazon daily deals, available and discounted items are periodically changing based on the current sale strategies. Users are attracted by the high discounts and show period-specific interactions. The frequent changes of interacted items provoke sparse interactions. Periodically changed interactions also represent users' period-specific preferences. Considering the periodic changes in user preferences, it is reasonable to learn users' preferences in a period by leveraging only the interactions in that period. However, previous works simply combine users' interaction data in all the past periods and learn only a uniform preference, making such works difficult to perform well in different sale periods. Therefore, modeling period-specific user preferences with only limited interactions is still a challenge in flash sale e-commerce.

ii) The second scenario is to effectively leverage rich public data to enhance local recommendations. In real-world online services, most users interact with only a small number of items (sparse interactions), occurring not only in new services and small companies but also in established services and large companies. Due to the data-hungry

nature of deep learning models, services with sparse interactions have a practical demand for leveraging rich public data to enhance their recommendations, where the public data are released by service providers for the purpose of organizing competitions and promoting researches, such as the Amazon Review data released by Amazon Inc.. Cross-domain recommender systems (CDRSs) are a promising approach that can facilitate the recommendations in the sparse target domain by transferring knowledge from the interaction data in an auxiliary source domain, where the sparse target domain and the source domain are, respectively, the local service and the external service that provides public data.

However, existing CDRSs cannot effectively leverage rich public data due to a so-called negative transfer issue which generally leads to worse recommendation accuracy compared to single-domain methods. This issue is caused by the misleading of the source interactions that present users' unique interests within the source domain, which is also called domain-specific preferences. Existing CDRSs solve this issue by merging individual interactions across domains. With the individually merged interactions, these systems can assign a minimal weight to such source interactions, thereby, alleviating their impact. However, merging individual interactions between domains requires these methods to identify the users having interactions in both domains. These users are commonly referred to as domain-shared users. Identifying such users has to match individuals (i.e. user matching) through shared user information, e.g., user profiles or other characteristics that facilitate individual identification, between domains. In public data, the user characteristics that can identify individuals are definitely unavailable due to the privacy issue. Therefore, it is impractical to identify domain-shared users in the case of leveraging public data. In other words, effectively leveraging public data to improve local recommendations is still a challenge because of negative transfer issue.

iii) The third practical scenario is to effectively leverage commercial data to enhance local recommendations. Public data may suffer from incompleteness or outdated information and may lack quality and accuracy due to limited support. Consequently, it is logical to utilize commercial data to enhance local recommendations by partnering with commercial corporations. However, addressing the negative transfer issue is more challenging in this scenario. While directly merging individual interactions is not feasible in the second scenario, it is possible to merge interactions from the source domain and the

target domain at a cluster level by jointly clustering items. Then, the weighting technology also can assign minimal scores for source interactions that present domain-specific preferences at a cluster level to handle the negative transfer issue. However, in the third scenario, merging interactions (i.e. sharing the information on interactions), even cluster-level interactions, is prohibited, as it may disclose personal preferences, behaviors, or patterns of individuals without their explicit consent to other companies. Due to the prohibition of merging interactions, it is impossible to explicitly identify and directly operate on source interactions that present domain-specific preferences when learning user preferences in the target domain. As a result, another type of entity is required to bridge domains and transfer knowledge, e.g., item clusters that contain items having similar textual features from both the source and the target domains. Without sharing interactions, the impact of source interactions has to be implicitly alleviated based on such entities. Different from the explicit and direct methods working on source interactions, it is hard to design implicit methods due to the difficulty in distinguishing how the operation on entities affects the impact of source interactions. This gives the reason why addressing the negative transfer issue is more challenging in this scenario.

In this thesis, we focus on improving recommendations with transfer learning-based methods and tackle the above-mentioned challenges. This thesis consists of five chapters. We introduce the research background and issues for improving recommendations in Chapter 1. In Chapter 2, we address the challenge of modeling period-specific preferences in flash sale e-commerce. In Chapter 3, we address the challenge of leveraging public data to improve target recommendations. In Chapter 4, we address the challenge of boosting target recommendations with external commercial data. Finally, in Chapter 5, we summarize this thesis and discuss our future work.

In Chapter 2, we introduce a novel meta-learning-based RS to model users' period-specific preferences. Moreover, we propose a new hierarchical meta-training algorithm to guide the learning of our recommendation model via user- and period-specific gradients. With the guidance of these gradients, the model can learn user- and period-share prior knowledge, supporting modeling users' period-specific preferences with only limited interactions and several updating steps. By doing so, our RS can quickly adapt to the recommendation tasks for new flash sale periods. Our experimental result on a real-world flash sale e-commerce dataset shows that our proposal remarkably outperforms

current state-of-the-art methods. This result also demonstrates the effectiveness of our proposal in modeling period-specific preferences.

In Chapter 3, to effectively leverage public data, we propose a novel CDRS that requires no domain-shared users and can handle the negative transfer issue. To remove the requirement of domain-shared users, we merge cluster-level interactions between domains by jointly clustering items from both domains. Besides, to handle the negative transfer issue, we construct a cross-domain interaction graph to transfer cluster-level interaction between domains, and propose a new debiasing graph convolutional layer to weight source cluster-level interactions. Constructing this graph only requires jointly clustering the items in the source domains and the ones in the target domains based on their semantic embeddings extracted from their text data, i.e., titles of products. This removes the requirement of domain-shared users to bridge the two domains. Our debiasing layer handles the negative transfer issue by adaptively weighting source interactions in the cross-domain graph to alleviate the impact of the source interactions that present users' unique interests within the source domain. Our experimental results on three public datasets and a pair of private datasets verify the advantages of our method over state-of-the-art models in terms of cross-domain recommendations and handling the negative transfer issue.

Different from Chapter 3, utilizing external commercial data requires no user interaction merging between domains due to privacy concerns. We, hence, develop a novel CDRS that does not merge any interactions between domains and can handle the more challenging negative transfer issue in Chapter 4. To remove the mergence of interactions, our CDRS constructs a similarity-based cross-domain graph to bridge domains and transfer knowledge. To handle the more challenging negative transfer issue, our CDRS transfers knowledge based on the cross-domain graph by focusing on the items that are from different domains and have similar textual features, i.e., similar titles. This alleviates the impact of the source items with irrelevant titles which cause the negative transfer issue. Our experimental results on real-world datasets show that our method significantly outperforms state-of-the-art cross-domain comparisons, which indicates the effectiveness of our proposal in the negative transfer issue.

Contents

Chapter 1

Introduction

With the explosive growth of online information and services, efficiently exploring useful content from the overwhelming amount of available services becomes a tough yet essential task in our daily lives. The personal recommendations provided by service providers can counterbalance the overload of information and are effective in reducing the time costs of users' exploration process.

Consequently, they are a central part of the overall user experiences on many online services, such as e-commerce and media streaming sites. In Figure 1.1, we sketch four types of online services, including e-commerce, video streaming services, social networking sites (SNS), and app stores. In these services, recommendations are central to the user experience. We also provide a screenshot of the personal recommendations provided by Amazon Japan. Recommender systems (RSs), which provide the above-mentioned personal recommendations, have become an essential component of many modern websites and online services. Therefore, they have been studied in academia and industry over the past few decades.

Figure 1.2 depicts a paradigm for an RS. To build an RS, service providers collect the history of user-item interactions (i.e., the observed interaction set in Figure 1.2) within their services, e.g., the records of what users have previously liked, purchased, or clicked. Then, the RS trains a learning model to predict users' preferences towards items (the item set in Figure 1.2) by using interaction records, where items include all the products, services, or content that the system is designed to suggest to users. After learning such a model, items are ranked based on the learned preferences, and the top

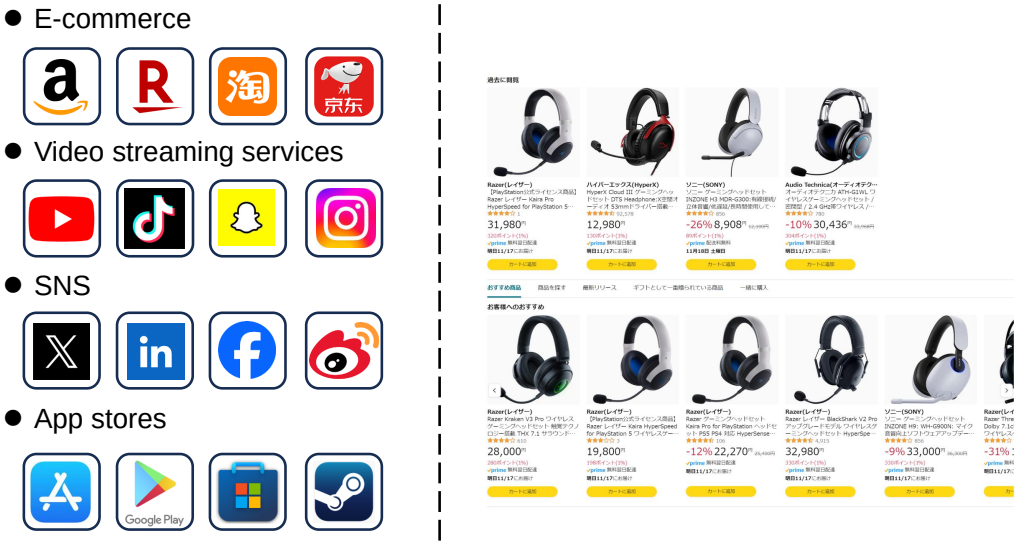


Figure 1.1: The left lists some leading online services providing different recommendations, including product recommendations in e-commerce, video recommendations in video streaming services, news recommendations in social networking sites (SNS), and app recommendations in app stores. The right shows an example of the personal recommendations provided by Amazon Japan, where the recommendations are generated based on historical browsing data.

part of items will be recommended to users engaged in these services (i.e., the user set in Figure 1.2). If preferences are learned precisely, recommendations are expected to satisfy the interests of users.

In industry, the performance of an RS is generally assessed by conducting an online A/B test, which measures the change of indicators in a certain period when adopting a new RS on a practical service. The indicators can be the click-through-rate (CTR), the conversion rate, or sales and revenue [1]. For example, in [2], the authors compared a "purchase-based" recommendation system with a baseline condition where no recommendations were provided to users on an online DVD retailer. Their comparison results revealed that the recommender system led to an increase in sales by 35%. Moreover, RSs producing higher performances are always demanded by service providers to im-

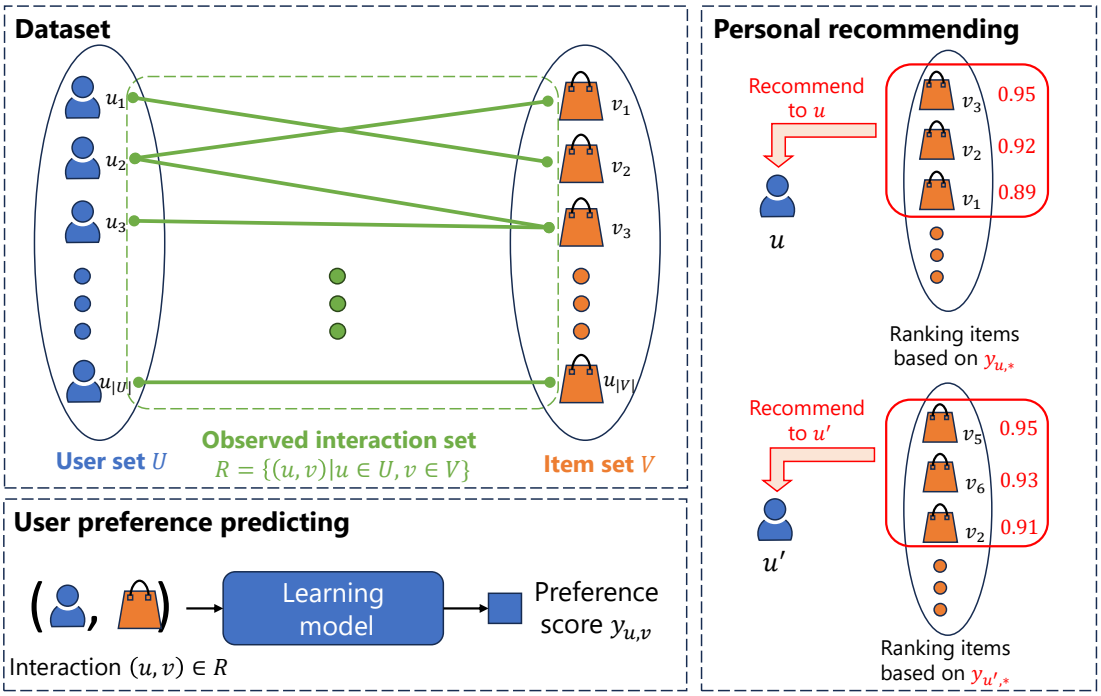


Figure 1.2: A paradigm of a recommender system. In such a system, recommended items for each user are generated by ranking all items based on preference scores. Such preference scores are predicted by a learning model given the input of all pairs of that user and items. To predict scores that can capture users' preferences, the learning model is trained based on users' interaction records (i.e., the interaction set) collected by service providers.

prove their commercial profits further. The enormous profits led by high-performance RSs have been reported in many studies [3, 4, 5, 6]. For example, in the study at YouTube [7], the difference in CTR between two RSs is as high as 200%. In another study at Google News [8], the improvement of CTR is 38% after replacing the method recommending the most popular items with their proposed recommendation method.

On the other hand, academic RSs are often assessed by offline experiments that measure the accuracy of their recommendations on held-out data. The offline experiments are based on an assumption: when an RS can rank items interacted by users higher in held-out data, the recommendations will better match user interests. Thus, RSs with higher offline accuracy are supposed to achieve a better performance in practi-

cal services [1]. The results in most of the above-mentioned literature studied by service providers also reveal a high level of alignment towards the performance of RSs between online tests and offline experiments.

We now see that improving the performance of RSs cannot only elevate the commercial profits for service providers but also reduce the costs in users' decision-making process. Therefore, it is necessary to study how to improve the performance of RSs, which motivates this thesis to focus on improving the performance of RSs.

In the past decade, the success of deep learning has drawn attention, motivating modern RSs that stem from deep learning and have empirically demonstrated their superiority in accurately modeling users' preferences and boosting the performance of recommendations [9]. Unfortunately, deep learning-based RSs always require a large amount of users' item-interacted data to learn their millions or even billions of parameters due to the data-hungry nature of deep learning [10, 11, 12, 13]. In other words, it is almost impossible to build a high-performance deep learning-based RS from scratch with only a limited number of interactions for most users. Such limited interactions are usually referred to as sparse interaction data. Sparse interactions (i.e., sparse scenarios) are ubiquitous and realistic in real-world services. This is because of a long tail phenomenon that most users interact with only a small number of items in real-world online services [14, 15]. This phenomenon is observed not only in new services and small companies but also in established services and large companies.

Transfer learning technology is the current mainstream solution to tackle the challenge of sparse interaction data. This technology can transfer knowledge from a well-learned task to a relevant new task, where the knowledge is defined as a recommendation model or embeddings for users and items [15, 16, 17, 18]. In established services and large companies (e.g., Amazon), even though most users have only a limited number of interactions (i.e., cold users), these services and companies still have a large amount of users having sufficient interaction data (i.e., warm users). Given this fact, previous studies transfer knowledge from warm users to cold users [15, 18, 19, 20, 21]. For example, transfer networks are developed to learn mapping functions for embeddings between warm and cold users [18, 21]. Unfortunately, these approaches are inapplicable in new services and small companies due to lacking sufficient warm users. To tackle this issue, Some studies transfer knowledge from external services with relatively sufficient inter-

actions to facilitate the learning of recommendation models in the sparse local service [16, 17].

However, current transfer learning-based approaches ignore the fact that user preferences towards items are changing. These preferences vary from service to service and differ even in different periods of one service. This significantly limits the performance of these approaches in accurately learning user preferences and providing high-performance recommendations. Since changing user preferences and sparse scenarios are ubiquitous in real-world services, the RSs that can handle the changing preferences and work well in sparse scenarios are practically required to provide high-performance recommendations.

1.1 Research Challenges and Motivation

Considering the practical demand and the lack of a RS that can handle the changing preferences and work well in sparse scenarios, our goal in this thesis is to develop such RSs. Recall that user preferences change in different periods of one service and vary from service to service, this thesis studies both types of changes in user preferences. We itemize scenarios that user preferences change below.

- Changes of user preferences in one service
  - The case where user preferences change slowly
    - General e-commerce services, e.g., Amazon.com and Rakuten Market.
    - Online movie watching services, e.g., Netflix and Amazon Prime Video.
  - The case where user preferences change very frequently
    - Flash sale e-commerce services, e.g., Daily sale in Amazon and the au PAY market. **(the scenario we studied in Chapter 2)**
    - Video streaming and online news, e.g., YouTube and Microsoft News.
- Changes of user preferences between two services
  - The case to leverage public data. **(the scenario we studied in Chapter 3)**

- The case to leverage private commercial data from another service. **(the scenario we studied in Chapter 4)**

For the first type of changes, we focus on challenging cases that user preferences change very frequently (every few weeks), e.g., the services of flash-sale e-commerce, video streaming, and online news. In such cases, previous RSs are inapplicable to predict users' current preferences with models learned in the past. The services of video streaming and online news are out of our sparse scenario scope because it is easy to collect large numbers of user interaction data, e.g., click records, in these services. Therefore, we study the scenario of flash sale e-commerce, which suffer from severe sparse interactions and practically requires a RS that can capture changing user preferences to boost recommendation performances.

For the scenarios that user preferences are different in two services, it is still a practical challenge to improve recommendations in the sparse local service by effectively leveraging the data from external services. we, hence, study RSs that can extract useful knowledge from external data to facilitate recommendations in the sparse target service. Since external data include public data and private commercial data, this thesis studies how to effectively leverage both types of external data.

After achieving our goal, our RSs are expected to provide recommendations that can better meet users' interests in the future by accurately predicting their future preferences in a sparse service. Besides, our RSs are promising to leverage knowledge from external services to improve the recommendations for the local sparse service, even when user preferences are different between external and local services. We next elaborate on the challenges in the three practical sparse scenarios we studied and introduce why these scenarios are still required to be studied.

1.1.1 Model Period-specific Preferences in Flash Sale E-commerce

In recent years, flash sales e-commerce, which sells items with high discounts within a limited time to attract users' buying interests, has achieved great success and has attracted more and more attention all over the world [11]. Given the great success achieved by recent flash sale e-commerce, it is reasonable to improve these services by developing RSs that are specific to flash sales. In such services, modeling users'

period-specific preferences is a practical challenge when developing RSs, and it impacts the performance of current RSs and their practical implementations. Available and discounted items vary significantly from period to period in flash sale e-commerce according to their current sale strategies, where a period is regarded as the time span of a flash sale event. Users of these services are easily attracted by high discounts and may buy items they would not usually buy, resulting in users' period-specific preferences. For example, when coupons or other price discounts are offered, users react positively by purchasing the discounted items because it is perceived to be a "good deal". However, previous RSs learn only a uniform user preference for each user by leveraging that user's interaction data from different past periods. Therefore, these works are impossible to model users' period-specific preferences due to ignoring the changes of user preferences and are hard to work well for future sale periods in which user preferences are different from the past. Besides, the frequent changes of interacted items provoke sparse interactions, which makes the challenge of modeling period-specific preferences more difficult in this scenario.

1.1.2 Improve Recommendations with Public Data

Limited by the data-hungry nature of deep learning, building high-performance deep learning-based RSs requires a large amount of training data, where the data here are usually the interaction records of user behaviors towards items, e.g., purchases and clicks [10, 11, 12, 13]. As a result, it is an essential and practical requirement for services that have sparse interaction data to improve the performance of their RSs with the help of rich public data. These public data, e.g., movie ratings in MovieLens25M[1] and product ratings in Amazon[2], are collected from real business services and are released by service providers to organize competitions and promote researches. Cross-domain recommender systems (CDRSs) are a promising approach that facilitates target (i.e., local) recommendations by transferring knowledge from an external source domain with relatively sufficient interactions to facilitate recommendations in the target domain with sparse interactions, where the source and the target domains are respectively an external

[1] grouplens.org/datasets/movielens/25m/

[2] jmcauley.ucsd.edu/data/amazon/

service that provides public data and the local service [9, 22].

However, effectively leveraging public data to improve target recommendations is still a challenge. This is because the existing efforts of CDRSs suffer from a negative transfer issue in the scenario of leveraging public data. This issue is caused by misguiding the source interactions that represent users' unique interests within the source domain, i.e., domain-specific preferences, which lead to a limited improvement or even a negative impact on recommendations [23, 24]. Existing CDRSs handle the impact of such source interactions by merging individual interactions between domains and operate directly on these interactions to reduce their impact [16, 17, 25, 26, 27]. For example, systems in [28, 29] assign a minimal weight to these interactions. Merging individual interactions requires identifying the users who have interaction data in both domains (i.e., domain-shared users), where this identification is impossible when one of the domains is public. Therefore, handling the negative transfer issue is still a challenge for effectively leveraging public data.

1.1.3 Improve Recommendations with External Commercial Data

Exploiting only public data is often insufficient, as such data tends to be incomplete, outdated, and lacking in quality and accuracy due to limited support. Companies and services that require more data to improve their RSs, therefore, have the practical demand to utilize commercial data from other companies (other domains) in the way of commercial corporations. Furthermore, companies with rich data also request such corporations to sell their data to boost profits.

Different from the scenario in Subsection 1.1.2, leveraging commercial data prohibits the exchange of user interaction data due to privacy concerns [30]. This makes the negative transfer issue harder to be addressed. For example, in the scenario of leveraging public data, interactions in different domains can be merged at a cluster level, including the cluster of users and the one of items from these domains. The work in [31] proposed a method to transfer such cluster-level interactions. However, this approach is prohibitive by privacy regulations (e.g., the General Data Protection Regulation in the European Union) in this scenario, as merging interactions, even at a cluster level, discloses personal preferences, behaviors, or patterns of individuals without their consent

to other companies [32, 33]. This forces the requirement of another type of entities, such as semantic similarities between item clusters in [34], to transfer knowledge and handle the impact of source interactions implicitly. However, distinguishing the rules governing how operations on entities impact source interactions is challenging, further complicating the issue of negative transfer.

1.2 Research Contents

We overcome each of these challenges and propose new RSs. Our systems, which utilize various transfer learning-based technologies, solve the research issues stated in Section 1.1. The outlines of the proposed RSs are as follows.

- Hierarchical meta-learning for flash sale recommendations.

  In Chapter 2, we address the challenge of modeling users' period-specific preferences to improve the recommendations in flash sale e-commerce. Meta-learning methods, e.g., Model-agnostic meta-leaning [35], learn cross-task knowledge from past tasks to guide the model fast adapt to similar unseen tasks with only a few updates. Inspired by this concept, we propose a novel hierarchical meta-training algorithm that guides the learning of our recommendation model via user- and period-specific gradients to achieve our goal. With the guidance of such gradients, the model can learn prior knowledge shared in different periods and among users, which significantly simplifies its learning process when adapting to different periods and users, including unseen ones. By doing so, the learned model can capture users' period-specific preferences with only a few interactions and several updating steps.

- Debiasing graph transfer for non-overlap cross-domain recommendations.

  In Chapter 3, we address the challenge of improving recommendations with public data. To overcome this challenge, we propose a novel CDRS that can handle the negative transfer issue and requires no domain-shared users. Our CDRS removes the requirement of domain-shared users by merging cluster-level interactions between domains. Besides, it handles the negative transfer issue at the

level of the cluster by adaptively weighting the source clusters to avoid users' source-specific preferences. After mitigating the influence of these clusters, an enhancement in recommendation performance is anticipated.

- Semantic relation transfer for privacy preserved cross-domain recommendations.

  In Chapter 4, we overcome the challenge of improving recommendations with external commercial data. Employing such data requires that RSs do not merge interactions between domains to preserve user privacy. We, therefore, propose a new CDRS that meets the above requirement and can handle the more challenging negative transfer issue. Our CDRS removes the mergence of interactions by bridging domains and transferring knowledge through a similarity-based cross-domain graph. To handle the negative transfer issue in this scenario, our CDRS transfer knowledge focuses on the items that have similar textual features, e.g., similar titles or similar descriptions. In this way, the impact of source items that have irrelevant textual features and tend to present users' source-specific preferences can be alleviated.

1.3 Organization

This thesis consists of five chapters. The remainder of the thesis is organized as follows.

In Chapter 2, we study the challenge of improving recommendation performance in flash sale recommendations. In Section 2.1, we introduce users' period-specific preferences in flash sale e-commerce and also the issue caused by these preferences. Section 2.2 reviews prior works related to the topic of this chapter. Section 2.3 introduces some preliminaries of this chapter. Our proposed framework is presented in Section 2.4. We first give an overview of our framework. Then, we elaborate on our model that predicts user preferences and our novel hierarchical meta-training algorithm which guides the training of our model with the user- and period-specific gradients to capture users' period-specific preferences. The experiment setup and the results, including the comparison with state-of-the-art methods and the ablation study, are described in Section 2.5. After that, Section 2.6 discusses the interests and limitations of our proposal in this chapter. Finally, a conclusion of this chapter is given in Section 2.7. The study in this

chapter is based on our works published in [11], [36], and [37].

In Chapter 3, we introduce and discuss the problem of improving recommendation performances by leveraging public data under the scenario of sparse interaction data. In Section 3.1, we describe the existing problem, i.e., requiring domain-shared users or suffering the negative transfer issue, held by prior works when leveraging public data. Next, we review some related works on this topic in Section 3.2. In Section 3.3, we give some preliminaries of this chapter. Our proposed framework in this chapter is presented in Section 3.4. First, we exhibit an overview of our framework. Then, we elaborate on the details of each component in our framework, including the semantic domain fusing, the debiasing graph convolutional predictor, and the restrictions for debiasing learning. In particular, the semantic domain fusing module removes the requirement of domain-shared users by transferring cluster-level interaction information between domains, and the debiasing graph convolutional predictor weights the interactions from the source domain that successfully alleviates the negative transfer issue. We compare our proposed method with existing methods and conduct an ablation study to evaluate the effectiveness of each component in our proposal in Section 3.5. Finally, this chapter is summarized in Section 3.6. The study in this chapter is based on our work published in [31].

In Chapter 4, we focus on improving recommendation performances by leveraging external commercial data for the cases of sparse interaction data. We describe the issue of prior works in leveraging commercial data from another company in Section 4.1. In Section 4.2, we review some related works on this topic. The problem studied in this chapter is formulated in Section 4.3. Section 4.4 presents our proposed framework. We first give an overview of our proposed framework. We also elaborate on the details of our methods that effectively address the issue of leveraging external commercial data. We conduct experiments to examine the performance of our proposed method, and the setting and results of experiments are described in Section 4.5. Finally, this chapter is concluded in Section 4.6. The study in this chapter is based on our work published in [34].

Finally, in Section 5, we summarize this thesis and discuss future work.

Chapter 2

Hierarchical Meta-learning for Flash Sale Recommendations

2.1 Introduction

As we described in Section 1.1, the great success and the enormous commercial benefits flourish the development of flash sale e-commerce. Besides holding flash sale events and providing flash sale services, some companies also started to develop platforms that focus on only flash sale services, such as Vip.com[1] (the biggest flash sale e-commerce platform in China) and Taoqianggou[2] (the flash sale e-commerce platform of Alibaba). RSs, which are essential components of modern e-commerce websites, have empirically shown their superiority in improving user experiences and online business revenue [36, 38, 39, 40, 41, 42]. Therefore, it is natural to develop RSs for flash sale e-commerce to improve their services.

In flash sale e-commerce services, users are easily attracted by periodically changed discounted items and show period-specific preferences, which can be observed in their purchases in different sale periods. Naturally, it is important for RSs in flash sale e-commerce to improve their recommendations by modeling period-specific preferences toward discounted items.

[1] https://www.vip.com/

[2] http://qiang.taobao.com/

2.1.1 Challenge

Most existing RS studies lose their superiority in flash sale e-commerce because they learn user preferences from all users' past interactions and ignore the differences in preferences between periods. Here, interactions mean user behaviors in the services such as purchases, carts, and reviews. Note that session-based recommendations [43, 44, 45, 46] are not suitable for flash sale recommendations because the temporal information in these systems is different from users' period-specific preferences defined in this chapter. The temporal information indicates the relation hidden in the order of interactions, and the relation is determined by the inherent feature of items. In flash sales, users' period-specific preferences are influenced by external factors, e.g., high discounts. Besides, Graph neural network-based RSs, such as GraphSAGE [47] and AMAN [48], learn embeddings for users and items by capturing structural knowledge from a graph composed of user-item interactions. However, the embeddings learned by these RSs are static and cannot handle period-specific preferences in flash sale recommendations.

Notice that the frequent changes of items, including available ones and interested ones, in flash sale scenarios provoke the cold-start problem. This problem means that the recommendation performance degrades because of limited interaction data [12, 13]. To alleviate this problem, many efforts [49, 50, 51] developed cross-domain recommender systems (CDRSs) that leverage some data from external domains as prior knowledge to support the learning of the target recommendation model. For example, in [49], the domain-invariant contextual features from external domains are leveraged as a regularization to alleviate overfitting and to improve the user and item representations learned in the target domain. Most CDRSs require domain-shared users or domain-invariant contextual features to bridge the domains and transfer knowledge from external domains to the target domain. Unfortunately, external domains that contain domain-shared users or domain-invariant features with flash sale e-commerce domains do not always exist. CDRSs that require no domain-shared users and features usually transfer the latent matrix learned via matrix factorization algorithms [52, 53]. However, they cannot provide recommendations for users and items that appear for the first time after the recommendation model has been trained and hence are not applicable for flash sale e-commerce recommendations.

In recent years, some works [54, 55] introduced optimization-based meta-learning [35] into RS due to its superiority in mitigating the cold-start problem and improving the model's generalizability. Meta-learning learns cross-task knowledge from past tasks to facilitate the model to fast adapt to similar unseen tasks with only a few updates [56]. Meta-learning-based RSs usually learn a user-shared model and fast generate user-specific models for each user [54, 55]. These user-specific models have empirically demonstrated their effectiveness in alleviating the recommendation accuracy decrease caused by individual differences. However, previous meta-learning-based RSs consider static user preferences and thus cannot handle users' period-specific preferences well. We below summarize the challenges we overcome in this chapter to enhance recommendation accuracy in flash sale scenarios.

- The first challenge is addressing the cold-start problem caused by the periodically changed items, including the provided and interacted ones, in flash sale e-commerce.
- The second challenge is modeling users' period-specific preferences accurately.

2.1.2 Contribution

We propose a novel hierarchical meta-learning-based RS named HML4Rec to tackle the cold-start problem and simultaneously handle users' period-specific preferences in flash sale e-commerce recommendations. Motivated by the effectiveness of existing meta-learning-based RSs in alleviating the cold-start problem and modeling the individual bias in user preferences, we employ the same local update in [54] as a user local update. However, different from existing meta-learning-based RSs that further customize user-specific models, we consider hierarchically combining the user local update with a new period local update, which enables HML4Rec to learn user- and period-shared knowledge with the guidance of user- and period-specific gradients generated by two types of local updates. By doing so, HML4Rec can quickly track users' period-specific preferences and alleviate the cold-start problem with the learned knowledge.

Specifically, HML4Rec learns a recommendation model that contains prior knowledge and is shared by users and periods, enabling the trained model to quickly adapt

to new users and new periods with only several steps of updates and a few interactions. To learn a model that contains prior knowledge, we propose a novel hierarchical meta-training algorithm that hierarchically combines the user local update and the period local update. In this algorithm, the period local update generates period-specific models for different periods to handle users' period-specific preferences. The user local update further generates user- and period-specific models for different users to prevent the recommendation accuracy decreases caused by individual differences. The recommendation model shared by users and periods is optimized by minimizing the losses calculated on different users and periods, where losses are measured by the user- and period-specific models. As a result, the learned recommendation model contains user- and period-shared knowledge and can quickly adapt to new users and periods. Our contributions in this chapter are summarized below.

- We propose a novel hierarchical meta-learning RS for flash sale recommendations. Our system can handle users' period-specific preferences in flash sales and mitigate the cold-tart problem. To the best of our knowledge, this is the first work that focuses on improving recommendation accuracy for flash sale e-commerce recommendations.
- We develop a novel hierarchical meta-training algorithm to learn the user- and period-shared knowledge and provide user- and period-specific recommendations. Moreover, we introduce some effective methods to improve this algorithm.
- We conduct extensive experiments on a private flash sale e-commerce dataset and a widely used benchmark dataset to evaluate the effectiveness of our framework in both flash sale and non-flash sale recommendations. The results show that HML4Rec outperforms the state-of-the-art methods in flash sale recommendations and non-flash sale cold-start recommendations.

The organization of this chapter is as follows. We introduce related works in Section 2.2. Next, we describe some preliminaries of this chapter in Section 2.3. Then, we present the detail of our proposed hierarchical meta-learning RS in Section 2.4. The experimental setup and results are presented in Section 2.5. After that, we discuss the interests and limitations of our proposal in Section 2.6. Finally, a conclusion of this chapter is given in Section 2.7.

2.2 Related Work

2.2.1 Cold-start Problem in RS

Deep learning-based RSs [57, 58, 59, 60] have empirically shown their superiority in improving recommendation accuracy. However, it is difficult for the above deep learning-based RSs to make decent recommendations for new users and new items that appear after the prediction model is trained (named cold users and cold items) due to their limited historical data and the bias in user preferences caused by personal differences. Such a cold-start problem generally exists in RSs. To alleviate the problem of lacking historical data, side information, e.g., user profiles, item content features, and auxiliary information from other domains, is utilized in RSs to describe cold users and cold items. With this method, side information can represent cold users and items by the representations learned from warm users and warm items, where warm users and warm items have relatively sufficient interactions. For example, Bansal et al. [61] applied word-level embedding to represent new items with their semantic features that are extracted from their descriptions. Li et al. [62] introduced the Behavior-Intensive Neural Network, which also utilizes textual contents of items to learn the semantic latent item vectors and represents new users by aggregating their interacted items. Besides the above textual feature-based RSs, Cheng et al. [39] exploited sparse features and proposed Wide&Deep to learn the linear and deep relations from these features jointly. To better learn the relations between feature embeddings, Cheng et al. [63] developed the Adaptive Factorization Network (AFN). AFN proposed a logarithmic transformation network to learn arbitrary-order cross-features from data effectively. However, the above content-based RSs always provide the same recommendations for users with the same side information, even when the users have different historical interaction data [54]. As a result, these RSs ignore the individual differences in users' interactions and cannot provide personal recommendations accurately.

Meanwhile, CDRSs leverage the data from external domains to supply the insufficient interactions in the target domain. CDRSs have been viewed as a promising solution to mitigate the problem of lacking interactions. Some efforts [28, 29, 49, 50, 51] have been proposed following this line. For example, in [49, 51], they employed domain adversarial learning to implicitly transfer the interaction patterns from support domains

to the target domain. Efforts, such as [28, 29], explicitly transferred the embedding of users and items as prior knowledge when the common users or items exist across domains. As aforementioned in Section 2.1, user-shared or contextual feature-shared external domains do not exist for our flash sale e-commerce domain. Therefore, these existing CDRSs are not applicable to our flash sale e-commerce recommendations.

2.2.2 Meta-Learning in RS

Meta-learning has empirically demonstrated its effectiveness in improving the generalization of the model. Representatively, Optimization-based meta-learning [35] learns cross-task knowledge, i.e., a global shared initialization of model parameters, and leverages the learned knowledge to initialize a model that can fast adapt to a new task with a few task-wise data. In RSs, optimization-based meta-learning usually generates customized models for every user to handle the bias in user preferences caused by individual differences and further improve personalized recommendations [54, 55, 64]. For example, Lee et al. [54] proposed a meta-learned User Preference Estimator (MeLU) upon optimization-based meta-learning technology. First, MeLU generates a global user preference estimator model that is shared among users. The global model is then updated with users' historical interactions to generate user-specific models for every user. Dong et al. [55] proposed Memory-Augmented Meta-Optimization (MAMO) to enhance the customization of their RS with the guidance of two memory networks. Specifically, compared with MeLU, MAMO further customizes the global initialization for each user with the guidance of a memory network before generating user-specific models.

In addition, optimized-based meta-learning is also introduced into RSs to solve other problems. For example, in [65], the author considered the bias in different items and developed a session-based meta-learning framework to facilitate the recommendation tasks for new items via a task embedding learned from the past recommendation tasks on warm items. Sequential Scenario-Specific Meta Learner (S^2Meta) [66] is developed to adapt to the bias in different scenarios, where scenarios are denoted as the different tags of movies. Song et al. [67] proposed a novel cluster-based meta-learning model (CBML) for session-based RSs, and CBML can better transfer shared knowledge

across similar sessions and preserve the characteristics of each session itself. Lu et al. [68] constructed two GNN-based aggregators and a temporal meta-learning method to extract users' global long-term and internal short-term preferences from two graphs.

Metric-based meta-learning, which learns task representations to facilitate model generalization, was also introduced into RSs. For example, Contextual Modulation Meta Learning [69] aggregates context embeddings from a user's past interactions to specify the learned user-item features effectively. In [65], the author considered the bias in different items and developed a session-based meta-learning framework to facilitate the recommendation tasks for new items via a task embedding learned from the past recommendation tasks on warm items. Meta Warm Up Framework [70] focused on the warm-up of the embedding of cold items with a warm-up model learned by the relationship between the cold and warm embedding of items. Meta-learning-based approaches have been proven to be promising in mitigating the cold-start problem. However, most of them assume that user preferences are static and lack modules to handle users' period-specific preferences in flash sale scenarios.

2.2.3 Session-Based Recommendations

To provide better recommendations, temporal features hidden in the order of user interactions are considered in session-based recommendations. In such recommendations, historical interactions are organized as sessions in an individual manner. Then, Recurrent Neural Network and its variants, i.e., Gated Recurrent Unit (GRU) [71] and Long Short-Term Memory [72], are usually applied to capture sequential information from sessions. For example, in [43], the author developed a session-based GRU model with ranking loss to capture the sequential information in users' historical interactions. Moreover, General Knowledge Enhanced Framework [73] incorporated sequential information with graph knowledge and empirically improved the performance and explainability of recommendations. In addition, some recent efforts [44, 74, 75] introduced attention mechanisms into session-based recommendations to weigh the impact of different items in a session. For example, Sun et al. [44] introduced bidirectional encoder representations from Transformer into session-based recommendations to capture the bi-direction sequential information. Zheng et al. [74] proposed an attentive

meta-graph embedding approach to learn graph information from the heterogeneous information network.

Graph neural network-based RSs [47, 48, 76, 77] also attracted much attention due to their effectiveness in capturing the structural information hidden in the user-item interaction graph. For example, As an instance, Adaptive Multi-Attention Network [48] constructed a graph attention network and a co-attention network to learn users' multi-behaviors jointly. Other deep learning approaches, such as convolutional networks and memory networks, are also studied in session-based recommendations. Tang et al. [45] proposed a convolutional sequence model that learns sequential information by using both horizontal and vertical convolutional filters. In [78] and [46], the memory network is also taken into consideration to improve session-based recommendations.

In general, session-based RSs represent users with the sessions of their interacted items. However, in flash sale scenarios, available items frequently change in different flash sale periods. When items are significantly changed, it is hard for session-based RSs to infer user preferences with the knowledge learned from previous items. Besides, session-based RSs learn unique embeddings for items in different periods and thus ignore users' period-specific preferences raised by the sales strategies. Therefore, session-based RSs are not suitable for flash-sale e-commerce recommendations.

2.3 Preliminaries

2.3.1 Problem Definition

In this chapter, a recommender task is denoted as top-K recommendations for a user in some period. We consider a flash sale e-commerce domain that contains U, V, R, and P, where U (V) denotes the user (item) set and R is the interaction set between them. P is the flash sale period set, where we collect a period $p \in P$ with the method described in Subsection 2.5.1. For each user $u \in U$ in each period $p \in P$, we predict a preference score $\hat{y}^p_{u,v}$ for each item $v \in \bar{V}^p_u$, where $\bar{V}^p_u$ denotes the item set that have not interacted with u in p. The preference score $\hat{y}^p_{u,v}$ is calculated by a user- and period-specific model $f_{\theta^{p,u}}$, given the user-item pair (u, v). $\hat{y}^p_{u,v}$ is formulated as:

$$\hat{y}^p_{u,v} = f_{\theta^{p,u}}(u, v). \tag{2.1}$$

Then, items in $\bar{V}_u^p$ are ranked based on their preference scores, and the top-K items in the ranking list are recommended to u. It is worth mentioning that we measure the preference scores with different models for different users in different periods to provide *user- and period-specific* recommendations.

2.3.2 Optimization-based Meta-learning

The optimization-based meta-learning algorithm [35] considers some learning tasks and a meta-learner f_θ, where θ is the parameter of f_θ. f_θ can quickly adapt to a new task with the guidance of the prior knowledge that is learned from previous tasks. In previous meta-learning-based RSs [54, 55], a learning task T^u is defined as making recommendations for a user u. First, for each learning task T^u, the observed interaction set $R^u = \{(u, v)|v \in V_u\}$ of user u is divided into a support set R_s^u and a query set R_q^u, where V_u contains items that are interacted with u. The model f_θ is locally updated to generate the user-specific model f_{θ^u} and adapt to T^u by the gradient descent:

$$\theta^u \leftarrow \theta - \alpha \nabla \mathcal{L}_u \left(f_\theta \right), \tag{2.2}$$

where $\mathcal{L}_u \left(f_\theta \right)$ is the task-related loss measured on R_s^u and α is a universal learning rate. $\mathcal{L}_u \left(f_\theta \right)$ is calculated by

$$\mathcal{L}_u \left(f_\theta \right) = \sum_{(u,v) \in R_s^u} \ell \left(f_\theta \left(u, v \right), y_{u,v} \right), \tag{2.3}$$

where ℓ is the loss between $f_\theta \left(u, v \right)$ and $y_{u,v}$, and $y_{u,v}$ is the ground truth of interaction between u and v. Then, θ is globally updated based on the meta-loss $\mathcal{L}' \left(f_{\theta^u} \right)$ for the task-wise fine-tuned parameter θ^u over task-provided R_q^u, where $\mathcal{L}' \left(f_{\theta^u} \right)$ is formulated by

$$\mathcal{L}' \left(f_{\theta^u} \right) = \sum_{(u,v) \in R_q^u} \ell \left(f_{\theta^u} \left(u, v \right), y_{u,v} \right), \tag{2.4}$$

and θ is then updated by gradient descent $\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}'$, where β is a learning rate. After training, when serving recommendations for a new user u', f_θ can fast customize the user-specific model $f_{\theta^{u'}}$ with a few update steps.

However, the learned user-specific model f_{θ^u} predicts the same user preference score given the same user-item pair in different periods. As a result, the user-specific model

cannot handle users' period-specific preferences in flash sale e-commerce recommendations. Therefore, RSs that can learn users' period-specific preferences and provide period-specific recommendations are required. Inspired by previous meta-learning-based RSs, we define a learning task T^p as making recommendations for a flash sale period p and locally update the model f_θ with the historical interaction data in p. By doing so, f_θ learns users' period-specific preferences and fast customizes the period-specific model f_{θ^p} to provide period-specific recommendations.

2.4 Proposed Framework

The details of our proposed framework, HML4Rec, are given in this section. First, we present an overview of our framework. We then elaborate on the details of the recommendation model. After that, we describe our new hierarchical meta-training algorithm, which optimizes our model and adapts the model to different users and flash sale periods. We next present the methods for improving the optimizing process of our model. Last, we introduce the top-K recommendations.

2.4.1 Overview of Framework

Motivated by the observations that existing RSs cannot handle users' period-specific preferences or suffer from the cold-start problem in flash sale recommendations, we propose a novel HML4Rec, which is depicted in Figure 2.1 and does not have these drawbacks. To handle users' period-specific preferences, HML4Rec generates user- and period-specific models for every user and period, where such a model learns preferences for a user in a specific period. Then, users' preferences in different periods can be captured by the corresponding models. Besides, we address the cold-start problem by learning a global recommendation model that contains prior knowledge shared among users and periods. Under the guidance of this knowledge, the global recommendation model can initialize the above user- and period-specific models to fast adapt to recommendations for cold-start users and future periods. This approach significantly simplifies the learning process of user- and period-specific models. By doing so, these models are expected to be learned well with only sparse interaction data and few steps

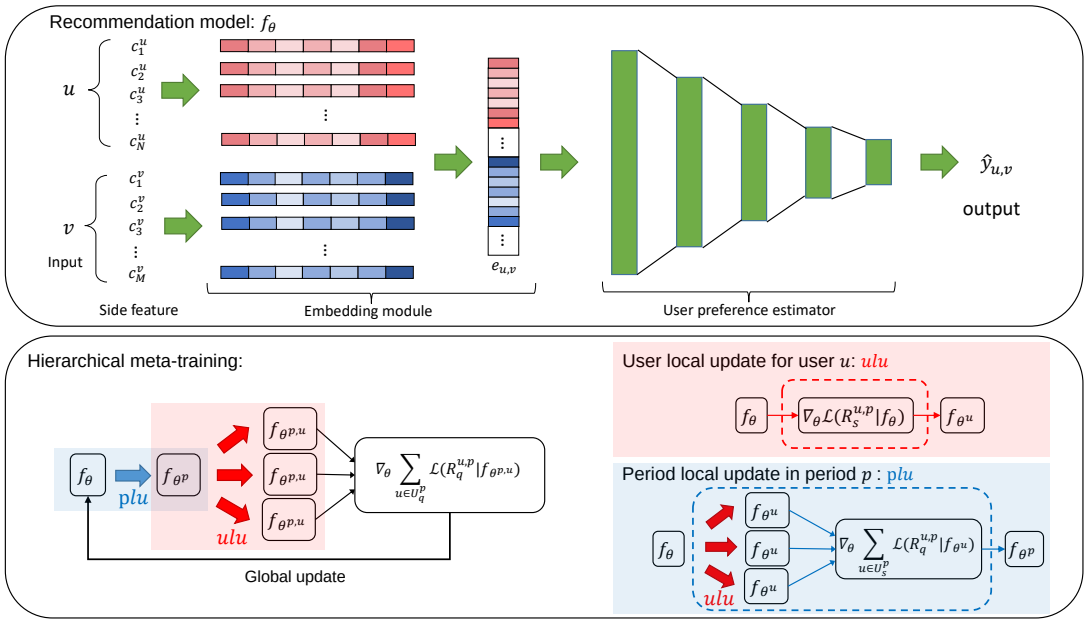


Figure 2.1: Overview of HML4Rec. The recommendation model infers user preferences and calculates a preference score by inputting the side information of a user-item pair. The hierarchical meta-training consists of the user local update and the period local update and is applied to learn the parameters in the recommendation model. The user local update adapts the recommendation model to a user by locally updating the model with this user's historical interactions. The period local update adapts the recommendation model to a flash sale period and contains several user local update operations.

of updates. Table 2.1 summarizes important notations used in this chapter.

2.4.2 Recommendation Model

The recommendation model predicts a preference score $\hat{y}_{u,v}$, given a user-item pair (u, v). That is, we aim at having $f_\theta(u, v) = \hat{y}_{u,v}$, where θ is the learnable parameters of the model. The recommendation model consists of a user & item embedding module and a user preference estimator. It takes one-hot vectors of user profiles (e.g., age and occupation) and item contents (e.g., genre and price) as input. First, the embedding module embeds the input to generate latent representations for each user-item pair.

Table 2.1: Summary of notations

Notation	Description
u	a user
v	an item
p	a flash sale period
I	number of the user preference estimator layer
$\mathbf{c}_n^u, \mathbf{c}_m^v$	categorical feature of u and that of v
$\mathbf{e}_{u,v}$	latent embedding of the user-item pair (u, v)
$\bar{V}_u^p$	item set that have not interacted with u in p
$R_s^{p,u}$, $R_q^{p,u}$	support set and query set of u's interactions in p
U_s^p, U_q^p	support user set and query user set in p
f_θ	global shared recommendation model
f_{θ^u}, f_{θ^p}	user-specific model and period-specific model
$f_{\theta^{u,p}}$	user- and period-specific model
$\mathbf{T}_i$	transformation matrix for layer i
H_p	entropy of interacted items in p.
β_p	entropy customized period local update learning rate in p

Then, from the representations, user preferences are inferred by utilizing a multi-layered fully connected neural network, namely user preference estimator, which is widely used to learn complex non-linear user preferences in previous studies [54, 55, 57]. Last, the model outputs a score to describe the probability that a user interacts with an item.

User & Item Embedding Module. In this part, the embedding process, which has empirically shown its advantages in improving recommendation performances in [79, 80], automatically extracts latent features from the contents of users and items. We use this embedding process for the following observations. First, the auxiliary side features, including user profiles and item contents, provide general information that can be used to represent users and items. In the cold-start scenario, this embedded process can provide representations for new users and items that appear after model training has finished. Second, traditional one-hot embedding represents users and items with their unique ID, making it impossible to describe new users and new items. Our user & item

embedding module learns the embedding vector for each categorical feature in user and item side information. Suppose that the number of user (item) categorical features is N (M). The embedding of the user-item pair (u, v) is denoted as $\mathbf{e}_{u,v}$, where $\mathbf{e}_{u,v}$ is an aggregation of all feature embedding vectors and can be written as:

$$\mathbf{e}_{u,v} = AGG\left(\mathbf{E}_1^u \mathbf{c}_1^u; \cdots; \mathbf{E}_N^u \mathbf{c}_N^u; \mathbf{E}_1^v \mathbf{c}_1^v; \cdots; \mathbf{E}_M^v \mathbf{c}_M^v\right), \tag{2.5}$$

where $\mathbf{c}_n^u \in \mathbb{R}^{d_n}$ and $\mathbf{c}_m^v \in \mathbb{R}^{d_m}$ are the one-hot vectors for categorical features $n \in \{1, \cdots, N\}$ of user u and for those $m \in \{1, \cdots, M\}$ of item v, respectively. $\mathbf{E}_{u,n} \in \mathbb{R}^{d_e \times d_n}$ is an adaptive embedding matrix for the corresponding categorical feature n, while $\mathbf{E}_{v,m} \in \mathbb{R}^{d_e \times d_m}$ is an adaptive embedding matrix for the corresponding categorical feature m. d_e, d_n, and d_m are the embedding dimension, the number of categories for feature n, and the number of categories for feature m, respectively. All the embedding matrices $\{\mathbf{E}_1^u \cdots \mathbf{E}_N^u, \mathbf{E}_1^v \cdots \mathbf{E}_M^v\}$ are learnable parameters in this embedding module. If numerical features exist, the normalized values are connected directly with $\mathbf{e}_{v,u}$. As for the aggregation function $AGG(\cdot)$, we adopt the simple concatenation method to keep the model backbone consistent with previous efforts [54, 55, 57] for a fair comparison. In addition, as our aggregation function, we employ the same logarithmic transformation network as AFN, for fair comparison with AFN. The HML4Rec with the logarithmic transformation network is denoted as HML4Rec-AFN.

User Preference Estimator. The user preference estimator is a fully connected neural network with I dense layers. This estimator learns the complex and non-linear user preferences. These layers are defined as:

$$\begin{aligned} \mathbf{x}_0 &= \mathbf{e}_{u,v}, \\ \mathbf{x}_1 &= \sigma\left(\mathbf{W}_1^\top \mathbf{x}_0 + \mathbf{b}_1\right), \\ &\vdots \\ \mathbf{x}_I &= \sigma\left(\mathbf{W}_I^\top \mathbf{x}_{I-1} + \mathbf{b}_I\right), \end{aligned} \tag{2.6}$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ ($i \in [1, I]$) are respectively the weight matrix and bias vector for the i-th preference estimating layer. σ is the activation function of these layers. In this chapter, σ is the rectified linear unit (ReLU) [81]. The learnable parameters of the user preference estimator are all the weight matrix and bias vector $\{\mathbf{W}_i, \mathbf{b}_i | i \in [1, I]\}$.

Output. The output layer outputs a score to describe the probability that a user u will interact with an item v. This layer is formulated as

$$\hat{y}_{u,v} = \sigma\left(\mathbf{W}_o^\top \mathbf{x}_V + b_o\right), \tag{2.7}$$

where $\mathbf{W}_o$ and b_o are respectively the weight matrix and the bias value for the output layer. For the activation function σ, we use the sigmoid function to match the binary implicit feedback. The learnable parameters of the output layer are $\mathbf{W}_o$ and b_o.

2.4.3 Hierarchical Meta-training Algorithm

In this section, we introduce a novel hierarchical meta-training algorithm to train the recommendation model in Subsection 2.4.2. Traditional neural networks initialize their parameters with the values randomly sampled from a Gaussian distribution. Then, the randomly initialized parameters usually converge to an optimum with sufficient training data. However, traditional training methods often lead to overfitting in the cold-start scenario due to the limited training data. Furthermore, in flash sale e-commerce scenarios, users' item-consumption histories in different periods can mislead the model due to frequent changes in users' interaction patterns. Inspired by the concept of optimization-based meta-learning [35], our hierarchical meta-training conducts a global update and a local update to handle the above problems. More precisely, the global update learns cross-task knowledge to generate a global initialization for recommendation model parameters. Then, in each task, the global initialization can fast adapt to an optimum, even with limited task-wise training data. The local update fine-tunes the parameter from the global initialization for each recommendation task. Our recommendation task has a two-level hierarchy: task (period-level recommendations) and sub-task (user-level recommendations). In light of this, we propose a hierarchical local update, including the user local update and the period local update, to adapt to each task and sub-task.

User Local Update. Inspired by [54], we apply the user local update to handle the bias in user preferences caused by individual differences. For each user in a period (i.e., sub-task), given the user's item-consumption history $R^{p,u}$, we randomly split $R^{p,u}$ into a support set $R_s^{p,u}$ and a query set $R_q^{p,u}$. The optimization goal in this phase is to learn a user-specific model with the local parameters θ^u. θ^u is customized from the

global shared initialization θ by minimizing the prediction loss $L(R_s^{p,u}|f_\theta)$ on $R_s^{p,u}$. We perform a pair-wise hinge loss following some previous works [82, 83]. It can soften the implicit hard labels to train our model. The loss function is formulated as

$$\mathcal{L}\left(R_s^{p,u} \mid f_\theta\right) = \sum_{\substack{(u,v)\in R_s^{p,u},\\ v'\in \bar{V}_u^p}} \left[z - f_\theta(u,v) + f_\theta(u,v')\right]_+ \tag{2.8}$$

where $[a]_+ = \max(a, 0)$ is the standard hinge loss, $z > 0$ is the safety margin size and v' is a negative item that is randomly sampled from the item set $\bar{V}_u^p$. $\bar{V}_u^p$ contains items that are not interacted by user u in period p. Thus, the user-specific parameters θ^u are updated by

$$\theta^u \leftarrow \theta - \alpha \cdot \nabla_\theta \mathcal{L}\left(R_s^{p,u}|f_\theta\right), \tag{2.9}$$

where α is the learning rate for the user local update. After updating θ, the prediction loss $L(R_q^{p,u}|f_{\theta^u})$ on $R_q^{p,u}$ is measured by the user-specific model f_{θ^u}.

Period Local Update. To capture users' period-specific preferences, we propose the period local update. For each flash sale period p (i.e., task), given the user set U^p in period p, we split U^p into a support set U_s^p and a query set U_q^p based on the time of the user's first interaction before and after the middle time of that period. The learning goal of this phase is to learn a period-specific model with the local parameters θ^p. θ^p is customized from the global shared initialization θ by minimizing all prediction losses measured on each support user's query interaction set by the user-specific models learned in Subsection 2.4.2. Precisely, for each user u in U_s^p, we conduct the user local update and calculate the prediction loss $L(R_q^{p,u}|f_{\theta^u})$ mentioned above. Then, we learn the local parameters θ^p by minimizing these prediction losses. The local parameters θ^P are updated by

$$\theta^p \leftarrow \theta - \beta \sum_{u\in U_s^p} \nabla_\theta \mathcal{L}\left(R_q^{p,u}|f_{\theta^u}\right), \tag{2.10}$$

where β is the learning rate for the period local update. Next, for each user u in U_q^p, we perform the user local update with the initialization of θ^p to learn a user- and period-specific model $f_{\theta^{p,u}}$, where $f_{\theta^{p,u}}$ captures this user's period-specific preferences. The above user local update is conducted on the support set $R_s^{p,u}$ of each user u in U_q^p. $f_{\theta^{p,u}}$ can capture u's period-specific preferences as it is locally updated by the period-specific

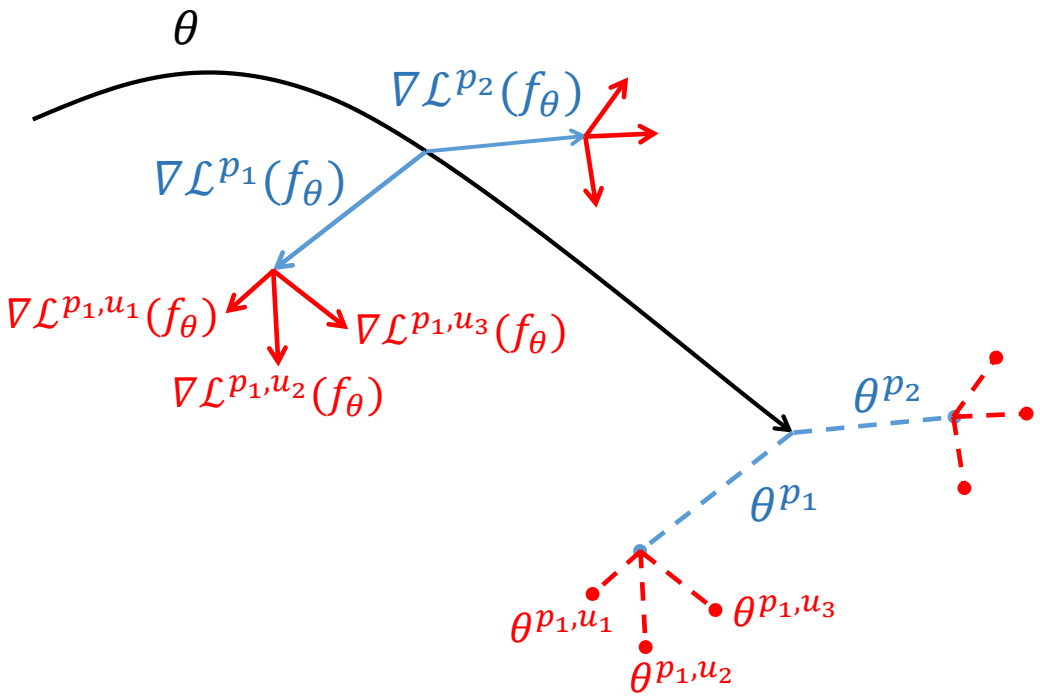


Figure 2.2: Optimization strategy of hierarchical local update. Red lines represent user local updates, and blue lines represent period local updates.

gradients. After learning user- and period-specific models, we measure the prediction loss $L(R_q^{p,u}|f_{\theta^{p,u}})$ on $R_q^{p,u}$ for each user u in U_q^p

Figure 2.2 illustrates the optimization strategy in the hierarchical local update. The period local update learns period-specific gradients (the solid blue lines) to generate period-specific parameters (the dashed blue lines) for each period (i.e., task). Then, the user local update learns user-specific gradients (the solid red lines) to generate user- and period-specific parameters (the dashed red lines) for each user (i.e., sub-task).

Global Update. In the global update phase, we learn the global initialization θ, which contains user- and period-shared knowledge and can fast adapt to new users and periods with only a few steps of updates. For each flash sale period p, we conduct the above two-level hierarchical local update to get the prediction loss $L(R_q^{p,u}|f_{\theta^{p,u}})$ on each user

u in query user set U_q^p. After that, the global initialization θ is trained by

$$\theta \leftarrow \theta - \gamma \sum_{p \in P} \sum_{u \in U_q^p} \nabla_\theta \mathcal{L} \left(R_q^{p,u} | f_{\theta^{p,u}} \right), \tag{2.11}$$

where γ is the learning rate for global updates and P is all the training periods.

We locally update θ via back-propagation in the hierarchical local update phase, and consequently the gradients $\nabla_\theta \mathcal{L} \left(R_q^{p,u} | f_{\theta^{p,u}} \right)$ involve high-order derivatives. The computation of high-order derivatives is expensive when the depth of the neural networks is deep. Therefore, we take a one-step gradient descent for the hierarchical local update to accelerate the training.

2.4.4 Optimization for Training

In order to mitigate the unstable gradient issue raised by the high-order derivatives, we introduce residual layers [84] and a meta-warped activation space method [85] into our model. Besides, we develop entropy-guided period local updates to customize the learning rate of the period local update to prevent recommendations from being concentrated on a few items.

Stabilize Gradient. The global update of the hierarchical meta-training algorithm involves high-order derivatives when calculating the gradients of θ. The gradients will be multiplied by the same set of parameters many times. After multiple back-propagation steps, the gradient will be small. In the worst case, this may ultimately stop the neural network from further training. Inspired by the remarkable effectiveness of the residual layers that alleviate the gradient vanishing problem in computer vision [84, 86], we incorporate a residual layer into our system to stabilize gradients. The residual layer skips the layers that are locally updated in the user local update phase to decrease the impact of high-order derivatives in gradient calculations. Additionally, for layers that should be locally updated, we warp each layer's activation space by inserting a meta-learned distance metric. The model with such warped activation spaces has been empirically demonstrated to be more sensitive to task identity in [85]. In this chapter, these warped activation spaces can guide the optimization of our hierarchical local updates to prevent the recommendation model from overfitting. For a clear explanation, supposing a recommendation model with an embedding layer and 4 dense layers, we conduct the

period local update for the first 2 dense layers and conduct the user local update for the last 2 dense layers. After incorporating the residual layer and the meta-learned distance metrics, the user preference estimator is reformulated as

$$\begin{aligned}
\mathbf{x}_0 &= \mathbf{e}_{u,v}, \\
\mathbf{x}_1 &= \sigma\left(\mathbf{T}_1\mathbf{W}_1^\top\mathbf{x}_0 + \mathbf{b}_1\right), \\
&\vdots \\
\mathbf{x}_4 &= \sigma\left(\mathbf{T}_4\mathbf{W}_4^\top\mathbf{x}_3 + \mathbf{b}_4\right) + \sigma\left(\mathbf{W}_r^\top\mathbf{x}_2\right),
\end{aligned} \tag{2.12}$$

where $\mathbf{W}_r$ is the weight matrix for the residual layer and $\mathbf{T}_i$ is the transformation matrix for layer i. $\mathbf{W}_r$ and $\mathbf{T}_i$ are updated only in the global update. Specifically, considering the limited user-wise historical interactions, we skip-connection $\mathbf{x}_2$ and $\mathbf{x}_4$ with the residual layer to simplify the learning in the user local update.

Entropy-guided Period Local Update. In the flash sale e-commerce scenario, the interactions can be concentrated on a few items or scattered across many items according to the sales strategy. The difficulties of learning different tasks are quite different. In the concentration case, most users' purchases move towards some items, and the concentration of purchases makes it easier to infer. Therefore, a small learning rate can prevent the recommendation model from overfitting. On the contrary, in the scattering case, a large learning rate is needed.

In light of this, we develop a novel entropy-guided period local update to customize the period learning rate β. Specifically, according to the definition of entropy, the entropy of interactions can describe the concentration of the interactions. For a period p, we denote the entropy of interactions in p by H_p. Considering that the period local update in p are conducted on the support interaction set $R_s^p = \{R_s^{p,u} | u \in U_s^p\}$, H_p is formulated by

$$H_p = -\sum_{v \in R_s^p} \mathrm{P}\left(v\right) \log_2 \mathrm{P}\left(v\right), \tag{2.13}$$

where $\mathrm{P}\left(v\right) = \frac{|R_s^{p,v}|}{|R_s^p|}$. $|R_s^{p,v}|$ is the number of interactions for item v. Then, the learning rate in period local updates β is customized by

$$\beta_p = H_p \beta. \tag{2.14}$$

Training Algorithm. After optimizing the hierarchical meta-training algorithm with the above methods, the advanced algorithm is applied to train our recommendation model and learn a global initialization for the model parameters. The details are summarized in Algorithm 1.

Algorithm 1: Hierarchical meta-training algorithm

Input: Period set P; User set $U = \{U^p | p \in P\}$; Hyper-parameters α, β, γ

Output: Global parameters θ

Randomly initialize the parameters θ
while *Not converge* **do**
  for *each period* $p \in P$ **do**
    Sample a batch of users $U_s^p \sim U^p$;
    for *each user* $u \in U_s^p$ **do**
      Evaluate $\nabla_\theta \mathcal{L}\left(R_s^{p,u} | f_\theta\right)$ by Eq.(2.8);
      User local update $\theta^u \leftarrow \theta - \alpha \cdot \nabla_\theta \mathcal{L}\left(R_s^{p,u} | f_\theta\right)$;
      Evaluate $\nabla_\theta \mathcal{L}\left(R_q^{p,u} | f_{\theta^u}\right)$ by Eq.(2.8);
    end
    Evaluate H_p by Eq. (2.13);
    Update $\beta_p \leftarrow H_p \beta$;
    Period local update $\theta^p \leftarrow \theta - \beta_p \sum_{u \in U_s^p} \nabla_\theta \mathcal{L}\left(R_q^{p,u} | f_{\theta^u}\right)$;
    Sample another batch of users $U_q^p \sim U_p$;
    for *each user* $u \in U_q^p$ **do**
      Evaluate $\nabla_\theta \mathcal{L}\left(R_s^{p,u} | f_{\theta^p}\right)$ by Eq. (2.8);
      User local update $\theta^{p,u} \leftarrow \theta_p - \alpha \cdot \nabla_\theta \mathcal{L}\left(R_s^{p,u} | f_{\theta^p}\right)$;
      Evaluate $\nabla_\theta \mathcal{L}\left(R_q^{p,u} | f_{\theta^{p,u}}\right)$ by Eq. (2.8);
    end
  end
  Global update θ by: $\theta \leftarrow \theta - \gamma \sum_{p \in P} \sum_{u \in U_q^p} \nabla_\theta \mathcal{L}\left(R_q^{p,u} | f_{\theta^{p,u}}\right)$;
end

2.4.5 Top-K Recommendations

Given a user u and a set of items $\bar{V}_u^p$ that have not interacted with u in period p, we first perform the hierarchical local update to generate the user-and period-specific model $f_{\theta^{p,u}}$. Then, $f_{\theta^{p,u}}$ measures the preference score $\hat{y}_{u,v}^p$ for each item in $\bar{V}_u^p$. Last, the top-K items with the largest $\hat{y}_{u,v}^p$ are recommended to u as her recommendation list in period p.

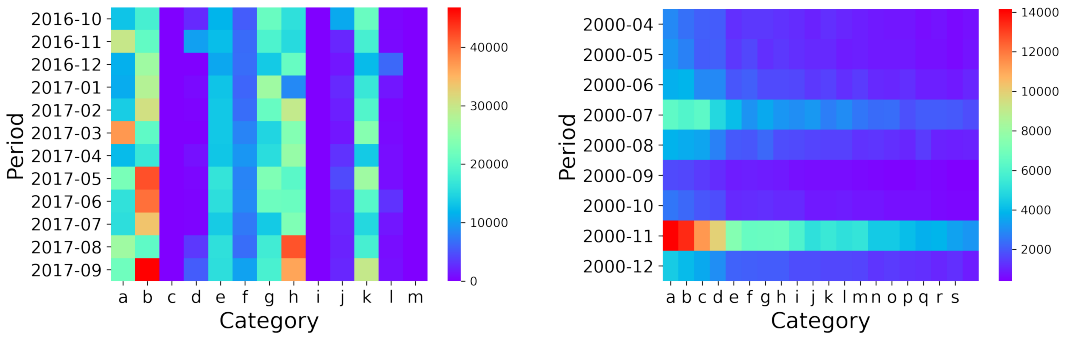
2.5 Experiments

In this section, we first introduce the datasets we used and then conduct experiments to discuss the effectiveness of our HML4Rec in both flash sale recommendations and non-flash sale recommendations.

2.5.1 Dataset

We used a private flash sale e-commerce dataset (E-commerce) and a widely used public dataset (MovieLens 1M) to evaluate the performance of our method under the flash sale scenario and non-flash sale scenario, respectively. E-commerce is a private commercial dataset from a real-world flash sale e-commerce platform providing both products (e.g., home electronics and make-up) and services (e.g., travel and restaurant reservations). It includes users' purchase records from *30/10/2016* to *31/7/2017* and some auxiliary features (4 user features and 6 item features). MovieLens 1M dataset is a well-known benchmark for movie recommendations. We applied the same MovieLens 1M dataset as [54]. It contains user ratings for movies, where we select the ratings from *1/4/2000* to *31/12/2000*. 4 user features and 4 item features are available in this dataset. Item features (i.e., movie content) were collected from IMDb[3]. To visualize interaction patterns, we counted monthly interactions of items at a category level. The result is shown in Figure 2.3, where each alphabet in the horizontal axis indicates a category (e.g., home electronics) and the vertical axis indicates time periods. Then, the following pre-processing was conducted for each dataset:

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



(a) Purchases on E-commerce

(b) Binary ratings on MovieLens

Figure 2.3: Category-level monthly interactions. A block represents the monthly interaction cases of a category.

E-commerce. This dataset includes some free items (e.g., coupons and gifts) and discounted items to attract users. Available items change significantly from period to period according to the provider's sale strategies, where the period is a flash sale period that lasts 28 days in this domain. Hence, users' purchase patterns are quite different in different periods, and the result in Figure 2.3 illustrates this phenomenon. From Figure 2.3 (a), we can see that users' purchase patterns change a lot before and after *5/2017*. In order to evaluate whether HML4Rec can adapt to the periods with different purchase patterns, we divided the purchase records in this manner: *30/10/2016* to *31/3/2017* as training set, *1/4/2017* to *30/4/2017* as validation set, and *1/5/2017* to *30/9/2017* as test set. We split the purchase records in the training set every 28 days and collected 5 flash sale periods for training. Similarly, we collected a validation period and 5 test periods from the validation and test sets, respectively. Then, for users in the first half of each period, we did a popular 6-core pre-processing stem [60, 87, 88] to select the users who have at least 6 purchase records as support users. Query users were selected similarly from the last half of the period. Then, we used the last 5 purchase records for each support and query user as the query records and the rest as the support records. After the 6-core stem, the dataset has 17,705 users, 45,057 items, and 569,394 interactions.

MovieLens 1M. We binarized the rating scores as implicit feedback by converting all observed rating scores as positive interactions and the remaining as negative interac-

tions. From Figure 2.3 (b), we observed that users' interaction patterns are nearly consistent among different months. This observation shows that MovieLens 1M is a non-flash sale scenario. We divided the binary ratings into three parts: *1/4/2000* to *31/8/2000* as training set, *1/9/2000* to *30/9/2000* as validation set, and *1/10/2000* to *31/12/2000* as test set. To run our HML4Rec, we supposed that a flash sale period lasts 28 days in this domain and collected 6 training periods, 1 validation period, and 3 test periods following the same manner applied in E-commerce. We also conducted the same 6-core processing stem to generate the support and query sets. Finally, the dataset has 6,014 users, 3,673 items, and 902,301 interactions.

2.5.2 Experiment Setting

Recommendation Scenarios. We considered the recommendations in two scenarios: i) recommendations for warm users as a warm scenario and ii) recommendations for cold users as a cold scenario. Specifically, we classified users into warm or cold users according to their first interaction time. Users who had interactions in the training set were warm users, and users who had no interactions in the training set were cold users.

Evaluation Criteria. In this chapter, we separately evaluated the recommendations in different periods, where we had 5 test periods in E-commerce dataset and 3 test periods in MovieLens dataset. In the test phase, for each test period, support users were sampled from the first half of the period. Recommendations were evaluated based on the query users sampled from the last half of the period. To evaluate the performance of recommendations, we followed the common strategy of top-K recommendation in [89, 90], which ranks the items that have not interacted with a user and recommends top-K items to her. For each query user, we randomly sampled 99 items that had no interaction with that user in this test period and ranked the target item among the 100 items based on the preference scores measured by the user- and period-specific models, where these models are generated by conducting the user local update and the period local update in Subsection 2.4.3. Then, we applied *Hit Ratio* (HR) and *Normalized Discounted Cumulative Gain* (NDCG) to evaluate the recommended top-k items.

- **Hit Ratio.** HR is a metric that is widely applied to measure recommendation performances [91, 92]. Given a user-item interaction in a test set, HR@K measures

Table 2.2: Some important characteristics of evaluated methods. Pref. indicates preferences.

Methods	DL-based	ML-based	Period-specific Pref.	Individual bias
Wide&Deep	✓			
NeuCF	✓			
AFN	✓			
MeLU	✓	✓		✓
MAMO	✓	✓		✓
s^2Meta-u	✓	✓		✓
s^2Meta-p	✓	✓		✓
CMML	✓	✓		✓
HML4Rec	✓	✓	✓	✓
HML4Rec-AFN	✓	✓	✓	✓

whether a test item is in the top-K recommendation list or not. If the target item appears in the top-K recommendation list, we obtain a *hit*. HR@K is calculated as follows:

$$HR@K = \frac{\text{Number of hits}@K}{|R|}, \tag{2.15}$$

where $|R|$ is the number of interactions in the test set.

- **Normalized Discounted Cumulative Gain.** The ranking quality of the recommendation list is usually evaluated with NDCG [93]. NDCG accounts for the position of the $hits$ by assigning higher scores to the $hits$ at top ranks and downgrading the scores to $hits$ at lower ranks . NDCG@K is defined as:

$$NDCG@K = \sum_{j=1}^{K} \frac{2^{r_j} - 1}{\log_2(j+1)}, \tag{2.16}$$

where r_j shows the graded relevance of target item at position j: $r_j = 1$ if the target item is ranked at the j-th position, otherwise $r_j = 0$.

Evaluated Methods. The experiments evaluated the following recommendation methods.

Deep learning-based (DL-based) RSs:

- **Wide&Deep** [39]. Wide&Deep jointly trains wide linear models and deep neural networks to combine the benefits of memorization and generalization for recommendations.
- **NeuCF** [57]. NeuCF jointly learns a neural network and a matrix factorization model. We integrate our embedding mechanisms into NeuCF for the cold-start recommendations.
- **AFN** [63]. AFN constructs logarithmic transformation layers to adaptively learn arbitrary-order cross features from user and item side information.

Meta-learning-based (ML-based) RSs:

- **MeLU** [54]. MeLU provides user-specific recommendations by locally customizing the global model to generate individual models for every user.
- **MAMO** [55]. This method further develops MeLU to provide personalized initialization for model parameters with the guidance of three memory networks.
- s^2**Meta-u** [66]. s^2Meta-u develops a meta-learning framework to generate individual models for different scenarios, where scenarios are denoted as users.
- s^2**Meta-p** [66]. s^2Meta-p considers periods as scenarios and provides period-specific recommendations.
- **CMML** [69]. CMML learns a context embedding from a user's past interactions to effectively specify the learned user-item features.
- **HML4Rec**. This is our proposed recommender system with the concatenating aggregation function described in Subsection 2.4.2.
- **HML4Rec-AFN**. This is our proposed recommender system with the logarithmic transformation network as described in Subsection 2.4.2.

For fair comparisons, we utilized the same base model for MeLU, MAMO, s^2Meta-u, s^2Meta-p, and HML4Rec, while CMML requires a more complex model to incorporate context information. In addition, we aligned the base model of HML4Rec-AFN with AFN. Some important characteristics of evaluated methods are listed in Table 2.2. As this table shows, our HML4Rec can model users' period-specific preferences as well as handle individual bias.

2.5.3 Implementation Detail

The codes of Wide&Deep[4], NeuCF[5], AFN[6], MeLU[7], MAMO[8], and s^2Meta[9] were obtained from the corresponding GitHub repositories. Our HML4Rec and CMML were implemented by using PyTorch framework. For our model, in E-commerce dataset, we learned a 16-dimensional embedding for each feature (32-dimension for MovieLens dataset). The learned vectors were concatenated and fed into a 5 layer multi-layer perceptron (MLP) with the shape of $\{128, 64, 32, 16, 8\}$ ($\{128, 64, 32, 8, 4\}$ for MovieLens dataset). The output layer was locally updated in the user local update step, and the last 3 layers of MLP were locally updated in the period local update step. In MovieLens dataset, we updated the output layer and the last 3 layers of MLP in the user local update step and the embedding layer and the first 2 layers of MLP in the period local update step. We also conducted the random search for all evaluated methods to tune their hyper-parameters based on the performance on the validation set, where the hyper-parameter set includes the learning rate of user local updates α from $\{5 \times 10^{-6}$, 5×10^{-5}, $5\times10^{-4}\}$, the learning rate of period local updates β from $\{5\times10^{-5}$, 5×10^{-4}, $5 \times 10^{-3}\}$, the learning rate of global update γ from $\{0.0005, 0.0001, 0.001, 0.0025, 0.0125\}$, the safety margin size z from $\{0.2, 0.6, 1\}$.

[4]https://github.com/jrzaurin/pytorch-widedeep
[5]https://github.com/yihong-chen/neural-collaborative-filtering
[6]https://github.com/shenweichen/DeepCTR-Torch
[7]https://github.com/hoyeoplee/MeLU
[8]https://github.com/dongmanqing/Code-for-MAMO
[9]https://github.com/THUDM/ScenarioMeta

Table 2.3: Comparison between HML4Rec and other methods on E-commerce. Performance ± 95% confidence intervals are reported.

Type	Method	HR@1	HR@3	NDCG@3
Warm scenario	Wide&Deep	0.0103 ± 0.0011	0.0278 ± 0.0017	0.0202 ± 0.0013
	NeuCF	0.1904 ± 0.0331	0.2936 ± 0.0394	0.2503 ± 0.0368
	AFN	0.2036 ± 0.0325	0.3193 ± 0.0398	0.2713 ± 0.037
	MeLU	0.0695 ± 0.0109	0.1374 ± 0.0126	0.1084 ± 0.0118
	MAMO	0.1595 ± 0.0374	0.2262 ± 0.0459	0.1985 ± 0.0425
	s^2Meta-u	0.0151 ± 0.0042	0.0413 ± 0.0086	0.0262 ± 0.0048
	s^2Meta-p	0.2155 ± 0.0089	0.2949 ± 0.0052	0.2641 ± 0.0048
	CMML	0.1662 ± 0.0384	0.2618 ± 0.0523	0.2222 ± 0.0466
	HML4Rec	**0.2386 ± 0.0422**	**0.3390 ± 0.0469**	**0.2975 ± 0.0452**
	HML4Rec-AFN	0.2116 ± 0.0374	0.3146 ± 0.0397	0.2718 ± 0.0385
Cold scenario	Wide&Deep	0.0090 ± 0.0020	0.0263 ± 0.0032	0.0187 ± 0.0025
	NeuCF	0.2080 ± 0.0416	0.3134 ± 0.0513	0.2692 ± 0.0472
	AFN	0.234 ± 0.0425	0.3561 ± 0.0517	0.3056 ± 0.0481
	MeLU	0.0593 ± 0.0124	0.1253 ± 0.015	0.0969 ± 0.0135
	MAMO	0.1042 ± 0.0278	0.1717 ± 0.0397	0.1430 ± 0.0347
	s^2Meta-u	0.0208 ± 0.0093	0.0474 ± 0.0187	0.052 ± 0.0141
	s^2Meta-p	0.2137 ± 0.0164	0.2945 ± 0.0172	0.2559 ± 0.0161
	CMML	0.1187 ± 0.0271	0.2047 ± 0.0358	0.1685 ± 0.0324
	HML4Rec	**0.2744 ± 0.0538**	**0.3703 ± 0.0618**	**0.3311 ± 0.0588**
	HML4Rec-AFN	0.2356 ± 0.0455	0.3454 ± 0.056	0.2999 ± 0.0512

2.5.4 Performance Comparison

Overall Performance. We report the average recommendation performance over test periods for all the evaluated methods. The comparison results on E-commerce are listed in Table 2.3. From this table, we can see that HML4Rec outperforms all comparisons in both warm and cold scenarios. It gains 10.72% HR@1, 6.17% HR@3, and 9.46% NDCG@3 improvements against the strongest baseline in the warm scenario and also achieves 16.47% HR@1, 3.99%, and 8.34% NDCG@3 improvements in the cold scenario. This is due to the strong capability of the hierarchical meta-training algorithm in capturing the users' period-specific preferences and alleviating the cold start problem.

Table 2.4: Comparison between HML4Rec and other methods on MovieLens. Performance ± 95% confidence intervals are reported.

Type	Method	HR@1	HR@3	NDCG@3
Warm scenario	Wide&Deep	0.0073 ± 0.0046	0.0245 ± 0.0083	0.0171 ± 0.0062
	NeuCF	0.0770 ± 0.0132	0.1875 ± 0.0144	0.1391 ± 0.0114
	AFN	0.0751 ± 0.0146	**0.1896 ± 0.0163**	**0.1405 ± 0.012**
	MeLU	0.0435 ± 0.011	0.1163 ± 0.0166	0.0847 ± 0.0117
	MAMO	**0.0780 ± 0.0176**	0.1808 ± 0.0185	0.1373 ± 0.0158
	s^2Meta-u	0.0294 ± 0.0179	0.0709 ± 0.0225	0.0528 ± 0.012
	s^2Meta-p	0.0075 ± 0.0081	0.0222 ± 0.0120	0.0094 ± 0.0087
	CMML	0.0667 ± 0.0112	0.1518 ± 0.0114	0.1166 ± 0.0083
	HML4Rec	0.0611 ± 0.0127	0.1613 ± 0.0172	0.1161 ± 0.0139
	HML4Rec-AFN	0.0757 ± 0.0119	0.1611 ± 0.0107	0.1245 ± 0.0083
Cold scenario	Wide&Deep	0.0101 ± 0.0028	0.0316 ± 0.004	0.0224 ± 0.0033
	NeuCF	0.0899 ± 0.0042	0.1868 ± 0.0053	0.1454 ± 0.0036
	AFN	**0.0938 ± 0.0037**	0.1921 ± 0.0044	0.1498 ± 0.0032
	MeLU	0.0570 ± 0.0037	0.1356 ± 0.0064	0.1018 ± 0.0049
	MAMO	0.0782 ± 0.0067	0.1853 ± 0.0091	0.1396 ± 0.0076
	s^2Meta-u	0.0364 ± 0.0052	0.0972 ± 0.0075	0.0582 ± 0.0046
	s^2Meta-p	0.0383 ± 0.0064	0.0883 ± 0.0102	0.0697 ± 0.0075
	CMML	0.0891 ± 0.0055	0.1934 ± 0.0056	0.149 ± 0.0052
	HML4Rec	0.0842 ± 0.0042	0.1926 ± 0.0054	0.1465 ± 0.0032
	HML4Rec-AFN	0.0882 ± 0.0058	**0.1997 ± 0.0048**	**0.1523 ± 0.0046**

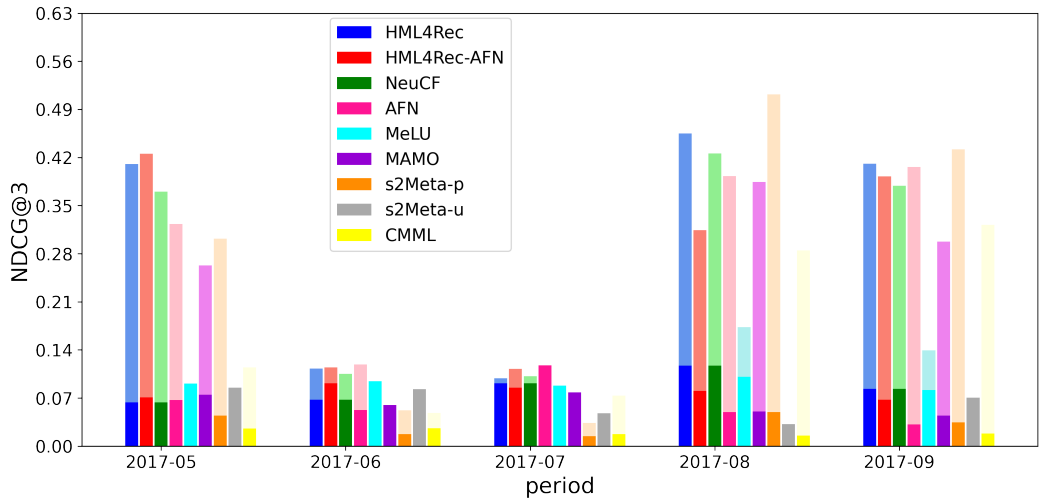
AFN outperforms other baselines due to its sophisticated logarithmic transformation layers. Besides, s^2Meta-p achieves the best performance among all meta-learning-based comparisons. This result demonstrates the effectiveness of capturing users' period-specific features in flash sale recommendations. By comparing the result of s^2Meta-u, MeLU, and MAMO, we observe that MAMO achieves the best performance. This observation demonstrates that simplifying the local update improves recommendations due to the limited user-wise interactions in this dataset. In addition, HML4Rec performs better than HML4Rec-AFN. This result also suggests that limited user-wise interaction data is hard to specify the sophisticated AFN model.

Table 2.5: Free item proportion on E-commerce.

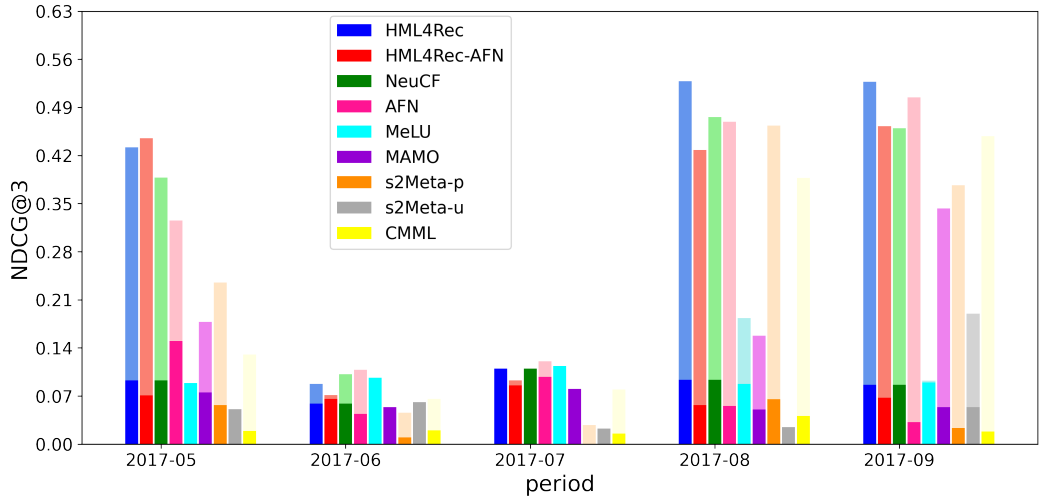
Period	May	Jun	Jul	Aug	Sup
Warm scenario	0.393	0.031	0.033	0.480	0.448
Cold scenario	0.407	0.015	0.008	0.544	0.634

The comparison results on MovieLens are reported in Table 2.4. We can find that the performances of all the methods are not as good as the performances on E-commerce dataset. This is because of the scarce user and item contents. HML4Rec-AFN and AFN outperform the other methods in most cases due to their well-designed model structure in learning cross-features from scarce content. Specifically, HML4Rec-AFN achieves the best HR@3 and NDCG@3 in the cold scenario due to its user local update that handles the bias in user preferences caused by individual differences. Inversely, HML4Rec-AFN needs to split the support users for the period local update and further split the support interactions for the user local update. As a result, the amount of data used to train the global model is reduced. This makes HML4Rec-AFN lose its superiority in the warm scenario where user preferences are relatively consistent. In conclusion, Although MovieLens is a non-flash sale dataset, HML4Rec-AFN still works well when making recommendations for cold users.

Period-wise Performance. To evaluate the effectiveness of HML4Rec in handling users' period-specific preferences, we further investigated the performance on NDCG@3 among different periods. The results on E-commerce are illustrated in Figure 2.4. In this figure, light and dark colors represent NDCG@3 on free and paid items, respectively. From Figure 2.4 and Table 2.5, we can observe that the ratio of free items mainly causes the variance of NDCG@3 among different periods. Moreover, HML4Rec outperforms other baselines when recommending free items in the periods of *5/2017*, *8/2017*, and *9/2017*. Such free items are selected by flash sale strategies so that they can describe users' period-specific preferences to some extent. Hence, the above result demonstrates that our hierarchical meta-training algorithm is critical to model and capture users' period-specific preferences in flash sale recommendations. In addition, for the periods 5/2017, 8/2017, and 9/2017, NDCG@3 of NeuCF on the cold scenario is larger than on



(a) NDCG@3 on warm scenario



(b) NDCG@3 on cold scenario

Figure 2.4: NDCG@3 on E-commerce.

the warm scenario. This indicates that warm users' interactions in past periods do not help infer their current preferences. To address this problem, our model considers the interactions in each period individually and locally updates our model to adapt to each period. Therefore, HML4Rec achieves a better performance in these periods than deep

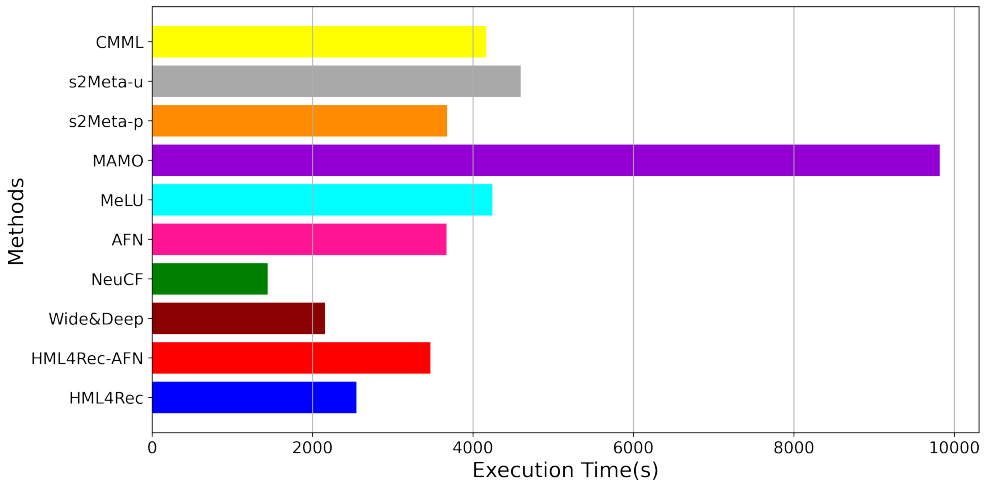


Figure 2.5: Execution time on E-commerce

learning-based baselines.

2.5.5 Execution Time Comparison

In this part, we compared the execution time for all comparisons on E-commerce dataset, where the execution time indicates the time to train a model until the loss converges on the validation set. All methods run on a Linux server with 64 Intel(R) Xeon(R) Platinum 8375C CPUs. Figure 2.5 illustrates the performance of execution time on E-commerce dataset. This figure shows that NeuCF and Wide&Deep require the least training time due to their light model and simple training algorithm. Inversely, MAMO tasks the longest time due to locally updating its two memory modules. HML4Rec converged within one hour and satisfied the time requirement of flash sale e-commerce recommendations.

2.5.6 Ablation Study

To better understand the contribution of different components in HML4Rec, we performed ablation experiments on E-commerce dataset with variants of HML4Rec, including 1) w/o PLU: HML4Rec without period local update, 2) w/o HLU: HML4Rec

Table 2.6: Performance on variants of HML4Rec. NDCG@3 ± 95% confidence intervals are reported.

	Warm scenario	Cold scenario
w/o PLU	0.2843 ± 0.0453	0.3151 ± 0.0597
w/o HLU	0.2754 ± 0.0384	0.3143 ± 0.0532
w/o T-net	0.2931 ± 0.0482	0.3289 ± 0.0558
w/o ResNet	0.2888 ± 0.0477	0.317 ± 0.0591
w/o EGplr	0.2824 ± 0.0432	0.3156 ± 0.061
HML4Rec	**0.2975 ± 0.0452**	**0.3311 ± 0.0588**

without the hierarchical local update, 3) w/o T-net: HML4Rec without warped activation spaces, 4) w/o ResNet: HML4Rec without residual layer, and 5) w/o EGplr: HML4Rec without entropy-guided period learning rate. The w/0 PLU variant is trained by setting the period local update learning rate β as 0, and the w/o HLU variant is trained by setting both the user local update learning rate α and the period local update learning rate β as 0.

Table 2.6 shows the results of HML4Rec and its variants in both warm and cold scenarios. From this Table, we can find that all the evaluated components boost recommendation accuracy. Specifically, NDCG@3 on warm scenario decreases most, i.e., about 8.02%, without the hierarchical local update. This result demonstrates the effectiveness of the hierarchical local update in modeling users' period-specific preferences, especially when users' interactions in previous flash sale periods are not helpful and even mislead the current recommendations. In addition, the residual layer and entropy-guided period learning rate are introduced to alleviate overfitting and work better in the cold scenario.

2.6 Discussion

2.6.1 Advantage of HML4Rec

In this chapter, we proposed a hierarchical meta-learning-based RS (HML4Rec) that handles users' period-specific preferences and alleviates the cold-start problem simultaneously. Motivated by the effectiveness of existing meta-learning-based RSs [54, 55, 69] in alleviating the cold-start problem and modeling the individual bias in user preferences, we applied the same local update as the user local update in Subsection 2.4.3. Moreover, HML4Rec improves existing meta-learning-based approaches by hierarchically combining a novel period local update and the widely used user local update to handle users' periodically changed preferences. This periodic change is caused by the update of available items and discounted items according to the provider's sales strategies in flash sale e-commerce scenarios. In addition, the hierarchical meta-training algorithm in HML4Rec learns prior knowledge shared by users and periods, enabling the trained model to fast adapt to new users and periods.

2.6.2 Limitation of HML4Rec

As mentioned in Subsection 2.5.4, Meta-learning-based RSs, including HML4Rec, use a part of data (i.e., the support set) for the local update, and the goal of such local update is to generalize the learned model to fit different tasks, rather than to improve the performance of the model for a particular task. As a result, these RSs lose their superiority when making recommendations for the scenarios in that users' preferences are relatively consistent. This is why meta-learning-based RSs perform worse than deep-learning-based RSs in the warm scenario (e.g., our tests on MovieLens dataset). One possible way to tackle this issue is to design a module that measures the difference between the past recommendation tasks and a new task. Then, the module automatically determines how much support data is required for this task based on the measured difference.

Besides, HML4Rec applies the one-step gradient descent approach to locally update parameters in both user and period local update steps. Although this approach alleviates the overfitting problem when the task-wise support data is limited, it also leads

to an under-fitting problem due to the limited support data. To address this problem, MAMO and CMML learn and provide more prior knowledge to simplify the learning task for their local update. From the experimental result in Table 2.3, we observe that MAMO and CMML outperform the baselines that apply the one-step gradient descent approach (i.e., MELU and s^2Meta-u). Due to the above observations, simplifying the local update is a promising direction to address the under-fitting problem and improve our HML4Rec.

2.7 Conclusion and Future Work

In this chapter, we developed a hierarchical meta-learning-based recommender system (HML4Rec) for flash sale e-commerce recommendations. HML4Rec can handle users' period-specific preferences caused by the periodically discounted item changes in flash sale e-commerce. Considering that previous deep learning-based RSs and meta-learning-based RSs ignore the above period-specific preferences, we introduced a period local update that can specify the recommendation model to estimate users' preferences in different periods accurately. Besides, inspired by the effectiveness of previous meta-learning-based RSs in alleviating the cold-start problem caused by limited user-wise interactions, we proposed a novel hierarchical meta-training algorithm by combining the period local update and a user local update. The hierarchical meta-training learns user- and period-shared knowledge and thus can fast adapt to new users and periods with only a few updates and limited training data.

To evaluate the effectiveness of HML4Rec, we conducted extensive experiments on a private flash sale e-commerce dataset and a public non-flash sale dataset. The experimental results on the flash sale e-commerce dataset show that our model outperforms all comparison methods in flash sale recommendations, especially when new interaction patterns appear. Moreover, the results on the public dataset demonstrate the effectiveness of our model in non-flash sale cold-start recommendations.

HML4Rec leverages a part of interaction data for locally updating the learned model to fit different users and periods. As a result, it loses advantages in scenarios where users' preferences are relatively consistent. Besides, the one-step gradient descent in the hierarchical local update of HML4Rec may cause an under-fitting problem. In the

future, we would like to improve HML4Rec in scenarios with consistent user preferences by designing modules that can automatically select support data according to the difference between new tasks and current tasks. In addition, we will attempt to develop simplified local update methods and fit the limited task-wise support data with the simplified local update to address the under-fitting problem.

Chapter 3

Debiasing Graph Transfer for Non-overlap Cross-domain Recommendations

3.1 Introduction

RSs predict users' future item-interacted behaviors, where the predictions are inferred by models learned from their past interaction behaviors [36, 94]. Deep learning has been employed in RSs, with its well-designed structures and its large number of learnable parameters, to better model users' complex interaction patterns. In real-world services, most users interact with only a few items, particularly in start-up companies and when companies develop new services. In such scenarios, deep learning-based RSs may lead to overfitting because of the sparse interaction data [10, 11], which significantly degrades their recommendation performances. As a result, it is an essential and practical requirement for services with sparse interaction data to improve their recommendation performances by leveraging rich public data. Cross-domain recommender systems (CDRSs) are a promising approach that exploits the data from an auxiliary domain (i.e., a source domain) to facilitate the inference process in a target domain [29, 51, 95, 96]. For example, RecSys-DAN [51] and CFAA [96] extract user preferences from the source domain and transfer the learned preferences to the target domain,

where preferences are defined as the learned user (item) embeddings or the distribution of predictions.

3.1.1 Challenge

However, existing CDRSs, including the above CDRSs, suffer from a negative transfer issue in the scenario of leveraging public data. The negative transfer issue means that the source knowledge degrades recommendation performances in the target domain. This issue is caused by the misleading source interactions that present users' unique interests within the source domain, i.e., domain-specific preferences.

Some CDRSs address this issue by merging personal interactions between the two domains and operating directly on source interactions to reduce the impact of the ones that present domain-specific preferences [16, 17, 25, 26, 27]. For example, systems in [28, 29] designed an attention-based and a graph-based approaches that adaptively assign minimal weights to these source interactions. However, merging personal interactions requires finding users having interactions in both domains (i.e., domain-shared users). Finding such users relies on characteristics that can identify individuals, e.g., user profiles. Due to privacy concerns, public data cannot provide such personal characteristics. As a result, a method that can handle the negative transfer in this scenario is demanded. The challenges we overcome in this chapter are summarized below.

- The first challenge is bridging domains and transferring knowledge without using domain-shared users.
- The second challenge is addressing the negative transfer issue without merging personal interactions between the source and the target domains.

3.1.2 Contribution

We develop a new CDRS, namely semantic clustering enhanced debiasing graph neural recommender system (shortly SCDGN), to address the above challenges. To remove the requirement of domain-shared users, SCDGN jointly clusters items in the source and the target domains and transfers knowledge between the two domains based on these clusters (i.e. bridging the two domains by clusters). By doing so, the source knowledge

can be transferred to the target domain without domain-shared users. Besides, SCDGN merges cluster-level interactions between the source and the target domains and assigns minimal weights to the source interactions that present domain-specific preferences at a cluster level. In this way, SCDGN is expected to handle the negative transfer issue. More precisely, we generate a cluster-level cross-domain graph to bridge the two domains, and the graph consists of users and item clusters. Item clusters are generated from textual information on items (e.g. movie titles and web page contents), so that the cross-domain graph can merge the semantic interaction information of different domains, which makes domain-shared users unnecessary.

Then, we extract knowledge from two interaction graphs, including an item-level graph in the target domain and the above cross-domain graph, by devising a CDRS variant of LightGCN [97]. This variant is inspired by the success of LightGCN in extracting complex high-hop neighbor information from graph structures. To handle the negative transfer issue, we develop a novel debiasing graph convolutional layer that adaptively weights the interactions in the cross-domain graph, particularly for the ones from the source domains. More precisely, we design adaptive debiasing vectors for users and item clusters to weight edges in the cross-domain user-cluster graph. Inspired by the effectiveness of debiasing learning [98], moreover, we develop two-level restrictions to learn the above debiasing vectors.

To summarize, our contributions are as follows:

- We propose a novel semantic cluster-based domain merge approach to make the interaction information transferable at an item-cluster level. By doing so, Our CDRS does not require domain-shared users.
- We develop a new debiasing graph convolutional layer to adaptively weight source interactions, alleviating the negative transfer issue.
- Our experimental results on three public datasets and two private datasets demonstrate that our proposal significantly outperforms state-of-the-art methods.

In Section 3.2, we review related works. Then, we introduce some preliminaries in Section 3.3. Section 3.4 elaborates on the details of our proposed method. The experimental results are summarized in Section 3.5. Section 3.6 presents our conclusion.

3.2 Related Work

3.2.1 Cross-domain Recommender Systems

To mitigate the data sparsity problem, CDRSs leverage data from an auxiliary source domain to facilitate recommendations in the sparse target domain. Some existing CDRSs require domain-shared users to bridge the source and the target domains [23, 25, 95, 99]. With these users, the source domain can transfer individual-level knowledge to the target domain. Learning transformation function and domain adversarial learning are two promising directions for cross-domain recommendations. The former learns transformation functions to transfer user or item embeddings from the source domain to the target domain. For example, CGN [25] proposed a novel generative adversarial network to transfer item embeddings in a set manner. DOML [99] learned a latent orthogonal metric mapping to transfer the user embedding between domains. PTUPCDR [100] considered the bias caused by personal differences and introduced a meta-learning method that learns user-specific transformation functions to handle the personal difference bias for cross-domain recommendations. Meanwhile, domain adversarial learning aligns source and target embedding spaces to transfer knowledge from the source domain to the target domain. For example, DARec [101] developed a deep domain adaptation model to transfer rating patterns. RecGURU [95] introduced a transformer network and minimized Kullback–Leibler divergence between the learned distributions of latent user representations to learn the domain-invariant embeddings for cross-domain sequential recommendations.

However, matching users is an arduous task and may involve privacy issues in most real-world applications. Considering privacy and the scalability of methods, some studies avoid user alignment and transfer distribution-level knowledge from the source domain to the target domain. MMT-DRR [49] regularized the target domain's user and item embedding space with the embedding space learned in source domains. However, MMT-DRR cannot work without domain-shared contextual information. ESAM [15] and CFAA [96] removed the requirement of domain-shared contextual information and aligned the attribution distribution and correlation between source and target embedding spaces to transfer knowledge. Besides, RecSys-DAN [51] proposed a novel discriminator and minimized the divergence of the predictions between the source domain and

the target domain for knowledge transformation. Unfortunately, users' interaction patterns (i.e., preferences) heavily depend on domains. The above-mentioned methods ignore domain-dependent preferences and are easily misled by specific preferences in the source domain (the negative transfer issue). This significantly degrades their recommendation performances. We hence propose a CDRS that can handle the negative transfer issue without using domain-shared users.

3.2.2 Graph Convolution in Recommendations

Recently, graph neural networks (GNNs) have been employed in RSs to guide the embedding learning by exploiting user-item graph structures [102, 103, 104, 105]. PinSage [106] and NGCF [107] defined the information propagation as an aggregation of the embeddings of neighbors to enhance the target node's (i.e., users' or items') embedding. Considering that recommender systems often use one-hot embedding (i.e., less information than images and text), SGCN [108] and LightGCN [97] further simplified and customized graph models to avoid overfitting. In addition, some GNN-based CDRs also alleviate the sparse problem by combining the complex high-order graph structural information from the source to the target [28, 29, 109, 110]. For example, GA-DTCDR [28] constructed heterogeneous graphs to learn user and item embeddings and developed an element-wise attention mechanism to combine the embeddings of users learned from both domains.

However, the above-mentioned GNN-based CDRs require domain-shared users to connect domains and ignore the negative transfer issue in user preference patterns. BiT-GCF [29] developed a domain-specific feature propagation layer to handle the negative transfer issue, but it still requires domain-shared users to fuse domain information. In light of the above causal view, we develop a GNN-based CDRS that requires no domain-shared users and can handle the negative transfer issue.

3.3 Preliminaries

3.3.1 Problem Formulation

In this chapter, we define the top-K recommendations in a sparse domain as our recommendation task. We consider an auxiliary source domain $\mathcal{D}_s$ and a sparse target domain $\mathcal{D}_t$. $\mathcal{D}_t$ contains $\mathcal{U}_t$, $\mathcal{V}_t$, and $\mathcal{R}_t$, where $\mathcal{U}_t$ ($\mathcal{V}_t$) denotes the user (item) set and $\mathcal{R}_t$ is the interaction set between them. Similarly, $\mathcal{D}_s$ contains $\mathcal{U}_s$, $\mathcal{V}_s$, and $\mathcal{R}_s$. There is no overlap between the user and item sets of $\mathcal{D}_s$ and $\mathcal{D}_t$.

To address the data sparsity problem, we consider the semantic clustering information of items $\mathcal{C}$ extracted from both the source and target domains, because this enhances the sparse interactions in the target domain. As a result, each interaction $r \in \mathcal{R}_t$ is a tuple $r = (u, v, c)$, where $u \in \mathcal{U}_t$, $v \in \mathcal{V}_t$, and $c \in \mathcal{C}$. For each user $u \in \mathcal{U}_t$, we predict a preference score $\hat{y}_{u,v}$ for each item $v \in \bar{\mathcal{V}}_u = \{v \in \mathcal{V}_t, v \notin \mathcal{V}_u\}$, where $\mathcal{V}_u$ is the items that interacted with u. We then rank the items in $\bar{\mathcal{V}}_u$ according to their preference scores and recommend the top-K items with the largest scores to u.

3.3.2 Simplified Graph Convolution for RS

LightGCN [97] is a graph convolution network that refines user and item embedding by extracting structural information, particularly high-hop neighbors, from the user-item interaction graph. ID-based embedding in RSs contains less available information than words in text and pixels in images. Hence, LightGCN removes the non-linear projection and the self-connection operations from its massage propagation. More precisely, the l-th simplified graph convolution (i.e., massage propagation) layer in LightGCN is defined as:

$$\begin{aligned} \mathbf{e}_u^{(l+1)} &= \sum_{v \in \mathcal{V}_u} \frac{1}{\sqrt{|\mathcal{V}_u|}\sqrt{|\mathcal{U}_v|}} \mathbf{e}_v^{(l)}, \\ \mathbf{e}_v^{(l+1)} &= \sum_{u \in \mathcal{U}_v} \frac{1}{\sqrt{|\mathcal{U}_v|}\sqrt{|\mathcal{V}_u|}} \mathbf{e}_u^{(l)}, \end{aligned} \tag{3.1}$$

where $\mathbf{e}_u^{(0)}$ and $\mathbf{e}_v^{(0)}$ are the ID embeddings of user u and item v, respectively. $\mathcal{U}_v$ is a set of users that interacted with v. This graph convolutional layer has been analytically and empirically proven to be effective in accelerating the training process and alleviating the

data sparsity problem.

3.4 Proposed Method

3.4.1 Overview

Motivated by the observations that existing CDRSs require user matching or suffer from the negative transfer issue, we propose a novel CDRS, which is depicted in Figure 3.1 and does not have these drawbacks. The numbers below correspond to the ones in Figure 3.1.

(1) Our idea for avoiding user matching is to merge the source and target domains by using *semantic information of items*. Such semantic information is extracted via a pre-trained representation extractor BERT [111]. (2) Then, items in source and target domains are clustered together based on their semantic information. (3) After that, we construct a cross-domain user-cluster graph that contains edges between users and item clusters. With this graph, the cluster-level interactions in source and target domains are merged together to transfer knowledge. (4) Our prediction model fuses knowledge from the cross-domain graph and a target interaction graph to improve the expression of users. To tackle the negative transfer issue, we introduce a novel debiasing graph convolutional layer that adaptively assigns minimal weights for source interactions representing domain-specific preferences. (5) Finally, thanks to the above novel ideas, we can expect a high inner product of a user and an item to which she would prefer, thus yielding an accurate recommendation list. Table 3.1 summarizes important notations used in this chapter.

3.4.2 Semantic Domain Fusing

To semantically fuse the source and target domains, we first embed all items in the source and target domains into a domain-shared embedding space. Then, we cluster items based on this embedding space and construct a cross-domain user-cluster graph to enhance the interaction information.

Semantic Item Embedding. Given textual information on items, such as a description

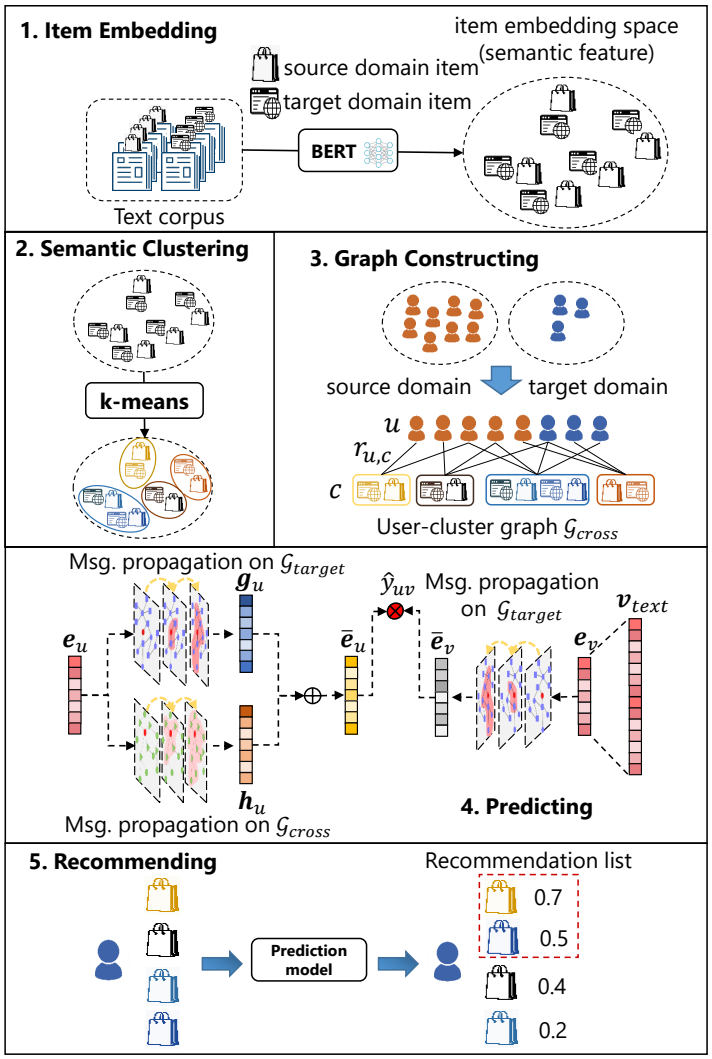


Figure 3.1: Overview of our proposed CDRS. (1) The item embedding learns semantic embeddings for all items in source and target domains. (2) All items in the source and target domains are then clustered by their semantic embeddings. (3) A cross-domain user-cluster graph is constructed to merge the two domains' interaction information at a semantic cluster level. (4) A debiasing graph convolutional neural network makes predictions by leveraging the interaction information from the target user-item and cross-domain user-cluster graphs. (5) Finally, recommendations are produced by the prediction results.

Table 3.1: Summary of notations

Notation	Description
u	a user
v	an item
c	a cluster (a set of items)
$\mathcal{G}_{cross}$	cross-domain user-cluster graph
$\mathcal{G}_{target}$	target user-item graph
$\mathcal{N}_u$	neighbor cluster set of u in $\mathcal{G}_{cross}$
$\mathcal{N}_c$	neighbor user set of c in $\mathcal{G}_{cross}$
$\mathcal{M}_u$	neighbor item set of u in $\mathcal{G}_{target}$
$\mathcal{M}_v$	neighbor user set of v in $\mathcal{G}_{target}$
$\mathbf{v}_{txt}$, $\mathbf{c}_{txt}$	semantic vectors of v and c
$\mathbf{e}_u$, $\mathbf{e}_v$, $\mathbf{e}_c$	embedding of u, v, and c
$\mathbf{a}_u$, $\mathbf{a}_c$	debiasing vectors of u and c
$\bar{\mathbf{e}}_u$, $\bar{\mathbf{e}}_v$, $\bar{\mathbf{e}}_c$	unbiased final embedding of u, v, and c
$\bar{\mathbf{e}}'_u$, $\bar{\mathbf{e}}'_c$	biased final embedding of u and c
$\mathbf{h}_u^{(l)}$, $\mathbf{h}_v^{(l)}$	$\mathcal{G}_{target}$'s l-th graph conv. layer outputs
$\mathbf{g}_u^{(l)}$, $\mathbf{g}_c^{(l)}$	$\mathcal{G}_{cross}$'s l-th debiasing graph conv. layer outputs
$\mathbf{g}_u'^{(l)}$, $\mathbf{g}_v'^{(l)}$	$\mathcal{G}_{cross}$'s l-th graph conv. layer outputs

of a product, we extract semantic features from the text information to represent items in the source and target domains. We apply the token embeddings from a pre-trained BERT [111] to represent tokens in item text because this model is learned by sufficient Wikipedia data and hence contains semantic information. The text of item v is denoted by $\text{text}(v)$, and an semantic embedding of item v is obtained by

$$\mathbf{v}_{txt} = \sum_{w \in \text{text}(v)} \phi_{tf\text{-}idf}(w) \cdot \phi_{\text{BERT}}(w), \tag{3.2}$$

where $\phi_{\text{BERT}}(w)$ and $\phi_{tf\text{-}idf}(w)$ are respectively the embedding and the tf-idf score of token $w \in \text{text}(v)$. Note that $\phi_{tf\text{-}idf}(w)$ is calculated based on the text corpus collected from both domains.

User-cluster Graph Construction. We next construct a user-cluster graph. This aims at merging the source and target domains *without* user, item, and side information alignments. In addition, high-hop neighbors in this user-cluster graph can yield useful knowledge to improve the recommendation accuracy [97]. In Subsection 3.4.3, we leverage this observation through modeling such structures from this graph, which also motivates building this user-cluster graph.

To construct the user-cluster graph, we first run the semantic clustering in Figure 3.1, that is, we cluster all items from the source and target domains in the learned semantic embedding space. We employ the empirically effective k-means clustering [112] method and leave the discussion of more clustering methods as a future work. After that, we construct the cross-domain user-cluster graph $\mathcal{G}_{cross} = \{(u, r_{u,c}, c)|u \in \mathcal{U}, c \in \mathcal{C}\}$ to merge the two domains' semantic-level interaction information, where $\mathcal{U} = \{\mathcal{U}_s, \mathcal{U}_t\}$ and $\mathcal{C}$ respectively denote the user and cluster sets. The link $r_{u,c} = 1$ indicates that there is an interaction between u and any item belonging to c; otherwise $r_{u,c} = 0$.

3.4.3 Debiasing Graph Convolutional Predictor

We here develop a cluster-enhanced debiasing graph convolutional model for recommendations in the sparse target domain. Different from existing CDRSs that transfer item interaction patterns directly, our model transfers the semantic clustering interaction patterns via the cross-domain user-cluster graph $\mathcal{G}_{cross}$. To achieve this, our model fuses $\mathcal{G}_{cross}$ and the target user-item graph $\mathcal{G}_{target}$ to refines the user and item embeddings with structural knowledge from graphs, where $\mathcal{G}_{target} = \{(u, r_{u,v}, v)|u \in \mathcal{U}_t, v \in \mathcal{V}_t, r_{u,v} \in \mathcal{R}_t\}$.

This model consists of three main components: (i) an embedding layer, which learns latent vectors for users and items, (ii) debiasing graph convolutional layers, which recursively propagate unbiased high-hop neighbor information to refine the user and item vectors, and (iii) a prediction layer, which aggregates the user and item representations from all propagation layers and outputs the predictions.

Embedding Layer. To alleviate the data sparsity problem, we propose a novel approach that projects users into the item embedding space learned in Subsection 3.4.2. Furthermore, we design a metric-invariant dimension reduction approach to control the scale

of parameters according to the difficulty of the recommendation task and the sparsity of the training data.

The item embedding is calculated by a dimension compression layer: $\mathbf{e}_v = \mathbf{W}\mathbf{v}_{txt} + \mathbf{b}$, where $\mathbf{W}$ and $\mathbf{b}$ are the parameters of this layer. The dimension of $\mathbf{e}_v$ is much smaller than that of $\mathbf{v}_{txt}$, in order to adapt to the sparse target domain. The cluster embedding is computed by the same layer: $\mathbf{e}_c = \mathbf{W}\mathbf{c}_{txt} + \mathbf{b}$, where $\mathbf{c}_{txt}$ is the semantic embedding of cluster c. $\mathbf{c}_{txt}$ is defined as the mean pooling of all item semantic embeddings in this cluster and formulated by

$$\mathbf{c}_{txt} = \frac{1}{|\mathcal{V}_c|} \sum_{v \in \mathcal{V}_c} \mathbf{v}_{txt}, \tag{3.3}$$

where $\mathcal{V}_c$ is the item set in cluster c. The user embedding is defined as the ID embedding $\mathbf{e}_u$, which has the same dimension as that of $\mathbf{e}_v$.

We measure the cosine similarities S_c between items and clusters and minimize the mean squared error of the cosine similarities calculated before and after dimension reduction to ensure the metric invariance, where the error is defined as

$$\begin{aligned} \mathcal{L}_{dr} = \frac{1}{|\mathcal{R}_t|} \sum_{(u,v,c) \in \mathcal{R}_t} \Big(& \big(S_c(\mathbf{e}_v, \mathbf{e}_{v^-}) \text{-} S_c(\mathbf{v}_{txt}, \mathbf{v}_{txt}^-)\big)^2 \\ & + \big(S_c(\mathbf{e}_c, \mathbf{e}_{c^-}) \text{-} S_c(\mathbf{c}_{txt}, \mathbf{c}_{txt}^-)\big)^2 \Big) . \end{aligned} \tag{3.4}$$

In this equation, v^- is a negative item randomly sampled from $\bar{\mathcal{V}}_u$ and c^- is the cluster to which v^- belongs. This approach adjusts the embedding dimension and maintains a consistent spatial relationship with the original embedding space.

Debiasing Graph Convolutional Layers. Because of the superiority of graph convolutional networks in capturing and modeling structural information from graphs, we develop graph convolutional modules for extracting structural information from the target user-item graph $\mathcal{G}_{target}$ and the cross-domain user-cluster graph $\mathcal{G}_{cross}$. More precisely, we employ the state-of-the-art "light graph convolution" layer [97] to propagate graph information because of its effectiveness in alleviating overfitting for our sparse target domain.

To alleviate the impact of user preferences that appear only in the source domain (domain bias in preferences) and extract unbiased knowledge from $\mathcal{G}_{cross}$, we propose a novel debiasing graph convolutional layer. For each user $u \in \mathcal{U}$, we set an adaptive

debiasing vector $\mathbf{a}_u = (a_{u,1}, a_{u,2}, a_{u,3}, ..., a_{u,|\mathbf{e}_u|})$ to model this user's individual domain bias in preferences, where $\mathbf{a}_u$ has the same shape with user embedding $\mathbf{e}_u$. For each cluster $c \in \mathcal{C}$, we also set an adaptive debiasing vector $\mathbf{a}_c = (a_{c,1}, a_{c,2}, a_{c,3}, ..., a_{c,|\mathbf{e}_c|})$, where $\mathbf{a}_c$ has the same shape with cluster embedding $\mathbf{e}_c$. By doing so, the debiasing factor of user-cluster interaction $r_{u,c}$ can be defined as the inner product between $\mathbf{a}_u$ and $\mathbf{a}_c$:

$$a_{uc} = \mathbf{a}_u \cdot \mathbf{a}_c. \tag{3.5}$$

The l-th debiasing graph convolutional layer for $\mathcal{G}_{cross}$ is then formulated as:

$$\begin{aligned} \mathbf{g}_u^{(l+1)} &= \sum_{c \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_c|}} a_{uc} \cdot \mathbf{g}_c^{(l)}, \\ \mathbf{g}_c^{(l+1)} &= \sum_{u \in \mathcal{N}_c} \frac{1}{\sqrt{|\mathcal{N}_c|}\sqrt{|\mathcal{N}_u|}} a_{uc} \cdot \mathbf{g}_u^{(l)}, \end{aligned} \tag{3.6}$$

where $\mathcal{N}_u = \{c | r_{u,c} = 1, r_{u,c} \in \mathcal{G}_{cross}\}$ is the neighbor cluster set of user u and $\mathcal{N}_c$ is the neighbor user set of cluster c. We define $\mathbf{g}_u^{(0)} = \mathbf{e}_u$ and $\mathbf{g}_c^{(0)} = \mathbf{e}_c$. It is worth mentioning that we detach the gradient computation of the debiasing vectors $\mathbf{a}_u$ and $\mathbf{a}_c$ here for computational efficiency. The learning of $\mathbf{a}_u$ and $\mathbf{a}_c$ is left to the proposed restrictions in Subsection 3.4.4.

For $\mathcal{G}_{target}$, we adopt the standard "light graph convolution" layer [97], where the l-th graph convolutional layer is formulated as:

$$\begin{aligned} \mathbf{h}_u^{(l+1)} &= \sum_{v \in \mathcal{M}_u} \frac{1}{\sqrt{|\mathcal{M}_u|}\sqrt{|\mathcal{M}_v|}} \mathbf{h}_v^{(l)}, \\ \mathbf{h}_v^{(l+1)} &= \sum_{u \in \mathcal{M}_v} \frac{1}{\sqrt{|\mathcal{M}_v|}\sqrt{|\mathcal{M}_u|}} \mathbf{h}_u^{(l)}. \end{aligned} \tag{3.7}$$

$\mathcal{M}_u = \{v | r_{u,v} = 1, r_{u,v} \in \mathcal{G}_{target}\}$ is the neighbor item set of user u and $\mathcal{M}_v$ is the neighbor user set of item v. Similarly, we define $\mathbf{h}_u^{(0)} = \mathbf{e}_u$ and $\mathbf{h}_v^{(0)} = \mathbf{e}_v$.

Prediction Layer. We next refine $\mathbf{e}_u$, $\mathbf{e}_v$, and $\mathbf{e}_c$ by using the extracted graph structure information. The final representation is produced by aggregating the embeddings

obtained at each graph convolutional layer:

$$\begin{aligned} \bar{\mathbf{e}}_u &= \sum_{l=0}^{P} \mathbf{g}_u^{(l)} + \sum_{l=0}^{Q} \mathbf{h}_u^{(l)}, \\ \bar{\mathbf{e}}_v &= \sum_{l=0}^{Q} \mathbf{h}_v^{(l)}; \quad \bar{\mathbf{e}}_c = \sum_{l=0}^{P} \mathbf{g}_c^{(l)}, \end{aligned} \tag{3.8}$$

where P and Q are the numbers of debiasing graph convolutional layers for $\mathcal{G}_{cross}$ and graph convolutional layers for $\mathcal{G}_{target}$, respectively. It is worth mentioning that $\mathbf{e}_u$ is refined by the structural information from both the target and cross-domain graphs, i.e., the knowledge from both the item- and cluster-level interactions.

Finally, the preference score is defined as the inner product of the final representations between the user and the item:

$$\hat{y}_{uv} = \bar{\mathbf{e}}_u \cdot \bar{\mathbf{e}}_v \tag{3.9}$$

3.4.4 Restrictions for Debiasing Learning

The previous debiasing learning [98] calculates their restrictions via domain-shared users and domain-shared item attributions, e.g., category, seller, brand, and price, resulting in a limited application. Besides, it directly sets adaptive debiasing factors for each user-item interaction and optimizes them separately. In other words, the learning of a debiasing factor only relies on the corresponding interaction and thus suffers from a severe overfitting issue.

Based on these findings, we get hints from the matrix factorization algorithm and define the debiasing factor a_{uc} by Equation 3.5. Our approach learns $\mathbf{a}_u$ and $\mathbf{a}_c$ via the restriction losses at both prediction and individual levels.

Restriction in Prediction Level. As a debiasing factor, a_{uc} is demanded to produce unbiased prediction $\hat{y}_{uc}$ from the biased version $\hat{y}'_{uc}$. To achieve this, we set a restriction loss $\mathcal{L}_{rsp}$ that measures the mean squared error between $\hat{y}_{uc}$ and $a_{uc} \cdot \hat{y}'_{uc}$. $\mathcal{L}_{rsp}$ is defined as:

$$\mathcal{L}_{rsp} = \frac{1}{|\mathcal{R}_t|} \sum_{(u,v,c) \in \mathcal{R}_t} \left(\hat{y}_{uc} - a_{uc} \cdot \hat{y}'_{uc}\right)^2, \tag{3.10}$$

where $\hat{y}_{uc} = \bar{\mathbf{e}}_u \cdot \bar{\mathbf{e}}_c$ and $\hat{y}'_{uc} = \bar{\mathbf{e}}'_u \cdot \bar{\mathbf{e}}'_c$. The biased user embedding $\bar{\mathbf{e}}'_u$ and the biased cluster embedding $\bar{\mathbf{e}}'_c$ aggregate the output of every graph convolutional layers and are formulated as:

$$\bar{\mathbf{e}}'_u = \sum_{l=0}^{P} \mathbf{g}'^{(l)}_u; \quad \bar{\mathbf{e}}'_c = \sum_{l=0}^{P} \mathbf{g}'^{(l)}_c, \tag{3.11}$$

where $\mathbf{g}'^{(l)}_u$ and $\mathbf{g}'^{(l)}_c$ are the user and cluster aggregation results of the l-th graph convolutional layer that can be computed by Equation 3.6 without the debiasing factor. By minimizing $\mathcal{L}_{rsp}$, we can ensure a consistent result between the unbiased prediction and the prediction produced by the debiasing graph convolutional layers. As a result, $\mathcal{L}_{rsp}$ constrains the embedding space of $\mathbf{a}_u$ and $\mathbf{a}_c$ and hence can alleviate overfitting.

Restriction in Individual Level. At the individual level, $\mathbf{a}_u$ and $\mathbf{a}_c$ are required to generate unbiased $\bar{\mathbf{e}}_u$ and $\bar{\mathbf{e}}_c$ from the biased $\bar{\mathbf{e}}'_u$ and $\bar{\mathbf{e}}'_c$, respectively. To meet this requirement, we introduce a user restriction loss $\mathcal{L}_{rsu}$ and a cluster restriction loss $\mathcal{L}_{rsc}$. $\mathcal{L}_{rsu}$ measures the Euclidean distance between $\bar{\mathbf{e}}_u$ and $\mathbf{a}_u \odot \bar{\mathbf{e}}'_u$, where $\odot$ is the element-wise product. Similarly, $\mathcal{L}_{rsc}$ measures the Euclidean distance between $\bar{\mathbf{e}}_c$ and $\mathbf{a}_c \odot \bar{\mathbf{e}}'_c$. $\mathcal{L}_{rsu}$ and $\mathcal{L}_{rsc}$ can be written as:

$$\begin{aligned} \mathcal{L}_{rsu} &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \|\bar{\mathbf{e}}_u - \mathbf{a}_u \odot \bar{\mathbf{e}}'_u\|_2^2, \\ \mathcal{L}_{rsc} &= \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \|\bar{\mathbf{e}}_c - \mathbf{a}_c \odot \bar{\mathbf{e}}'_c\|_2^2. \end{aligned} \tag{3.12}$$

Minimizing $\mathcal{L}_{rsu}$ and $\mathcal{L}_{rsc}$ forces $\mathbf{a}_u$ and $\mathbf{a}_c$ to mitigate the bias in preferences for user u and cluster c. Therefore, $\mathbf{a}_u$ and $\mathbf{a}_c$ can be learned as the debiasing vectors.

3.4.5 Model Optimization

Because of removing the non-linear projection in the graph convolutional layers, the trainable parameters of our model θ are the user embedding $\mathbf{e}_u$, the parameters of the dimension reduction layer ($\mathbf{w}$ and b), the user debiasing embedding $\mathbf{a}_u$, and the cluster debiasing embedding $\mathbf{a}_c$. We consider these to optimize our model. We use the Bayesian Personalized Ranking (BPR) loss [113] to learn users' item preference scores. The BPR loss is obtained as:

$$\mathcal{L}_{bpr} = - \sum_{(u,v,c) \in \mathcal{R}_t} \ln \sigma \left(\hat{y}_{uv} - \hat{y}_{uv^-} \right), \tag{3.13}$$

where v^- is a negative item randomly sampled from $\bar{\mathcal{V}}_u$.

The total loss is measured by combining the dimension reduction loss $\mathcal{L}_{dr}$, the restriction loss $\mathcal{L}_{rs}$, and the BPR loss $\mathcal{L}_{bpr}$, that is,

$$\mathcal{L} = \mathcal{L}_{bpr} + \lambda_1 \mathcal{L}_{rs} + \lambda_2 \mathcal{L}_{dr} + \lambda_3 \|\theta\|^2, \tag{3.14}$$

where $\mathcal{L}_{rs} = \mathcal{L}_{rsp} + \mathcal{L}_{rsu} + \mathcal{L}_{rsc}$. λ_1, λ_2, and λ_3 are hyper-parameters used to balance the weight between losses. We employ a gradient descent algorithm to optimize θ by minimizing $\mathcal{L}$.

3.5 Experiments

The objective of our experiments is to answer the following research questions:

- **RQ1**: How does SCDGN perform on recommendations compared with state-of-the-art methods?
- **RQ2**: Does the proposed debiasing learning framework benefit recommendations?
- **RQ3**: Does the semantic clustering facilitate recommendations by fusing the knowledge from another domain?
- **RQ4**: Does the metric-invariant dimension reduction approach work in improving recommendation performance?
- **RQ5**: How does K (the recommendation list size) affect the recommendation accuracy of SCDGN?

3.5.1 Experiment Setting

Dataset. We conducted experiments on two proprietary datasets and three widely used public datasets to investigate the recommendation performance of SCDGN in practical applications and for benchmarking purposes.

The public datasets contain a subset of MovieLens25M[1] and two subsets of Amazon[2]. The subset of MovieLens25M (ML) contains movie ratings from *30/9/2016* to *1/10/2018*, where the movie descriptions in ML were collected from the public API of TMDB[3]. The two subsets of Amazon include an AmazonBook (AB) dataset and an AmazonMovie (AM) dataset. AB and AM contain book and movie ratings from *30/9/2016* to *3/10/2018*, respectively, as well as textual descriptions of the books and movies. The language of textual descriptions in public datasets is English.

The private datasets have an online advertisement dataset (ADs) [114] and an e-commerce dataset (E-com). ADs contains web browsing records from *1/8/2017* to *31/8/2017* on an ads platform and the textual content of Web pages. E-com provides purchase records from an e-commerce platform and the textual descriptions of products, where the purchase records in E-com have the same period as that of ADs. The language of textual content in private datasets is Japanese.

We measured three cross-domain recommendation tasks, where each recommendation task contains an auxiliary source domain and a relatively sparse target domain. We defined A→B as a cross-domain recommendation task, where A is the source domain, and B is the target domain. Considering the scenario of leveraging public data, we measure cross-domain tasks that source and target domains are from different services (i.e., companies). Besides, we experiment with the domains having the text contents of the same language for jointly clustering item embeddings in different domains. As a result, our recommendation tasks include (1) ADs→E-com, (2) ML→AM, and (3) ML→AB. Besides, we also measured the source-target inversion version of the above tasks: (4) E-com→ADs, (5) AM→ML, and (6) AB→ML. For each source domain, we selected users who have 3 to 10 interaction records and items that have 10 to 15 interaction records to fit a dense setting. Inversely, for each target domain, we selected users who have 3 to 5 interactions and items that have 5 to 15 interactions to form a relatively sparse environment. Some basic information about the pre-processed datasets is summarized in Table 3.2.

Evaluation Criteria. For each user in the target domains, we took this user's last and

[1] grouplens.org/datasets/movielens/25m/

[2] jmcauley.ucsd.edu/data/amazon/

[3] www.themoviedb.org/documentation/api

Table 3.2: Basic information on the datasets we used. #Int./U is the average number of interactions per user.

	Dataset	#Users	#Items	#Interactions	#Int./U
As Source	ML	18,232	14,435	421,803	23.14
	AM	22,046	7,814	104,216	4.73
	AB	27,662	12,708	129,899	4.70
	ADs	18,829	12,253	360,880	19.17
	E-com	17,418	6,142	81,499	4.68
As Target	ML	6,298	9,873	31,445	4.99
	AM	8,566	6,752	39,696	4.63
	AB	13,350	10,477	61,004	4.57
	ADs	11,010	12,031	55,050	5.00
	E-com	12,558	5,118	46,871	3.73

second-last interactions to form the test and validation sets, respectively. The remaining interactions were used as the training set. Then, we randomly sampled 99 items that had no interaction with this user and ranked the target item among the 100 items. The result for the top-K recommendations was measured by the same *Hit Ratio* (HR) and *Normalized Discounted Cumulative Gain* (NDCG) in Subsection 2.5.2.

Evaluated Methods. To measure the validity of the semantic information coming from the source data, we compared our method with the following state-of-the-art methods:

Single-domain recommendations (SDRs)

- **NeuCF** [57] jointly learns a neural network and a matrix factorization model.
- **LightGCN** [97] is a light graph convolutional network that enhances the user and item embeddings with the learned structural information from the user-item interaction graph.

Cross-domain recommendations (CDRs)

Table 3.3: Some important characteristics of evaluated methods.

Methods	SDR	CDR	Require User Map.	Address Neg. Trans.
NeuCF	✓			
LightGCN	✓			
s^2-Meta		✓		
RecSys-Dan		✓		
ESAM		✓		
CFAA		✓		
GCN		✓	✓	
BiTGCF		✓	✓	✓
SCDGN (ours)		✓		✓

- **s^2-Meta** [66] develops a meta-learning framework to generate individual models for different scenarios, where scenarios are denoted as domains. We represented users by the average of their interacted items to run s^2-Meta between domains with no domain-shared users and items.
- **RecSys-DAN** [51] trains a source user preference predicting model via the source domain data and then transfers the learned user preference patterns by aligning user preference patterns between source and target models.
- **ESAM** [15] adopts attribute correlation alignment to improve long-tail recommendation performance by suppressing inconsistent distribution between items from source and target domains.
- **CFAA** [96] proposes an embedding attribution alignment module to reduce the discrepancy of attribution distributions and relations between source and target domains.

For fair comparisons, we aligned the base model for all cross-domain methods with LightGCN, where this base model is equal to our SCDGN without the cross-domain user-cluster graph part. Besides, we replaced the randomly initialized item embedding

with our pre-trained semantic item embedding in Subsection 3.4.2 for all cross-domain comparisons and LightGCN, where these methods are identified with (wt). Table 3.3 summarizes some important characteristics of evaluated methods, including the methods in Subsection 3.5.3. From this table, we can see that only our SCDGN can effectively tackle the negative transfer issue without the need for user mapping.

Implementation Details. The codes of NeuCF[4], LightGCN[5], and s^2-Meta[6] were obtained from the corresponding GitHub repositories. Our SCDGB, ESAM, and CFAA were implemented by using PyTorch framework and can be found in a GitHub repository[7]. We used Adam to optimize the model parameters and speed up the training process with the mini-batch trick. For hyper-parameters, the learning rate was 0.001 for the recommendation tasks on private datasets and 0.01 for the cases on public datasets. The cluster number was 200. The embedding size of $\mathbf{e}_u$ was 32. The mini-batch size was 1024. The restriction loss balance factor λ_1 was set to 1, 0.001, and 0.0001 for the recommendation task on ML→AM, E-com→ADs, and ADs→E-com, respectively. λ_1 was set to 0.01 for the recommendation task on AM→ML, ML→AB, and AB→ML. The dimension reduction loss balance factor λ_2 was set to 1 for the recommendation task on private datasets, ML→AM, and ML→AB, where it was set to 10 for the recommendations on AM→ML and AB→ML. The weight of the regularization term λ_3 was set to 0.01 for the recommendation task on private datasets and 0.1 for the case on public datasets. The user-cluster graph convolutional layer number P was set to 2 for recommendations on public datasets and 1 for private datasets. For fair comparisons, we set the same user-item graph convolutional layer number Q as 3 for all comparisons except NeuCF. All these hyper-parameters were tuned on the validation set.

3.5.2 Performance Comparison (RQ1)

We report the average recommendation performances on the test set of each target domain. The comparison results are listed in Table 3.4. From this table, we found that the overall performances on private recommendation tasks (i.e., ADs→E-com and E-

[4]github.com/yihong-chen/neural-collaborative-filtering
[5]github.com/gusye1234/LightGCN-PyTorch
[6]github.com/THUDM/ScenarioMeta
[7]github.com/ZL6298/SCDGN

Table 3.4: Comparison between our proposal and state-of-the-art by using HR@K and NDCG@K. Performances ± 95% confidence intervals are reported. Bold shows the winner (the best one).

		ADs → E-com			E-com → ADs		
	Method	HR@1	HR@5	NDCG@5	HR@1	HR@5	NDCG@5
singe-domain RS	NeuCF	0.323±0.024	0.442 ± 0.025	0.348 ± 0.022	0.058 ± 0.008	0.136 ± 0.019	0.080 ± 0.010
	LightGCN	0.368±0.005	0.435 ± 0.005	0.384 ± 0.006	0.264 ± 0.005	0.352 ± 0.006	0.282 ± 0.005
	LightGCN (wt)	**0.383 ± 0.007**	0.466 ± 0.005	0.400 ± 0.008	**0.322 ± 0.007**	0.472 ± 0.006	**0.354 ± 0.006**
cross-domain RS	s^2-Meta (wt)	0.282 ± 0.032	0.372 ± 0.011	0.337 ± 0.020	0.033 ± 0.009	0.106 ± 0.022	0.067 ± 0.013
	RecSys-DAN (wt)	0.254 ± 0.016	0.352 ± 0.019	0.277 ± 0.016	0.043 ± 0.006	0.114 ± 0.010	0.065 ± 0.007
	ESAM (wt)	0.355 ± 0.020	0.459 ± 0.017	0.378 ± 0.019	0.200 ± 0.013	0.379 ± 0.014	0.248 ± 0.012
	CFAA (wt)	0.359 ± 0.018	0.485 ± 0.019	0.387 ± 0.020	0.128 ± 0.009	0.300 ± 0.015	0.181 ± 0.010
	SCDGN (ours)	0.380 ± 0.014	**0.496 ± 0.012**	**0.410 ± 0.012**	0.312 ± 0.006	**0.489 ± 0.005**	**0.354 ± 0.005**
		ML → AM			AM → ML		
	Method	HR@1	HR@5	NDCG@5	HR@1	HR@5	NDCG@5
singe-domain RS	NeuCF	0.075 ± 0.021	0.154 ± 0.013	0.097 ± 0.016	0.074 ± 0.022	0.198 ± 0.024	0.113 ± 0.018
	LightGCN	0.160 ± 0.003	0.218 ± 0.002	0.175 ± 0.002	0.139 ± 0.009	0.266 ± 0.011	0.174 ± 0.009
	LightGCN (wt)	0.168 ± 0.002	0.244 ± 0.003	0.189 ± 0.002	0.145 ± 0.014	0.300 ± 0.018	0.189 ± 0.013
cross-domain RS	s^2-Meta (wt)	0.059 ± 0.002	0.126 ± 0.001	0.094 ± 0.001	0.027 ± 0.001	0.103 ± 0.001	0.063 ± 0.001
	RecSys-DAN (wt)	0.108 ± 0.006	0.163 ± 0.007	0.124 ± 0.006	0.045 ± 0.008	0.111 ± 0.010	0.066 ± 0.008
	ESAM (wt)	0.095 ± 0.016	0.169 ± 0.010	0.115 ± 0.012	0.144 ± 0.017	0.301 ± 0.021	0.189 ± 0.017
	CFAA (wt)	0.111 ± 0.010	0.189 ± 0.008	0.133 ± 0.008	0.124 ± 0.015	0.274 ± 0.022	0.168 ± 0.016
	SCDGN (ours)	**0.181 ± 0.005**	**0.260 ± 0.006**	**0.200 ± 0.005**	**0.180 ± 0.012**	**0.356 ± 0.013**	**0.229 ± 0.011**
		ML → AB			AB → ML		
	Method	HR@1	HR@5	NDCG@5	HR@1	HR@5	NDCG@5
singe-domain RS	NeuCF	0.091 ± 0.011	0.186 ± 0.022	0.116 ± 0.013	0.074 ± 0.022	0.198 ± 0.024	0.113 ± 0.018
	LightGCN	0.175 ± 0.006	0.269 ± 0.005	0.197 ± 0.005	0.139 ± 0.009	0.266 ± 0.011	0.174 ± 0.009
	LightGCN (wt)	0.173 ± 0.007	0.278 ± 0.006	0.199 ± 0.006	0.145 ± 0.014	0.300 ± 0.018	0.189 ± 0.013
cross-domain RS	s^2-Meta (wt)	0.056 ± 0.001	0.170 ± 0.002	0.114 ± 0.001	0.025 ± 0.001	0.103 ± 0.001	0.063 ± 0.001
	RecSys-DAN (wt)	0.067 ± 0.005	0.136 ± 0.009	0.087 ± 0.007	0.043 ± 0.006	0.114 ± 0.010	0.065 ± 0.007
	ESAM (wt)	0.099 ± 0.019	0.212 ± 0.019	0.129 ± 0.015	0.138 ± 0.019	0.299 ± 0.022	0.185 ± 0.018
	CFAA (wt)	0.111 ± 0.016	0.228 ± 0.016	0.141 ± 0.014	0.132 ± 0.015	0.292 ± 0.024	0.179 ± 0.017
	SCDGN (ours)	**0.199 ± 0.011**	**0.321 ± 0.008**	**0.228 ± 0.010**	**0.181 ± 0.011**	**0.350 ± 0.013**	**0.227 ± 0.010**

com→ADs) are better than the ones on public recommendation tasks (i.e., ML→AM, AM→ML, ML→AB, and AB→ML). A reason raised this result can be that the behavior of purchasing products and browsing web pages in e-commerce and online advertisements inherently reflects user preferences, whereas users may not consistently rate movies or books even if they are interested in them. Due to this reason, it is reasonable to consider that predicting unknown ratings (i.e., the recommendation task in ML, AM, and AB) is more challenging than predicting purchases and browsing behaviors (i.e., the recommendation task in E-com and ADs). Besides, we can see that SCDGN outper-

forms other competitors on HR@5, NDCG@5, and HR@1 (in most cases). It achieves a remarkable improvement on four public recommendation tasks. This observation empirically demonstrates that our SCDGN effectively leverages the semantic information on the source domains to improve the recommendations in the target domains. For single-domain RSs, we find that LightGCN (wt) achieves the best performance, LightGCN the second best, and NeuCF the worst. This is because the target semantic information and the graph convolutional network yield a better performance. For cross-domain RSs, although they transfer interaction patterns or align embedding space from the source domain to the target domain, they perform worse than the single-domain method, i.e., LightGCN, in most cases. This result indicates that source-specific preferences cause the negative transfer issue and an inferior performance.

Besides, based on the results from two private datasets, we observed that SCDGN outperforms LightGCN (wt) in terms of HR@5 and NDCG@5 but lags behind LightGCN (wt) in terms of HR@1. This result indicates that SCDGN provides more true recommendations (i.e., interacted items in the test set) with the guide of source knowledge. However, the source knowledge sometimes misleads the recommendations in the top rank. This is because SCDGN balances the impact of interactions in source and target domains at the item-cluster level, In other words, it is hard for SCDGN to distinguish the impact of source interactions on individual target items.

3.5.3 Vs. CDRS with Domain-shared Users (RQ1)

To further investigate the effectiveness of the proposed method, we identified domain-shared users between AM and AB and conducted experiments to compare our SCDGN with CDRSs that require domain-shared users. Some basic information of the datasets used in this experiment is summarized in Table 3.5.

Evaluated Methods. We compared our method with the following state-of-the-art CDRSs:

- **CGN** [25] develops generative models for each domain to produce users' interacted itemset. Then, the domain knowledge is transferred via mapping the generated users' interacted itemset between domains.

Table 3.5: Basic information on the datasets with only the domain-shared users. #Int./U is the average number of interactions per user.

	Dataset	#Users	#Items	#Interactions	#Int./U
As	AM	1,315	5,458	15,169	11.54
Source	AB	722	2,894	6,485	8.99
As	AM	722	3,337	5,870	8.13
Target	AB	1,315	4,246	7,458	5.67

- **BiTGCF** [29] is a GNN-based CDRS that learns domain-specific feature propagation layers to alleviate the negative transfer issue. It transfers knowledge at an individual level by matching the domain-shared users from both domains.

Comparison Results. Table 3.6 shows the comparison results on HR@1 and HR@5. We observe that our SCDGN remarkably outperforms CGN and achieves a competitive performance with BiTGCF. CGN transfers users' interaction patterns between domains and neglects the domain bias in user preferences, i.e., domain-specific preferences. As a result, it yields a degraded performance. Both BiTGCF and SCDGN propose approaches to alleviating the impact of the bias and thus outperform CGN. In addition, it is worth mentioning that SCDGN involves no user assignment, suggesting that SCDGN has a broader application than CGN and BiTGCF.

3.5.4 Ablation Study (RQ2 & RQ3 & RQ4)

To study the impact of different components of SCDGN, we conducted ablation studies on ML→AM and ML→AB with some variants of SCDGN, including (1) w/o SI: SCDGN without user-cluster graph information, which is equal to LightGCN (wt), (2) w/o DRloss: SCDGN without the dimension reduction loss $\mathcal{L}_{dr}$, and (3) w/o DB: SCDGN without debiasing learning mechanism.

Table 3.7 shows HR@5 and NDCG@5 of SCDGN and its variants. From this table, we can see that all the information from the user-cluster graph, the metric-invariant dimension reduction, and the debiasing learning boost recommendation accuracy. Specif-

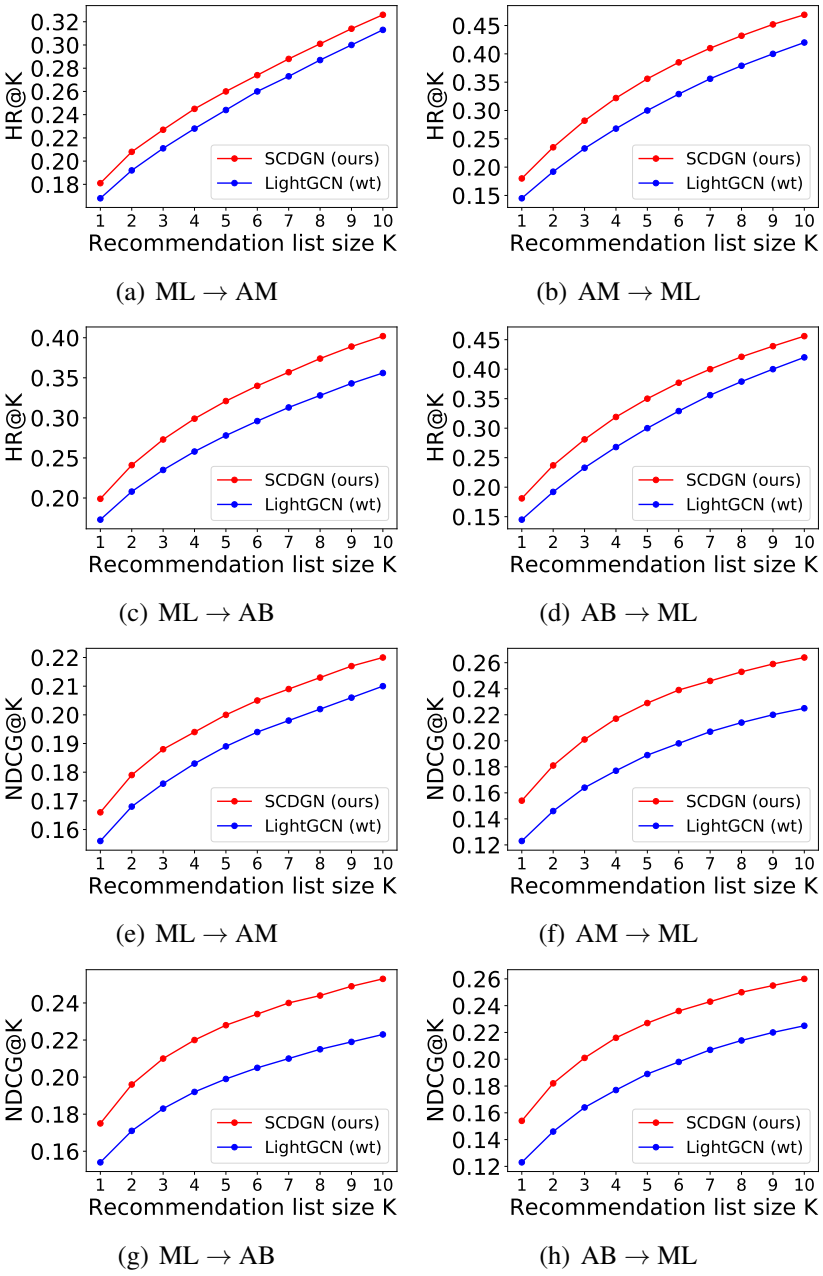
Table 3.6: Comparison between our proposal and CDRSs that require domain-shared users

Scenario	Method	HR@1	HR@5
AM→ AB	CGN	0.036 ± 0.001	0.131 ± 0.002
	BiTGCF	0.059 ± 0.002	**0.217 ± 0.002**
	SCDGN (ours)	**0.094 ± 0.003**	0.171 ± 0.003
AB→ AM	CGN	0.022 ± 0.001	0.173 ± 0.006
	BiTGCF	0.087 ± 0.003	**0.266 ± 0.003**
	SCDGN (ours)	**0.127 ± 0.002**	0.209 ± 0.003

Table 3.7: Performances of variants of SCDGN

Dataset	Method	HR@5	NDCG@5
ML → AM	w/o SI	0.244 ± 0.003	0.189 ± 0.002
	w/o DRloss	0.240 ± 0.003	0.186 ± 0.002
	w/o DB	0.229 ± 0.003	0.177 ± 0.002
	SCDGN	**0.260 ± 0.006**	**0.200 ± 0.005**
ML → AB	w/o SI	0.278 ± 0.006	0.199 ± 0.006
	w/o DRloss	0.314 ± 0.008	0.223 ± 0.010
	w/o DB	0.253 ± 0.006	0.183 ± 0.007
	SCDGN	**0.321 ± 0.008**	**0.228 ± 0.01**

ically, the results decrease the most without the proposed debiasing learning approach. This observation demonstrates that it is necessary to handle negative transfer issues even when transferring the semantic cluster-level interaction information. Besides, the decrement of results on w/o DRloss indicates the effectiveness of constraining the metric relationship when reducing dimension in a sparse domain.



(a) ML → AM (b) AM → ML

(c) ML → AB (d) AB → ML

(e) ML → AM (f) AM → ML

(g) ML → AB (h) AB → ML

Figure 3.2: Impact of K

3.5.5 Impact of Recommendation List Size (RQ5)

To investigate the impact of the recommendation list size K, we conducted experiments on public datasets by varying K. We used LightGCN (wt) as a competitor, as it is the

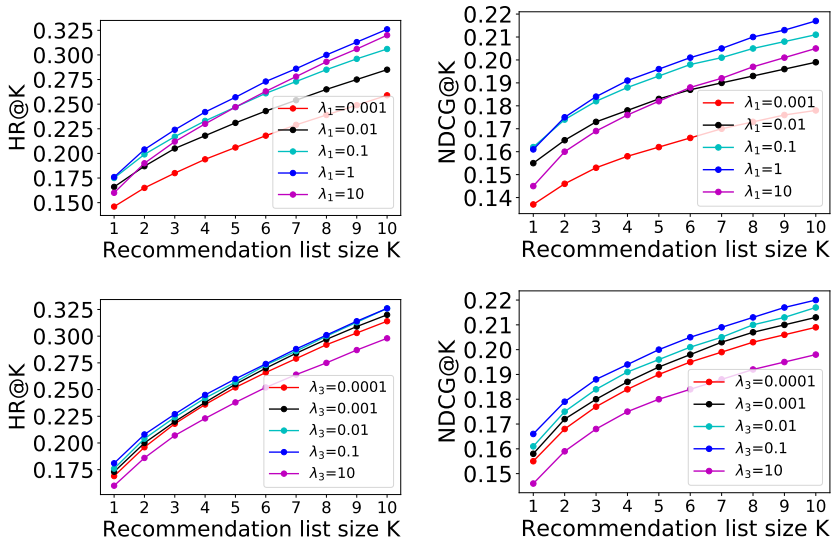


Figure 3.3: Impact of λ_1 and λ_3

best baseline.

Figure 3.2 shows the results on HR@K and NDCG@K. From these figures, we can see that SCDGN outperforms LightGCN (wt) consistently. This finding indicates that SCDGN successfully extracts unbiased structural knowledge from the cross-domain cluster-level graph, where this knowledge is effective in producing a better recommendation. For the cross-domain recommendations between ML and AB, our SCDGN achieves a greater improvement than LightGCN (wt) over both HR@K and NDCG@K. This result demonstrates that the debiasing learning mechanism in SCDGN facilitates cross-domain recommendations, especially for domains with different user behaviors.

3.5.6 Impact of Loss Balance Factors λ_1 and λ_3

In this part, we conducted experiments on ML→AM to discuss the impact of the hyper-parameter λ_1 and λ_3, where λ_1 and λ_3 are the factors to balance the restriction loss and the regularization term, respectively.

Figure 3.3 reports the results on HR@K and NDCG@K with varying λ_1 and λ_3. From this figure, we found that SCDGN achieves the best performance when $\lambda_1 = 1$

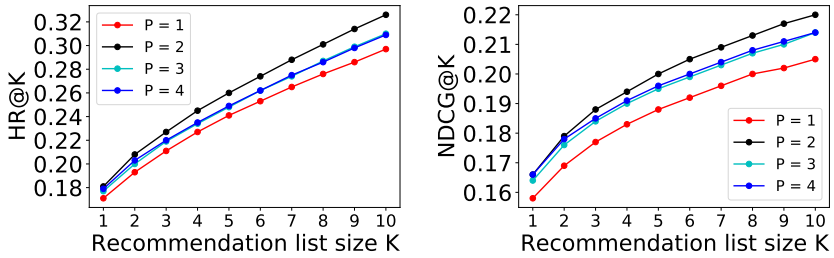


Figure 3.4: Impact of P

and $\lambda_3 = 0.001$. A small λ_1 produces an under-fitting issue when learning user and item debiasing vectors, resulting in an inferior performance. Inversely, a large λ_1 may introduce noise information from the source domain to mislead the user preference prediction of the target domain. Besides, a proper λ_3 is necessary to prevent the optimization of SCDGN from overfitting and under-fitting issues.

3.5.7 Impact of Hyper-parameter P

We conducted experiments on ML→AM to empirically investigate the impact of P, the number of the debiasing graph convolutional layer for the cross-domain user-cluster graph.

Figure 3.4 shows the results of SCDGN with varying P in a set $\{1, 2, 3, 4\}$. From this result, we can see that the performance reaches its peak when $P = 2$. This result indicates that a two-hop connected sub-graph can provide the best structural information to boost recommendations.

3.6 Conclusion

In this chapter, we proposed a novel semantic clustering enhanced debiasing graph neural recommender system (SCDGN) that effectively leverages public data to improve target recommendations. SCDGN jointly clusters source and target items based on their semantic features and makes the interaction information transferable between domains at an item-cluster level. By doing so, SDCGN avoids domain-shared users who are im-

possible to be identified between local and public domains. Besides, SCDGN addresses the negative transfer issue without merging personal interactions between the source and target domains. To achieve this, SCDGN constructs a cross-domain user-cluster graph and develops a new debiasing graph convolutional layer that adaptively weights source interactions at the item-cluster level. With this debiasing layer, SCDGN can alleviate the impact of source interactions that present users' source-specific preferences to handle the negative transfer issue. Furthermore, we developed a metric-invariant dimension reduction approach to alleviate overfitting caused by the sparse target data.

To verify the recommendation performance of SCDGN, we conducted extensive experiments on two public datasets and two private datasets. We compared our SCDGN with current state-of-the-art CDRSs under six cross-domain scenarios in which domain-shared users are not identified. The experimental result shows that our model outperforms all comparisons. Besides, to evaluate the effectiveness of SCDGN in addressing the negative transfer issue, we compared SCDGN with single-domain methods under the same six cross-domain scenarios. The comparison result demonstrates that SCDGN outperforms single-domain methods in four scenarios and achieves a comparable performance compared to single-domain methods in the other two scenarios. This indicates the effectiveness of SCDGN in handling the negative transfer.

SCDGN relies on textual features of items, i.e., titles and textual descriptions, to jointly cluster source and target items, which limits SCDGN from leveraging the public data without such features. Therefore, it is reasonable to extend SCDGN by designing methods that can cluster items based on other types of features, such as images of items. Moreover, the clustering process and the learning of the prediction model in SCDGN are separated. Therefore, an end-to-end cluster method that clusters items under the supervision of the prediction is demanded to further improve the performance of SCDGN.

Chapter 4

Semantic Relation Transfer for Privacy Preserved Cross-domain Recommendations

4.1 Introduction

The cold-start problem, i.e., a limited recommendation caused by training models with insufficient interaction data, is a general challenge in practical recommender systems [11, 12, 13]. Therefore, it is natural to leverage external data to improve recommendation performances. Cross-domain recommendation (CDR) is a promising solution to alleviate the cold-start problem. It utilizes sufficient data from an external (source) domain as prior knowledge to support the recommendation in the sparse target domain. In Chapter 3, we have proposed a CDRS that can effectively leverage public data to improve target recommendations. However, relying solely on public data is often inadequate, as such information tends to be incomplete, outdated, and deficient in quality and accuracy because of limited support. This raises practical demands for leveraging external commercial data by developing commercial operations.

4.1.1 Challenge

Different from the scenario of leveraging public data, merging interactions between domains, even in a cluster way, is prohibitive when leveraging external commercial data. This is because the source interactions, including the cluster-level interactions, disclose personal preferences without their consent to external service providers. Therefore, our proposed CDRS in Chapter 3, which merges cluster-level interactions between domains, is not available in the scenario of leveraging commercial data.

To make use of data from external services without merging interactions between domains, non-overlapped CDRSs were developed. These systems transfer distribution-level knowledge and hence remove the requirement of interaction mergence. Domain adversarial learning (DAL) and embedding attribution alignment (EAA) are two mainstream approaches to constructing these CDRSs [15, 51, 96]. DAL basically introduces a domain discriminator to extract domain-independent user and item embeddings [51]. By doing so, users' preferences are better modeled by utilizing both source and target interactions. EAA aligns the attribution of user and item embeddings between source and target domains to guide the target embedding learning with source knowledge [15, 49, 96]. Given the fact that user preferences vary from domain to domain, the ideal user and item embeddings that represent users' preferences should also meet the characteristic of domain-dependent [29]. DAL and EAA methods, hence, are easily misled by preferences that only appear in the source domain (i.e., domain-specific preferences) when learning target preferences. This produces a degraded performance on recommendations, which is known as the negative transfer issue.

Previous CDRSs and our CDRS in Chapter 3 respectively address the negative transfer issue based on merging individual interactions and cluster-level interactions between domains. Because the exposure of user behaviors across services, particularly across companies, is prohibitive for privacy concerns, user interactions cannot be merged when they are from different services. Therefore, existing CDRSs cannot handle the negative transfer issue in this scenario. This poses a challenge to develop a new method that can handle the negative transfer issue and does not merge interactions between domains. Furthermore, when interactions are not shared, the impact of source interactions must be implicitly measured, which makes dealing with the negative transfer issue harder.

Below summarize the challenges we address in this chapter.

- The first challenge is bridging the source and the target domains and transferring knowledge without merging interactions between the two domains.
- The second challenge is addressing the negative transfer in the scenario where source interactions cannot be merged to the target domain. Note that cluster-level interactions also cannot be merged here. This differentiates the negative transfer issue here and the one in Chapter 3.

4.1.2 Contribution

We propose a novel semantic relation-based knowledge transfer framework (shortly SRTrans) to address the above challenges. Recall that, previous CDRSs (including the one proposed in Chapter 3) handle the negative transfer issue by merging interactions between the source and the target domains. Different from these CDRSs, SRTrans bridges the domains by transferring relational knowledge through a similarity-based graph. Besides, SRTrans handles the negative transfer issue by focusing on only the items that have similar textual features, e.g., similar titles and similar descriptions, when transferring knowledge. In this sense, the knowledge is called relational knowledge. In this way, the impact of source items, which have irrelevant textual features and tend to present users' source-specific preferences, can be alleviated. To achieve the above knowledge transfer, SRTrans introduces a new two-tier graph transfer framework. Besides, SRTrans combines a prediction loss with a new task-oriented knowledge distillation supervision. With this combined supervision, the domain-shared preferences can be adaptively distilled from the source knowledge transferred by the two-tier graph transfer network. In summary, our contributions are three-fold:

1 We propose a semantic relation-based graph transfer framework, namely SRTrans, for privacy preserved non-overlapped CDRSs. SRTrans requires no mergence of user interactions between domains and can handle the negative transfer issue.

2 To mitigate the negative impact of source-domain-specific preferences, we introduce a new task-oriented knowledge distillation supervision and combine it with

a prediction loss. This approach improves the final performance of SRTrans.

3 We conduct extensive experiments on real-world datasets, and the results demonstrate that SRTrans outperforms the state-of-the-art methods.

Section 4.2 gives a brief overview of related works. Section 4.3 formulates the problem studied in this chapter. Section 4.4 presents our proposed method, and experimental results are illustrated in Section 4.5. Finally, this chapter is concluded in Section 4.6.

4.2 Related Work

4.2.1 User-overlapped CDR

Most CDRSs bridge source and target domains through domain-shared users. These systems transfer individual-level knowledge learned in the source domain to the target domain. For example, BiTGCF [29] and GA-DTCDR [28] respectively introduced a graph neural network and an attention mechanism as a cross-domain feature transfer layer to fuse domain-shared users' source and target latent features. However, the assumption of domain-shared users limits their applications.

Another research direction is to learn mappings between source and target user embeddings. EMCDR [115] proposed a multi-layer perceptron to map source user embeddings to target embeddings. PTUPCDR [100] further improved EMCDR by learning personalized meta-transfer mappings.

4.2.2 Non-overlapped CDR

To remove the limitation of domain-shared users, non-overlapped CDRSs were developed, and domain adversarial learning (DAL) for non-overlapped CDRSs was introduced. For example, RecSys-Dan [51] devised a discriminator and minimized the divergence of the predictions between source and target domains. Some studies proposed embedding attribution alignment (EAA), which aligns embedding attributions between source and target domains. MMT-Net [49] developed a contextual CDRS and regularized the contextual-jointed target user and item embedding learning with learned source

embedding distributions. ESAM [15] and CFAA [96] removed the requirement for domain-shared contextual features and aligned the attribution distribution and correlation between source and target domains.

However, existing DAL and EAA approaches ignore the relations between source and target and leverage all source data to transfer knowledge, resulting in an impaired transfer or even a negative transfer. These approaches, moreover, learn from both domain-shared and domain-specific preferences and yield sub-optimal performances. The above drawbacks motivate us to propose SRTrans, which extracts relational knowledge and alleviates the negative impact of the domain-specific preferences in the source domain.

4.3 Problem Formulation

We first formulate the cross-domain recommendation task as the top-K recommendation in a sparse target domain $\mathcal{D}_t$, under the assumption of the existence of an auxiliary dense domain $\mathcal{D}_s$ that is considered to be the source. Let $\mathcal{U}_t$ and $\mathcal{V}_t$ denote sets of users and items in $\mathcal{D}_t$, respectively. The interaction set between $\mathcal{U}_t$ and $\mathcal{V}_t$ is denoted as $\mathcal{R}_t = \{(u, v)|u \in \mathcal{U}_t, v \in \mathcal{V}_t\}$. Analogously, we denote the user set, item set, and interaction set in $\mathcal{D}_s$ as $\mathcal{U}_s$, $\mathcal{V}_s$, and $\mathcal{R}_s$, respectively. It is important to note that there is no overlap of user and item between $\mathcal{D}_s$ and $\mathcal{D}_t$. Given a user $u \in \mathcal{U}_t$, the top-K recommendations aim to predict a preference score $\hat{y}_{u,v}$ for each item $v \in \bar{\mathcal{V}}_u = \{v|v \in \mathcal{V}_t, v \notin \mathcal{V}_u\}$, where $\mathcal{V}_u$ is the set of items that interacted with u. The preference score $\hat{y}_{u,v}$ is defined as $\hat{y}_{u,v} = f(\mathbf{e}_u, \mathbf{e}_v)$, where $f(\cdot)$ is a user preference estimator, $\mathbf{e}_u$ is the user embedding, and $\mathbf{e}_v$ is the item embedding. The K items with the largest scores are recommended to u.

4.4 Proposed Method

4.4.1 Overview

Motivated by the previous non-overlapped CDRSs that ignore the relations among interactions and are easily misled by the source-domain-specific preferences, we propose

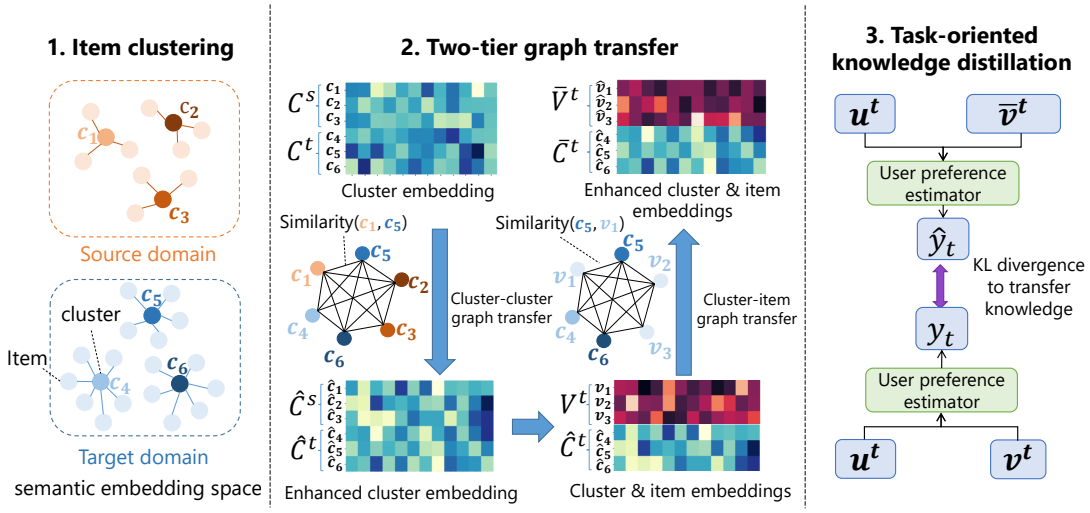


Figure 4.1: Overview of our proposed SRTrans

a novel semantic relation-based graph transfer framework (SRTrans) that extracts and transfers cluster-based relational knowledge while alleviating the misleading source-domain-specific preferences. SRTrans is depicted in Figure 4.1. The numbers below correspond to the ones in Figure 4.1.

(1) We first semantically cluster items and calculate cluster embeddings through an adaptive cluster approach. (2) Then, we construct a cluster-cluster cross-domain graph based on the similarities between the above item clusters to transfer knowledge from source clusters to target clusters, which avoids merging interactions between domains. We also propose a novel two-tier graph transfer network that transfers relational knowledge on the cross-domain graph by focusing on the source clusters that have larger similarity scores. By doing so, SRTrans can alleviate the impact of irrelevant source clusters to handle the negative transfer issue. (3) After that, we design a new task-oriented knowledge distillation supervision that can be combined with a prediction loss to avoid learning the source-domain-specific preferences.

4.4.2 Adaptive Semantic Item Cluster

To learn semantic item clusters, we first compute the semantic embeddings of items. Given the item texts, i.e., item descriptions, we apply the semantic token embeddings

from a pre-trained BERT model [111] to represent the tokens in the item texts. Let T_v denote the text of an item v, and the semantic embedding of v is the *tf-idf* weighted aggregation of token embeddings:

$$\mathbf{v}_{txt} = \sum_{w \in T_v} tf\text{-}idf(w) \cdot \mathrm{BERT}(w), \tag{4.1}$$

where $\mathrm{BERT}(w)$ and $tf\text{-}idf(w)$ are respectively the embedding and the *tf-idf* score for token $w \in T_v$. The *tf-idf* score is calculated with a combination of the source and target item text corpora. By doing so, we can focus on important tokens and alleviate the negative impact of noisy tokens. The dimension of the BERT semantic embeddings (768) is so high and may lead to tremendous computing costs and overfitting issues in our sparse target domain. We hence assign a dense layer ϕ_{emb} with output dimension $d \ll 768$ to encode $\mathbf{v}_{txt}$. The encoded item embedding $\mathbf{v}$ is given by

$$\mathbf{v} = \phi_{emb}(\mathbf{v}_{txt}). \tag{4.2}$$

After that, we calculate semantic embeddings for all source and target items, which are denoted as $\mathbf{V}^s$ and $\mathbf{V}^t$, respectively.

To get semantic item clusters that can facilitate the final recommendation task, we borrow the idea of DE-RRD [116]. Formally, given the cluster number $\mathcal{N}_c$, we calculate the cluster assignment probability vector $\mathbf{p} \in \mathbb{R}^{\mathcal{N}_c}$ with a small network ϕ_{ca}, where the vector of an item v is given by

$$\mathbf{p}_v = \phi_{ca}(\mathbf{v}). \tag{4.3}$$

Each element $p_{v,j}$ of $\mathbf{p}_v$ represents the probability that the item v is assigned to the cluster j. With the help of $\mathbf{p}_v$, we assign a binary gradient vector $\mathbf{m}_v \in \mathbb{R}^{\mathcal{N}_c}$ to indicate the cluster of item v, where the element j of $\mathbf{m}_v$ is given by

$$m_{v,j} \sim \mathrm{Bern}\left(\frac{\exp(p_{v,j})}{\sum_{k \in \mathcal{N}_c} \exp(p_{v,k})}\right), \tag{4.4}$$

where $\mathrm{Bern}(\cdot)$ represents the Bernoulli distribution. We use the Gumbel-Softmax reparameterization trick to differentiate through the Bernoulli sampling process:

$$m_{v,j} = \frac{\exp((p_{v,j}) + g_j/\tau)}{\sum_{k \in \mathcal{N}_c} \exp((p_{v,k} + g_k)/\tau)}, \quad \mathbf{g} \sim \mathrm{Gumbel}(0, \mathbf{1}) \tag{4.5}$$

where $\mathbf{g} \in \mathbb{R}^{\mathcal{N}_c}$ is the Gumbel noise drawn from $\mathrm{Gumbel}(0, \mathbf{1})$ distribution. Note that g_j is the j-th element of $\mathbf{g}$ and τ is the temperature parameter. The sampling process is separated from $\mathbf{p}_v$, so the cluster assigning network ϕ_{ca} can be end-to-end updated through backpropagations. Let $\mathbf{M^s} \in \mathbb{R}^{|\mathcal{V}^s| \times \mathcal{N}_c}$ ($\mathbf{M^t} \in \mathbb{R}^{|\mathcal{V}^t| \times \mathcal{N}_c}$) denote the binary gradient matrix for the source (target) domain, where each row in $\mathbf{M^s}$ ($\mathbf{M^t}$) indicates the cluster of a source (target) item and can be computed by Equations (4.3) and (4.5).

We next declare source cluster embeddings $\mathbf{C}^s$ and target cluster embeddings $\mathbf{C}^t$ as the average semantic embeddings of the items assigned to the corresponding cluster:

$$\mathbf{C}^s = (\tilde{\mathbf{M}}^s)^T \mathbf{V}^s \quad \text{and} \quad \mathbf{C}^t = (\tilde{\mathbf{M}}^t)^T \mathbf{V}^t, \tag{4.6}$$

where $\tilde{\mathbf{M}}^s$ and $\tilde{\mathbf{M}}^t$ respectively are assignment matrices normalized by the number of items in each cluster, i.e., $\tilde{\mathbf{M}}^s_{[:,i]} = \mathbf{M}^s_{[:,i]} / \sum_i \mathbf{M}^s_{[:,i]}$.

4.4.3 Two-tier Graph Transfer

We propose a novel two-tier graph transfer network to extract and transfer cluster-based relational knowledge from the source to the target. This network consists of a cluster-level relational graph transfer and a cluster-item relational graph transfer. The former transfers knowledge from the source to the target clusters by constructing a relational graph whose adjacency matrix is denoted as the similarities between the source and target clusters. Analogously, the latter transfers knowledge from the target clusters to the target items by building a graph whose adjacency matrix is denoted as the similarities between the target clusters and items. With these similarity-based adjacency matrices, the graph transfer network can transfer the most relevant source knowledge from source clusters into the target items and thus alleviate the negative impact of irrelevant noises. Therefore, we can remove the drawbacks of the state-of-the-art methods (transferring all source knowledge, including domain-specific knowledge).

Cluster-level Relational Graph Transfer. To extract and transfer cluster-level relational knowledge from the source to the target, we first define a cluster-based relational graph $\mathcal{G}_c$ to represent the relations between clusters from different domains. Let $\mathbf{C} = (\mathbf{C}^s; \mathbf{C}^t) \in \mathbb{R}^{(2\mathcal{N}_c) \times d}$ denote the fused source and target cluster representations and serve as vertices in graph $\mathcal{G}_c$. We further define the edges and the corresponding

edge weights. The edge weight $\mathcal{S}(\mathbf{c}_i, \mathbf{c}_j)$ between clusters $\mathbf{c}_i$ and $\mathbf{c}_j$ is measured by the cosine similarity with softmax and is formulated by:

$$\mathcal{S}(\mathbf{c}_i, \mathbf{c}_j) = \text{softmax}(\frac{\mathbf{c}_i \cdot \mathbf{c}_j}{\|\mathbf{c}_i\|\|\mathbf{c}_j\|}). \tag{4.7}$$

For simplicity, we denote the cluster-based relational graph as $\mathcal{G}_c = (\mathbf{C}, \mathcal{A})$, where $\mathbf{C} = \{\mathbf{c}_i | i \in [1, 2\mathcal{N}_c]\} \in \mathbb{R}^{2\mathcal{N}_c \times d}$ represents the set of vertices and each vertex corresponds to a cluster, whereas $\mathcal{A} = \{\mathcal{S}(\mathbf{c}_i, \mathbf{c}_j) | \mathbf{c}_i \in \mathbf{C}, \mathbf{c}_j \in \mathbf{C}\}$ is the adjacency matrix, which indicates the relations between clusters. The adjacency matrix $\mathcal{A}$ also can be written by:

$$\mathcal{A} = \{\mathcal{S}(\mathbf{C}^s, \mathbf{C}^s), \mathcal{S}(\mathbf{C}^s, \mathbf{C}^t); \mathcal{S}^T(\mathbf{C}^s, \mathbf{C}^t), \mathcal{S}(\mathbf{C}^t, \mathbf{C}^t)\}. \tag{4.8}$$

From this equation, we can see that $\mathcal{A}$ contains cluster relations within and across domains.

After constructing the cluster-based relational graph $\mathcal{G}_c$, we use the graph convolutional network (GCN) [117] to transfer relational knowledge from source clusters $\mathbf{C}^s$ to target clusters $\mathbf{C}^t$, where the GCN is formulated as:

$$\mathbf{C}^{(l+1)} = \text{ReLU}\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{C}^{(l)}\mathbf{W}_c^{(l)}\right). \tag{4.9}$$

Here $\tilde{\mathcal{A}} = \mathcal{A} + \mathbf{I}$, where $\mathbf{I}$ is the identity matrix, corresponds to adding self-loops to the graph. Also, $\tilde{\mathbf{D}}$ is the degree matrix with elements $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, and $\mathbf{W}_c^{(l)}$ is a trainable weight matrix for layer l. The input $\mathbf{C}^{(0)} = \mathbf{C}$. After aggregating L_c GCN layers, we get the relational enhanced target cluster representations as the last $\mathcal{N}_c$ rows of $\mathbf{C}^{(L_c)}$, which is denoted as $\mathbf{C}^{t(L_c)} = \{\mathbf{c}_i^{t(L_c)} | i \in [1, \mathcal{N}_c]\}$. By fusing and aggregating source and target clusters together, the enhanced target clusters can refine relational knowledge within and across domains.

Cluster-item Relational Graph Transfer. After calculating the enhanced target clusters $\mathbf{C}^{t(L_c)}$, we further construct a cluster-item relational graph $\mathcal{G}_v$ to measure the relations between enhanced target clusters $\mathbf{C}^{t(L_c)}$ and the target items $\mathbf{V}^t$. With $\mathcal{G}_v$ and cluster-item relations, the useful knowledge from the most relevant $\mathbf{C}^{t(L_c)}$ is aggregated into the target items, while the irrelevant noise is alleviated. To fuse knowledge from both target clusters and items, we concatenate $\mathbf{C}^{t(L_c)}$ and $\mathbf{V}^t$ to serve as vertices in the graph $\mathcal{G}_v$, where the vertex set is denoted as $\mathbf{H} = (\mathbf{C}^{t(L_c)}; \mathbf{V}^t) \in \mathbb{R}^{(\mathcal{N}_c + |\mathcal{V}_t|) \times d}$. Analogously, the edge weight $\mathcal{S}(\mathbf{h}_i, \mathbf{h}_j)$ between vertex $\mathbf{h}_i$ and $\mathbf{h}_j$ is defined as the cosine

similarity with softmax:

$$\mathcal{S}(\mathbf{h}_i, \mathbf{h}_j) = \mathrm{softmax}(\frac{\mathbf{h}_i \cdot \mathbf{h}_j}{\|\mathbf{h}_i\|\|\mathbf{h}_j\|}). \tag{4.10}$$

The cluster-item relational graph $\mathcal{G}_v$ is denoted as $\mathcal{G}_v = (\mathbf{H}, \mathcal{B})$, where the vertex set and the adjacency matrix are $\mathbf{H} = \{\mathbf{h}_i | i \in [1, \mathcal{N}_c + |\mathcal{V}_t|]\}$ and $\mathcal{B} = \{\mathcal{S}(\mathbf{h}_i, \mathbf{h}_j) | \mathbf{h}_i \in \mathbf{H}, \mathbf{h}_j \in \mathbf{H}\}$, respectively.

By leveraging $\mathcal{G}_v$, we next transfer relational knowledge via a GCN, which is formulated as:

$$\mathbf{H}^{(l+1)} = \mathrm{ReLU}\left(\tilde{\mathbf{D}}_v^{-\frac{1}{2}} \tilde{\mathcal{B}} \tilde{\mathbf{D}}_v^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}_v^{(l)}\right). \tag{4.11}$$

Here, $\tilde{\mathcal{B}}$ is the adjacency matrix with self loops, and $\tilde{\mathbf{D}}_v$ is the degree matrix on $\tilde{\mathcal{B}}$. $\mathbf{W}_c^{(l)}$ is a trainable weight matrix for layer l. Also, we define $\mathbf{H}^{(0)} = \mathbf{H}$. After aggregating L_v GCN layers, we get the knowledge-enhanced target item embeddings $\mathbf{V}^{t(L_v)}$ as the last $|\mathcal{V}_t|$ rows of $\mathbf{H}_c^{(L_v)}$.

4.4.4 Task-oriented Knowledge Distillation

Existing non-overlapped CDRSs [15, 51, 96] are easily misled by domain-specific preferences, because the source and target embedding space is directly aligned. Motivated by this finding, we combine a knowledge distillation with the target prediction by introducing a new task-oriented knowledge distillation supervision. In this way, the target prediction loss can supervise the knowledge distillation to learn the domain-*share* preferences. As a result, the negative transfer issues incurred by the misleading source-domain-specific preferences can be alleviated.

Formally, we first define the user embeddings of source and target domains as trainable parameter matrices $\mathbf{U}^s$ and $\mathbf{U}^t$, respectively. Each row in $\mathbf{U}^s$ ($\mathbf{U}^t$) corresponds to a source (target) user and can be retrieved with the user ID. The prediction score of the target user u^t for the target item v^t before and after knowledge enhancement in Subsection 4.4.3 are then given by:

$$\hat{y}_{u,v}^t = f(\mathbf{u}^t, \mathbf{v}^t) \quad \text{and} \quad \tilde{y}_{u,v}^t = f(\mathbf{u}^t, \mathbf{v}^{t(L_v)}), \tag{4.12}$$

where $\mathbf{v}^t$ and $\mathbf{v}^{t(L_v)}$ are respectively the item embedding before and after knowledge enhancement. For a user preference estimator $f(\cdot)$, we adopt inner product [113] and LGC [97] to evaluate the performance of our SRTrans on different models.

After that, the task-oriented knowledge distillation is defined as the KL-divergence between $\hat{y}^t_{u,v}$ and $\tilde{y}^t_{u,v}$, which is formulated by:

$$\mathcal{L}_{KD}(\hat{y}^t|\tilde{y}^t) = \sum_{(u,v)\in\mathcal{R}^t} \left(\hat{y}^t_{u,v} \log \frac{\hat{y}^t_{u,v}}{\tilde{y}^t_{u,v}} + (1-\hat{y}^t_{u,v}) \log \frac{(1-\hat{y}^t_{u,v})}{(1-\tilde{y}^t_{u,v})} \right). \tag{4.13}$$

We use the pair-wise BPR loss [97] to measure the loss of predictions. To achieve this, we randomly sample a negative item for each source and each target interaction. Taking the target domain as an example, a new interaction $r^t \in \mathcal{R}_t$ is a triplet $r^t = (u, v, v')$, where $u \in \mathcal{U}_t$, $v \in \mathcal{V}_t$, and $v' \in \bar{\mathcal{V}}_u$. Then, the pair-wise BPR loss is given by

$$\mathcal{L}_{BPR} = -\left(\sum_{(u,v,v')\in\mathcal{R}_t} \ln \sigma\left(\hat{y}_{uv} - \hat{y}_{uv'}\right) + \sum_{(u,v,v')\in\mathcal{R}_s} \ln \sigma\left(\hat{y}_{uv} - \hat{y}_{uv'}\right) \right). \tag{4.14}$$

The total loss is measured by combining the knowledge distillation loss $\mathcal{L}_{KD}$ and the prediction loss $\mathcal{L}_{BPR}$, that is

$$\mathcal{L} = \mathcal{L}_{BPR} + \lambda \mathcal{L}_{KD}, \tag{4.15}$$

where λ is a hyper-parameter used to balance the weights of different losses.

By combining $\mathcal{L}_{KD}$ and $\mathcal{L}_{BPR}$, the negative impact from the source-domain-specific preferences can be alleviated under the supervision of the prediction loss.

4.5 Experiment

4.5.1 Experiment Setting

Datasets. In this chapter, we used the same four datasets in Subsection 3.5.1, including two public datasets, i.e., MovieLens25M (ML) and AmazonBook (AB), and two private datasets, i.e., the online advertisement dataset (AD) and the e-commerce dataset (E-com). Then, we can evaluate the recommendation performance of SRTrans in the scenario of leveraging external commercial data on these datasets which is from different services. The recommendation performance of SRTrans in the scenario of leveraging external commercial data can be measured by them.

Table 4.1: Basic information on the datasets we used. #Int./U is the average number of interactions per user.

	Dataset	#Users	#Items	#Interactions	#Int./U
As source	ML	18,232	14,435	421,803	23.14
	AB	27,662	12,708	129,899	4.70
	AD	18,829	12,253	360,880	19.17
	E-com	17,418	6,142	81,499	4.68
As target	ML	6,298	9,873	31,445	4.99
	AB	13,350	10,477	61,004	4.57
	AD	11,010	12,031	55,050	5.00
	E-com	12,558	5,118	46,871	3.73

From the CDR scenarios in Subsection 3.5.1, we utilized those containing domains from different services, including: (1) AD→E-com and E-com→AD; (2) ML→AB and AB→ML. Also, we adopted the same pre-processing in Subsection 3.5.1 for all the source and the target domains. Some basic information on the pre-processed dataset is shown in Table 4.1

Evaluation Criteria. We evaluated the recommendations by leveraging the same evaluation criteria in Subsection 3.5.1 on each user in target domains.

Baseline. We compared SRTrans with the following state-of-the-art non-overlapped cross-domain recommendation models:

- **BASE** is the base model trained with target domain data.
- **RecSys-DAN** [51] jointly trains source and target user preference prediction model and introduces a domain discriminator to extract domain-invariant user preferences. In this chapter, we added the target prediction task for a fair comparison.
- **ESAM** [15] develops an attribute correlation alignment method to improve long-tail recommendations by suppressing inconsistent distribution between items from the source and target domains.

Table 4.2: Some important characteristics of evaluated methods.

Methods	SDR	CDR	Address	Neg. Trans.
BPR-MF	✓			
LightGCN	✓			
RecSys-Dan		✓		
ESAM		✓		
CFAA		✓		
SRTrans (ours)		✓	✓	

- **CFAA** [96] proposes an embedding attribution alignment module to reduce the discrepancy of attribution distributions and relations between source and target domains.

Besides, considering all baselines and our SRTrans are model-agnostic frameworks, we adopted two following base models for all comparisons:

- **BPR-MF** [113] combines the matrix factorization method with a binary personalized ranking loss.
- **LightGCN** [97] designs a light graph convolutional network to learn the structural information and alleviate the overfitting issue in traditional GNN-based recommender systems.

For a fair comparison, the embedding module of the baselines was replaced with our semantic encoded item embedding that can be calculated by Equation 4.2. Some important characteristics of evaluated methods are listed in Table 4.2. From this table, we can see that only our SRTrans can effectively tackle the negative transfer issue among non-overlapping cross-domain methods.

Implementation Details. We adopted the same L2 penalty and mini-batch trick for the evaluated methods and set them to 0.01 and 2048, respectively. The GCN layer for the base model LightGCN was set to 3. The embedding dimension d was 32 and 16 for public and private datasets, respectively. For SRTrans, the temperature parameter τ was

0.0001 and the weight of the knowledge distillation loss λ was 0.1. The number of graph transfer layers L_c and L_v was 1. The number of clusters $\mathcal{N}_c$ was 32. We implemented MF- and LightGCN-based SRTrans, RecSys-DAN, ESAM, and CFAA with PyTorch framework, using the Adam optimizer where the learning rate was set to 0.01. The above hyper-parameters were fine-tuned according to the performance on the validation set.

4.5.2 Comparison Results

Table 4.3 shows the comparison results. We find that: (1) SRTrans outperforms the baselines w.r.t. HR@5 and NDCG@5 in most cases, especially when the base model is LightGCN. This is because SRTrans transfers relational knowledge into the individual item, and GNN-based models can further fuse knowledge in these items by structurally aggregating them. (2) Although cross-domain baselines transfer user preferences (RecSys-DAN) or align embedding spaces (ESAM and CFAA) from the source domain to the target domain, they often perform worse than the single-domain method (BASE). This result indicates that domain-specific preferences and noisy source data incur negative transfer issues. (3) SRTrans achieves comparable performance or outperforms the single-domain model (BASE) in most scenarios. This observation confirms that SRTrans remarkably alleviates the negative transfer issue.

4.5.3 Visualization

To better show the knowledge transfer process and explain the learned cluster relations, we visualize the cluster-based relational graph by sampling a mini-batch of interactions from the training data. The result of ML→AB is shown in Figure 4.2 and Figure 4.3.

Figure 4.2 depicts a heatmap of an adjacency matrix that describes the relation between the source and target clusters. The colors indicate cosine similarities between clusters. From this figure, we can find that highly related source and target clusters (indicated by light colors) and irrelevant ones (indicated by dark colors) are identified.

Figure 4.3 gives a sub-graph constructed by using 10 most related source-target cluster pairs. This figure shows how the knowledge in source clusters is transferred to target clusters. For example, target cluster 23 receives more knowledge from its highly

Table 4.3: Comparison between our proposal and state-of-the-art. Performances ± 95% confidence intervals are reported. Bold shows the winner.

		AD→E-com		E-com→AD	
	Method	HR@1	NDCG@5	HR@1	NDCG@5
MF	BASE	0.224±0.033	0.310±0.014	0.180±0.008	0.299±0.006
	RecSys-DAN	0.269±0.025	0.356±0.015	0.046±0.010	0.102±0.014
	ESAM	**0.283±0.035**	**0.365±0.016**	0.125±0.012	0.239±0.013
	CFAA	0.251±0.036	0.336±0.014	0.140±0.008	0.257±0.007
	SRTrans (ours)	0.243±0.028	0.331±0.011	**0.187±0.005**	**0.307±0.004**
LightGCN	BASE	0.297±0.012	0.372±0.009	0.190±0.008	**0.315±0.008**
	RecSys-DAN	0.254±0.016	0.277±0.016	0.043±0.006	0.065±0.007
	ESAM	0.216±0.033	0.314±0.023	0.114±0.010	0.221±0.012
	CFAA	0.282±0.035	0.375±0.021	0.042±0.004	0.114±0.007
	SRTrans (ours)	**0.312±0.011**	**0.384±0.009**	**0.191±0.010**	0.306±0.009
		ML→AB		AB→ML	
	Method	HR@1	NDCG@5	HR@1	NDCG@5
MF	BASE	0.072±0.003	0.136±0.004	0.031±0.002	0.076±0.002
	RecSys-DAN	0.021±0.006	0.048±0.007	0.005±0.001	0.019±0.002
	ESAM	0.034±0.006	0.079±0.008	**0.032±0.003**	**0.083±0.004**
	CFAA	0.055±0.004	0.109±0.006	0.029±0.003	0.076±0.004
	SRTrans (ours)	**0.075±0.005**	**0.142±0.005**	0.030±0.002	0.075±0.003
LightGCN	BASE	0.122±0.008	0.194±0.006	0.085±0.007	0.165±0.008
	RecSys-DAN	0.042±0.009	0.084±0.009	0.013±0.001	0.041±0.004
	ESAM	0.046±0.010	0.119±0.011	0.070±0.011	0.166±0.018
	CFAA	0.049±0.010	0.121±0.012	0.016±0.001	0.049±0.002
	SRTrans (ours)	**0.129±0.011**	**0.204±0.009**	**0.107±0.008**	**0.202±0.009**

relevant source clusters 13, 30, 7, and 14, compared to other irrelevant clusters. This example highlights the mechanism of alleviating negative transfer issues.

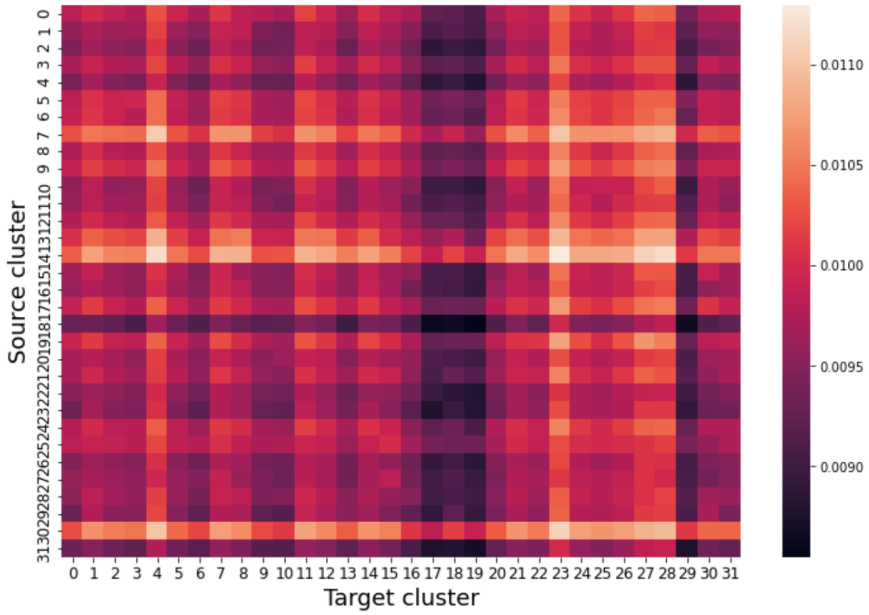


Figure 4.2: Heatmap of cosine similarities between source clusters and target clusters, where the visualization results are calculated by sampling a mini-batch of data from the ML $\rightarrow$ AB scenario

4.5.4 Ablation Study

To study how each module of SRTrans contributes to the final performance, we compared SRTrans with its several variants, namely (1) w/o SI, which replaces semantic item embeddings with randomly initialized ones, (2) w/o AC, which removes the adaptive cluster module and directly calculates similarities between single source and target items to build a knowledge transfer graph, and (3) w/o KD, which is SRTrans without knowledge distillation and is equal to the single-domain BASE model. Table 4.4 reports the result.

We see that w/o SI has the lowest performance. This result indicates that the semantic features are essential to extracting the relational knowledge. Moreover, w/o AC shows comparable performances to SRTrans in some cases, suggesting that transferring item-based relational knowledge can also alleviate the performance degradation caused by source noise data.

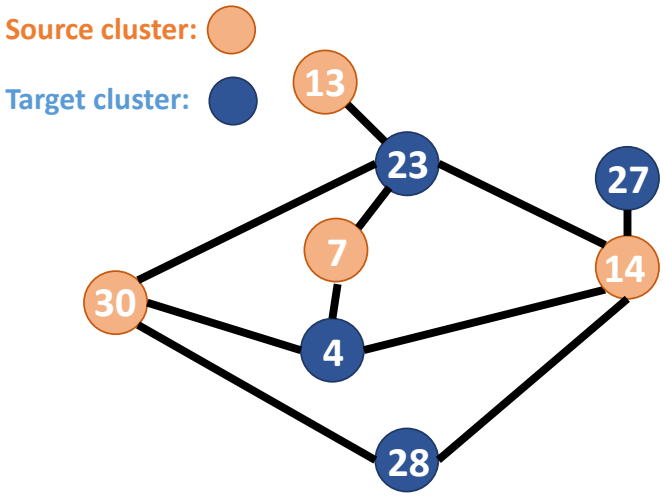


Figure 4.3: Relation graph between source clusters and target clusters, where edges in this graph identify the 10 cluster pairs that come from different domains and have the biggest cosine similarities. The visualization results are calculated by sampling a mini-batch of data from the ML → AB scenario

Table 4.4: Performances of variants of SRTrans

	AD→E-com		E-com→AD	
Method	HR@1	NDCG@5	HR@1	NDCG@5
w/o SI	0.292 ± 0.008	0.351 ± 0.006	0.090 ± 0.005	0.140 ± 0.005
w/o AC	0.310 ± 0.012	0.376 ± 0.009	**0.194 ± 0.009**	0.310 ± 0.008
w/o KD	0.297 ± 0.012	0.372 ± 0.009	0.190 ± 0.008	**0.315 ± 0.008**
SRTrans	**0.312 ± 0.011**	**0.384 ± 0.009**	0.191 ± 0.01	0.306 ± 0.009
	ML→AB		AB→ML	
w/o SI	0.101 ± 0.008	0.159 ± 0.007	0.060±0.004	0.115±0.005
w/o AC	**0.130 ± 0.011**	**0.207 ± 0.009**	0.097±0.007	0.188±0.009
w/o KD	0.122 ± 0.008	0.194 ± 0.006	0.085±0.007	0.165±0.008
SRTrans	0.128 ± 0.011	0.201 ± 0.009	**0.107 ± 0.008**	**0.202 ± 0.009**

4.6 Conclusion

In this chapter, we studied improving recommendations with external commercial data for services with sparse interaction data. Employing commercial data from other com-

panies requires no mergence of interactions between domains as well as solving a negative transfer issue raised by users' source-specific preferences. Therefore, we proposed a novel semantic relation-based knowledge transfer framework (SRTrans) that does not merge interactions between domains and can effectively leverage external commercial data. To avoid merging interactions, SRTrans transfers knowledge between domains by constructing a similarity-based cross-domain graph. To address the negative transfer issue, SRTrans transfers knowledge by focusing on the source item clusters that are similar to target clusters. Moreover, SRTrans combines a task-oriented knowledge distillation loss with a prediction loss to alleviate the impact of source-specific preferences.

To evaluate the effectiveness of SRTrans, we conducted experiments on two public data and two private data and compared SRTrans to current state-of-the-art CDRSs and single-domain baselines under four cross-domain scenarios that contain domains from different companies. Note that there is no merged interaction between domains, the above scenarios are, hence, the same as the ones using external commercial data. The experiment results demonstrate that SRTrans significantly outperforms all comparisons in most cases, which indicates the effectiveness of our SRTrans in improving recommendations with external commercial data.

Although SRTrans is effective in leveraging external commercial data, it can only leverage data from a single source domain. This finding reveals a future work that extends SRTrans for leveraging the data from multiple domains. Leveraging data from multiple domains is practical for real-world services. For example, when local services provide different categories of items, such as books and movies, it is reasonable to leverage data from external book domains and movie domains.

Chapter 5

Summary

5.1 Summary of Contributions

In this thesis, we have discussed the methods to improve the performance of recommendations in three realistic scenarios with sparse interaction data.

In Chapter 1, we explained why enhancing the performance of RSs is essential. Then, we introduced three challenges that limit the effectiveness of existing RSs and hinder their applications. These challenges include modeling period-specific user preferences in flash sale e-commerce, improving recommendations with public data, and improving recommendations with external commercial data. Subsequently, we developed methods that overcome these challenges.

In Chapter 2, we addressed the challenge of modeling period-specific user preferences in flash sale e-commerce. Period-specific preferences arise from periodically changing sale strategies in flash sale services, and they are revealed through interactions in different sale periods. Existing RSs, which model only a uniform user preference from all of the user's interaction data in different periods, cannot capture these period-specific preferences. In flash sale environments, this approach significantly degrades the performance of these RSs and makes them difficult to adapt to new periods.

To address this issue, we proposed a novel hierarchical meta-learning-based RS that models users' period-specific preferences accurately. Our model utilizes a novel hierarchical meta-training algorithm to guide its learning process with user- and period-

specific gradients. These gradients enable our model to learn user- and period-shared prior knowledge, facilitating quick adaptation to different periods and users, including new ones. Experimental results on real-world datasets have demonstrated that our proposed method outperforms existing state-of-the-art methods in flash-sale recommendations.

In Chapter 3, we addressed the challenge of improving recommendations with public data for services that lack sufficient training data. Learning a large number of parameters in deep learning-based RSs with small data usually causes the overfitting problem, degrading recommendation performances. Most existing CDRSs leverage data from external domains relying on domain-shared users to transfer individual knowledge. Domain-shared users indicate the users who have interactions in both local (target) and external (source) domains. This makes these CDRSs unable to leverage public data because it is impossible to find domain-shared users between local and public data. Others that require no domain-shared users, however, suffer from a negative transfer issue due to ignoring the impact of source interactions that show users' unique interests within the source domain.

We, therefore, proposed a CDRS (i.e., SCDGN) to track the above challenges. SCDGN jointly clusters items in source and target domains, and merges cluster-level interactions between the two domains based on these cross-domain clusters. By doing so, SCDGN avoids domain-shared users. Besides, SCDGN adaptively weights source interactions at the cluster level. This handles the impact of interactions containing users' source-specific preferences and addresses the negative transfer. Experimental results have shown that our proposal outperforms all cross-domain comparisons and significantly alleviates the negative transfer issue in scenarios of leveraging public data.

In Chapter 4, we addressed the challenge of improving recommendations with external commercial data for services that have only sparse interactions. Learning high-performance RSs require a large number of training interaction data. Therefore, it is essential and practical for services with only sparse interactions to improve their recommendations with the help of external data. Since employing only public data is usually insufficient due to its limited supports, there is a practical demand to develop CDRSs that can effectively leverage external commercial data. Existing CDRSs, including SCDGN proposed in Chapter 3, cannot meet the above demand. This is because

emerging interactions between domains in this scenario is prohibitive due to privacy concerns.

We proposed a CDRS (i.e., SRTrans) to handle the negative transfer issue in this scenario for meeting the demand of effectively using external commercial data without merging interactions between domains. SRTrans bridges domains and transfers knowledge through a similarity-based cross-domain graph. Besides, SRTrans handles the negative transfer issue by transferring knowledge while focusing on the items that have similar textual features. In this way, the impact of source items that have irrelevant textual features and tend to present users' source-specific preferences can be alleviated. Experiments on both public and commercial data demonstrated that our proposal significantly outperforms cross-domain comparisons in terms of recommendation accuracy and greatly alleviates the negative transfer issue.

Considering the benefits offered by our proposed methods and the challenges they addressed, we believe that our approaches will enhance recommendation performances for services that have sparse interaction data, e.g., flash sale e-commerce, new services, and small companies. Furthermore, we believe that they will play a significant role in advancing commercial corporations' efforts to build CDRSs.

5.2 Future Work

Through the work in this thesis, we identified the following remaining issues.

5.2.1 Quantify the Negative Transfer Issue with Transfer Loss

User preferences toward items vary from domain to domain. These domain-dependent preferences cause a negative transfer issue, as observed when comparing the recommendation performance between a CDRS with transferred source knowledge and a single-domain RS without source knowledge (e.g., our experimental results in Chapters 3 and 4). Although our proposed methods in Chapters 3 and 4 have experimentally alleviated the negative transfer issue in most cases, they still require a quantifiable criterion to measure the degree of this issue. By defining such a criterion as a transfer loss, we can explicitly alleviate the negative transfer issue by minimizing the transfer loss. Addi-

tionally, with such a criterion, comparison experiments are expected to provide a more convincing result to demonstrate the effectiveness of our methods in alleviating the negative transfer issue and also in handling domain bias in user preferences. Therefore, we plan to design a learnable transfer loss that measures the degree of the negative transfer issue.

5.2.2 Effectively Leverage Data from Multiple Domains

Although our methods in Chapters 3 and 4 can leverage data from an external domain, they can leverage external data from only one domain. This poses a practical challenge in leveraging data from diverse source domains. This challenge is prevalent across various industries and applications. For example, when local items contain multiple categories, such as books and movies, it is reasonable to leverage public data from both book and movie domains. Besides, in a commercial corporation involving more than two companies, a CDRS that can effectively leverage commercial data from multiple companies is desirable. Therefore, a part of our future work is to extend our proposed methods in Chapters 3 and 4 so that they can work well on multiple domains.

Acknowledgment

This thesis encapsulates my academic endeavors at Osaka University and a significant chapter of my life spent studying and residing in Japan over these years. The successful completion of this thesis is attributed to the selfless support and hard work of a number of individuals.

First and foremost, I extend my deepest gratitude to my advisor, Professor Takahiro Hara, who has provided me with countless opportunities since the beginning of my journey in this laboratory. He has been a profound motivator for me to start the Ph.D. course and grow as a researcher. It has been an honor to be his Ph.D. student. His guidance and mentorship have supported me throughout my master's and doctoral journey. His insightful feedback and encouragement have been invaluable in shaping my research.

I am greatly indebted to my associate advisor, Assistant Professor Daichi Amagata, who has meticulously guided every detail in each step of my research and all the drafts of my papers. Without his kind guidance and valuable advice, it would have been impossible for me to finish my research and this thesis.

I gratefully acknowledge the help of my instructive professor, Assistant Professor Takuya Maekawa, for his valuable instructions and suggestions on my research. Also, I am grateful for the precious opportunities for tutorial presentations at international conferences provided by Professor Takuya Maekawa.

I would like to thank a researcher in our research group, Specially Appointed Associate Professor Yihong Zhang, for his enduring support and much-appreciated advice throughout my dissertation.

In addition, I would like to express my appreciation to Mr. Mori Kurokawa, Mr. Kei Yonekawa, Dr. Duc Nguyen, Dr. Shuichiro Haruta, and Dr. Hao Niu from KDDI Research Inc. Without the foundational equipment and data provided by them, many

evaluations would have been impossible to accomplish. Besides, they always gave me helpful guidance and technological support throughout my research.

I would like to acknowledge my research team, both past and present members, Dr. Qingxin Xia, Mr. Bunki Cao, Mr. Yuan Lyu, Mr. Hanxin Wang, Mr. Yuya Takahashi, Mr. Takuma Kaiho, Ms. Yue Zhao, Ms. Hikaru Nomura, Mr. Jun Murao, Mr. Kentaro Shiga, Mr. Rikuto Tsubouchi, Mr. Takuma Yamashita, Mr. Yuma Dose, Mr. Yoshiyuki Maekawa, and Mr. Keita Matsunaka. Also, it is my pleasure to work with all the smart members of Hara laboratory. All of them have provided me with cooperative and active support throughout my study.

Finally, this thesis would not have taken place without the support of my family and friends who gave me unconditional love and motivation. I give my thanks to all involved.

REFERENCE

[1] Dietmar Jannach and Markus Zanker. Value and impact of recommender systems. In *Recommender Systems Handbook*, pages 519–546. 2022.

[2] Dokyun Lee and Kartik Hosanagar. Impact of recommender systems on sales volume and diversity. In *ICIS*, 2014.

[3] Dietmar Jannach and Michael Jugovac. Measuring the business value of recommender systems. *ACM Trans. Manag. Inf. Syst.*, 10(4):16:1–16:23, 2019.

[4] M. Benjamin Dias, Dominique Locher, Ming Li, Wael El-Deredy, and Paulo J. G. Lisboa. The value of personalised recommender systems to e-business: a case study. In *RecSys*, pages 291–294, 2008.

[5] Richard D. Lawrence, George S. Almási, Vladimir Kotlyar, Marisa S. Viveros, and Sastry Duri. Personalization of supermarket product recommendations. *Data Min. Knowl. Discov.*, 5(1/2):11–32, 2001.

[6] Ye Chen and John F. Canny. Recommending ephemeral items at web scale. In *SIGIR*, pages 1013–1022, 2011.

[7] Renjie Zhou, Samamon Khemmarat, and Lixin Gao. The impact of youtube recommendation system on video views. In *SIGCOMM*, pages 404–410, 2010.

[8] Abhinandan Das, Mayur Datar, Ashutosh Garg, and Shyamsundar Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW*, pages 271–280, 2007.

[9] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1):1–38, 2019.

[10] Trong Dinh Thac Do and Longbing Cao. Metadata-dependent infinite poisson factorization for efficiently modelling sparse and large matrices in recommendation. In *IJCAI*, pages 5010–5016, 2018.

[11] Zhi Li, Daichi Amagata, Yihong Zhang, Takuya Maekawa, Takahiro Hara, Kei Yonekawa, and Mori Kurokawa. Hml4rec: Hierarchical meta-learning for cold-start recommendation in flash sale e-commerce. *Knowl. Based Syst.*, 255:109674, 2022.

[12] Hanxin Wang, Daichi Amagata, Takuya Makeawa, Takahiro Hara, Niu Hao, Kei Yonekawa, and Mori Kurokawa. A dnn-based cross-domain recommender system for alleviating cold-start problem in e-commerce. *IEEE Open J. Ind. Electron*, 1:194–206, 2020.

[13] Hanxin Wang, Daichi Amagata, Takuya Maekawa, Takahiro Hara, Hao Niu, Kei Yonekawa, and Mori Kurokawa. Preliminary investigation of alleviating user cold-start problem in e-commerce with deep cross-domain recommender system. In *ECNLP*, pages 398–403, 2019.

[14] Adit Krishnan, Ashish Sharma, Aravind Sankar, and Hari Sundaram. An adversarial approach to improve long-tail performance in neural collaborative filtering. In *CIKM*, pages 1491–1494, 2018.

[15] Zhihong Chen, Rong Xiao, Chenliang Li, Gangfeng Ye, Haochuan Sun, and Hongbo Deng. Esam: Discriminative domain adaptation with non-displayed items to improve long-tail performance. In *SIGIR*, page 579–588, 2020.

[16] Chenglin Li, Yuanzhen Xie, Chenyun Yu, Bo Hu, Zang Li, Guoqiang Shu, Xiaohu Qie, and Di Niu. One for all, all for one: Learning and transferring user embeddings for cross-domain recommendation. In *WSDM*, pages 366–374, 2023.

[17] Jiangxia Cao, Shaoshuai Li, Bowen Yu, Xiaobo Guo, Tingwen Liu, and Bin Wang. Towards universal cross-domain recommendation. In *WSDM*, pages 78–86, 2023.

[18] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Junchi Yan, and Hongyuan Zha. Towards open-world recommendation: An inductive model-based collaborative filtering approach. In *ICML*, volume 139, pages 11329–11339, 2021.

[19] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. Dropoutnet: Addressing cold start in recommender systems. In *Proceedings of the International Conference on Neural Information Processing Systems*, pages 4957–4966, 2017.

[20] Shaoyun Shi, Min Zhang, Yiqun Liu, and Shaoping Ma. Attention-based adaptive model to unify warm and cold starts recommendation. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 127–136, 2018.

[21] Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks. In *SIGIR*, pages 1167–1176, 2021.

[22] Muhammad Murad Khan, Roliana Ibrahim, and Imran Ghani. Cross domain recommender systems: A systematic literature review. *ACM Comput. Surv.*, 50(3):36:1–36:34, 2017.

[23] Jiangxia Cao, Xixun Lin, Xin Cong, Jing Ya, Tingwen Liu, and Bin Wang. Disencdr: Learning disentangled representations for cross-domain recommendation. In *SIGIR*, pages 267–277, 2022.

[24] Yoonhyuk Choi, Jiho Choi, Taewook Ko, Hyungho Byun, and Chong-Kwon Kim. Review-based domain disentanglement without duplicate users or contexts for cross-domain recommendation. In *CIKM*, pages 293–303, 2022.

[25] Yinan Zhang, Yong Liu, Peng Han, Chunyan Miao, Lizhen Cui, Baoli Li, and Haihong Tang. Learning personalized itemset mapping for cross-domain recommendation. In *IJCAI*, pages 2561–2567, 2020.

[26] Weiming Liu, Xiaolin Zheng, Jiajie Su, Mengling Hu, Yanchao Tan, and Chaochao Chen. Exploiting variational domain-invariant user embedding for partially overlapped cross domain recommendation. In *SIGIR*, pages 312–321, 2022.

[27] Junda Wu, Zhihui Xie, Tong Yu, Handong Zhao, Ruiyi Zhang, and Shuai Li. Dynamics-aware adaptation for reinforcement learning based cross-domain interactive recommendation. In *SIGIR*, pages 290–300, 2022.

[28] Feng Zhu, Yan Wang, Chaochao Chen, Guanfeng Liu, and Xiaolin Zheng. A graphical and attentional framework for dual-target cross-domain recommendation. In *IJCAI*, pages 3001–3008, 2020.

[29] Meng Liu, Jianjun Li, Guohui Li, and Peng Pan. Cross domain recommendation via bi-directional transfer graph collaborative filtering networks. In *CIKM*, pages 885–894, 2020.

[30] Chaochao Chen, Huiwen Wu, Jiajie Su, Lingjuan Lyu, Xiaolin Zheng, and Li Wang. Differential private knowledge transfer for privacy-preserving cross-domain recommendation. In *WWW*, pages 1455–1465, 2022.

[31] Zhi Li, Daichi Amagata, Yihong Zhang, Takahiro Hara, Shuichiro Haruta, Kei Yonekawa, and Mori Kurokawa. Debiasing graph transfer learning via item semantic clustering for cross-domain recommendations. In *IEEE BigData*, 2022.

[32] Xinting Liao, Weiming Liu, Xiaolin Zheng, Binhui Yao, and Chaochao Chen. Ppgencdr: A stable and robust framework for privacy-preserving cross-domain recommendation. In *AAAI*, pages 4453–4461, 2023.

[33] Gaode Chen, Xinghua Zhang, Yijun Su, Yantong Lai, Ji Xiang, Junbo Zhang, and Yu Zheng. Win-win: A privacy-preserving federated framework for dual-target cross-domain recommendation. In *AAAI*, pages 4149–4156, 2023.

[34] Zhi Li, Daichi Amagata, Yihong Zhang, Takahiro Hara, Shuichiro Haruta, Kei Yonekawa, and Mori Kurokawa. Semantic relation transfer for non-overlapped cross-domain recommendations. In *PAKDD*, volume 13937, pages 271–283, 2023.

[35] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70, pages 1126–1135, 2017.

[36] Zhi Li, Daichi Amagata, Takuya Maekawa, Kei Yonekawa, Mori Kurokawa, and Takahiro Hara. Trends-enhanced attention & memory networks for e-commerce recommendation. In *SIGIReCom*, 2022.

[37] Zhi Li, Boqi Gao, Daichi Amagata, Takuya Maekawa, Takahiro Hara, Hao Niu, Kei Yonekawa, and Mori Kurokawa. Trends tracking memory recommender networks for product recommendations in e-commerce. In *DEIM*, 2020.

[38] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *KDD*, pages 839–848, 2018.

[39] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *DLRS*, pages 7–10, 2016.

[40] Daichi Amagata and Takahiro Hara. Reverse maximum inner product search: How to efficiently find users who would like to buy my item? In *RecSys*, pages 273–281, 2021.

[41] Mori Kurokawa, Hao Niu, Kei Yonekawa, Arei Kobayashi, Daichi Amagata, Takuya Maekawa, and Takahiro Hara. Virtual touch-point: trans-domain behavioral targeting via transfer learning. In *IEEE BigData*, pages 4762–4767, 2018.

[42] Duc Nguyen, Hao Niu, Kei Yonekawa, Mori Kurokawa, Chihiro Ono, Daichi Amagata, Takuya Maekawa, and Takahiro Hara. On the transferability of deep neural networks for recommender system. In *IAL*, pages 22–37, 2020.

[43] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

[44] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, pages 1441–1450, 2019.

[45] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, pages 565–573, 2018.

[46] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. Improving sequential recommendation with knowledge-enhanced memory networks. In *SIGIR*, pages 505–514, 2018.

[47] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, pages 1024–1034, 2017.

[48] Jingjing Qiao and Li Wang. Modeling user micro-behaviors and original interest via adaptive multi-attention network for session-based recommendation. *Knowl. Based Syst.*, 244:108567, 2022.

[49] Adit Krishnan, Mahashweta Das, Mangesh Bendre, Hao Yang, and Hari Sundaram. Transfer learning via contextual invariants for one-to-many cross-domain recommendation. In *SIGIR*, pages 1081–1090, 2020.

[50] Pan Li and Alexander Tuzhilin. DDTCDR: deep dual transfer cross domain recommendation. In *WSDM*, pages 331–339, 2020.

[51] Cheng Wang, Mathias Niepert, and Hui Li. Recsys-dan: Discriminative adversarial networks for cross-domain recommender systems. *IEEE Trans. Neural Netw. Learn. Syst.*, 31(8):2731–2740, 2020.

[52] Bin Li, Qiang Yang, and Xiangyang Xue. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, pages 2052–2057, 2009.

[53] Ting Yu, Junpeng Guo, Wenhua Li, and Meng Lu. A mixed heterogeneous factorization model for non-overlapping cross-domain recommendation. *Decis. Support Syst.*, 151:113625, 2021.

[54] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. Melu: Meta-learned user preference estimator for cold-start recommendation. In *KDD*, pages 1073–1082, 2019.

[55] Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. Mamo: Memory-augmented meta-optimization for cold-start recommendation. In *KDD*, pages 688–697, 2020.

[56] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *ICLR*, 2020.

[57] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017.

[58] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. Deep matrix factorization models for recommender systems. In *IJCAI*, pages 3203–3209, 2017.

[59] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *WSDM*, pages 153–162, 2016.

[60] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *WWW*, pages 689–698, 2018.

[61] Trapit Bansal, David Belanger, and Andrew McCallum. Ask the gru: Multi-task learning for deep text recommendations. In *RecSys*, pages 107–114, 2016.

[62] Zhi Li, Hongke Zhao, Qi Liu, Zhenya Huang, Tao Mei, and Enhong Chen. Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors. In *KDD*, pages 1734–1743, 2018.

[63] Weiyu Cheng, Yanyan Shen, and Linpeng Huang. Adaptive factorization network: Learning adaptive-order feature interactions. In *AAAI*, pages 3609–3616, 2020.

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

[65] Yujia Zheng, Siyi Liu, Zekun Li, and Shu Wu. Cold-start sequential recommendation via meta learner. In *AAAI*, pages 4706–4713, 2021.

[66] Zhengxiao Du, Xiaowei Wang, Hongxia Yang, Jingren Zhou, and Jie Tang. Sequential scenario-specific meta learner for online recommendation. In *KDD*, pages 2895–2904, 2019.

[67] Jiayu Song, Jiajie Xu, Rui Zhou, Lu Chen, Jianxin Li, and Chengfei Liu. CBML: A cluster-based meta-learning model for session-based recommendation. In *CIKM*, pages 1713–1722, 2021.

[68] Ruobing Xie, Yalong Wang, Rui Wang, Yuanfu Lu, Yuanhang Zou, Feng Xia, and Leyu Lin. Long short-term temporal meta-learning in online recommendation. In *WSDM*, pages 1168–1176, 2022.

[69] Xidong Feng, Chen Chen, Dong Li, Mengchen Zhao, Jianye Hao, and Jun Wang. CMML: contextual modulation meta learning for cold-start recommendation. In *CIKM*, pages 484–493, 2021.

[70] Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks. In *SIGIR*, pages 1167–1176, 2021.

[71] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[72] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural. Comput. Appl.*, 9(8):1735–1780, 1997.

[73] Zuoxi Yang, Shoubin Dong, and Jinlong Hu. GFE: general knowledge enhanced framework for explainable sequential recommendation. *Knowl. Based Syst.*, 230:107375, 2021.

[74] Fenfang Xie, Angyu Zheng, Liang Chen, and Zibin Zheng. Attentive meta-graph embedding for item recommendation in heterogeneous information networks. *Knowl. Based Syst.*, 211:106524, 2021.

[75] Chaoqun Feng, Chongyang Shi, Shufeng Hao, Qi Zhang, Xinyu Jiang, and Daohua Yu. Hierarchical social similarity-guided model with dual-mode attention for session-based recommendation. *Knowl. Based Syst.*, 230:107380, 2021.

[76] Lin Liu, Li Wang, and Tao Lian. Case4sr: Using category sequence graph to augment session-based recommendation. *Knowl. Based Syst.*, 212:106558, 2021.

[77] Zhiqiang Pan, Fei Cai, Wanyu Chen, Chonghao Chen, and Honghui Chen. Collaborative graph learning for session-based recommendation. *ACM Trans. Inf. Syst.*, 40(4):72:1–72:26, 2022.

[78] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. Sequential recommendation with user memory networks. In *WSDM*, pages 108–116, 2018.

[79] Oren Barkan and Noam Koenigstein. Item2vec: neural item embedding for collaborative filtering. In *MLSP*, pages 1–6, 2016.

[80] Wenjie Pei, Jie Yang, Zhu Sun, Jie Zhang, Alessandro Bozzon, and David M. J. Tax. Interacting attention-gated recurrent networks for recommendation. In *CIKM*, pages 1459–1468, 2017.

[81] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[82] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge J. Belongie, and Deborah Estrin. Collaborative metric learning. In *WWW*, pages 193–201, 2017.

[83] Qinyong Wang, Hongzhi Yin, Zhiting Hu, Defu Lian, Hao Wang, and Zi Huang. Neural memory streaming recommender networks with adversarial training. In *KDD*, pages 2467–2475, 2018.

[84] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[85] Yoonho Lee and Seungjin Choi. Gradient-based meta-learning with learned layerwise metric and subspace. In *ICML*, volume 80, pages 2933–2942, 2018.

[86] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[87] Ruining He and Julian J. McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*, pages 507–517, 2016.

[88] Thanh Tran, Kyumin Lee, Yiming Liao, and Dongwon Lee. Regularizing matrix factorization with user and item embeddings for recommendation. In *CIKM*, pages 687–696, 2018.

[89] Thanh Tran, Xinyue Liu, Kyumin Lee, and Xiangnan Kong. Signed distance-based deep memory recommender. In *WWW*, pages 1841–1852, 2019.

[90] Guangneng Hu, Yu Zhang, and Qiang Yang. Conet: Collaborative cross networks for cross-domain recommendation. In *CIKM*, pages 667–676, 2018.

[91] Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456, 2011.

[92] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *KDD*, pages 1235–1244, 2015.

[93] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[94] Daichi Amagata, Takahiro Hara, and Chuan Xiao. Dynamic set knn self-join. In *ICDE*, pages 818–829, 2019.

[95] Chenglin Li, Mingjun Zhao, Huanming Zhang, Chenyun Yu, Lei Cheng, Guoqiang Shu, Beibei Kong, and Di Niu. Recguru: Adversarial learning of generalized user representations for cross-domain recommendation. In *WSDM*, pages 571–581, 2022.

[96] Weiming Liu, Xiaolin Zheng, Mengling Hu, and Chaochao Chen. Collaborative filtering with attribution alignment for review-based non-overlapped cross domain recommendation. In *WWW*, pages 1181–1190, 2022.

[97] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*, pages 639–648, 2020.

[98] Siqing Li, Liuyi Yao, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Tonglei Guo, Bolin Ding, and Ji-Rong Wen. Debiasing learning based cross-domain recommendation. In *KDD*, pages 3190–3199, 2021.

[99] Pan Li and Alexander Tuzhilin. Dual metric learning for effective and efficient cross-domain recommendations. *IEEE Trans. Knowl. Data Eng.*, 35(1):321–334, 2023.

[100] Yongchun Zhu, Zhenwei Tang, Yudan Liu, Fuzhen Zhuang, Ruobing Xie, Xu Zhang, Leyu Lin, and Qing He. Personalized transfer of user preferences for cross-domain recommendation. In *WSDM*, page 1507–1515, 2022.

[101] Feng Yuan, Lina Yao, and Boualem Benatallah. Darec: Deep domain adaptation for cross-domain recommendation via transferring rating patterns. In *IJCAI*, pages 4227–4233, 2019.

[102] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. Self-supervised graph learning for recommendation. In *SIGIR*, pages 726–735, 2021.

[103] Lianghao Xia, Yong Xu, Chao Huang, Peng Dai, and Liefeng Bo. Graph meta network for multi-behavior recommendation. In *SIGIR*, pages 757–766, 2021.

[104] Huiyuan Chen, Lan Wang, Yusan Lin, Chin-Chia Michael Yeh, Fei Wang, and Hao Yang. Structured graph convolutional networks with stochastic masks for recommender systems. In *SIGIR*, pages 614–623, 2021.

[105] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. KGAT: knowledge graph attention network for recommendation. In *KDD*, pages 950–958, 2019.

[106] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *KDD*, pages 974–983, 2018.

[107] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *SIGIR*, pages 165–174, 2019.

[108] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying graph convolutional networks. In *ICML*, volume 97, pages 6861–6871, 2019.

[109] Lei Guo, Li Tang, Tong Chen, Lei Zhu, Quoc Viet Hung Nguyen, and Hongzhi Yin. DA-GCN: A domain-aware attentive graph convolution network for shared-account cross-domain sequential recommendation. In *IJCAI*, pages 2483–2489, 2021.

[110] Kun Xu, Yuanzhen Xie, Liang Chen, and Zibin Zheng. Expanding relationship for cross domain recommendation. In *CIKM*, pages 2251–2260, 2021.

[111] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019.

[112] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297, 1967.

[113] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[114] Kei Yonekawa, Hao Niu, Mori Kurokawa, Arei Kobayashi, Daichi Amagata, Takuya Maekawa, and Takahiro Hara. Advertiser-assisted behavioral ad-targeting via denoised distribution induction. In *IEEE BigData*, pages 5611–5619, 2019.

[115] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. Cross-domain recommendation: An embedding and mapping approach. In Carles Sierra, editor, *IJCAI*, pages 2464–2470, 2017.

[116] SeongKu Kang, Junyoung Hwang, Wonbin Kweon, and Hwanjo Yu. DE-RRD: A knowledge distillation framework for recommender system. In *CIKM*, pages 605–614, 2020.

[117] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.