IMPROVING CONTENT-AWARE RECOMMENDATION: RELATIONSHIP MINING, INTEGRATING, AND DISENTANGLING

A Dissertation

by

YIN ZHANG

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2022

Major Subject: Computer Engineering

ABSTRACT

Explosive growth in the amount of information has infiltrated every aspect of our lives. Recommender systems, as an effective tool to filter information, have become prevalent and influential in many domains. This dissertation seeks to improve recommender systems by learning the unique properties and complex relationships of the rich content information associated with both users (who seek recommendations) and items (that are recommended to users). However, to do so, there are key research challenges: (i) the extreme sparsity of the observed item relations; (ii) high heterogeneity between user-user relations and user-item interactions; (iii) duplication between content and collaborative signals; and (iv) the highly skewed long-tail distribution of user feedback towards items. With these challenges, this dissertation research makes four unique contributions:

- The first research contribution of this dissertation is to mine and integrate the item-side relations. Concretely, we begin by investigating different types of item relations (e.g., complementary and substitute relations), and build a novel neural item-relationship based model to uncover the item relationships. We then integrate these item multi-relations into user sequences through a hierarchical temporal graph to enhance sequential recommendation.

- The second research contribution of this dissertation is to mine and integrate the user-side relations. We are one of the first to investigate the socio-behavioral phenomenon of social resonance to closely connect user relations with user interactions towards items. Then we integrate the user social relations in sequential recommendations to improve fashion recommendations. We also provide a new large visual dataset of dynamic visual posts by key fashion bloggers that contain both high-quality dynamic fashion preference shifts over time.

- The third research contribution of this dissertation is to disentangle the content and collaborative features to address the duplication problem in content-aware recommendations. We propose a novel two-level disentanglement approach that supports both content-collaborative disentanglement and feature disentanglement based on a variational auto-encoder.

- The fourth research contribution of this dissertation is to decouple the learning process that tackles the skewed long-tail distribution problem. We propose a novel cross decoupling framework that utilizes cumulative learning and multi-experts to decouple the learning of long-tail distribution from prior and conditional knowledge aspects.

DEDICATION

To my family.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. James Caverlee, for his mentorship, guidance, encouragement, help, and support. All of those are greatly meaningful to me. It is my fortune to have him as my advisor. He is my role model. After years of working with him, I learned what the research is, how to find a well-defined problem, how to do research and most of all, how to become a better person. His full of energy always encourages me to be enthusiastic and passionate about my research as him.

I was fortunate to work with many great people during my three internships. First of all, I would like to thank my mentor Zhiyuan Derek Cheng, for offering me two amazing internship experiences. His wide knowledge and professions inspired me to come up with good ideas for projects. I also benefit a lot from his advice and suggestions. I would also like to thank my mentor Ruoxi Wang, for her patient and kindness. She helped me dig into details to make the project go smoothly and I also learned a lot about how to successfully manage a project. I also would like to thank Tiansheng Yao, Xinyang Yi, Lichan Hong and Ed H. Chi, for their help and advice on my remote internship projects. Particularly, many thanks to Lichan Hong and Ed H. Chi, for their guidance of my future career. I also enjoyed the discussion with Maciej Kula, Jiaxi Tang, Wang-Cheng Kang, Bo Chang, Minmin Chen and many other members at Google, who provide many useful ideas and suggestions. I am grateful to John Zheng, Mao Ye, Yitong Zhou, Aditya Pal, Peifeng Yin for their help during my internship at Pinterest.

I am deeply indebted to my parents Yazeng Zhang and Yanxia Gao, for their boundless love and continuous support. Words can not express how much I love them. They are my role models, always encouraging me no matter I am in ups and downs in my life. My grandparents also give me many supports and encouragement. Many thanks to my husband Yiming and my little girl Ellie. You are the inspirations of my life.

I also would like to thank my awesome labmates. First, thanks to my senior labmates Wei Niu, Haokai Lu, Majid Alfifi, Hancheng Ge and Cheng Cao. Wei Niu and Haokai Lu gave me many useful suggestions for my first Ph.D. paper. It is a very special and valuable experience for me. They also gave me many great suggestions for my Ph.D. career. Majid Alfifi helped me a lot to maintain the server. He is really an expert on that! Many thanks to my labmates Yun He, Ziwei Zhu, Xing Zhao, Jianling Wang, Parisa Kaghazgaran, Zhuoer Wang, Haiping Xue, Xiangjue Dong and Allen Lin for working together during my Ph.D. journey. I enjoyed every discussion, brainstorming and collaboration with them. They inspired me and had a tremendous impact on my Ph.D. career. Thank you!

I would also like to thank Dr. Ruihong Huang, Dr. Theodora Chaspari and Dr. Yu Ding for being my Ph.D. committee. Their continuous support and advice motivated me to successfully complete my dissertation.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor James Caverlee, Professor Ruihong Huang and Professor Theodora Chaspari of the Department of Computer Science and Engineering and Professor Yu Ding of the Department of Industrial & Systems Engineering and Electrical & Computer Engineering.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by a fund from Texas A&M University, NSF and DARPA grants.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

Explosive growth in the amount of information over the past decades has infiltrated every aspect of our lives. Users are over-exposed to products, news, jobs, media, and more. One effective and powerful tool to make sense of this deluge of information is the *recommender system*. Recommender systems have blossomed, including popular recommenders like those that power Amazon, Netflix, and YouTube, as well as in domains like education [1] and medical science [2]. Furthermore, recommender systems can be viewed as an important area of study for developing personalized machine learning methods with broad impact.

The recommendation algorithms that power these systems are typically based on either *collaborative* or *content-based* approaches. Collaborative approaches seek to exploit collaborative features from user behavior data (e.g., clicks or likes) to identify items of interest. For example, the neighbourhood-based collaborative filtering methods [3] usually find a neighborhood of similar users around a target user (where similarity may be based on similar ratings over a common set of items), and then calculate the weighted sum of ratings of those similar users to identify items to recommend to the target user. In contrast, content-based approaches aim to explore content information about the users and items (e.g., user ages or item images). For example, VBPR [4] leverages the images associated with items to discover the visual patterns that users prefer (e.g., striped shirts or solid shirts) to find good items to recommend. Content-aware recommenders are ubiquitous in practice and have gained extensive attention due to their strong and robust performance [4–7]. Compared with purely collaborative-based methods, content-aware recommenders can utilize both user and item properties to improve recommendation performance, especially for cold-start users, and also offer good interpretability at the same time (e.g., to explain that a shirt was recommended due to a particular pattern).

This dissertation seeks to improve recommender systems by exploiting the unique properties and complex relationships of the rich content information associated with both users (who seek recommendations) and items (that are recommended to users). Let's consider each of these perspectives in turn: item-side content and user-side content.

Item-side Content. Item-side content describes item properties such as the images associated with an item, its reviews, and multi-relations (e.g., complementary or substitute relations between items). Item-side content is especially useful for identifying what particular aspects of an item a user prefers to make recommendations more transparent. Particularly, the relations among items play an important and unique role [8]. For example, knowing if one item is a substitute for another (i.e. is interchangeable, as in the case of a camera) or is a complement (i.e. goes well with one another, as in a specific lens that works with a camera) can facilitate many recommendation applications, such as helping users find related items, discover new items, and promoting bundle purchases. Further, when we consider the order of user interaction history (known as sequential recommendation [9]), item relations play a significant role to influence the user's next actions [8, 10–12]. An example is, in a short time period, users are more likely to purchase a phone and its complementary accessories, while later she may tend to purchase another updated phone to replace her old ones. Therefore, item relations provide further understanding of user sequence patterns and could help uncover user sequence patterns.

User-side Content. User-side content like the user's age, location, or social connections can be helpful to describe users and is of particular significance for improving user cold-start recommendation. User relations, such as friends, colleagues, followers and followees, can heavily influence the user preference towards items. Indeed, social effects such as social homophily and the importance of social connections in the presence of limited attention have been shown to enhance recommendation quality [13, 14]. Of special interest in this dissertation of user-side content is the impact of fashion-specific influencers on recommendation. Fashion is a core cultural phenomenon with large social impact [15, 16]. We can view recommendation in this context as a special type of sequential recommendation that is complicated by changing trends, the importance of visual signals, and the impact of fashion leaders. Hence, careful consideration of user-side content is critical for the development of effective recommenders.

With these two perspectives for content-based recommendation, it motivates us to carefully mine the rich user-side and item-side content relationships and then integrate these relationships with collaborative signals for improved recommendation algorithms. However, to do so, there are two key research challenges we face:

- **Sparsity of Relations.** First, from the item-side, while item relationships may be a valuable signal for improving recommendation, in many scenarios the observed data points of item relations are quite sparse, especially comparing with the huge number of items. For example, in a public Amazon dataset, the observed item relations are only present in ~0.01% among all items [17]. Furthermore, when we consider the time factor, both user sequences and item relations are even sparser since users interact with very few items. That is, these item-level relations become less useful as items are updated over time (e.g., an older iPhone being replaced by a new model). How could we uncover the unobserved item relations and further integrate those relations with user sequences for improved sequential recommendation?

- **High heterogeneity.** Second, from the user-side, there is high heterogeneity in terms of the types of relationships: some platforms support user-user connections based on social relationships, while some platforms (such as eCommerce ones) may not support direct user-user relationships but do support user-item connections (via purchasing, rating, or interacting). How can we mine the mutual interactions of these relationships across these fundamentally different perspectives? Furthermore, when we consider the time factor, there is high heterogeneity in terms of the influence from user social connections over time. For example, in fashion recommendation, a user could follow fashion bloggers and the fashion preference of those fashion blogger's could be evolving over time. How could we integrate those dynamic changes with user preference towards items in different time periods?

If we can successfully deal with these challenges, we expect to observe improvements gained in recommendation by mining user-side and item-side content and integrating them with collaborative signals. However, naively combining content-based methods with collaborative ones may lead to unintended negative consequences. That is, there are still critical challenges in how we model content:

- **Duplication.** Many recommendation approaches simply concatenate content information with collaborative signals to learn user and item features. The learned user and item features derived from these combined sources can be entangled by intermixing the influence from each, harming recommendation quality. For example, a user, and also many similar users, may prefer a dress because of its visual appearance, price, and high quality. The known content information is the dress images. If the user-item interactions and the dress image are considered separately to learn the features that influence user preference towards items, the collaborative features and content features could be highly correlated. In essence, both the collaborative and content features could redundantly encode the visual characteristics of the dress, meaning there is less capacity to focus on learning other features (like price) that could influence user preference towards items. How could we disentangle this duplicated information to gain more diverse knowledge for enhanced content-aware recommendation?

- **Skewed distribution.** For different items, existing content-aware recommendation methods usually treat content equally important. However, user interactions towards items generally exhibit the long-tail distribution: a small fraction of popular items receive most of the user feedback (which we refer to as head items), while most items only have few user feedback (which we refer to as tail items). We find that many content-aware recommendation methods that treat content information equally only bring improvements for tail items, but the recommendation quality decreases for the head items. Hence, an important question is how to learn the content information while also considering the distribution gap between head and tail items, to improve the recommendation performance?

These additional two challenges motivate us to explore novel methods to disentangle and decouple representations for improved recommendation. That is, we aim to isolate and augment the signal contained in content, so that it is not duplicated nor drowned out by other (collaborative) signals. Indeed, disentanglement and decoupling strategies have been widely studied in computer vision [18] and natural language processing [19] due to their robust performance and interpretability. Few if any methods have considered the decoupling and disentanglement problem in the context of recommendation with both user behavior data and content information, which is especially important for building robust and high-quality user-item joint representations for content-aware recommendation in practice.

1.1 Contributions

With these challenges in mind, this dissertation focuses on improving content-based recommendation through careful mining and integration of the complex relationships among users and items, and further disentanglement and decoupling of the learned representations. In sum, this dissertation makes unique contributions towards content-aware recommendation as shown in Figure 1.1:

- **Mining Item-side Relationships**: We first investigate different types of item relations (e.g., complementary and substitute relations), and build a neural item-relationship based model to learn the complex relationships between items. Based on that, we further apply those item multi-relations into sequential recommendation to enhance the learning of sequence patterns. By investigating and aggregating the multi-relations among items in user interactions among items, we can (i) provide a sound sequential recommender that dynamically adapts item different relations to user sequence behavior for improved next-item recommendation; (ii) effectively uncover high-quality sequence patterns in highly-personalized user sequences; and (iii) alleviate the sparsity issue of items in user sequences for improved recommendation.
- **Investigating User-side Social Influence**: We propose to investigate the socio-behavioral phenomenon of *social resonance* wherein users are more influenced by opinions that “strike a chord”. In recommendation, this social resonance can bridge the gap between a user’s social connections with other users on an eCommerce platform, to better capture the social influence for user preference towards items. We then consider the social relations in sequential recommendation. Concretely, we present a key opinion leaders-aware recommendation that fully

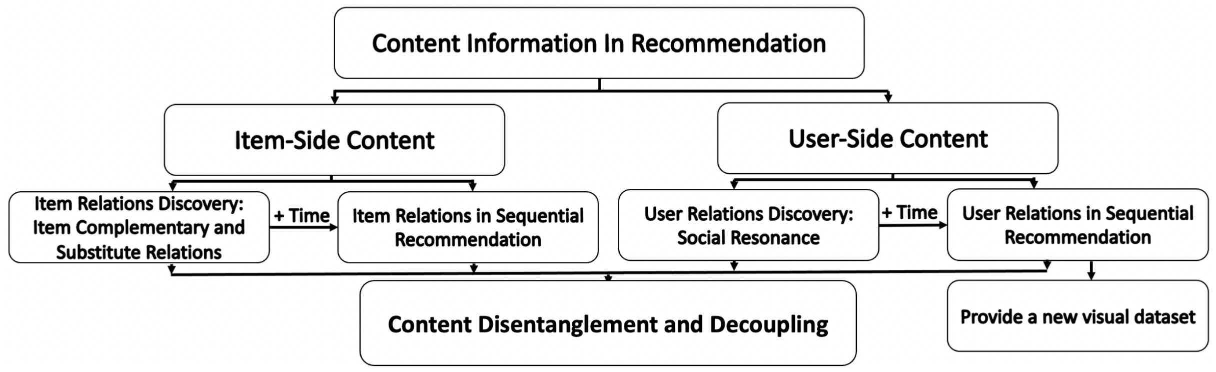

Content Information In Recommendation
Item-Side Content
User-Side Content
Item Relations Discovery: Item Complementary and Substitute Relations
+ Time
Item Relations in Sequential Recommendation
User Relations Discovery: Social Resonance
+ Time
User Relations in Sequential Recommendation
Content Disentanglement and Decoupling
Provide a new visual dataset

Figure 1.1: Dissertation overview: contributions include the modeling of both user and item content information and their applications in sequential recommendation, and how to disentangle and decouple content when both content and collaborative information is considered.

investigates those key opinion influencers on user dynamic preference towards items. We also *provide a new large visual dataset* of dynamic visual posts by key fashion bloggers that contains both high-quality fashion features and captures the dynamic aesthetic preference shifts of users over time.

- **Disentangling Diverse Content Information**: Then we explore how disentanglement can help learn different aspects of features that influence user preference towards items. We propose a two-level disentanglement approach that supports both content-collaborative disentanglement and feature disentanglement based on the structure of a variational auto-encoder. We successfully tackle the duplication problem by theoretically showing that each extracted feature in our proposed model is disentangled with other extracted features via statistical independence properties. Such disentanglement representation learning can improve the user and item representations, alleviate the high feature correlation influence, and offer a high-quality, stable and more interpretable recommendation.
- **Decoupling Memorization and Generalization**: Finally, we propose a cross decoupling method that separately considers memorization and generalization of items in the highly skewed long-tail distribution. Concretely, we provide a theoretically analysis and experimental investigation of previous methods that address the skewed distribution problem. With a compre-

hensive understanding of the limitations of previous methods, we utilize cumulative learning and multi-experts to decouple the learning of long-tail distribution from prior and conditional knowledge aspects. Therefore, head items (with lots of user feedback) can be separately learned from tail items (with little user feedback) with careful consideration of the distribution gap between them. The proposed method shows improvements for both head and tail items. The learned item representations also preserve better semantics of items.

1.2 Dissertation Overview

The remainder of this dissertation is organized as follows:

- **Chapter 2: Related Work.** In this chapter, we discuss related work, especially the content information utilized in item and user side, with a focus on the item relations and user relations in recommendation, and their implementation in sequential recommendation. We also discuss recent advances in decoupling and disentanglement learning.
- **Chapter 3: Quality-Aware Neural Complementary Item Recommendation.** In this chapter, we propose Encore that learns the item complementary relations based on both item visual (stylistic) and textual (functional) content information, and further gives a high quality complementary item recommendation based on a novel neural network model.
- **Chapter 4: Adaptive Hierarchical Translation-based Sequential Recommendation.** In this chapter, we further consider the item complementary and substitute relations in sequential recommendation settings. Concretely, we propose a novel translation-based sequential recommendation HierTrans that integrates item multi-relations to user sequences through a hierarchical temporal graph. The learned user preference towards items can adaptively change based on both user interacted item relations and the user own preference.
- **Chapter 5: Vibe Check: Social Resonance Learning for Enhanced Recommendation.** In this chapter, we investigate the user social relations for improved item recommendation. Specifically, social resonance effect is explored and we further propose a first social resonance-aware recommender ResRec. ResRec can closely connect user social connections with user

interactions towards items through preference-based resonance and multi-hop relation-based resonance, to futher facilitate the prediction of social influence on user preference towards items.

- **Chapter 6: Instagrammers, Fashionistas, and Me: Recurrent Fashion Recommendation with Implicit Visual Influence.** In this chapter, we further consider the user social relations in sequential recommendation settings. Particularly, we focus on the social influence in fashion recommendation. We propose to consider the fashion focused key opinion bloggers and model their influence on user preference towards items. Concretely, we build a BiLSTM that integrates those fashion blogger posts to user purchase history, and find the influenced fashion bloggers to different users for further fashion recommendation. We also release a large scale time-aware visual dataset of further research.
- **Chapter 7: Content-Collaborative Disentanglement Representation Learning for Enhanced Recommendation.** In this chapter, we propose DICER, a novel disentanglement generative recommendation model that disentangled the content aware recommendation from two level: content-collaborative disentanglement and feature disentanglement, to address the duplication problem in content recommendation. Therefore, different types of features can be effectively learned for user and item representation learning and further give an improved recommendation.
- **Chapter 8: Cross Decoupling: Learning Item Embeddings based on Long-tail Item Distribution.** In this chapter, we propose CDN, a cross decoupling method to address the skewed long-tail distribution problem in recommendation. CDN decouples the memorization and generalization through a novel regularized Bilateral-Branch Network and multi-experts structure to separately consider the head and tail items in the long-tail item distributions. CDN brings significant recommendation improvement for both head and tail items.
- **Chapter 9: Conclusions and Future Directions.** We conclude the dissertation with a summary of contributions, and also discuss potential research extensions.

2. RELATED WORK

In this chapter, we highlight related work to the themes of this dissertation. We begin with a discussion of item-side and user-side relationships, and how these relationships can inform the design of content-based recommender systems. Then we examine decoupling and disentangling methods.

2.1 Item-side Relations in Recommendation

As we have discussed in the previous chapter, item relations vary greatly. Here we mainly discuss two important types of relations: complementary and substitute relations. We also highly how these relations can be impactful in sequential recommendation settings.

2.1.1 Item Relations based Recommendation.

Item recommendation often focuses on finding related items that are similar to an item of interest. The intuition is that a recommender should aim to find items that are very close to a target item (e.g., Harry Potter Book 2 is similar to Harry Potter Book 1). Such item-to-item recommendation often uses collaborative filtering [20–24] with similarity functions [25–27] such as Pearson similarity [28], cosine-based similarity [29], conditional probability-based similarity [30], or simultaneous regression [SLIM] [31]. Recently, Kabbur et al. [32] introduced a model called FISM that uses latent factor matrices to learn item-item similarity. Shambour [33] used Euclidean distance to measure item-item similarity and showed such a method is better than traditional similarity approaches. Li et al. [34] used the proportion of same users who rated items to measure item similarity. Moreover, many approaches seek to find similar items by incorporating user ratings [35–37] or images [16, 38, 39]. Similarly, context-based recommenders [40] and phrase-level sentiment analysis [41] have been proposed to capture additional item features for improved recommendation.

Recent research has focused on detecting relationships between items – such as substitutes or complements [10–12,42,43] – that go beyond traditional item similarity. Substitute items are those that are interchangeable, such as cameras in different brands. Complementary items are those that "go well" with one another. Examples include a camera that requires a specific lens or a laptop that works well with only certain chargers. Those relations can facilitate many recommendation applications, such as recommendations in different context – 'users who viewed X also viewed Y' (substitutes), 'users who bought X also bought Y' (complementary), as illustrated in Amazon and many other ecommerce platforms [11].

For example, [44] employed an association rule to find implicit relationships between items and used it as a regularization term in matrix factorization. [43] used user reviews to find relationships between items such as "albums that are similar with Taylor Swift's 1989". McAuley et al. showed how complementary fashion items like dresses and shoes can be recommended by projecting item images into a common visual style space [10]. Image-based recommendations are also discussed to discover substitutable items in a style space [11]. Another improved model based on images of items was proposed by He et al. [45], in which a mixtures-of-experts framework is built to model the relative importance of different image aspects. Most existing methods are based on a single source of item information – such as images or textual information. However, in practice, the item relationships are complex and the interaction of items in the relationship varies according to different categories. For example, style-based methods do well for clothing but not for books.

2.1.2 Item Relations in Sequential Recommendation

Sequential dynamics play a key role in many modern recommenders [9, 46–48]. Typically, there are two major types of sequence models: (i) the order-based models (shown in Figure 4.2 (a)) consider user sequences as *item orders* and focus on uncovering diverse patterns from these orders, such as using Markov Chains [49, 50], Recurrent Neural Networks (RNNs) [51–53], Convolutional Neural Networks (CNNs) [46, 47, 54] and attention mechanism [9]; (ii) the translation-based models (shown in Figure 4.2 (b)) treat user sequences as *user translation behavior* to connect items [55–57]. These translation-based methods can capture higher-order user-item interactions [55] and are more scalable compared with neural network-based models. However, most assume each user translation vector is static and identical, thus the translation behavior is the same across time; (iii) Our proposed model, HierTrans, as shown in Figure 4.2 (c), adaptively aggregates item high-order multi-relations at the category-level and dynamic user preferences at the item-level for next-item recommendation. Thus the translation vector can adaptively change.

Different types of item relations have gained attention to improve recommendation [58–60]. Recently, many works focus on those item relations due to their pervasive real-world application, such as inferring complementary and substitute relations based on content information since these relations are very sparse [8, 12, 61]. Here, we mainly focus on the category-level relations. Furthermore, few of these methods explore such complement and substitute relations for *dynamic* sequential recommendation.

Many research efforts have investigated graph structures [62] for link prediction [63–69]. Specifically, for translation-based models [70, 71], TransE [72] first proposed the core idea that items were connected by translation vectors in their vector space. The model structure is simple but achieves powerful performance in many situations. In follow-up work, various methods (such as TransH [73] and TransR [74]) extend TransE. Different from these methods that are mainly based on *generic* graphs, we investigate the specific structure of a user's dynamic sequence (which is a path) inside an item's heterogeneous relational graph.

2.2 User-side Relations in Recommendation

In this section, we discuss about the user-side relations, which is known as the social connections. Especially, we explore one widely used social effect – social resonance – and also discuss the social influence in sequential recommendation settings – known as the fashion recommendation.

2.2.1 Social-aware Recommendation

Social-aware recommendation is based on the common assumption that users can be influenced through their social connections to have similar preferences [75–78]. For example, Reshma *et al.* [79] considered directed and transitive trust relations among users to overcome the data sparsity problem in recommendation; Wu *et al.* [80] investigated user influence propagation inside the social network to improve the recommendation performance.

Recently, many studies further explore different social effects in a social network to enhance the learning of social influence. For example, Wu *et al.* [13] explored the social homophily properties to more accurately estimate the social influence for user preference towards items; Wang *et al.* [14] investigated the social network property that users can only accept a limited amount of social information, thus only focused on a subset of friends to improve recommendation. However, most of those methods mainly focus on the internal properties of the social network (e.g., social homophily), ignoring the mutual social resonance effect and its reinforced influence on user preference towards items. Furthermore, they usually directly estimate the user social influence on the user preference towards items (e.g., through an attention mechanism), with less consideration of the high heterogeneity between user-user social connections and user-item interactions. Both of them could limit the learning of user social network influence on user preference towards items.

In recent years, many studies have shown that users who have rated or reviewed an item could heavily influence the preferences of other users towards the item, especially when the explicit social network is absent [81–84]. For example, Amazon users may refer to the previous ratings and reviews of an item to help make a purchase decision. We refer to such an influence network for each item as the item-aware user influence network. Exploiting the latent influence in item-aware user influence networks has attracted significant attention in recommendation. For example, Mukherjee *et al.* [83] utilized users who rate the same item to explore the influence in the item-aware user influence network and showed great improvement in recommendation. Guo *et al.* [81] empirically compared different ways to infer implicit trust among users. Lin *et al.* [85] illustrated how potentially influential experts can implicitly influence user preference. However, few of these works consider the connections between the item-aware user latent influence and the user social network, which we find especially helpful to bridge the gap between user-user social connection and user-item interactions to estimate user preference towards items.

2.2.2 Fashion Recommendation.

With the rapid expansion of online shopping for fashion, recommending personalized fashion items has gained increasing attention [15, 86–90]. Different from traditional item-based recommendation, visual information plays a significant role in fashion recommendation [15, 91]. For example, He *et al.* [15] extracted fashion trends from user's purchase history and built a visual-time aware matrix factorization to recommend clothing. Jagadeesh *et al.* [16] built a visual-aware complementary recommender to find items of a similar style based on the user's purchase history. Recently, Yu *et al.* [91] used aesthetic visual features extracted from the AVA activities dataset [92] to improve Amazon clothing recommendation. Gabale *et al.* [88] explored community influence on fashion trends and identified the importance of social media on fashion evolution. Our work exploits trends revealed through fashion bloggers, in contrast to most existing approaches that use purchase history [15, 16] or static visual datasets [91].

Since fashion evolves over time, time-aware recommendation can be used to model user and item temporal dynamics [93–96]. Considerable prior works focus on RNN-based models in these cases. For example, Wu *et al.* [94] built a recurrent recommender network that can achieve high performance with fewer parameters for rating recommendation. Beutel *et al.* [96] used a latent cross recurrent neural model to effectively model contextual information in neural recommender systems. Hidasi *et al.* [53] used a session-based RNN recommender to achieve an improvement for implicit recommendation. Ko *et al.* [97] proposed a collaborative sequence model based on RNNs to capture a user's contextual state as a personalized hidden vector. Sun *et al.* analyzed the importance of user social dynamic influence and built a recurrent recommender with utilizing user explicit static social network. Different from these works, considering the significant importance of key fashion opinion leaders (fashion bloggers) in the fashion area [98, 99], we focus on fashion bloggers and use the implicit influence of their visual posts as a dynamic fashion signal for user clothing recommendation.

Previous research has shown that fashion bloggers can influence fashion preferences, and even directly influence user purchase preference, especially for young women [98–100]. For example, Vineyard [101] examined the relations between fashion bloggers and consumer purchase (e.g. "I buy one or more products which I have browsed on a blog") and the results show they are strongly positively connected (Cronbach's $\alpha = 0.931$). Zain [102] interviewed consumers who had commented on fashion blogs, finding that their purchase preferences are strongly influenced by fashion bloggers and their posts. Marwick [100] interviewed fashion bloggers to show the high aesthetic quality of their posts and their commercial value. McQuarrie *et al.* [98] highlighted the influence of fashion bloggers on consumption. Among those work, Instagram is regarded as the platform with the largest number of influential fashion bloggers with a large reach [103]. Many brands specifically utilize Instagram to promote their clothing [104], with around £1 billion spent per year to sponsor Instagram posts [99]. We see many commenters show their strong willingness to buy similar clothing as the blogger posted.

2.3 Content Disentanglement and Decoupling

Previous sections mainly focus on integrating the content information to enhance the recommendation. However, simply aggregating content information usually suffer suboptimal problem. In the following, we give a detailed discussion and show the recent proposed decoupling and disentanglement techniques.

2.3.1 Disentanglement Learning in Latent Factor-based Recommendation

One of a main approach for content-aware recommendation is latent factor-based models. They (e.g., matrix factorization and recent neural approaches) [105–110] typically map both users and items to a latent factor space with a low latent dimension. However, since only user-item interactions are considered, these models usually suffer from sparsity and cold-start problems. Thus, latent factor models for recommendation have been augmented to incorporate additional content information of items and users [4,50,111–113]. For instance, Li et.al. [114] combined item content information and user-item feedback into a variational autoencoder to learn user preference towards items. Lv et al. [115] proposed a multimodal item similarity-based framework that learned visual and textual features for recommendations. However, most of these existing content-based latent factor models extract the content and collaborative features independently and then simply concatenate them to learn representations for users and items, without fully considering the correlation between user-item feedback and content information. For example, VBPR [4] directly concatenates the visual features learned from item images with collaborative features as the joint item representation and multiplies it with user representations to predict the user preference. This learned representation may contain duplicated/highly correlated features between images and collaborative features, leading to less robust models of user preference.

Recently, disentanglement representation learning has attracted increasing attention due to its robust performance and interpretability [116–119]. Disentanglement learning aims to identify each feature that is relatively not influenced by other feature changes. For example, disentanglement learning on a visual dataset might learn the shape, the color, and the position features of the object [120], where each feature is not easily influenced by other feature changes. One popular method to capture disentangled features is based on *statistical independence* [121], which has demonstrated good performance in many applications [122, 123]. For example, β-VAE shows that disentanglement can be achieved if the KL term in the evidence lower bound (ELBO) is highly penalized. Based on that, Chen *et al*. [121] introduced β-TCVAE which provides a further decomposition of the ELBO to explain the penalty of KL for feature disentanglement. For recommendation, Ma *et al*. [124] learned disentangled latent representations for users and items based on user-item feedback and showed great improvement in recommendation.

2.3.2 Decoupling Learning for Long-tail Distributions

The large-scale datasets usually exhibit long-tail distributions, which poses many critical challenges to many tasks in different areas [125–128]. Previous works mainly focus on the three directions: re-sampling, losses engineering and transfer learning. Re-sampling aims to create a more balanced data distribution during training, including over-sampling for tail classes and under-sampling for head classes. However, those methods could easily cause over-fitting/under-fitting problem to tail/head classes. Losses engineering aims to modify the loss functions, such as refining the weighting or adding regularization [129], to put more focus on the training of tail classes. For example, the widely used logQ [130] adds more weights to tail samples based on the class frequency. However, Those methods is argued to cause unstable training for the highly skewed datasets [125]. Another way to deal with the long-tail distribution problem is to transfer the learned knowledge from head classes to tail classes, such as transferring the semantic features [131] and learning the relation types from head to tail classes [132]. There are also many methods incorporate the meta learning with transfer learning to better learn the connections between head and tail classes, such as meta-transferring the model parameters [133] and learning weight differences in domain adaption [134]. Recently, studies found that those jointly learning re-balancing framework has different effects on representation and classification. Based on that, they [18, 135] proposed a decouple learning mechanism that separately consider the representation learning and classification. Those methods achieves powerful performance in image classification and recognition tasks.

In recommendation, the long-tail distribution problem has attracted many attentions [136, 137], especially for the item side with the rapid increase of number of items. Traditional methods usually separately consider the tail items as the cold-start items [138, 139]. They focus on investigating efficient ways to incorporate content information, such as transfer learning from different domains [140], graph learning by leveraging knowledge graph [69] and meta learning [141, 141]. Recently, there are increase efforts focus on the long-tail distribution, instead of purely consider the tail items, like adding more weights to tail items [142], linking different long-tail distributions [143], and transfer learning from head items to tail items [144, 145]. However, those works jointly train user and item embedding learning and their final recommendation, which makes it unclear how the long-tail distributions influence the recommendation performance. Therefore, we study the potential opportunities to decouple the learning in recommendation to deal with the long-tail distribution problem in recommendation.

The interplay between memorization and generalization has been widely studied in different areas [146, 147]. They are closely related to feature mining, model design and model performance. In recommendation, according to [148], the memorization refers to the learning of co-occurrence in historical datasets, and generalization is to learn the new feature combinations which rarely occurred in the historical dataset. The wide and deep learning [148] shows the importance of achieve both the memorization and generalization, and the wide and deep network has achieved great success in real world application.

3. QUALITY-AWARE NEURAL COMPLEMENTARY ITEM RECOMMENDATION*

In this chapter, we tackle the challenges of sparsity of item relations by mining item complementary relations and giving a complementary item recommendation. We propose a new neural complementary recommender ENCORE that can jointly learn complementary item relationships and user preferences. Specifically, ENCORE (i) effectively combines and balances both stylistic and functional evidence of complementary items across item categories; (ii) naturally models item latent quality for complementary items through Bayesian inference of customer ratings; and (iii) builds a novel neural network model to learn the complex (non-linear) relationships between items for flexible and scalable complementary product recommendations. Through experiments over large Amazon datasets, we find that ENCORE effectively learns complementary item relationships, leading to an improvement in accuracy of 15.5% on average versus the next-best alternative.

3.1 Introduction

Complementary items that "go well" with one another abound. Examples include a camera that requires a specific lens or a laptop that works well with only certain chargers (see Figure 3.1). While these complements are strictly compatible – that is, they have particular requirements that allow them to work together – other complements are more loosely related. For example, an aesthetically matching shirt and pants outfit. Different from substitutes items that are interchangeable, complementary items are those that might be purchased together [10].

And yet, it can be challenging to identify complementary items, especially considering large and varied item populations (e.g., Amazon boasts around 500 million unique items). For example, one method is to first find exactly compatible items [45]. However, a sample of 500,000 items from Electronics on Amazon finds only 20% explicitly mention compatibility with other items [10, 12], with even rarer occurrences of such mentions in categories like books, movies, and fashion. Hence,

*Reprinted with permission from "Quality-Aware Neural Complementary Item Recommendation" by Yin Zhang, Haokai Lu, Wei Niu, James Caverlee, 2018. Proceedings of the 12th ACM Conference on Recommender Systems. Copyright 2018 by ACM. DOI:https://doi.org/10.1145/3240323.3240368 in this chapter we aim to create new methods for *complementary item recommendation* that can uncover complementary items across items and categories.

In particular, we identify three critical challenges for accurately recommending complementary items: First, the dimensions of how items complement each other vary by item and by category. For example, previous work has shown how to uncover complements based on visual style [11], but some items match based on size or on specific common interfaces. Identifying these features requires adaptive methods that can integrate multiple (possibly conflicting) sources of evidence like images and product descriptions. Second, even if a set of complementary candidate items can be identified, which ones will actually be preferred by users? For example, dozens of iPhone adapters may be identified as complements to an iPhone, meaning that complementary item recommenders must carefully model item quality to discern user preferences. Third, complementary relationships among items are complex, with potential non-linear relationships among item features and item quality. Yet, many existing methods rely on linear combinations of single source features (like visual style) [10–12], meaning that adapting these methods to complementary items may lead to poor performance.

With these challenges in mind, we address the problem of uncovering complementary item relationships through the creation of a new Neural COmplementary REcommender called ENCORE. The proposed model is characterized by three unique features:

- ENCORE can effectively model both stylistic and functional evidence of complementary items through careful balancing of high-level visual features learned by a convolutional neural network and text-based embeddings of titles and descriptions;
- ENCORE naturally models latent item quality through Bayesian inference over user ratings, leading to an item-relationship based quality-aware ranking method; and
- ENCORE builds a novel neural item-relationship based model to learn the complex complementary relationships between items. That is, the interplay of style, function, and quality can be learned for different categories of items, leading to more flexible and scalable complementary item recommendations.

Through experiments over large Amazon datasets, we quantitatively and qualitatively evaluate the performance of ENCORE versus a suite of state-of-the-art baselines. We find that ENCORE effectively learns complementary relationships between items, leading to an improvement in accuracy of 15.5% on average versus the next-best alternative across multiple categories of items. We further evaluate how different aspects of the model (e.g., images, text, ratings, neural recommendation) impact the final complementary item recommendation in different categories. We also show examples to illustrate the recommended items by ENCORE. Ultimately, we find that ENCORE's careful combination of different sources of complementary evidence is necessary for high-quality recommendation.

3.2 Overall Approach: ENCORE

We assume we have a set of items $\mathcal{I} = \{I_1, I_2, ...I_{|\mathcal{I}|}\}$ and sets of links $\mathcal{C}_i = \{l_{ic_1}, l_{ic_2}, ...l_{ic_{|\mathcal{C}_i|}}\}$, $i \in \{1, 2, ...|\mathcal{I}|\}$ that describe relationships between a query item I_i and its complementary items $I_{c_1}, I_{c_2}, ..I_{|\mathcal{C}_i|} \in \mathcal{I}$. Inspired by [11], our goal is to design a *complementary distance function* $d_{j|i}(I_i, I_j)$ that captures a user's preferences for complementary items given $I_i \in \mathcal{I}$. Specifically, we propose the quality-aware Neural Complementary Item Recommendation framework ENCORE that decomposes the problem into three phases (see Figure 3.2):

- **Detect Complementary Items.** First, we aim to construct a distance function $d^{(c)}_{j|i}(I_i, I_j)$ that assesses how well an item $I_j \in \mathcal{I}$ complements the seed item I_i based on stylistic properties (via the embedding $\mathbf{E}_M$) and functional properties (via the embedding $\mathbf{E}_T$).

- **Quality-Aware Recommendation.** Second, we augment the first distance with a quality-aware distance $d^{(r)}_{j|i}(I_i, I_j|\theta_j)$ to capture user preferences. Specifically, We show how to estimate the latent item quality θ_j and then asymmetrically incorporate it for ranking candidate complementary items (i.e. given an item I_i, find the nearest high-quality complementary items I_j).

- **Transform via Neural Model.** Finally, to capture the complex relationships between item properties and ratings, we build a neural-based distance $d^{(n)}_{j|i}(I_i, I_j)$ that can jointly learn complementary item relationships ($d^{(c)}_{j|i}(I_i, I_j)$) and user preference in $d^{(r)}_{j|i}(I_i, I_j|\theta)$, leading to high-quality complementary item recommendation.

3.2.1 Detecting Complementary Items

In this section, we focus on detecting complementary items from two perspectives: style and function. Our aim is to construct a distance function $d^{(c)}_{j|i}(I_i, I_j)$ that balances these two perspectives across different categories. For example, complementary fashion items may mainly match on style (that is, they go well visually with each other). In contrast, a Mac Pro and its charger need to functionally match based on a common interface (that is, the charger needs to fit specifically with the laptop, regardless of style). In practice, these notions of style and function vary across categories and can both be necessary in many cases. For example, while complementary fashion items may need to be stylistic matches, they also need to have similar functional sizes (e.g., identifying a woman's shirt and not one for a toddler). Ultimately, we propose a joint embedding model that captures both perspectives and the model can be customized for different product categories.

Style-Based Complements. As the first step, we exploit the image-based relationship between complementary items to find stylistically-related items. Following [11], we first use the high-level visual features extracted from a convolutional neural network (CNN) proposed by [149]. The CNN is pre-trained by Caffe 1.2 million ImageNet (ILSVRC12 challenge). Particularly, the features that we use are the output of the second fully connected layer in CNN based on their strong performance in previous work [11, 45], and the feature vector length is $f_m = 4096$. After extracting high-level image features, we can learn a low-rank Mahalanobis transformation for image embedding [11] (we refer to this method as **LMT**) and then calculate the Euclidean distance between the high-level image feature vector $\mathbf{m}_i$ and $\mathbf{m}_j$ in the embedding space. The image distance is used to represent the distance between items I_i and I_j, that is:

$$d^{(cm)}_{j|i}(I_i, I_j) = ||(\mathbf{m}_i - \mathbf{m}_j)^T \mathbf{E}_M||_2^2, \tag{3.1}$$

where $\mathbf{E}_M \in R^{f_m \times f_{em}}$ is the low-rank Mahalanobis transformation matrix and f_{em} is the embedding dimension of image. Based on the distance, a shifted sigmoid function is used to calculate the probability that two items belong to a certain relationship:

$$P(l_{ij} \in \mathcal{C}_i) = \sigma(-d^{(cm)}_{j|i}(I_i, I_j)) = \frac{1}{1 + e^{d^{(cm)}_{j|i}(\mathbf{I}_i, \mathbf{I}_j) - \eta_d}}.$$

Based on this probability, we can use maximum likelihood to train $\mathbf{E}_M$ so that it can identify style-based complements [11].

Functional Complements. Such an image-based approach is well suited for fashion-related items that demonstrate clear visual style. However, since it relies solely on image-based features, there may be significant errors introduced for complementary relationship when it is applied to other product categories. For example, Figure 3.3 shows several items that can confuse image-only approaches, such as a Panda USB battery and Baymax flash drive which are complementary with laptops. If they are mis-classified as toys according to their visual appearance, they will be deemed complementary with other toys rather than a laptop. And even when the images themselves are identical (as in Figure 3.3), an image-only recommender could mistakenly recommend uncomplementary MacBook Pro chargers for a MacBook Air.

Since the complement criteria varies across products (recall Figure 3.1), instead of using existing methods to find functional topics of each product, we propose to directly learn the functional complementary features.[*] Specifically, we propose to exploit text-based embeddings which can model these more nuanced relationships. That is, we aim to find a compatible text distance $d^{(ct)}_{j|i}(I_i, I_j)$ by $\mathbf{t}_i$ and $\mathbf{t}_j$, where $\mathbf{t}_i$ includes the title and description of item I_i.

Based on that, we propose to extract $\mathbf{t}_i$ by using a distributed representation [150, 151]. Specifically, we train a representation with a window size of 20 and learning rate 0.1. The final representation is a fixed-length feature vector $\mathbf{t}_i \in R^{f_t}$. Through experimental validation, we find that different dimensions of the text vector, such as 4096, don't strongly impact the overall results, so

[*]One initial idea is to mine mentions of complements directly from the text of each product description – e.g., to seek phrases such as "this charger is compatible with MacBook Pro". However, only 20% of products in Electronics (99,304/498,196) contain such explicit mentions [10], with even rarer occurrences of such mentions in categories like Digital Music and Clothing. The method also can not cover all situations for complementary relationships.

we use $f_t = 100$. So the distance between items I_i and I_j is calculated by:

$$d^{(ct)}_{j|i}(I_i, I_j) = ||(\mathbf{t}_i - \mathbf{t}_j)^T \mathbf{E}_T||_2^2. \tag{3.2}$$

where matrix $\mathbf{E}_T \in R^{f_t \times f_{et}}$ is the trained text embedding to learn text features that are related to the complementary relationship and f_{et} is the embedding dimension of text.

3.2.2 Quality-Aware Recommendation

By carefully combining $d^{(cm)}_{j|i}(I_i, I_j)$ and $d^{(ct)}_{j|i}(I_i, I_j)$, we could immediately begin to recommend the nearest complementary items. E.g., we could combine the two factors as $d^{(c)}_{j|i}(I_i, I_j) = qd^{(cm)}_{j|i}(I_i, I_j) + \rho d^{(ct)}_{j|i}(I_i, I_j)$, where q and ρ are hyper-parameters. In practice, however, users may choose a relatively high-rated complementary item [152], rather than the strictly nearest complementary one. For example, Figure 3.4 shows three complementary pants for a user who bought a Bella Ladies hoodie. The nearest pants – (A) and (B) – found by $d^{(c)}_{j|i}(I_i, I_j)$ are from the same fashion line Bella Ladies. However, the user's actual choice is (C), a pair of Spandex pants that are more distant by $d^{(c)}_{j|i}(I_i, I_j)$ but that are rated more highly than (A) and (B) (as shown on the right of Figure 3.4).

Therefore we hypothesize that these complementary items purchase decisions are driven by both perceived match (stylistic and functional) and by item ratings. However, in practice, item ratings are noisy and the number of ratings is different across items, which makes it hard to capture user purchase preference. So we propose to model each item's latent quality through careful consideration of item rating distributions [44, 152, 153]. Concretely, we model *item latent quality* as the expectation θ that a user will highly rate an item, and so it may be easier to capture user purchase preference. Then based on this θ, we propose a quality-aware distance function $d^{(r)}_{j|i}(I_i, I_j)$ that can provide rich user preference information. In the following, we first show an example (Figure 3.5 and 3.6) to illustrate why Bayesian inference is preferred to estimate θ here. Then we discuss details of θ_i estimation and build a quality-aware complementary distance $d^{(r)}_{j|i}(I_i, I_j|\theta_i)$.

Figure 3.5 shows ratings for three items. Item 1 and item 2 have same average ratings, while

Table 3.1: Notation of ENCORE.

Notation	Explanation
$\mathcal{I}$	item set, where $\mathcal{I} = \{I_1, I_2, ...I_{\lvert\mathcal{I}\rvert}\}$
$\mathbf{m}_i$,$\mathbf{t}_i$	image/text feature vector for item I_i
r_{ik}, q_{ik}	the k_{th} rating for item I_i and its binary
η_d, η_r	the thresholds for distance and ratings
θ_i	the expected value of q_{ik} after observing ratings
l_{ij}	the relationship (link) between two items $\mathbf{I}_i$ and $\mathbf{I}_j$
$\mathcal{C}_i$	the set of items that are complementary with I_i
$\mathbf{E}_M$, $\mathbf{E}_T$	embedding matrix for image and for text
$\mathbf{W}_k$,$\mathbf{b}_k$	neural network weight matrix/bias term in layer k
$\mathbf{d}^{(n)}_{j\mid i}(I_i, I_j\mid\theta_j)$	neural item distance from query item I_i to I_j

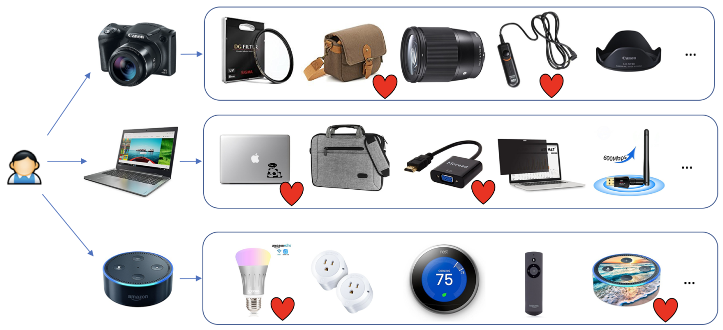


Figure 3.1: Complementary item examples and high-quality complementary items (with red mark).

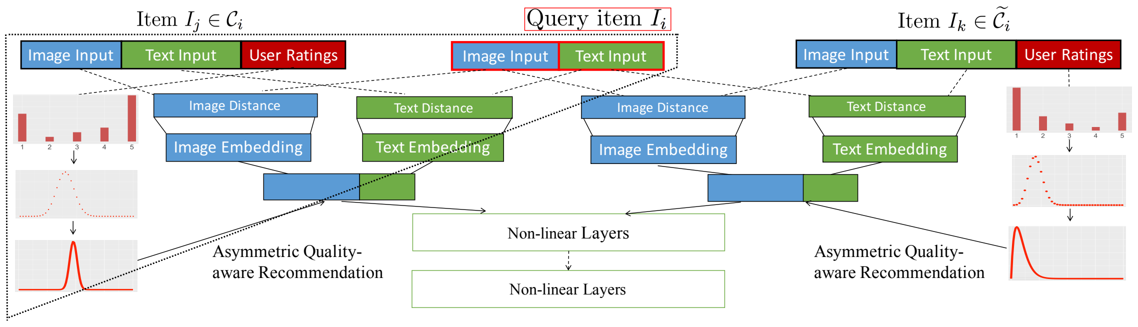


Figure 3.2: Overall ENCORE model framework.

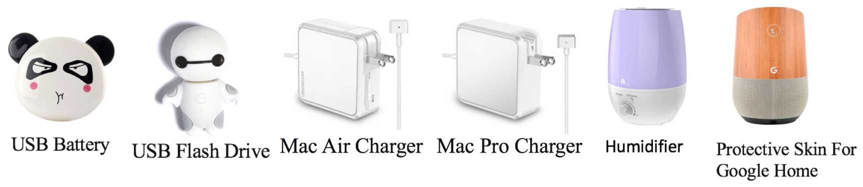
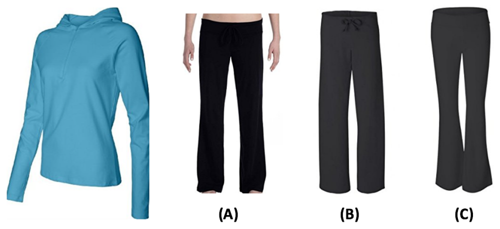
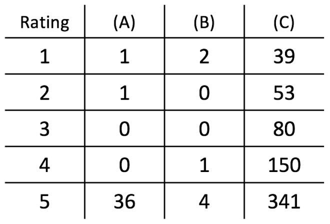


Figure 3.3: Image-confusing items.





Rating	(A)	(B)	(C)
1	1	2	39
2	1	0	53
3	0	0	80
4	0	1	150
5	36	4	341

Figure 3.4: A "Bella Ladies" hoodie and three complementary pants (A,B,C) with their ratings (on a 1-5 scale).

item 2 has been rated many more times (yielding more confidence in its underlying quality). So there is high probability that users would prefer item 2 than item 1. By Bayesian inference, the posterior distributions for the three example items are shown in Figure 3.6. We find that the posterior rating distribution for item 2 is more narrow and close to 1 while the distribution of item 1 is more spread out, which means the posterior rating distribution can properly indicate users have a higher probability to highly rate item 2 than item 1. So we leverage it to estimate θ_i. Concretely, the steps for generalizing $d_{j|i}^{(r)}(I_i, I_j|\theta_i)$ are:

Ratings-Based Bayesian Inference. First, suppose item I_i has ratings $r_{i1}, r_{i2} ... r_{i|I_i|}$ by different users and we treat each $r_{ik} \in \mathbb{Z}$ as a random variable for product I_i ratings. Since users have different evaluation scales and there is no big difference between 4-star or 5-star when a 5-scale rating is used [154], we first smooth r_{ik} as a binary random variable q_{ik}. Let η_r be a binary threshold to separate good ratings and bad ratings. If $r_{ik} > \eta_r$, it means r_{ik} is a good rating; otherwise it means the user thinks there are drawbacks of item I_i. So the probability (q_{ik}) that "the k_{th} rating

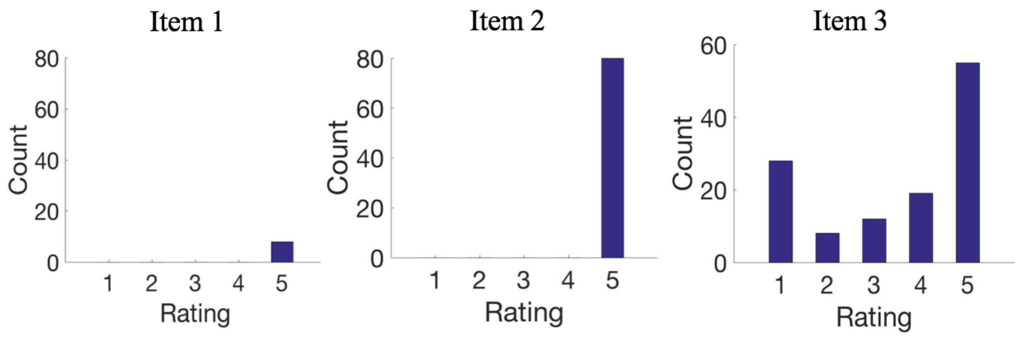


Figure 3.5: Ratings distributions for three example items.

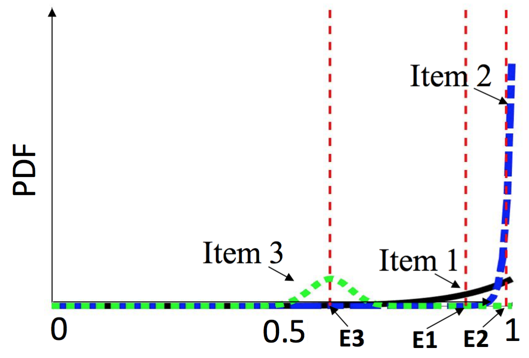


Figure 3.6: Posterior distribution for the example items. Ei is the expectation of the i^{th} item.

of item I_i is a good rating" can be represented as:

$$q_{ik} = \begin{cases} 1 & \text{if } r_{ik} > \eta_r \\ 0 & \text{otherwise,} \end{cases}$$

and the p.d.f. of q_{ik} for item I_i is:

$$f(q_{ik}|\theta_i) = \begin{cases} \theta^{q_{ik}}(1-\theta)^{1-q_{ik}} & \text{for } q_{ik} = 0, 1 \\ 0 & \text{otherwise.} \end{cases}$$

Thus q_{ik} is Bernoulli distributed $q_{ik} \sim \mathbf{B}(1, \theta_i)$. The expectation that item I_i can get a good rating is $\mathbb{E}(q_i|\theta_i) = \sum_{k=1}^{\infty} q_{ik} f(q_{ik}|\theta_i) = \theta_i$ and $\theta_i \in [0,1]$ for each item I_i. So θ_i can be used to measure user expectations (refer as quality) towards item I_i. If the value of θ_i is high, it means there is a high probability that the item I_i can get a good rating (i.e. item I_i has high quality).

But how can we estimate θ_i? Many previous methods do not consider ratings for recommendation, so they assume the quality of each item is randomly distributed, that is $\theta_i \in U[0,1]$ for all $I_i \in \mathcal{I}$. In contrast, we use Bayes' theorem [153, 155] to estimate the posterior p.d.f of θ_i of item I_i based on users' ratings (as we previously illustrated), where this uniform distribution is treated as a prior for items when we have no rating information:

$$\begin{aligned}\xi(\theta_i|q_{i1}, q_{i2}...q_{i|I_i|}) &= \frac{f_i(q_{i1}, q_{i2}...q_{i|I_i|}|\theta_i)\xi(\theta_i)}{h_i(q_{i1}, q_{i2}...q_{i|I_i|})} \\ &\propto f_n(q_{i1}, q_{i2}...q_{i|I_i|}|\theta_i)\xi(\theta_i),\end{aligned} \tag{3.3}$$

where $h_i(\cdot)$ is the marginal joint p.d.f of $q_{i1}, ...q_{in_i}$. $\xi(\theta_i)$ is the prior p.d.f. of θ_i. Here it is the p.d.f of uniform distribution $U[0,1]$. $f_i(q_{i1}, q_{i2}...q_{i|I_i|}|\theta_i)$ is the likelihood function:

$$f_i(q_{i1}, q_{i2}...q_{i|I_i|}|\theta_i) = \prod_{k\in[1,...|I_i|]} f(q_{ik}|\theta_i) = \theta_i^{\sum_{k=1}^{|I_i|} q_{ik}}(1-\theta_i)^{|I_i|-\sum_{k=1}^{|I_i|} q_{ik}}. \tag{3.4}$$

Let $\Phi(q_{i1}, q_{i2}...q_{i|I_i|}) := \frac{\Gamma(|I_i|+2)}{\Gamma(\sum_{k=1}^{|I_i|} q_{ik}+1)\Gamma(|I_i|-\sum_{i=1}^{|I_i|} q_{ik}+1)}$, where $\Gamma(z) = \int_0^1 x^{z-1}e^{-x}dx$. According to Equation 3.3, the p.d.f of posterior distribution $\xi(\theta_i|q_{i1}, q_{i2}...q_{i|I_i|})$ is:

$$\xi(\theta_i|q_{i1}, q_{i2}...q_{i|I_i|}) = \Phi(q_{i1}, q_{i2}...q_{i|I_i|})\theta_i^{\sum_{k=1}^{|I_i|} q_{ik}}(1-\theta)^{|I_i|-\sum_{k=1}^{|I_i|} q_{ik}}.$$

So the posterior distribution of θ_i is a Beta distribution

$$\theta_i \sim \textbf{Beta}(\sum_{k=1}^{|I_i|} q_{ik}+1, |I_i| - \sum_{k=1}^{|I_i|} q_{ik}+1). \tag{3.5}$$

Based on the posterior distribution, the expectation of θ_i is $\mathbb{E}(\theta_i|q_{i1}, q_{i2}...q_{i|I_i|}) = \frac{\sum_{k=1}^{|I_i|} q_{ik}+1}{|I_i|+2}$, which we can use to estimate users expectation for item I_i (same as Figure 3.6 examples).

Recommendation with Asymmetric Ratings. In practice, users care more about the quality of a complementary candidate item I_j, rather than the quality of a query item I_i. So we consider the latent quality for item I_j in $d^{(r)}_{j|i}(I_i, I_j)$. Given I_j's quality estimate, the recommended item quality should be inversely proportional to item distances: the lower the quality of the candidate I_j, the larger the distance from the query item. That is:

$$d^{(r)}_{j|i}(I_i, I_j|\theta_j) \propto \mathbb{E}(1-\theta_j|q_{j1}, q_{j2}...q_{j|I_j|}) = \frac{|I_j| + 1 - \sum_{k=1}^{|I_j|} q_{jk}}{|I_j| + 2},$$

when item I_i is queried and I_j is recommended. But how do we incorporate $\mathbb{E}(1-\theta_i|q_{i1}, q_{i2}...q_{i|I_i|})$ to item relationship distance for our complementary recommendation?

3.2.3 Neural Recommendation

As shown in Figures 3.3 and 3.4, complementary relationships vary greatly across categories. Moreover, users may choose a relatively high quality complementary item rather than the strictly nearest complementary one according to previous analysis of Figure 3.4. Hence, instead of directly combining three sources of information, we propose a neural-based complementary recommender that can bring some attractive characteristics for complementary item recommendation: (i) Neural methods may capture the variability of complementary relationships for different categories; (ii) Neural models can also offer more flexibility in balancing the style/functional complementary matches with item quality through activations; and (iii) Many neural methods may be easily parallelized for scalable computation [156], which can be beneficial for large item populations with high-dimensional visual and text features. Concretely, we transform ENCORE's complementary distance into a non-linear space $d^{(n)}_{j|i}(I_i, I_j|\theta_j)$ to capture these complex complementary relationships (see Figure 3.2).

To calculate $d^{(n)}_{j|i}(I_i, I_j|\theta_j)$ between I_i and I_j, we first extract features of query item I_i and its candidates complementary item I_j in each space, then use embedding to learn visual and functional complementary features separately. Instead of directly using their distances, we concatenate the embedded features with the expectation quality of candidate item I_j into a multi-modal space. So ENCORE can learn complementary relationships by feature differences in each source:

$$\mathbf{c}_{j|i}(I_i, I_j|\theta_j) = [(\mathbf{m}_i - \mathbf{m}_j)^T\mathbf{E}_M, (\mathbf{t}_i - \mathbf{t}_j)^T\mathbf{E}_T, \mathbb{E}(1 - q_j|\theta_j))]^T,$$

where $\mathbf{c}_{j|i}(I_i, I_j|\theta_j) \in \mathbb{R}^{(f_{em}+f_{et}+1)}$. We use it as our asymmetric merged layer of our neural network (because we only consider the quality of candidate item I_j). Then with adding $\mathbf{W}_1 \in \mathbb{R}^{(f_{em}+f_{et}+1)\times m_1}$ and bias $\mathbf{b}_1 \in \mathbb{R}^{m_1}$ in the layer:

$$\mathbf{h}_{j|i}(I_i, I_j|\theta_j) = \mathbf{c}_{j|i}(I_i, I_j|\theta_j) \times \mathbf{W}_1 + \mathbf{b}_1.$$

Then we add the activation function and now the distance becomes:

$$d^{(n)}_{j|i}(I_i, I_j|\theta_j) = ||\tanh(\mathbf{h}_{j|i}(I_i, I_j|\theta_j))\mathbf{W}_2||^2, \tag{3.6}$$

where $\tanh(-\mathbf{h}_{j|i}(I_i, I_j|\theta_j)) = \frac{e^{-\mathbf{h}_{j|i}(I_i,I_j|\theta_j)} - e^{-\mathbf{h}_{j|i}(I_i,I_j|\theta_j)}}{e^{-\mathbf{h}_{j|i}(I_i,I_j|\theta_j)} + e^{-\mathbf{h}_{j|i}(I_i,I_j|\theta_j)}} \in \mathbb{R}^{m_1}$. And $\mathbf{W}_2$ in Equation 3.6 is a weight vector when features are put into non-linear space. So we can calculate the probability that item I_i and I_j are complementary when I_i is queried as: $P(l_{ij} \in \mathcal{C}_i) = \frac{1}{1+e^{d^{(n)}_{j|i}(I_i, I_j\theta_j) - \eta_d}}$, where η_d is a learned complementary threshold.

Based on the probability, we use the maximum likelihood function to find the maximum observed complementary relationship of set $\mathcal{C}_i$ for each I_i. Then the complementary relationship for the item set $\mathcal{I}$ is $\mathcal{C}_{\mathcal{I}} = \{\mathcal{C}_1, \mathcal{C}_2, ...\mathcal{C}_{|\mathcal{I}|}\}$. The log-likelihood function for all items in $\mathcal{I}$ is:

$$\begin{aligned} l(\mathbf{E}, \mathbf{W}, \mathbf{b}, \eta_d|\mathcal{C}_{\mathcal{I}}, \widetilde{\mathcal{C}_{\mathcal{I}}}) &= -\sum_{I_i\in\mathcal{I}}\sum_{l_{ij}\in\mathcal{C}_i} lnP(l_{ij} \in \mathcal{C}_i|I_i, I_j) - \sum_{I_i\in\mathcal{I}}\sum_{l_{ij}\in\widetilde{\mathcal{C}_i}} (1 - lnP(l_{ij} \in \widetilde{\mathcal{C}_i}|I_i, I_j))) \\ &= -\sum_{I_i\in\mathcal{I}}\sum_{l_{ij}} (y_{ij} lnP(r_{ij} \in \mathcal{C}|I_i, I_j) + (1 - y_{ij})(1 - lnP(l_{ij} \in \widetilde{\mathcal{C}}|I_i, I_j))), \end{aligned} \tag{3.7}$$

where y_{ij} indicates whether there is a complementary relationship between item I_i and I_j. If $l_{ij} \in \mathcal{C}_i$, then $y_{ij} = 1$; otherwise it is 0. In $l(\mathbf{E}, \mathbf{W}, \mathbf{b}, \eta_d|\mathcal{C}_{\mathcal{I}}, \widetilde{\mathcal{C}_{\mathcal{I}}})$, $\mathbf{E}$ represents $\{\mathbf{E}_M, \mathbf{E}_T\}$. $\mathbf{W}$

represents $\{\mathbf{W}_1, \mathbf{W}_2\}$. $\tilde{\mathcal{C}_{\mathcal{I}}} = \{\tilde{\mathcal{C}_1}, \tilde{\mathcal{C}_2}, ...\tilde{\mathcal{C}_{|\mathcal{I}|}}\}$. Each $\tilde{\mathcal{C}_i}$ is a randomly selected negative set of non-complementary items. To train parameters, we generate $\tilde{\mathcal{C}_i}$ such that $|\tilde{\mathcal{C}_i}| = |\mathcal{C}_i|$ [11].

3.3 Experiments

In this section, we evaluate ENCORE's complementary item recommendations over large Amazon datasets in comparison with state-of-the-art baselines. Especially, we seek to address the following key research questions:

- How well does ENCORE perform versus baselines? And does this performance vary by item types? And also across complementary relationships (i.e. also-bought versus bought-together)?
- What impact do the design choices of ENCORE have? For example, is textual-added complement more impactful than image-driven complement across different categories? What impact does the neural recommender model have versus a linear model for complementary recommendation?

Finally, we explore the complementary recommendations of ENCORE through several case studies.

3.3.1 Amazon Dataset

Concretely, we adopt a large real-world dataset from Amazon recently introduced in [11,45]. The complete dataset contains over 1 million products and 42 million co-purchase relationships across around 20 top-level product categories. We focus on six main categories that display different complementary aspects: Electronics, Cell Phones & Accessories (C & A), Clothing, Books, Digital Music, and Movies (see Table 3.2 for details) [11]. Specifically, following with previous work [11,12,45], we adopt two relationships in Amazon data: the "Bought-together (BT)" relationship, where users bought item I_i and I_j simultaneously, and the "Also-bought (AB)" relationship, where users who bought item I_i also bought I_j' [20].

3.3.2 Experimental Setup

We make recommendation based on a single category at a time. We use item title and description for the functional embedding, and item image for the style embedding (which was collected

Table 3.2: Amazon datasets. The second column is the number of subcategories. The Avg column is the average number of linked items for each query item. For example, users also-bought 32 complementary items on average in Electronics and bought-together 1.39 items on average.

Dataset	**# Cat.**	**Also-bought**			**Bought-together**		
		# Items	# Edges	Avg	# Items	# Edges	Avg
Digital Music	198	164,440	6,912,348	42	5,552	9,590	1.73
Movies	345	118,351	5,248,530	44	80,922	130,640	1.61
Cell phones & Accessory	81	122,031	2,985,220	24	100,567	138,815	1.38
Books	2,752	65,024	2,806,544	43	35,638	54,146	1.52
Electronics	786	140,922	4,446,609	32	140,020	194,309	1.39
Clothing	1,993	658,304	20,546,119	31	569,714	1,645,219	2.89

in [11]). For non-complementary item, we randomly select a negative set such that $|\widetilde{\mathcal{C}_i}| = |\mathcal{C}_i|$. All experiments are trained using Nvidia GeForce GTX Tian X GPU with 12GB memory and 3072 cores using Tensorflow. Since it takes around one week to train a model over the full dataset, we randomly select 11,000 items from each training set as query items to do the five fold cross-validation for model training. We find similar performance between models trained over the full data and this approach [11].

Baselines. We consider a suite of state-of-the-art baselines. To evaluate the model structure of ENCORE, for fairness, we extend each approach to be trained over the exact same input as ENCORE – images, product text, and ratings:

- *Logistic Regression with Average Rating (LR$_A$)*: Our first baseline is a straightforward application of logistic regression. We concatenate the differences of images, text, and ratings between queried item I_i and item I_j as input: $\mathbf{f}'_{j|i}(I_i, I_j|\theta_j) = [(\mathbf{m}_i - \mathbf{m}_j)^T, (\mathbf{t}_i - \mathbf{t}_j)^T, \mathbb{E}'(1-\theta_j)]^T$, and calculate the probability that two items are complementary. Here $\mathbb{E}'(1-\theta_j)$ is the average ratings without using Bayesian approach.
- *Logistic Regression with Bayesian Rating (LR$_B$)*: This variant is similar to the previous but uses the Bayesian ratings inference to find $\mathbb{E}(1-\theta_j|q_j)$ rather than the average ratings. Hence, the input is $\mathbf{f}_{j|i}(I_i, I_j|\theta_j) = [(\mathbf{m}_i - \mathbf{m}_j)^T, (\mathbf{t}_i - \mathbf{t}_j)^T, \mathbb{E}(1-\theta_j|q_j)]^T$.
- *Weighted Nearest Neighbor (WNN)*: This method uses a weighted Euclidean distance to mea-

sure complement between items I_i and I_j: $d = ||\mathbf{f}_{j|i}(I_i, I_j|\theta_j) \circ \mathbf{w}||_2^2$ where $\circ$ is Hadamard product and $\mathbf{w}$ is a weight vector.

- *Feedforward Neural Network (FNN)*: We use a 3-layer neural network to measure the non-linear relationships of complement, where the input is the same as in logistic regression LR$_B$. We use $tanh$ and $softmax$ as activation functions for the second and third layer. We set the hidden and final dimensions to 10 in keeping with the other methods.
- *Low-rank Mahalanobis Transform (LMT) [11]*: This state-of-the-art method uses low-ranked Mahalanobis embedding matrix parameters [11]. Whereas the original approach in [11] relies on images only, we adapt it to use images, text, and ratings. The distance between queried item I_i and item I_j is calculated as $d = ||\mathbf{f}_{j|i}(I_i, I_j|\theta_j)^T \times \mathbf{E}_f||_2^2$ where $\mathbf{E}_f$ is the low-ranked Mahalanobis transform matrix. We set the embedding dimension $K = 10$. [11] also further splits top-categories into smaller categories (such as splitting Clothing into Men's, Women's, Boys, Girls).
- *Monomer [45]*: Another state-of-the-art method – Mixtures of Non-metric Embeddings method [45] that learns low-rank embeddings to uncover different aspects of complementary distance and uses a mixture of experts to find the final complementary distance. We adapt the original method to consider images, text and ratings as input: $\mathbf{f}_i^T = [\mathbf{m}_i^T, \mathbf{t}_i^T, \mathbb{E}(1 - \theta_i|q_i)]$ rather than just images for each item I_i. The distance between queried I_i and I_j is calculated by $d = \sum P(n)d_n$ with mixture of weighted experts $P(n)$. $d_n = ||\mathbf{f}_i^T \mathbf{E}_{0f} - \mathbf{f}_j \mathbf{E}_{nf}||_2^2$ where $\mathbf{E}_{nf}$ is the n_{th} embedding matrix. Empirically, we set parameters $K = 10$ and $N = 3$.

Variations of ENCORE**.** In order to evaluate the impact of images, text, ratings, plus the appropriateness of adopting a neural approach, we consider several variations of our proposed approach:

- ENCORE$_{-M}$: This image-only method is based on [11] and uses the low-ranked Mahalanobis embedding matrix parameters in Equation 3.1. Note that we do not consider this refinement in any of the following alternatives.
- ENCORE$_{-MT}$: This method combines images and text, while ignoring ratings. The comple-

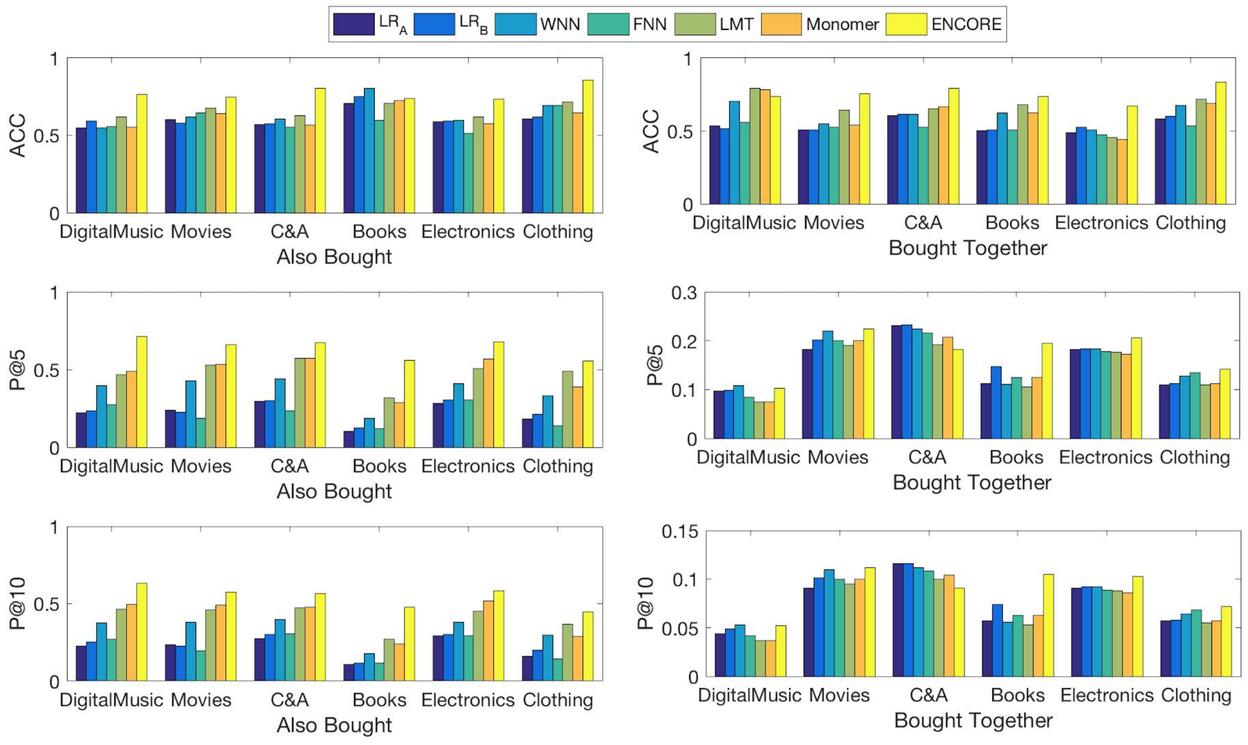


Figure 3.7: Accuracy and Precision of ENCORE and baselines.

mentary distance between I_i and I_j is $q||(\mathbf{m}_i - \mathbf{m}_j)^T\mathbf{E}_M||_2^2 + \rho||(\mathbf{t}_i - \mathbf{t}_j)^T\mathbf{E}_T||_2^2$. q and ρ is decided by cross-validation with grid search in the range of {0.1, 0.5, 1, 1.5, 10, 100} in each datasets.

- ENCORE$_{-MTCos}$: This method is a simplified version of the previous one, replacing the text-based embeddings with a simpler cosine-based approach over the original text itself.
- ENCORE$_{-MTR}$: This method considers images, text, and ratings, but uses a linear model: $d_{j|i}^{(r)}(I_i, I_j|\theta_j) = [||(\mathbf{m}_i - \mathbf{m}_j)^T\mathbf{E}_M||_2^2,\ ||(\mathbf{t}_i - \mathbf{t}_j)^T\mathbf{E}_T||_2^2,\ \mathbb{E}(1-\theta_j|q)]^T\mathbf{w}$, where $\mathbf{w} \in R^3$ is a model parameter to leverage the contributions of each source of information.
- ENCORE: Finally, we consider the full-blown ENCORE model that incorporates images, text, and ratings in a non-linear model as shown in Equation 3.6.

Metrics. For each method, we predict whether pairs of items are complementary or not, and

Table 3.3: Accuracy and precision increase of ENCORE comparing with the next-best alternative.

Δ	Digital Music		Movies		C & A		Books		Electronics		Clothing		Average
	AB	BT	AB	BT	AB	BT	AB	BT	AB	BT	AB	BT	
ACC	23%	-7%	10%	17%	29%	19%	-8%	9%	18%	40%	19%	16%	15.5%
P@5	46%	-5%	24%	2%	17%	-22%	76%	33%	20%	12%	13%	4%	18.4%
P@10	27%	-1%	18%	2%	18%	-22%	76%	42%	13%	12%	21%	6%	17.8%

measure the accuracy as:

$$ACC := \frac{\sum_i(\sum_j \mathcal{S}(P(l_{ij} \in C_i) - 0.5) + \sum_j \mathcal{S}(0.5 - P(l_{ij} \in \widetilde{\mathcal{C}}_i)))}{\sum_i(|C_i| + |\widetilde{\mathcal{C}}_i|)},$$

where $\mathcal{S}(\cdot)$ is a thresholding operator defined as: if $x > 0$, then $\mathcal{S}(x) = 1$; otherwise $\mathcal{S}(x) = 0$.

We also use Precision@k to measure the fraction of correctly predicted complementary items for each query item:

$$P@k := \frac{1}{|\mathcal{I}|}\sum_i \frac{|GT(C_i) \cap Pred(C_i)@k|}{k}, \tag{3.8}$$

where $GT(C_i)$ is the ground truth set of items that are complementary with I_i and $Pred(C_i)@k$ is the predicted top-k recommended complementary items.

Parameter settings. For all models, the image and text latent factor dimensions, output dimensions are set to 10 empirically for a trade-off between performance and computational complexity, as well as for fair comparison across methods. For ratings, the threshold is $\eta_r = 3$. Other parameters are fine-tuned for all methods. Particularly, in each experiment, five fold cross-validation is used. Model parameters are first randomly initialized according to truncated normal distributions with mean 0. The standard deviation is decided by grid search in $\{0.1, 0.01, 0.001\}$, and updated by conducting stochastic gradient descent (SGD). The corresponding learning rate is determined by grid search in the range of $\{0.1, 0.05, 0.01, ..., 0.000001\}$. Generally, training for different categories of items converges within 30 iterations.

3.3.3 Evaluating Complementary Recommendation

We begin by investigating the model quality of ENCORE versus each baseline. Since each approach is built over the same information – images, text, and ratings – we can explore how each approach models and combines these factors for complementary item recommendation. We report the accuracy, precision@5, and precision@10 in Figure 3.7 for all methods. Table 3.3 shows the increase of ENCORE comparing with the next-best alternative ("AB" means "Also Bought". "BT" means "Bought Together". Again, here we have modified these original methods to incorporate text and ratings, beyond their original image-only approaches).

Focusing on accuracy (the top row of Figure 3.7), we observe that ENCORE results in the highest accuracy across both also-bought and bought-together items for all categories except for Books and Digital Music, resulting in an average improvement versus the next-best alternative of 15.5% (ACC row in Table 3.3). Since Books and Digital Music demonstrate a fairly weak notion of compatibility (e.g., phone chargers match with specific phones, but books of course can be bought with any other books), we see that ENCORE has difficulty, though performing as well as other sophisticated models like LMT and Monomer. Additionally, ENCORE outperforms the next-best alternative of the state-of-the-art LMT and Monomer by 16.9% on average. Since all methods consider same information, these results show the structure of ENCORE introduced via our neural framework results in an even greater improvement. We also observe that LR_A outperforms LR_B, which indicates the number of ratings closely influences a user's preference for complementary items (as in Figure 3.4).

Next, we focus on precision – see the middle and bottom rows of Figure 3.7, and increase of ENCORE comparing with the next best baselines in Table 3.3 row P@5 and P@10. We observe that for all also-bought categories, ENCORE results in the highest precision@5 and precision@10. ENCORE improves versus the next-best alternative an average of 18.4% for precision@5 and 17.8% for precision@10, and versus the best state-of-the-art alternative an average of 27.5% for precision@5 and 26.9% for precision@10 shown in Table 3.3. Here, we see further evidence of the importance of low-rank embedding and neural transformation in comparison with models like Logistic Regression and Weighted Nearest Neighbors. And for those models that do consider those factors, we see the importance of careful modeling of ratings and integrating each sources information separately in a lower rank non-linear spaces. Observe that precision values are low for all methods in bought-together categories – the data in this case is extremely sparse, with most items having fewer than three ground truth items in the complementary set.

3.3.4 Impact of Encore Model Choices

Given the good performance of ENCORE versus baselines, what impact do the specific design choices have on complementary item recommendation? Does the functional complement derived from text improve upon image-only approaches? Does adding a ratings-based recommender improve the quality of prediction? And what impact does the neural approach have? To anwser those questions, we focus here on accuracy as shown in Table 3.4; note that similar results hold over precision@5 and precision@10. We additionally report the relative improvement versus the image-only $Encore_M$.

Table 3.4: Prediction accuracy of Encore variations. Δ is the change in accuracy compared with Encore$_{-M}$. Encore outperforms the other methods in each experiment for both also-bought and bought-together relationships.

Dataset		ENCORE$_{-M}$	ENCORE$_{-MT}$		ENCORE$_{-MTR}$		ENCORE	
		ACC	**ACC**	Δ	**ACC**	Δ	**ACC**	Δ
Digital Music	AB	0.661	0.660	0%	0.755	14%	**0.763**	15%
	BT	0.525	0.693	32%	0.722	37%	**0.738**	40%
Movies	AB	0.686	0.722	5%	0.733	7%	**0.746**	9%
	BT	0.680	0.736	8%	0.748	10%	**0.756**	11%
C & A	AB	0.780	0.791	1%	0.797	2%	**0.806**	3%
	BT	0.722	0.749	4%	0.784	9%	**0.791**	10%
Books	AB	0.702	0.712	1%	0.725	3%	**0.738**	5%
	BT	0.580	0.706	22%	0.726	25%	**0.737**	27%
Electronics	AB	0.713	0.703	−1%	0.712	0%	**0.733**	3%
	BT	0.503	0.583	16%	0.623	24%	**0.670**	33%
Clothing	AB	0.844	0.845	0%	0.855	1%	**0.855**	1%
	BT	0.757	0.810	7%	0.827	9%	**0.833**	10%

Overall, we see the full-blown ENCORE improves upon all of its variations across all categories, with an average accuracy of 14.0%.

From column ENCORE$_{-MT}$ in Table 3.4, Text-based evidence is a strong indicator of functional complement in addition to what images can provide, especially in Books (Δ 22%), Digital Music (Δ 32%), and Electronics (Δ 16%) for the bought-together relationship. For also-bought, the relative accuracy improvement is smaller, with Electronics being the one category with worse accuracy (Δ -1%).

Our careful modeling of ratings makes a key positive impact for both also-bought and bought-together items. The accuracy improvement of ENCORE$_{-MTR}$ shows that Electronics, Cell Phones & Accessories, and Digital Music are all highly influenced by user ratings. Indeed, we calculate the average rating for predicted complementary items in Electronics and find that ENCORE$_{-MTR}$ recommends items with ratings higher than ENCORE$_{-M}$ by 4.6%, ENCORE$_{-MTCos}$ by 6.1% and ENCORE$_{-MT}$ by 1.2%.

Finally, we see that the non-linearity of ENCORE plays a significant role to identify complementary relationships. On average, ENCORE results in an improvement of 6.29% for also-bought and 21.91% for bought-together in comparison over $Encore_M$. The impact is especially large in the Electronics categories since the complement relationship is quite complex as discussed.

3.3.5 Encore Recommendations

We also generate predictions by ENCORE for several other items to give additional insights. For a domain like electronics, we see in Figure 3.8 that ENCORE generates different recommendations for different query items. For example, for the computer in first row, it recommends a keyboard cover, laptop sleeve, and external DVD writer. For an iPhone 5 in last row, ENCORE can recommend iPhone 5 screen protectors and cases.

3.4 Summary

In this chapter, we have focused on finding "complementary" relationships of items based on user preferences. We proposed a new neural item relationship-based recommender – ENCORE – which carefully combines multiple sources of complement evidence. We saw how stylistic complements (via images) and functional complements (via text-based titles and descriptions) could be

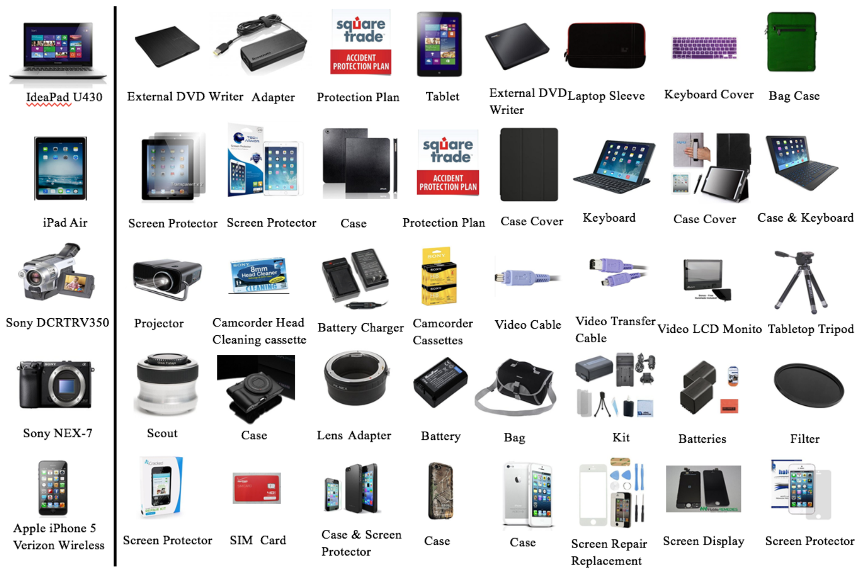


Figure 3.8: Encore predictions examples for a computer, iPad Air, Camcorder, Camera and iPhone 5. Query items are to the left of the line. Predictions are on the right.

combined in a quality-aware framework for uncovering high-quality complementary recommendations. Quantitative and qualitative results show that ENCORE improves upon a state-of-the-art baseline by 15.5% on average, even when all models are built over the exact same input. In our continuing work, we are interested in personalizing ENCORE for considering individual user personality, in addition to the aggregate perspective in the current version. We are also interested to explore more nuanced models of functional complements to improve the quality of our recommendations.

4. ADAPTIVE HIERARCHICAL TRANSLATION-BASED SEQUENTIAL RECOMMENDATION[†]

By mining the item relations, we could get a better understanding about the item connections. In this chapter, we consider the time factor to integrate item relations with user sequences. We propose an adaptive hierarchical translation-based sequential recommendation called HierTrans that first extends traditional item-level relations to the category-level, to help capture dynamic sequence patterns that can generalize across users and time. Then unlike item-level based methods, we build a novel hierarchical temporal graph that contains item multi-relations at the category-level and user dynamic sequences at the item-level. Based on the graph, HierTrans adaptively aggregates the high-order multi-relations among items and dynamic user preferences to capture the dynamic joint influence for next-item recommendation. Specifically, the user translation vector in HierTrans can adaptively change based on both a user's previous interacted items and the item relations inside the user's sequences, as well as the user's personal dynamic preference. Experiments on public datasets demonstrate the proposed model HierTrans consistently outperforms state-of-the-art sequential recommendation methods.

4.1 Introduction

Sequential recommendation aims to recommend new items based on a user's recent behaviors, e.g., to recommend a smart home device after a user purchases a smart home hub [9, 55]. Existing sequential recommenders mainly focus on modeling sequential patterns by using user activity sequences, such as Markov Chains [49], Recurrent Neural Networks, and Convolutional Neural Networks [53, 54]. However, purely sequence-based recommendation usually faces challenges in capturing *general item relations* that are not easily discovered from highly-personalized user sequences. For example, Figure 4.1 shows how a purely sequence-based recommender will treat

[†]Reprinted with permission from "Adaptive Hierarchical Translation-based Sequential Recommendation" by Yin Zhang, Yun He, Jianling Wang, James Caverlee, 2020. Proceedings of The Web Conference 2020. Copyright 2020 by ACM. DOI:https://doi.org/10.1145/3366423.3380067

the two user sequences as fundamentally different, even though there are clear patterns among the kinds of items being purchased (in this case, laptops and accessories). Though the specific items are different in each sequence, some items are complements to each other, while others are substitutes. Hence, there is growing interest in capturing these kinds of multi-relations for improved recommendation [59, 60, 63] and in particular, of leveraging complementary and substitute relations for their important influence on user purchases [8, 12, 61].

While encouraging, such relation-aware sequential recommendation still faces several key challenges: (i) *Sparsity and Temporal Generalization:* Previous works mainly use item-level relations to improve non-time aware user recommendation. However, both item relations and user sequences are typically very sparse since users interact with very few items. More importantly, these item-level relations become less useful as items are updated over time (e.g., an older iPad being replaced by a new model). Thus, we explore how to view item relations at the category-level as well, since these categorical relations are denser and more stable over time; (ii) *Hierarchical Structure:* While most previous methods can directly connect item-level relations with user interactions (since sequences are viewed from an item perspective), a categorical-level perspective introduces a hierarchical structure of category-level item relations and user-based item-level sequences. Hence, an important question is how to organize the hierarchical connections so that we can extract the complex multi-relations among items that are revealed inside user dynamic sequences; (iii) *Personalized Dynamic Adaptation:* Translation-based recommendation has received lots of attention for strong performance with high scalability to large, real-world datasets [9, 57]. These methods treat users as translation vectors to connect items in a translation space, a natural fit for capturing the interaction between users and the relations among items. However, most translation-based methods model user translation behavior identically. In practice, user translation behavior can be influenced not only by a user's previous interacted items and the item relations among those items, but also the user's personal preference towards items and item relations. Thus, this interplay is crucial for relation-aware sequential recommendation.

Considering these challenges, we propose a hierarchical translation-based recommendation

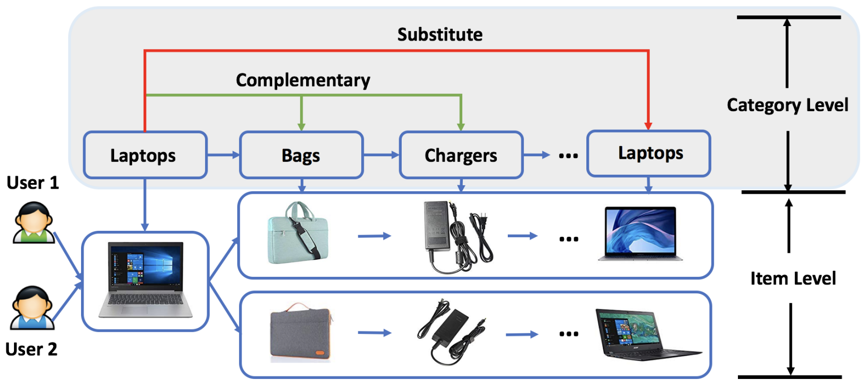


Figure 4.1: The hierarchical structure for two user sequences. Though the two users purchase different items in sequences, they have the same sequence patterns at the category-level and with respect to category-level item relations.

method called HierTrans. HierTrans has three unique properties: **First**, HierTrans extends traditional item-level relations to the category-level, to help capture dynamic sequence patterns that can generalize across users and across time. Furthermore, these category-level item relations can effectively alleviate the sparsity problem in both item relations and user sequences; **Second**, HierTrans is built on a hierarchical temporal graph $\mathcal{G}$ that contains item multi-relations at the category-level and user dynamic sequences at the item-level. The hierarchical graph structure enables us to more easily extract the high-order complex relation patterns among items that are revealed inside user dynamic sequences; **Third**, based on $\mathcal{G}$, we propose a novel hierarchical translation-based recommendation method that adaptively aggregates item multi-relations at the category-level and dynamic user preferences at the item-level for next-item recommendation. Specifically, the user translation vector in HierTrans can *adaptively change* based on both a user's previous interacted items and the item relations inside the user's sequences, as well as the user's personal dynamic preference.

To the best of our knowledge, this work is one of the first to aggregate category-level item relations to dynamically adapt user translation behavior in translation-based recommendation. Through extensive experiments on three public datasets, HierTrans consistently outperforms state-of-the-art methods by 7.02% in recall and 7.72% in NDCG (on average against the next-best alter-

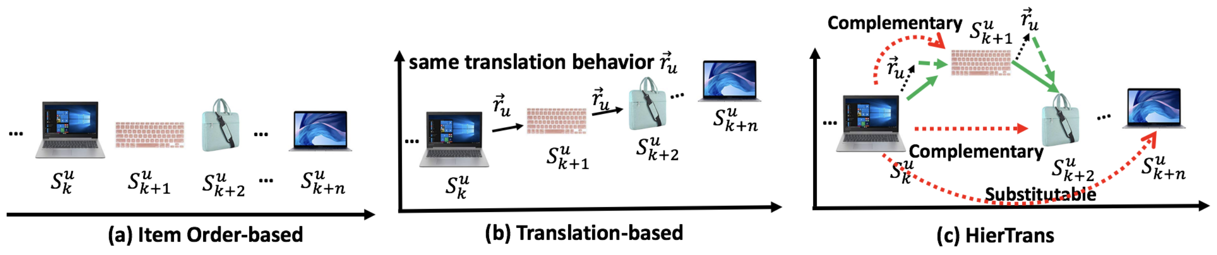


Figure 4.2: Different sequence models. S^u_k denotes that user u interacted with items in sequence order. $\vec{r}_u$ is the user translation vector. (a) Item order-based models, such as RNN, CNN, and Markov Chains, mainly focus on the diverse patterns of item orders in user sequences; (b) Translation-based sequence models treat users as translation vectors to connect items, which models 'higher-order' interactions between a user, her previously interacted items and the next interacted item. Since $\vec{r}_u$ stays the same in the user sequence, these models assume a user's translation behavior connecting items is the same across time; (c) Our proposed HierTrans considers both user personal preferences and item relations. The translation behavior (green line) can adaptively change according to both user preference and the relations of her recent interacted items, making the model more flexible to capture user complex dynamic preferences over time.

native). We also evaluate different components of HierTrans to better understand their impact on sequential recommendation.

4.2 Proposed Method: HierTrans

We aim to provide a personalized sequential recommendation that takes advantage of multi-relations between items.

Problem Statement. Formally, we assume a set of users $\mathcal{U}$, items $\mathcal{I}$ and categories $\mathcal{C}$ where $c^i \in \mathcal{C}$ denotes the category of item i. For each user, we have a sequence of items $\mathcal{S}^u = \{S^u_1, ...S^u_{|S^u|}\}$ that u has interacted with. Besides user-item interaction sequences, we assume there are also multi-types of item relations r_k in relation set $\mathcal{R}$. Here we focus on complementary r_c and substitutes r_s relations. A triple (i, r_k, j) denotes there exists a type of relation $r_k \in \mathcal{R}$ between item i and j from $\mathcal{I}$. Our task is: given a user sequence $\mathcal{S}^u = \{S^u_1, ...S^u_{|S^u|}\}$, we seek to predict the next item for the user.[*]

Based on the task, we face two key questions: (1) First, how can we *organize* the item multi-

[*]Notations are shown in Table 4.1. The category of an item is denoted by c with upper corner of the item letter (e.g., item i's category is c^i). The vector embedding of items and relations are denoted by the same letters with $\vec{\cdot}$. That is, the vector of item i is denoted as $\vec{i}$ and the relation r_k embedding vector is denoted as $\vec{r}_k$. Matrices are represented by boldface uppercase characters.

Table 4.1: Notation of HierTrans.

Notation	Explanation
$\mathcal{U}, \mathcal{I}, \mathcal{C}$	user set, item set, category set
$\mathcal{S}^u$	historical sequence for user u, $\{S_1^u, S_2^u, ...S_{\lvert S^u \rvert}^u\}$
$S_{:n,T}^u$	the user u previous T interacted items $S_{n-T+1}^u ... S_n^u$
$\mathcal{G}$	the hierarchical graph, contains $\mathcal{G}^C$ and $\mathcal{G}^I$
$\mathcal{G}^C$	category-level item relation graph
c^i, $\vec{c}^i$	item i category and the category embedding vector
(c^i, r_k, c^j)	category of item i and category of item j are connected by the r_k relation
$\mathcal{G}^I$	item-level user sequence graph
i, $\vec{i}$	the item i and the item embedding vector
(i, r_u, j)	item i and j are connected by the r_u relation

relations with user sequences to facilitate modeling the dynamic user sequential behavior? (2) Second, user behavior can change based on both the relations of previously interacted items and the user's dynamic personalized preference. How can we *model* the joint influence of user preference and item multi-relations, that at the same time, can also *adaptively adjust* to the user's dynamic personal preferred item/item relations? We deal with each of these questions in the following.

4.2.1 Hierarchical Temporal Graph

We organize/build a hierarchical temporal graph $\mathcal{G}$ to facilitate modeling the dynamic joint influence of the item relations and user personalization. Specifically, the graph $\mathcal{G}$ contains three parts, as shown in Figure 4.3(a)(b): the graph $\mathcal{G}^C$ captures item multi-relations at the category-level; the graph $\mathcal{G}^I$ captures each user's dynamic sequence at the item-level; and finally, the connections between the two graphs to integrate the hierarchical connections. In the following, we detail each step of this construction in order.

Item Multi-Relations Graph: The first graph $\mathcal{G}^C = (V^C, E^C)$ is built by the category-level item connections to facilitate exploiting item high-order semantic multi-relations. The nodes are item categories $V^C = \cup_{i \in I} c^i$. Based on [12], we extend the item relations into category-level relations with the following rule: if item i complements/substitutes item j, then the category c^i of item i complements/substitutes category c^j of item j. That is [12]:

- For complementary: if $(i, r_c, j) \Rightarrow (c^i, r_c, c^j) \in \mathcal{G}^C$;
- For substitutable: if $(i, r_s, j) \Rightarrow (c^i, r_s, c^j) \in \mathcal{G}^C$;

We use the category-level relations since the choices of specific items are highly personalized in user sequences and item-level relations in existing datasets are extremely sparse [8,11]. We observe that, category-level relations [12] are highly relevant to user sequences, and can usually provide denser and generic item relation information that can be applied for different user sequences. Furthermore, for sequential recommendation, we should ensure the model can be *generalized with time drift*. Considering that items can be updated over time, the category-level relations are more stable in time dimension. For example, an iPad2 may be updated to iPad3, but they both belong to the same tablet category. Thus the category-level relations of the iPad2 can also be considered to the iPad3 [12], such as being complementary to the earbud category.

Therefore, we extend the item-level relations to category-level based on [12] in order to capture category-level *semantic information* to help sequential recommendation. That it, we want to capture if item i and j are related, their categories probably share similar semantic information under r_k, and their representations are closer connected by r_k [12]. For example, if a laptop and mouse are complementary, we can infer the categories of the laptop and mouse are closer correlated in complementary relations. Furthermore, relations among different items also contain rich information. For example, if we know both a keyboard and mouse are complementary to a laptop, then keyboard and mouse are closer correlated. The built graph structure smooths the way to investigate them.

Dynamic User Interactions Graph: The second graph $\mathcal{G}^I = (V^I, E^I)$ is built by user sequences to facilitate exploiting user dynamic preference towards specific items. The nodes in $\mathcal{G}^I$ are specific items $V^I = \cup_{i \in I} i$. Concretely, we treat user sequences as a series of transitions the user u has made between each two adjacent items in S^u. Each user represents one type of relation as r_u. In sum,

- For user $u \in \mathcal{U}$, if item j is next to item i in the user sequence S^u, then $(i, r_u, j) \in \mathcal{G}^I$;

Thus, the node connections in $\mathcal{G}^I$ can dynamically change with user sequences changes.

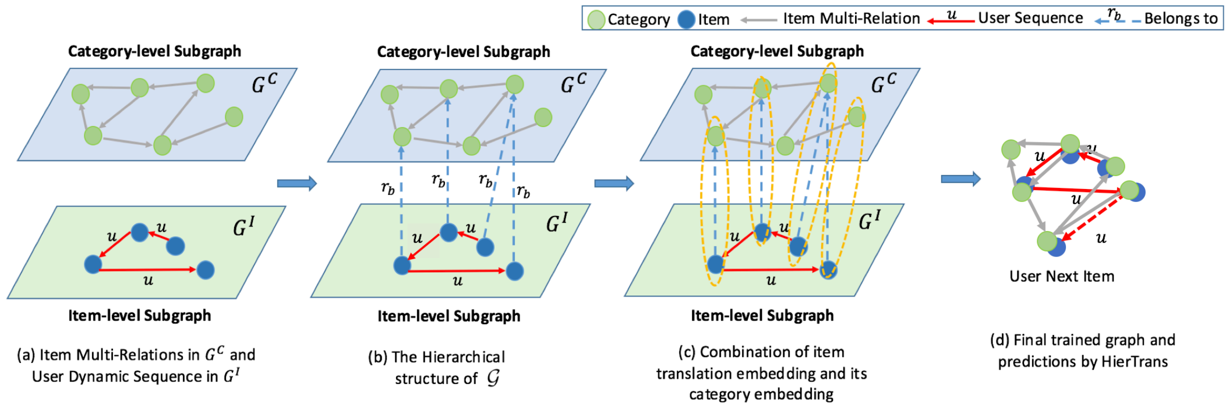

Category
Item
Item Multi-Relation
User Sequence
Belongs to
Category-level Subgraph
G^C
G^I
Item-level Subgraph
u
r_b
User Next Item
(a) Item Multi-Relations in G^C and User Dynamic Sequence in G^I
(b) The Hierarchical structure of $\mathcal{G}$
(c) Combination of item translation embedding and its category embedding
(d) Final trained graph and predictions by HierTrans

Figure 4.3: Overview of HierTrans on the hierarchical temporal graph $\mathcal{G}$. Dots in $\mathcal{G}^C/\mathcal{G}^I$ represent categories/items. Category nodes are connected (grey lines) by item relations in $\mathcal{G}^C$. Each user connects item nodes by their sequences (red lines) in $\mathcal{G}^I$. The item-category connections (in blue dotted lines) represent the "belongs to" relations shown in (c). With HierTrans, we can predict a user's next item based on both user preference and the item relations inside the user sequence, as shown in (d).

Connecting the Two Graphs: Last, we connect $\mathcal{G}^C$ and $\mathcal{G}^I$:

- For item i, if item i belongs to category c^i, then we connect them by the *belongs to* relation: (i, r_b, c^i);

where r_b represents the *belongs to* relation. The connections naturally aggregate the complex influence from both item category-level relations and user sequences as shown in Figure 4.3(b).

4.2.2 Recommendation with HierTrans

This section introduces HierTrans that explores the dynamic joint influence between user personal preference and item multi-relations for next item prediction based on the graph $\mathcal{G}$, as shown in Figure 4.3(c)(d). Since translation-based methods have shown success at capturing user-item interactions [55], it motivates us to utilize its structure to investigate the user-item relation interactions in dynamic sequences. The basic idea of traditional translation-based recommendation is: user sequences, e.g., $S^u = \{S_1^u, S_2^u ... S_{|S^u|}^u\}$, are composed by triples <head, translation relation, tail>, where the head represents the item that user has previously interacted with, and the tail is the next item. They satisfy: $\vec{S}_n^u + \vec{r_u} \approx \vec{S}_{n+1}^u$.

HierTrans adaptively aggregates the embedding based on $\mathcal{G}^C$ and $\mathcal{G}^I$. The high-level idea of HierTrans can be formulated as:

$$\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T}) + \vec{Trans}(r_u | S^u_{:n,T}, r_k) \approx \vec{S}^{*u}_{n+1}, \tag{4.1}$$

where $S^u_{:n,T} = [S^u_{n-T+1} ... S^u_n]$ is the user u's previous T interacted items and S^u_{n+1} is the next item. For each term in Equation (4.1): (i) The head $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$ captures both user personal dynamic preferred patterns through $\mathcal{G}^I$ and item multi-relation patterns through $\mathcal{G}^C$ inside the user's recent sequences $S^u_{:n,T}$ by function $\vec{Head}$; (ii) We propose a novel user translation vector $\vec{Trans}(r_u | S^u_{:n,T}, r_k)$, which models a user's dynamic personal preferred item relations. Different from previous translation-based recommendation where the user translation vector $\vec{r_u}$ is static and identical across time, this proposed user translation vector can *adaptively change* based on both a user's previous interacted items and item relations r_k inside the user's sequences, as well as the user's personal preference. (iii) Similar to translation-based methods, the tail is the embedding of user next item learned by HierTrans. In the following, we show the detailed construction.

4.2.2.1 Construction of Relation-aware Head

The head of HierTrans contains both item relations from $\mathcal{G}^C$ and user personal preference from $\mathcal{G}^I$.

I. Item Category-level Multi-relation in $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. To extract item relations inside user sequences, we first learn the item category relation-aware embeddings in $\mathcal{G}^C$. TransE is leveraged (note that TransH and TransR can also be easily applied):

$$\vec{c^i} + \vec{r}_k \approx \vec{c^j}, \quad k \in \{c, s\}. \tag{4.2}$$

This means that when (c^i, r_k, c^j) holds, the item j category embedding $\vec{c^j}$ should be the nearest neighbor of $\vec{c^i} + \vec{r}_k$.

II. User Personal Preference in $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. In $\mathcal{G}^I$, different from item relations which *pair* each two nodes, items in user sequences are connected in *order*. Thus, we explore the *unified items* effects. That it, we consider the recent T items $S^u_{:n,T} = [S^u_{n-T+1} ... S^u_n]$ together to extract the user preference. Based on $S^u_{:n,T}$, we can follow recent models to extract the different patterns (such as the union and skip patterns by CNN [54]) in $S^u_{:n,T}$. Thus, $\vec{Head}$ here is a function which returns a vector that has the same dimension as the user translation vector, such as the attention and CNN. We discussed the choice of $\vec{Head}$ in Section 4.3.2.

III. Relation-aware Pattern Learned by $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. With the item embedding $\vec{i}$ based on $\mathcal{G}^I$ and item category-level relation-aware embedding $\vec{c^i}$ based on $\mathcal{G}^C$, how can we incorporate them to capture the dynamic item relations *inside* user sequences? Different from other relations, the item and its category are very closely related to each other. So we connect $\mathcal{G}^I$ and $\mathcal{G}^C$ by adding the corresponding embedding vectors: if item i's category is c^i, then

$$\vec{i^*} = \vec{i} + \vec{c^i}, \tag{4.3}$$

as shown in Figure 4.3(c)(d). That is, for each item in $S^u_{:n,T}$, we apply Equation (4.3) to construct $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. Thus HierTrans considers the previous T items relation-aware patterns that contain both dynamic user preference and item multi-relations.

4.2.2.2 Construction of Adaptive Translation

Many existing translation-based recommendations assume user translation vector is static and identical [55, 57]. However, in practice, user translation behaviors are dynamic based on both (1) previously interacted items and those item relations; (2) personal preference towards specific items and personal preference towards item relations. For example, some users frequently change cellphones while others do not. For those users, substitutable relation is frequently utilized. Considering that, we propose a *novel adaptive translation vector* that considers these influence factors.

We first construct the candidates of user translation choice based on user preference and item relations:

$$\vec{r}_{uk} := \vec{t} + \vec{t}_u + \vec{r}_k, \quad k \in \{c, s, n\}, \tag{4.4}$$

where $\vec{t}$ denotes the global transition dynamics across all users [55]. $\vec{t}_u$ represents user personal preference translation. $\vec{r}_k$ represents embedding of different item relations ($\vec{r}_n$ is "not related").

Then we use the attention mechanisms [157, 158] to capture the relation that the user would choose for the next item:

$$\begin{aligned}
\vec{h}_{uk} &= \vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T}) + \vec{t}_u + \vec{r}_k \\
\vec{u}_{uk} &= tanh(\mathbf{W}_a \vec{h}_{uk} + \vec{b}_a) \\
\alpha_{uk} &= \frac{exp(\vec{u}_{uk}^\top \vec{u}_w)}{\sum_k exp(\vec{u}_{uk}^\top \vec{u}_w)} \\
\vec{Trans}(r_u | S^u_{:n,T}, r_k) &= \sum_k \alpha_{uk} \vec{r}_{uk},
\end{aligned} \tag{4.5}$$

where $\mathbf{W}_a$, $\vec{b}_a$ and $\vec{u}_w$ are learnable [157]. Based on Equation (4.5), through attention mechanisms and the construction of $\vec{r}_{uk}$, user's translation behaviors are *adaptively changed* according to user previous interacted items and their relation patterns (by $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C |$
$S^u_{:n,T})$), her personal preference (by $\vec{t}_u$) and her preferred item relations (by $\vec{r}_k$). Hence, with different previous interacted items, the translation vector is adaptively changed.

4.2.3 Optimization

The loss function of HierTrans contains the user dynamic personal preference learning, item multi-relations learning and regularizer:

$$\arg\min_{\Omega} \mathcal{L}^I(\mathcal{G}^I) + \mathcal{L}^C(\mathcal{G}^C) + R(\Omega).$$

Table 4.2: Amazon datasets including item sparsity. Sparsity is item sparsity in the user-item matrix.

Dataset	Users	Items	Feedback	Categories	Relations	Item Sparsity
Electronics	11,965	22,791	303,125	622	174,255	0.111%
C & A	23,539	11,170	173,464	50	4,704	0.066%
H & K	19,231	20,098	251,825	853	222,911	0.065%

$\mathcal{L}^I(\mathcal{G}^I)$ is user item-level personal preference learning and $\mathcal{L}^C(\mathcal{G}^C)$ is item category-level relation learning. Ω is the set of model parameters. $R(\Omega)$ is L_2 regularizer here [55]. For $\mathcal{L}^I(\mathcal{G}^I)$ [55]:

$$\mathcal{L}^I(\mathcal{G}^I) = -\sum_{u\in\mathcal{U}}\sum_{j\in\mathcal{S}^u}\sum_{j'\notin\mathcal{S}^u} ln\ \sigma(\hat{p}_{u,S^u_{:n,T},j} - \hat{p}_{u,S^u_{:n,T},j'}),$$

where j denotes the next item of S^u_n in user sequence S^u. $\sigma(\cdot)$ is the sigmoid function. $\hat{p}_{u,S^u_{:n,T},j}$ means the probability that u will prefer j given previous T items, which is calculated by:

$$\hat{p}_{u,S^u_{:n,T},j} \propto \beta_j - d(\vec{Head}(\mathcal{G}^I, \mathcal{G}^C|S^u_{:n,T}) + \vec{Trans}(r_u|S^u_{:n,T}, r_k), \vec{j}^*), \tag{4.6}$$

where β_j is item bias. We take $||\vec{x} - \vec{y}||^2$ as the dissimilarity measure $d(\vec{x}, \vec{y})$ to calculate the probability. For $\mathcal{L}^C(\mathcal{G}^C)$ [72]:

$$\mathcal{L}^C(\mathcal{G}^C) = \sum_{k\in\{c,s\}}\sum_{(c^i,r_k,c^j)\in\mathcal{R}}\sum_{(c^{i'},r_k,c^{j'})\in\mathcal{R}'} max(0, \gamma + d(\vec{c^i} + \vec{r}_k, \vec{c^j}) - d(\vec{c^{i'}} + \vec{r}_k, \vec{c^{j'}})),$$

where $\mathcal{R}'$ represents the negative sets $\mathcal{R}' = \cup_{k\in\{c,s\}}(\{(i', r_k, j) \cup (i, r_k, j')\})$. (i', r_k, j) means $i', j \in \mathcal{I}$ but they are not related by r_k. Here $d(\cdot,\cdot)$ uses the same measurement as Equation (4.6). The $\gamma > 0$ is a margin hyperparameter [72].

4.3 Experiments

We adopt three public datasets from Amazon [11]: Electronics (Elec), Cell Phones & Accessories (C & A), and Home & Kitchen (H & K), as shown in Table 4.2. They contain rich types of item relations and user purchase sequences. Following prior work [9, 54], we convert all numeric ratings to implicit feedback of 1. We discard users having less than n feedbacks (n is 20 for Elec, 5 for C & A, and 5 for H & K to obtain different sparsity and number of item relations). We also remove items having less than 3 feedbacks to keep the original user sequence patterns and alleviate the cold-start problem. Category-level relations are trained based on the item-level relation frequency to alleviate noise.

Similar to prior work in this area [9], we split the user sequences S^u into three parts: the most recent interacted item in each user sequence ($S^u_{|S^u|}$) for testing; the second most recent interacted item ($S^u_{|S^u|-1}$) as the validation data; the remaining items are used as training data.

Evaluation Metrics. To be consistent with prior work in sequential recommendation [9, 54], we adopt two common top-k metrics – recall@k and NDCG@k [9] – for evaluating recommendation performance. The recall at top-k measures the fraction of user next items that have been predicted over all purchased items. NDCG@k considers the position of correctly recommended items. Following the setup used in prior work [9, 159, 160], we randomly sample 100 negative items for each user and rank these items with the ground-truth items to avoid heavy computation on all user-item pairs. The evaluation metrics can be calculated based on the 101st items.

Table 4.3: Evaluating HierTrans versus baselines over three datasets (for all metrics, higher is better). The percentage improvement compares HierTrans versus the next-best alternative.

Dataset	Metric	BPR	TransE	TransFM	GRU4Rec	Caser	TransRec	SASRec	HierTrans	Improv.
Elec	R@1	0.1073	0.1161	0.1296	0.1153	0.1570	0.1678	0.1799	**0.1927**	7.1%
	R@5	0.2800	0.2828	0.3200	0.2959	0.3830	0.4028	0.3954	**0.4551**	13.0%
	R@10	0.3942	0.3941	0.4410	0.4162	0.4997	0.5313	0.5117	**0.5891**	10.9%
	N@5	0.1996	0.2020	0.2267	0.2029	0.2739	0.2892	0.2925	**0.3285**	12.3%
	N@10	0.2363	0.2390	0.2653	0.2435	0.3117	0.3308	0.3303	**0.3721**	12.5%
C&A	R@1	0.1359	0.1197	0.1328	0.1393	0.1423	0.2153	0.1848	**0.2300**	6.8%
	R@5	0.3145	0.2974	0.3295	0.3455	0.3615	0.4660	0.4146	**0.4775**	2.5%
	R@10	0.4144	0.4006	0.4419	0.4742	0.4397	0.5925	0.5327	**0.6031**	1.8%
	N@5	0.2285	0.2114	0.2344	0.2202	0.2615	0.3459	0.3043	**0.3591**	3.8%
	N@10	0.2603	0.2472	0.2703	0.2635	0.2867	0.3867	0.3426	**0.3994**	3.3%
H&K	R@1	0.0660	0.0688	0.0860	0.0695	0.0809	0.1053	0.0973	**0.1158**	10.0%
	R@5	0.2058	0.2355	0.2467	0.1815	0.1914	0.2817	0.2585	**0.2984**	5.9%
	R@10	0.3041	0.3580	0.3610	0.2712	0.2717	0.3898	0.3696	**0.4099**	5.2%
	N@5	0.1368	0.1539	0.1592	0.1374	0.1403	0.1955	0.1796	**0.2096**	7.2%
	N@10	0.1686	0.1945	0.1976	0.1687	0.1677	0.2304	0.2149	**0.2454**	6.5%

Baselines. We compare HierTrans with the following baselines:

- *BPR* [161]. This is the standard Bayesian personalized ranking (BPR) framework using matrix factorization;
- *TransE* [72]. We use user sequences to build the graph and recommend items based on a nearest neighbor search of the item embeddings. Notice other translation based methods, e.g., TransH and TransR, can be easily adapted for HierTrans, thus here we focus on TransE as a representative method;
- *TransFM* [57]. For the recent TransFM, we consider the item's category relation as side information;
- *TransRec* [55]. It treats each user as a translation vector to connect items by user sequences and learn item embeddings;
- *GRU4Rec* [53]. This is a session-based recommender based on RNN. We treat each user's sequence as a session;
- *Caser* [54]. A state-of-the-art sequential recommendation method based on a CNN to deal with high-order Markov Chains;
- *SASRec* [9]. The recent state-of-the-art sequential recommendation method uses a self-attention mechanism to capture useful user sequence patterns. We also use two self-attention blocks;

Parameter settings. The number of latent dimensions is empirically set to be 200 and attention dimension is 100 for all methods. $T = 10$. $\gamma = 1.0$. The regularizer is chosen by grid search from {0.5, 0.1, 0.05, 0.01, ... 0.00001}, drop out rate is from {0.1, 0.3, 0.5, 0.7, 0.9} and the learning rate is from {0.0001, 0.001, 0.01, 0.1}. The negative sampling ratio is 1. For HierTrans, other hyperparameters are tuned based on the validation data. $\vec{Head}(\cdot)$ is the average function for the three datasets. We discussed the choices of $\vec{Head}(\cdot)$ (such as attention and convolution) in Section 4.3.2.

4.3.1 Recommendation Performance

Table 4.3 shows the recommendation performance. R@1 and N@1 are the same and precision can be calculated based on recall for the user's next purchased item prediction [9]. The last column (Improv.) shows the percentage improvement of HierTrans over the next-best alternative. Overall, we observe the full-blown HierTrans improves upon all the baselines on all datasets in recall@k and NDCG@k. Concretely, HierTrans outperforms the next-best alternative by 7.02% in recall and 7.72% in NDCG on average.

Specifically, HierTrans consistently outperforms TransRec, which shows the importance of modeling item multi-relations inside user sequences and considering the T previous items. More importantly, comparing with TransFM, HierTrans consistently achieves a better performance. This confirms HierTrans can more effectively utilize item category-level information for sequential recommendation. For different datasets, HierTrans obtains a large improvement in Electronics and H&K while the improvement in C&A is relatively small. We attribute the good performance of HierTrans to the rich item relations in both Electronics and H&K based on the Relations column in Table 4.2. Furthermore, the proposed HierTrans outperforms all baselines on both sparse and dense datasets. One likely reason is that the incorporated item information in HierTrans can (1) provide more evidence for user sequential patterns and thus can alleviate the sparsity problem in user sequences; (2) also give more insights to effectively learn sequential patterns among the complex user interactions in dense datasets. We also check different latent dimensions ([50,100,150,200]). HierTrans consistently outperforms the other methods (omit this part due to space limitation).

4.3.2 Ablation Study

The section introduces several variants of HierTran to analyze their effects: (1) *TransRecC*: we only incorporate item category-level relations in TransRec without considering attention of user translations and multiple previous items; (2) *TransRecI*: we directly apply item-level relations in the user sequence graph rather than category-level; (3) *Concat*: instead of using $\vec{i}^* = \vec{i} + \vec{c}^i$, we concatenate the category embedding to item embedding; (4) *Connection*: we introduce "belongs to" relation embedding to connect $\vec{i}$ and $\vec{c^i}$; (5) *No Pre-train*: we forgo the pre-training phase for $\mathcal{G}^C$ and $\mathcal{G}^I$; (6) *No Attention*: we remove the attention (Equation (4.5)) and use the unified transition vectors; (7) *No Multi-Items*: Only the nearest recent item is used: $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S_n^u) + \vec{Trans}(r_u | S_n^u, r_k) \approx \vec{S}_{n+1}^{*u}$; (8) *Item Attention/Convolution*: we use self-attention /CNN for $\vec{Head}(\cdot)$. Hyper-parameters are fine tuned based on the validation datasets.

The results are shown in Table 4.4. The *Default* row shows HierTrans results. We observe methods that consider item relations (such as TransRecC and TransRecI) outperform TransRec, which supports that (category-level) item relations play an important role in sequential recommendation. Specifically, TransRecI achieves higher performance than TransRecC. One likely reason is that there can be information leakage since the item-level relations in the Amazon dataset are captured from user purchase history [11]. We also argue that category-level item information is more general and stable for sequential recommendation.

Table 4.4: Ablation study on Electronics. Similar results hold for the other two datasets.

Setup	R@5	N@5	Setup	R@5	N@5
Default	0.4551	0.3285	No Pre-train	0.4249	0.3034
TransRecC	0.4150	0.2999	No Attention	0.4497	0.3232
TransRecI	0.4282	0.3122	No Multi-Items	0.4147	0.3016
Concat	0.3994	0.2845	Item Attention	0.4406	0.3161
Connection	0.3931	0.2825	Item Covolution	0.4517	0.3267

The way to incorporate item multi-relations and dynamic user interactions heavily impacts HierTrans performance. Comparing *Concat* and *Connection*, directly adding category embeddings achieves the best performance. Furthermore, results of *No Attention* and *No Multi-Items* show both the attention and multi-items consideration play an important role to improve sequential recommendation, which confirms our intuition that user sequence patterns are collaboratively influenced by both item relations and previous T purchased items. For the choice of $\vec{Head}(\cdot)$, we observe using average performs better than using attention/CNN here. One possible reason is that the three datasets are very sparse while attention/CNN brings many parameters, which makes the corresponding method easier to overfit.

4.4 Summary

We propose a novel hierarchical translation-based sequential recommendation that *adaptively* aggregates item multi-relations and dynamic user preferences from both a user's interacted item patterns and a user's dynamic translation behavior. Experiments on different datasets show HierTrans consistently outperforms state-of-the-art sequential recommenders. In the future, we are interested in exploring the influence of different types of item relations (e.g., movies with the same director) on *dynamic* user behaviors.

5. VIBE CHECK: SOCIAL RESONANCE LEARNING FOR ENHANCED RECOMMENDATION[‡]

In this chapter, we consider the user relations and tackle the high heterogeneity by exploring the powerful social resonance effect between social connections and other users in an eCommerce platform to improve recommendation. Social Resonance is a common socio-behavioral phenomenon in which users are more influenced by opinions that have similar vibes. That is, opinions from two different groups of users can *mutually reinforce* (or resonate with) each other to have an even stronger impact on the user. To model the social resonance, we first formulate an item-aware user influence network that connects users who rate the same item. With the social network and item-aware user influence network, a novel graph-based mutual learning framework is proposed, which captures the resonance influence from both user local correlations and global connections. We then fuse these influence paths to predict the resonance-enhanced user preference towards items. Experiments on three public benchmarks show the proposed approach outperforms state-of-the-art social recommendation methods, and also achieves the best performance for cold-start users. Further, we show how the social network and item-aware influence network are complementary in improving item recommendation under different scenarios.

5.1 Introduction

Social Resonance is a common socio-behavioral phenomenon in which users are more influenced by opinions that have similar vibes [162–166]. That is, opinions from two different groups of users can *mutually reinforce* (or resonate with) each other to have an *even stronger* impact on the user. One of the unique properties for resonance is that it can expand in intensity with mutual re-enforcement between different sources, leading to more powerful influence on user intentions and actions. This social resonance effect has been widely studied in many areas such as

[‡]Reprinted with permission from short version “Vibe Check: Social Resonance Learning for Enhanced Recommendation” by Yin Zhang, Yun He, James Caverlee, 2021. International Conference on Advances in Social Network Analysis and Mining. Copyright 2021 by ACM. DOI:http://dx.doi.org/10.1145/3487351.3488335

marketing [162, 165, 166], communication [163, 167], and human behavior [168]. These studies further show social resonance can heavily influence a user's attitude towards items, impact a user's propensity to buy, and also provoke other desired actions.

For example, compared with the dress that is only recommended by Amy's friends, when *both* Amy's friends and other users in an eCommerce platform strongly recommend the same dress, the similar mutual "vibe" (or resonance) can potentially strengthen Amy's intensity of her preference towards the dress (e.g., leading to a higher chance of purchase). The resonance effect is especially *manifest* in eCommerce platforms like Epinions and Ciao [169], where users can make friends to help them discuss and choose items [170]. The resonance also widely impacts engagement with online media (e.g., resonating between a user's friends on Facebook and other users on a platform like YouTube) and apps (e.g., resonating between communities of reddit users and other users on the Google Play Store), among others.

However, there is little if any work on *explicitly* exploring the powerful social resonance effect on user preference towards items in recommendation. The closest work is social-aware recommendation that utilizes a social network to improve item recommendation. However, most existing social-aware recommendation methods mainly focus on the *intrinsic internal* properties of the social network (e.g., social homophily [13, 171] or social influence diffusion [80]) and their single directional influence from the social network to an eCommerce platform, ignoring the close *mutual interactions* with users on the eCommerce platform and the reinforced resonating influence effect. Hence, this gap between the user-user social network and the user-item interactions on an eCommerce platform may limit the learning capacity of social effects on user preferences towards items.

In this work, we propose the first investigation of the powerful social resonance effect to improve recommendation. However, learning the influence of social resonance on user preference is challenging and largely unexplored. Particularly, it poses three key challenges: (i) There is high heterogeneity between the social network and the eCommerce platform: the social network captures user-user connections, while the eCommerce platform exhibits user-item interactions. How can we model the mutual interactions of the social resonance effect across these fundamentally different perspectives? (ii) Social resonance captures the similar mutual vibe between two groups of users. Therefore, different from traditional user influence that typically focuses on how one user impacts on a single target user, the resonance effect measures the correlation between two groups of users, and their correlation's impact. Hence, an important question is how to model the *correlation influence* of social resonance on user preference towards items? (iii) The degree of resonance could vary greatly depending on both the strength of user connections in the social network and also the users on the eCommerce platform. Thus another key challenge is: how can we jointly consider the user relations in both the social network and an eCommerce platform for their correlations estimation?

With these challenges in mind, we propose a social Resonance Recommendation approach called ResRec that builds a novel graph-based mutual learning framework to learn social resonance for improved recommendation. Concretely, ResRec first builds an item-aware user influence network that connects users who rate the same item in the eCommerce platform. Through such user-user connections, it can naturally bridge the gap between the social network and user behaviors in the eCommerce platform to facilitate the exploration of their mutual interaction. Based on this, ResRec explores the resonance effect thoroughly from two perspectives: (i) preference-based resonance between a user's directly connected friends; and (ii) multi-hop relation-based resonance with distant connected users. We argue that the proposed graph-based back-and-forth mutual learning about social resonance through preference-based and multi-hop relation-based resonance can more comprehensively capture the correlation influence and the degree of resonance between users in a social network and an eCommerce platform, to enhance the learning of user preference towards items. As *one of the first works* to explicitly explore the social resonance effect in eCommerce recommendation, this chapter finds that ResRec consistently outperforms state-of-the-art social recommendation methods and also achieves the best performance for cold-start users over three real-world benchmarks. We further show how the social network and item-aware influence network in ResRec are complementary to each other, with each augmenting the other for improved item recommendation.

5.2 Problem Formulation

In this section, we first introduce the problem setting and key notations toward capturing social resonance. Suppose we have n users $U = \{u_1, u_2, ...u_n\}$ and m items $P = \{p_1, p_2, ...p_m\}$. Users can rate and review items to show their preferences and opinions towards those items [75]. Let $\mathbf{R} \in R^{n \times m}$ denote the rating matrix, where $r_{ai} \in \mathbf{R}$ is user u_a's rating of item p_i. Our goal is to predict a user's unknown preference for items, i.e., the missing ratings in $\mathbf{R}$. The key notations are shown in Table 5.1.

To model social resonance, we need to consider two groups of users that can mutually reinforce each other. Here, we view resonance between a social network that connects users and an eCommerce platform where users can engage with items. Concretely, we model the social network and users in the eCommerce platform as follows:

Social Network G^S. First, we assume there is a trust (or friendship) social network which can be represented as a user-user directed graph $G^S = (U, E^S)$. In G^S, similar as many social recommendation settings, the nodes are users U and the edges $E^S \in R^{n \times n}$ are based on the social connections: if u_a trusts (or follows) u_b, then $e_{ab} = 1$, otherwise it is 0. The connections mean u_a could be influenced by u_b. If the social connections are undirected (such as in a friendship network), the social graph can be treated as a bidirectional graph.

Item-Aware User Influence Network $G^P(p_i)$. For the eCommerce platform, we define a network that connects all the users who have rated or reviewed the same item p_i. Similar as [83], users who rate p_i could potentially form a latent influence network, which we denote as the *item-aware user influence network* $G^P(p_i) = (U(p_i), E^P(p_i))$ for each item p_i. Concretely, the nodes $U(p_i)$ are the users who rate item p_i, and the edges $E^P(p_i)$ connect users who rate *before* the target user. That is, to give a recommendation for a target user u_a, the user u_a is connected to users $G^P(p_i, u_a) = \{u_b \in U(p_i) | \exists\, r_{ai} \text{ and } r_{bi}, \text{and } (t(u_b, p_i) < t(u_a, p_i))\}$, where $t(u_b, p_i)$ is the time u_b rates p_i, since a user $u_a \in U(p_i)$ could only be influenced by other users who rate the item *before*

u_a. Then, similar as [83], we can use models (e.g. attention mechanisms) to estimate the user actual influence in $G^P(p_i)$.

An example of such an item-aware user influence network is shown in Figure 5.1. Users u_1, u_3, u_6, u_5 have rated the item p_3 before u_2. So for u_2 connections, she is connected to each of them since they could potentially influence user u_2's preference towards p_3. The connections between u_1, u_3, u_6, u_5 are omitted here for simplicity. Note that different from collaborative filtering which treats users who rate before and after the targeted user as equally important, we argue that users who rate before the targeted users can explicitly influence the targeted user's preferences towards items [81, 83]. For simplicity in presentation, we refer to $G^P(p_i)$ as an *item-aware influence network* in the rest of the chapter.

Based on these two perspectives, **social resonance in recommendation** can be defined as: Given user u_a social network G^S and users who have rated item p_i, the social resonance effect for a user u_a towards item p_i (i.e. a user-item pair (u_a, p_i)) is the mutual correlation between users in user u_a's social network (G^S) and users who have rated the item p_i ($G^P(p_i, u_a)$), which influences u_a's preference towards items.

5.3 Proposed Approach: ResRec

In this section, we propose a novel graph-based mutual learning framework called ResRec that learns the reinforced resonance effect between the social network G^S and the item-aware influence network G^P for improved recommendation. The overall model architecture of ResRec is shown in Figure 5.1. The three main components for ResRec are:

- **Preference-based resonance.** First, the social resonance often occurs when users express similar opinions [166]. In our context of item recommendation, if users who rate an item express similar preferences as my friends, then I am more likely to be influenced. This preference-based resonance captures the preference correlations between a user's friends (in the social network G^S) and other users in the eCommerce platform (in the item-aware influence network G^P). That is, if a user observes other users in $G^P(p_i)$ express *similar preference* as her friends in G^S do, then those opinions are more likely to influence the user's preference towards p_i.

- **Multi-hop relation-based resonance.** Second, while preference-based resonance focuses on a user's direct friends, more distant connections could have a latent influence on a user's preferences and show the degree of resonance. For example, opinions from directly connected users in the social network G^S are easier to resonate with the user than users many hops away. Therefore, we further utilize the global graph structure of G^S to explore user influence in the item-aware influence network G^P via multi-hop relation-based resonance.
- **Social resonance enhanced recommendation.** Lastly, we build a resonance enhanced user embedding (see Figure 5.1) by integrating the mutual resonance effect alongside the internal influence from each graph to give the final item recommendation.

5.3.1 Preference-based Resonance

We begin by examining the preference-based resonance between a user's friends (i.e. G^S) and the group of users who have rated the item p_i (i.e. $G^P(p_i, u_a)$). How do these two sources resonate with each other? To do so, in this section, we introduce a preference mutual learning module that utilizes the user embeddings to match user preference correlations based on the local graph structure of G^S and $G^P(p_i, u_a)$, as shown in Figure 5.2. The learning module finally builds a *preference resonance matrix* $\Gamma(u_a, p_i)$ that connects a user's general preference in G^S with item-aware preference in $G^P(p_i, u_a)$.

Concretely, as in many recommendation models, we first apply a user embedding lookup layer to describe the user u_a as an embedding vector $\mathbf{u}_a$ that represents the user's personal preferences. Similarly, item p_i is represented by an embedding vector $\mathbf{p}_i$. With the user embedding, to capture the preference connections, we define a cross correlation scoring function $\gamma^{relation}(\cdot, \cdot)$ that exploits the Euclidean distance to learn the correlation between user preference expressed in both the social network G^S and the item-aware influence network $G^P(p_i)$: $\gamma^{relation}(\mathbf{u}_s, \mathbf{u}_p) = \frac{1}{1+|\mathbf{u}_s - \mathbf{u}_p|}, \quad \forall u_s \in G^S(u_a), u_p \in G^P(p_i, u_a)$, where $\mathbf{u}_s$ is u_a's friends and $\mathbf{u}_p$ is a user who has rated item p_i before u_a. With the correlation scoring function, the preference mutual resonance matrix $\Gamma(u_a, p_i)$ for user u_a towards item p_i can be formed by her social-oriented and item-aware connections as:

Table 5.1: Notation of ResRec.

Notation	**Explanation**
$\mathcal{U}, \mathcal{P}$	user set, item set
$\mathbf{R}$	rating matrix
r_{ai}	ratings of user u_a to item p_i
$P(u_a)$	subset of items rated by u_a
$U(p_i)$	subset of users rated item p_i
G^S	social network
$G^S(u_a)$	social friends who u_a directly connected with
$G^P(p_i)$	item p_i aware latent social network
$G^P(p_i, u_a)$	users who interacted with item p_i before u_a
$u_a,\ u_s,\ u_p$	target user, user from G^S, user from G^P
d	latent dimension
$\mathbf{u}_a \in R^d$	user a latent factor
$\mathbf{p}_i \in R^d$	item i latent factor

$$\Gamma(u_a, p_i) = \gamma^{relation}(G^S(u_a), G^P(p_i, u_a)). \tag{5.1}$$

Each element in $\Gamma(u_a, p_i) \in R^{|G^S(u_a)| \times |G^P(p_i,u_a)|}$ denotes the preference correlation between u_a's friends and a user who rates p_i. Through the matrix, $\Gamma(u_a, p_i)$ gathers all the correlations of the user's friends and users who rate the item p_i based on both G^S and $G^P(p_i, u_i)$. Furthermore, rather than directly connecting users in social network with items, $\Gamma(u_a, p_i)$ explores user correlations. Hence, $\Gamma(u_a, p_i)$ can closely connect the social network with items, bridging the gap of heterogeneity between users and items to better capture the social influence on user preference.

Specifically, each row $\Gamma(u_a, p_i)[s, :]$ represents the preference similarity between the user's s_{th} friend and users who potentially influence the user at the item-level. Each column $\Gamma(u_a, p_i)[:, p]$ indicates the similarity between the p_{th} user in $G^P(p_i, u_a)$ and user u_a's social connections. Therefore, the preference resonance scores $z^S(u_b|u_a, p_i)$ of user $u_b^S \in G^S(u_a)$ to user u_a, and the resonance scores $z^P(u_c|u_a, p_i)$ of $u_p \in G^P(p_i, u_a)$ to user u_a is:

$$\begin{aligned} z^S(u_s|u_a, p_i) &= \sum \Gamma(u_a, p_i)[s, :], \\ z^P(u_p|u_a, p_i) &= \sum \Gamma(u_a, p_i)[:, p]. \end{aligned} \tag{5.2}$$

Equation (5.2) shows: to predict the preference resonance effect of her friend $u_s \in G^S$ to u_a, ResRec measures the preference similarities between u_s and all the users who rate item p_i, denoted as $z^S(u_s|u_a, p_i)$; mutually, to predict the resonance effect of $u_p \in G^P(p_i, u_a)$ to u_a, ResRec measures the preference similarities between u_p and all the user u_a's friends to find whether u_p has similar opinions as u_a's friends, denoted as $z^P(u_p|u_a, p_i)$.

5.3.2 Multi-hop Relation-based Resonance

The preference-based resonance uncovers the preference similarities between a user's directly-connected friends and users who rate item p_i based on the local graph structure. But of course, other users beyond those directly connected to our target user could also wield influence on her preferences, e.g., through information diffusion. Hence, in this section, we further explore the connections between the social network G^S and the item-aware influence network G^P from the perspective of these multi-hop relations. The key idea is to view the importance of each user $u_c \in G^P$ based on the multi-hop relations in the social network G^S to infer the resonance influence in G^P. Different from the preference-based resonance which learns the preference correlations between users in the local graph structure, this relation-based resonance explores the multi-hop relation influence of resonance effect based on the global graph structure (via node positions [172] in the global graph).

To do so, we propose a novel *relation-aware mutual module* to *inject* the graph $G^P(p_i, u_a)$ to the G^S graph structure, to explore the relation-based resonance, as shown in Figure 5.3. The relation-aware mutual module treats the users in $G^P(p_i, u_a)$ as the anchor set for G^S and computes the distance between each user in the anchor set to u_a based on the graph structure of G^S. Then a message construction function is used to compute the influence diffusion weights from user u_p to u_a based on their position distance. We introduce each step as follows.

5.3.2.1 *Mutual Anchor Set Construction*

We propose to incorporate the social network to help infer the user influence in G^P. That is, since user influence is related to their connections in the social network, we can leverage the node positions [172] in the social network to infer user influence in G^P. To do so, as shown in Figure 5.3, we first gather all the users who have rated the item p_i before u_a, i.e. the nodes in $G^P(p_i, u_a)$ to build the mutual anchor set S_{mutual} for user social graph G^S:

$$S_{mutual} = \{u_p | u_p \in G^P(p_i, u_a) \text{ and } u_p \in G^S\}.$$

Since all the users in S_{mutual} are both from $G^P(p_i, u_a)$ and G^S, we can utilize the relative distance in G^S to impact their influence in $G^P(p_i, u_a)$ based on the social resonance effect. More importantly, since $G^P(p_i, u_a)$ is item-aware, the learned influence in $G^P(p_i, u_a)$ is specific for the item and helpful to estimate user preference towards the item.

5.3.2.2 Relative Distance Inference

Based on the mutual anchor set, inspired by [172, 173], we then capture the node positions in G^S to infer each user's influence in $G^P(p_i, u_a)$. To reveal a node's position in a graph, we compute the shortest path distance between the target user u_a and users in S_{mutual}. Since the computation time for shortest path distance is high ($O(|U|^3)$), to make ResRec scalable, we adapt the t-hop shortest path.

Concretely, for each user $u_p \in S_{mutual}$ in the mutual anchor set, as shown in Figure 5.3 orange color, the t-hop shortest path distance in G^S can be computed as:

$$d^t(u_a, u_p | G^S) = \begin{cases} d(u_a, u_p | G^S) & \text{if } d(u_a, u_p | G^S) \leq t, \\ \infty & \text{otherwise,} \end{cases}$$

where $d(u_a, u_p | G^S)$ is the shortest path distance between user u_p to user u_a in G^S. That is, we only consider the influence if the user in $G^P(p_i, u_a)$ is within t-hops to the target user u_a in the shortest path of G^S, which makes ResRec easy to scale.

5.3.2.3 Resonance Message Computation

With the relative distance, we then build the message passing from users in S_{mutual} to u_a, to learn the relation-based resonance influence in G^P based on the graph structure in G^S. As indicated by many works [174], there will be more influence from u_p to u_a if user u_p is close to u_a in graph G^S. Therefore, the message construction function from u_p to u_a is defined as:

$$m^P(u_p|u_a, G^S) = \frac{1}{d^t(u_a, u_p|G^S) + 1}, \tag{5.3}$$

where $m^P(u_p|u_a, G^S)$ is the message transformation weight from u_p to u_a. Since the calculated message transformation weight $m^S(u_p|u_a, G^S)$ is based on the mutual positions in G^S, it can give a reasonable transformation from the social influence to the item-aware local influence. Additionally, for different items, the mutual anchor sets are different. Therefore, $m^P(u_p|u_a, G^S)$ can be adapted for *both items* (based on S_{mutual}) and *user social connections* (based on the relative distance in G^S), which makes ResRec more flexible to capture the user's varying preference towards items.

For the message transformation of users in G^S based on G^P, which is denoted as $m(u_s|u_a, G^P)$, since G^P does not have explicit social structures, it would be noisy if we utilize G^P's graph structure to infer the influence of u_s. Thus we simply count it as 0.

Integrated Resonances Effect. The preference-based resonance and the multi-hop relation-based resonance jointly influence the user through the resonance effect from two perspectives (one emphasizing local graph structure, while the other emphasizes global graph structure). We finally aggregate the two types of resonance to represent the joint resonance effect between graph G^P and G^S:

$$\begin{aligned} reson(G^P|G^S) &= AGG(z^P(u_p|u_a, p_i), m^P(u_p|u_a, G^S))\mathbf{u}_p, \\ reson(G^S|G^P) &= AGG(z^S(u_s|u_a, p_i), m^S(u_s|u_a, G^P))\mathbf{u}_s, \end{aligned} \tag{5.4}$$

where AGG is an aggregation function such as MEAN, MAX, SUM, which is permutation invariant. We find using a simple SUM aggregation function experimentally provides good results.

5.3.3 Recommendation with Social Resonance

With the resonance effect between G^P and G^S, how can we predict the user preference towards items? In this section, we finally build a resonance-enhanced user embedding and then show how ResRec makes recommendations.

To build the user embedding for recommendation, in most cases, besides the mutual influence of resonance effect between G^S and G^P, users could also be influenced from their friends (what we refer to as the internal influence of G^S), especially when the item is new or cold-start. Similarly, a user with few friends could rely more on the opinions from users who have rated the items (what we refer to as the internal influence of G^P). Therefore, for each user $\mathbf{u}_b \in G^S$ and $\mathbf{u}_c \in G^P$, we define their influence as:

$$\mathbf{u}_s^+ = AGG(reson(G^S|G^P), inter(G^S)),$$
$$\mathbf{u}_p^+ = AGG(reson(G^P|G^S), inter(G^P)),$$

which captures both the mutual resonance effect and the internal influence from G^S and G^P, as shown in Figure 5.1. We apply the same AGG as Equation (5.4). Many methods [13, 83, 170] can be applied to estimate $inter(G^P)$ and $inter(G^S)$. Since it is not our focus, here we simply implement the commonly used methods in [13, 83] for $inter(G^P)$ and $inter(G^S)$ estimation. In the experiments, we also consider [13, 83] as baselines to evaluate the resonance effect for recommendation.

With the influence from G^S and G^P, the final user resonance-aware embedding is formulated as:

$$\tilde{\mathbf{u}}_a = \sigma(\mathbf{W}_U[\mathbf{u}_a^P, \mathbf{u}_a^S, \mathbf{u}_a^O] + \mathbf{b}_U) \tag{5.5}$$

where $\mathbf{W}_U$, $\mathbf{b}_U$ are the weight matrix and bias vector. $[\mathbf{u}_a^P, \mathbf{u}_a^S, \mathbf{u}_a^O]$ are concatenated by row. $\mathbf{u}_a^S = f_{u_s \in G^S}(\mathbf{u}_s^+)$ that aggregates all the influence from user u_a social friends in G^S. $\mathbf{u}_a^P = f_{u_p \in G^P}(\mathbf{u}_p^+)$ that aggregates user influence to u_a in $G^P(p_i, u_i)$. Here we utilize the graph attention mechanism [13, 170] as the aggregation function. It also can help us better evaluate the resonance effect for recommendation through the attention mechanism. Note that other methods can also be applied. $\mathbf{u}_a^O$ represents the user's own preference which is calculated by $\mathbf{u}_a^O = Pool_d(\mathbf{p}_j \otimes \mathbf{p}_i, p_j \in P(u_a)) \otimes \mathbf{u}_a$. With the element-wise product between interacted items p_j and the candidate item p_i, the updated user latent embedding $\mathbf{u}_a^O$ captures item aspects that are more important to the user with respect to item p_i to further facilitate prediction of user preference towards items [13]. In sum, based on Equation (5.5), the learned resonance-aware user embedding $\tilde{\mathbf{u}}_a$ in ResRec can be very flexibly adapted to different user-item pairs based on both G^S and G^P mutual interactions and their own influence, as well as the user own preference.

By capturing the resonance effect, ResRec can be applied to different recommendation methods to learn item embeddings and thus recommend items that resonate with users. Here we use the commonly applied method in [13] as the item embedding to better evaluate the effect of mutual resonance for recommendation. Then with the user and item embedding, the dot product [107] is used to estimate the user preference towards items.

5.4 Evaluation

In this section, we conduct experiments on three real-world datasets to evaluate ResRec, with a focus on the mutual learning of social resonance.

5.4.1 Experimental Setup

Datasets. We choose three widely used datasets (see Table 5.2): Epinions [169], LibraryThing (LThing) [175] and Ciao [169]. The three datasets are publicly accessible and vary in terms of size, ratings and social link density. To avoid any information leakage in our setup, we split the data into three parts based on items: we use the most recent rating for each item to construct the test dataset, the one rating before the most recent as the validation set, and all the other ratings for training.

Evaluation Metrics. To be consistent with previous work in social recommendation [13, 75], we use two popular metrics to evaluate rating prediction accuracy: MAE (Mean Absolute Error) and RMSE (Root Mean Square Error). Smaller values of MAE/RMSE mean the predicted ratings are

Table 5.2: Summary of the three datasets including social networks.

	Epinions	**LThing**	**Ciao**
# Users	22,164	73,882	2,248
# Items	296,277	337,561	16,861
# Item Interaction	922,267	979,053	36,065
# Social Connection	355,800	120,536	57,544
Rating Density	0.014%	0.004%	0.095%
Link Density	0.072%	0.002%	1.139%

closer to the real ratings. *Note here, as indicated by [75,160], a small decrease of MAE and RMSE can bring a significant impact on the quality of top-k recommendation.*

Baselines. We select representative and state-of-the-art social recommendation methods as the baselines:

- *PMF* [107]. Probabilistic Matrix Factorization is widely used in recommendation due to its robust and strong recommendation performance [94]. PMF is based on matrix factorization without considering social influence.
- *TrustSVD* [176]. TrustSVD is a trust-based matrix factorization method that uses the social network to overcome the sparsity and cold-start problems.
- *GraphRec* [75]. This recent state-of-the-art method formulates the social network and user-item interactions as graphs, and adopts graph neural networks to learn the user and item latent factors for user rating prediction.
- *DANSER* [13]. This is another recent state-of-the-art method that captures homophily and influence from social and item aspects. DANSER is based on graph convolutional and graph attention networks. It also uses the social network to overcome the sparsity and cold-start problems.
- *GhostLink-S, GhostLink-G and GhostLink-D.* These three methods are based on the recently introduced GhostLink [83], that aims to predict ratings with estimated item-aware latent influence. Specifically, we first apply GhostLink to infer the item-aware influence network, i.e. $G^P(p_i)$. Then to compare the effect of item-aware influence network with social network, we apply TrustSVD with the learned item-aware influence for recommendation (denoted as

Table 5.3: Performance comparison of ResRec with other methods. The numbers in the parentheses show the relative improvements of ResRec comparing with the corresponding baselines. The ‘**’ indicates that the improvements over all baselines pass the significance test with p-value < 0.001. ResRec significantly outperforms the other methods in terms of both MAE and RMSE.

		Social Network				item-aware Latent Network			
		PMF	**TrustSVD**	**GraphRec**	**DANSER**	**GhostLink-S**	**GhostLink-G**	**GhostLink-D**	**ResRec**
Epinions	MAE	0.8764 (10.01%)	0.8104 (2.75%)	0.8374 (6.18%)	0.7980 (1.18%)	0.8088 (2.55%)	0.8312 (5.39%)	0.8033 (1.85%)	**0.7887****
	RMSE	1.1427 (8.33%)	1.0775 (2.86%)	1.0653 (1.70%)	1.0596 (1.16%)	1.0769 (2.81%)	1.0781 (2.92%)	1.0680 (1.96%)	**1.0475****
LThing	MAE	0.8364 (14.42%)	0.7686 (7.38%)	0.7423 (3.70%)	0.7268 (1.54%)	0.7694 (7.49%)	0.7606 (6.26%)	0.8075 (12.81%)	**0.7158****
	RMSE	1.0373 (11.55%)	1.0093 (10.00%)	0.9626 (4.92%)	0.9248 (0.80%)	1.0085 (9.92%)	0.9914 (8.05%)	1.0056 (9.60%)	**0.9175****
Ciao	MAE	0.7844 (11.68%)	0.7291 (5.24%)	0.7212 (4.10%)	0.7134 (2.97%)	0.7351 (6.11%)	0.7399 (6.79%)	0.7163 (3.39%)	**0.6928****
	RMSE	1.0276 (10.50%)	0.9607 (4.46%)	0.9399 (2.19%)	0.9321 (1.35%)	0.9672 (5.17%)	0.9473 (3.01%)	0.9389 (2.09%)	**0.9197****

Table 5.4: Performance of methods on cold-start users in terms of MAE and RMSE for all three datasets. The ‘**’ indicates p-value < 0.001 compared with the best baseline.

		Social Network				item-aware Latent Network			
		PMF	**TrustSVD**	**GraphRec**	**DANSER**	**GhostLink-S**	**GhostLink-G**	**GhostLink-D**	**ResRec**
Epinions	MAE	0.9734 (10.66%)	0.8829 (1.49%)	0.9164 (5.10%)	0.8811 (1.30%)	0.8794 (1.10%)	0.9171 (5.17%)	0.8910 (2.39%)	**0.8697****
	RMSE	1.2726 (9.94%)	1.1966 (4.22%)	1.1590 (1.11%)	1.1590 (1.11%)	1.1951 (4.09%)	1.1889 (3.60%)	1.1597 (1.17%)	**1.1461****
LThing	MAE	0.9359 (12.74%)	0.8550 (4.47%)	0.8803 (7.23%)	0.8463 (3.49%)	0.8722 (6.37%)	0.8774 (6.91%)	0.9490 (13.94%)	**0.8167****
	RMSE	1.1510 (11.16%)	1.0992 (6.97%)	1.1001 (7.05%)	1.0447 (2.12%)	1.1028 (7.28%)	1.1266 (9.24%)	1.1296 (9.48%)	**1.0225****
Ciao	MAE	0.7859 (10.55%)	0.7434 (5.43%)	0.7292 (3.59%)	0.7214 (2.54%)	0.7441 (5.52%)	0.7588 (7.35%)	0.7416 (5.20%)	**0.7030****
	RMSE	1.0217 (8.72%)	0.9913 (5.92%)	0.9485 (1.67%)	0.9477 (1.59%)	0.9939 (6.17%)	0.9627 (3.13%)	0.9725 (4.11%)	**0.9326****

GhostLink-S). Similarly, we also apply GraphRec [75] and DANSER [13] with the learned item-aware influence (as GhostLink-G and GhostLink-D) to evaluate the effectiveness of the implicit network for recommendation.

Reproducibility. We implement ResRec in Tensorflow. The latent dimensions d for all methods are fixed to be 10 empirically for a trade-off between performance and computational complexity, as well as for fair comparison across methods. We use mini-batch gradient descent with a batch size of 64. The learning rate is determined by grid search in the range of {0.1, 0.01, 0.001}. The regularization parameter is selected from {0.1, 0.01, 0.001, 0.0001, 0.00001} and dropout ratio is in {0.0,0.1,...0.8}. Average pooling is used. For methods using policy gradient descent, the gradient period is experimentally set to be 1000 for all the methods for fair comparison with corresponding learning rate 0.1.

5.4.2 Overall Comparison

We begin by investigating the overall performance of ResRec comparing with the alternatives as shown in Table 5.3. The ‘**’ indicates that the improvements over all baselines pass the significance test with p-value < 0.001. In Table 5.3, the numbers in the parentheses show the improvements of ResRec comparing with the corresponding baselines. Overall, ResRec consistently outperforms all the baselines in all three datasets. Concretely:

First, both TrustSVD and Ghostlink-S achieve better performance than PMF. The improvement confirms that both social networks and item-aware user influence network can help improve item recommendation, since all those methods are based on matrix factorization. Specifically, for the same methods with different networks (e.g., GraphRec vs Ghostlink-G), they have similar performance. This further verifies that the item-aware influence network indeed can strongly impact the user preference towards items. We also observe that the performance of GhostLink-G/D methods are not good in LThing dataset. One possible reason is that LThing is very sparse, thus the learned item-aware influence network introduce noise that impairs the performance of graph-based methods.

Second, comparing with the MF-based methods, graph neural network-based models (GraphRec and DANSER) generally give a better performance. The results demonstrate the power of graph neural networks in modeling the social network for ratings prediction, since the GNN-based methods can naturally capture the topological structure of the social network.

Third, ResRec significantly outperforms the other methods, especially the state-of-the-art methods GraphRec and DANSER, and also Ghostlink-G and Ghostlink-D. Since all of these methods use graph neural networks, the improvement of ResRec verifies that the modeled social resonance indeed can help improve the prediction of user preference towards items. Moreover, through the resonance mutual learning layer, ResRec can simultaneously utilize the two networks to enhance the learning of each influence.

We also analyze the effect of the key hyperparameter – the number of latent factors – on ResRec (details omitted due to space constraints). We find that ResRec consistently outperforms the other methods, with an optimal embedding size dependent on the particular dataset.

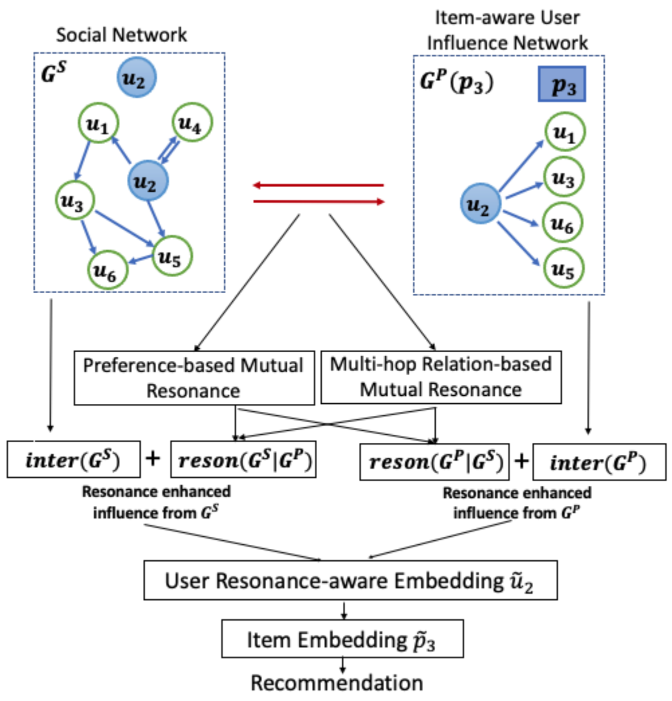

Social Network
G^S
u_1 u_2 u_3 u_4 u_5 u_6
Item-aware User Influence Network
$G^P(p_3)$
p_3
Preference-based Mutual Resonance
Multi-hop Relation-based Mutual Resonance
$inter(G^S)$ + $reson(G^S|G^P)$
$reson(G^P|G^S)$ + $inter(G^P)$
Resonance enhanced influence from G^S
Resonance enhanced influence from G^P
User Resonance-aware Embedding $\tilde{u}_2$
Item Embedding $\tilde{p}_3$
Recommendation

Figure 5.1: ResRec framework. By taking the user-item pair (u_2, p_3) as an input example. ResRec explores the mutual resonance effect between users in social network G^S and users (u_1, u_3, u_6, u_5) who have purchased p_3 through both preference-based and multi-hop relation-based resonance (the connections between u_1, u_3, u_6, u_5 in $G^P(p_3)$ are omitted here for simplicity.).

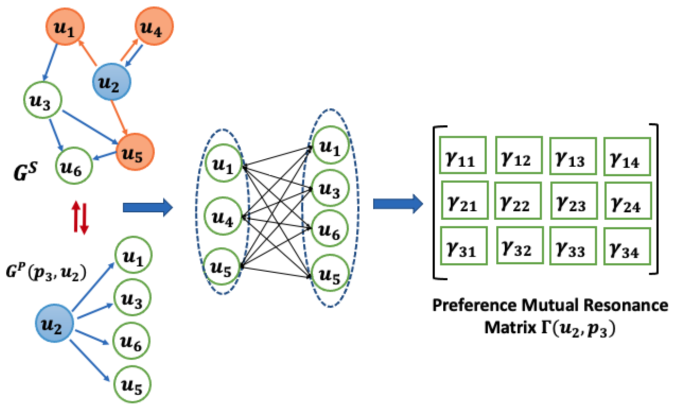

G^S
$G^P(p_3, u_2)$
u_1 u_2 u_3 u_4 u_5 u_6
γ_{11} γ_{12} γ_{13} γ_{14}
γ_{21} γ_{22} γ_{23} γ_{24}
γ_{31} γ_{32} γ_{33} γ_{34}
Preference Mutual Resonance Matrix $\Gamma(u_2, p_3)$

Figure 5.2: Preference-based resonance effect for user u_2 towards item p_3. We explore the preference similarities between a user's friends (in orange) and users who have rated the same item.

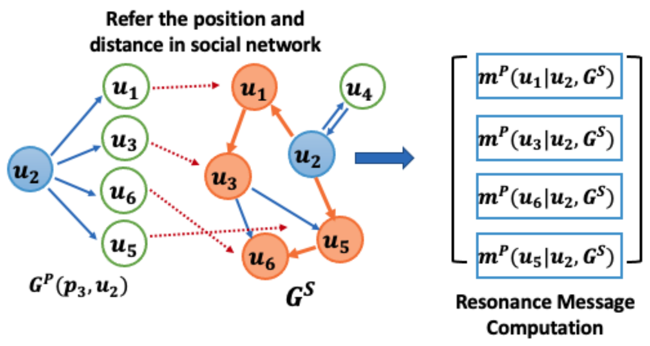


Figure 5.3: Multi-hop relation-based resonance of user u_2 towards item p_3. With the resonance effect, users in orange color are constructed as the mutual anchor set to estimate the user influence in $G^P(p_3, u_2)$ based on the G^S graph structure.

5.4.3 Cold-Start Users

In this section, we drill down to the performance of different methods on cold-start users. Based on the datasets shown in Table 5.2 and previous methods [14, 177, 178], we define the users who rate fewer than ten items as cold-start users for Epinions and LThing, and fewer than five items as the cold-start users for Ciao. Results are shown in Table 5.4. Overall, ResRec consistently significantly outperforms the other methods. Concretely:

First, for cold-start users, the consistent outperformance of ResRec further verifies the importance to consider item-aware influence and resonance effect in social recommendation. Second, comparing with the improvement in Table 5.3, for most of the cases ResRec has a higher improvement for cold-start users. This indicates that considering the influence from both G^S and G^P is even more beneficial for cold-start users. There are also situations where the overall improvements are higher. One possible reason is that the resonance could also heavily influence some active users. This further verifies the benefits to incorporate item-aware influence and resonance in social recommendation.

5.4.4 Ablation Study

In our ablation study, we evaluate the design choices of ResRec from both accuracy and efficiency aspects, with focus on the item-aware influence and graph-based mutual learning of social

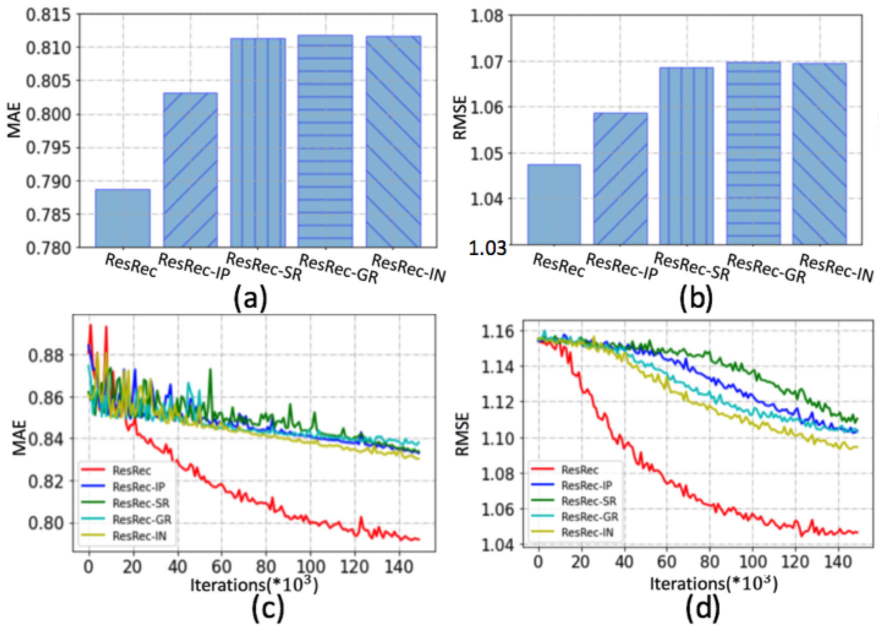


Figure 5.4: (a)(b) MAE and RMSE of variants of ResRec on Epinions dataset. Other datasets have similar performance. (c)(d) MAE and RMSE vs Iterations on Epinions dataset. Other datasets have similar performance.

resonance. The variants are: (1) *ResRec-IP* is ResRec without the item-aware influence network G^P, which is similar to traditional social recommendation; (2) *ResRec-SR* removes the preference resonance shown in Section 5.3.1; (3) *ResRec-GR* forgoes the graph-based resonance in Section 5.3.2; (4) *ResRec-IN* removes the internal influence term $inter(G^P)$.

Figures 5.4(a)(b) show the performance of ResRec and its variants on the Epinions dataset. Figures 5.4(c)(d) show the prediction accuracy of ResRec and its variants with respect to training iterations. Similar results hold for the other two datasets. Overall, we see the full-blown ResRec improves upon all of its variations and ResRec converges much faster than the other methods. In particular, ResRec significantly outperforms methods that remove each type of resonance (ResRec-SR and ResRec-PR). This confirms that both kinds of resonances play a critical role to help improve recommendation. Another finding is that ResRec-SR, ResRec-GR and ResRec-IN show similar performance. This indicates all three influence factors are important and can not replace each other.

5.5 Summary

We have explored the potential of social resonance to enhance the learning of user preference towards items. The proposed ResRec framework comprehensively learns resonance from preference and multi-hop relation-based aspects. Furthermore, the learned influence from the two networks can be flexibly adapted based on both user's social connections and different items. Extensive experimental results show that ResRec significantly improves upon state-of-the-art social recommenders and also achieves the best performance for cold-start users. In our continuing work, we are exploring additional sources of resonance, e.g., from user reviews.

6. INSTAGRAMMERS, FASHIONISTAS, AND ME: RECURRENT FASHION RECOMMENDATION WITH IMPLICIT VISUAL INFLUENCE[§]

By modeling the social resonance, we find user social network can powerfully influence the user preference towards items. In this chapter, we consider the item factor. Especially, we consider the social influence of fashion-focused key opinion bloggers for fashion recommendation. Fashion-focused key opinion bloggers on Instagram, Facebook, and other social media platforms are fast becoming critical influencers. They can inspire consumer clothing purchases by linking high fashion visual evolution with daily street style. We build the *first* visual influence-aware fashion recommender (FIRN) with leveraging fashion bloggers and their dynamic visual posts. Specifically, we extract the dynamic fashion features highlighted by these bloggers via a BiLSTM that integrates a large corpus of visual posts and community influence. We then learn the implicit visual influence funnel from bloggers to individual users via a personalized attention layer. Finally, we incorporate user personal style and her preferred fashion features across time in a recurrent recommendation network for dynamic fashion-updated clothing recommendation. Experiments show that FIRN outperforms state-of-the-art fashion recommenders, especially for users who are most impacted by fashion influencers, and utilizing fashion bloggers can bring greater improvements in recommendation compared with using other potential sources of visual information. We also release a large *time-aware high-quality visual* dataset of fashion influencers that can be exploited for future research.

6.1 Introduction

Opinion leaders can impact consumer purchase and consumption behaviors in a variety of different markets [83, 179, 180]. Among these, *fashion* opinion leaders wield outsize influence on fashion trends [181–184]. And with the rise of visual media platforms like Instagram and Pinterest,

[§]Reprinted with permission from "Instagrammers, Fashionistas, and Me: Recurrent Fashion Recommendation with Implicit Visual Influence" by Yin Zhang and James Caverlee, 2019. Proceedings of The 28th ACM International Conference on Information and Knowledge Management. Copyright 2019 by ACM. DOI:https://doi.org/10.1145/3357384.3358042 influential fashion leaders are not just celebrities and famous designers, but also *fashion bloggers* who have built a name and reputation within the platform itself [98–100]. These fashion bloggers link high fashion with daily wear through their appealing posts. For example, many bloggers attend high profile fashion shows, such as New York Fashion Week, to keep up with the frontiers of fashion trends (like dress design and fashionable colors) [98]. At the same time, connecting these fashion trends with our daily clothing choices through visual social media, fashion bloggers can directly disseminate their fashion choices to consumers, as illustrated in Figure 6.1. Typical example posts on Instagram include "#10 pieces every woman should have in her wardrobe", "#OOTD" (outfit of the day), and "#Top Trends of the season", which are very useful for clothing choices.

Since those influencers can play a significant role in fashion adoption [181] and consumer aesthetic evaluation is largely based on current fashion trends [88, 185, 186], we explore in this chapter the potential of enhancing fashion recommendation by carefully modeling the visual influence of these fashion bloggers. We collect more than 130,000 Instagram posts by influential female fashion bloggers, and connect this visual style to Amazon item purchases over time. This recommender can extract the current hottest fashion clothing based on a user's visual taste, as well as capture trends reflected in the choices of these fashion bloggers. While incorporating influencers into recommendation has great potential value, there are a number of key challenges.

First, the fashion tastes on platforms like Instagram is diffused across millions of posts, and these tastes vary across bloggers. Moreover, their styles are always in flux, since fashion bloggers adapt to new trends. How can we extract each fashion blogger's unique dynamic fashion features from a large corpora of posts? Second, in practice, it is extremely difficult to directly capture the explicit connections/influence from a fashion blogger to a user's purchase [83]. This influence can also be complicated: users can be directly influenced by a blogger's posts or indirectly influenced by the blogger through their friends or communities. Besides, each user's visual preference is personal and some users may be strongly influenced by fashion bloggers while others may not be at all. Hence, how can we learn such personal *implicit visual influence* funnel from fashion bloggers to users for fashion recommendation? Third, the visual influence from bloggers to users could change over time, as an example shown in Figure 6.1. How can we effectively *learn these temporal dynamics* for visual influence-aware fashion recommendation?

In this chapter, our main goal is to address these three challenges to build a personalized visual influence-aware fashion recommender that can learn both fashion trends and user visual preference evolution across time. Specifically, we propose a Fashion visual Influence-aware Recurrent Network (FIRN) that is characterized by three unique features:

- FIRN uncovers fashion features in each time period through a bidirectional LSTM that captures each fashion blogger's style over time as well as the trends in the overall fashion community;
- FIRN naturally models each user's personal visual taste towards these fashion features by learning an implicit visual influence funnel from the extracted fashion features to individual users;
- FIRN builds a novel visual influence-aware recurrent neural network that effectively models temporal dynamics of fashion features from bloggers, users, and their visual preferences.

To our knowledge, this is *the first work to leverage influential fashion bloggers and their visual posts as a dynamic visual signal for user clothing recommendation.* Through experiments over bloggers sampled from Instagram and purchases on Amazon, we quantitatively and qualitatively evaluate the performance of FIRN. We find that FIRN significantly outperforms the state-of-the-art fashion recommendation FSVD [15] by 8.38% on average in RMSE with an even greater improvement (14.05%) for users who have previously consistently purchased items that are similar to posts by fashion bloggers. Furthermore, compared with using other potential sources of visual fashion influence – i.e. the images of a user's previous purchases [11] and a dataset of static aesthetic images (AVA) [92] – fashion bloggers can bring larger improvements in recommendation. These results further confirm that fashion bloggers can provide strong fashion visual signals across time and important dynamic influence towards user clothing purchase decisions.

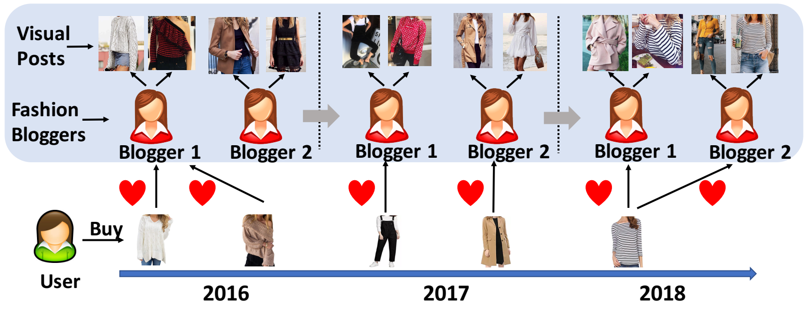


Figure 6.1: Fashion bloggers and their implicit influence funnel. The top two rows show bloggers and their posts. The bottom row shows a user and her purchases.

Table 6.1: Notation of FIRN.

Notation	Explanation
$\mathcal{U}, \mathcal{P}$	user set, item set
$r_{u,p}(t)$	ratings of user u to item p at time t
$\boldsymbol{\theta}_u(t)$	visual influence-aware hidden state of user u at time t
$\boldsymbol{\theta}_p(t)$	hidden state of item p at time t
$\mathbf{m}(p)$	image vector of item p
$P_u(t)$	users bought item at time t
Π	bloggers set
$\mathcal{B}_k$	blogger k, $\Pi = \{\mathcal{B}_1, \mathcal{B}_2, ...\}$
$\mathbf{m}_i(\mathcal{B}_k\|t)$	image vector of i_{th} post for blogger k at time t
$\mathcal{I}(\mathcal{B}_k\|t)$	post set of blogger k at time t
$\mathbf{v}(\mathcal{B}_k\|t)$	visual vector delivered by blogger k at time t

6.2 Visual Influence-Aware Fashion Recommendation

Inspired by these works of fashion bloggers and observations, we explore in this chapter the potential of integrating fashion bloggers for user clothing recommendation.

Problem Statement. Formally, we assume a set of users $\mathcal{U}$, a set of fashion items $\mathcal{P}$, and their ratings in time period T. Specifically, $r_{u,p}(t)$ is the rating that user $u \in \mathcal{U}$ rates $p \in \mathcal{P}$ at time $t \in T$. We further assume a set of fashion influential bloggers $\Pi = \{\mathcal{B}_1, \mathcal{B}_2, ...\}$ and a set of their visual posts $\mathcal{I}(\mathcal{B}_k|t)$ for each blogger $\mathcal{B}_k$, which contains fashion features at time t. Notice here that $\mathcal{U}$ and Π are two different groups of people. By leveraging visual posts in $\mathcal{I}(\mathcal{B}_k|t)$ for each $\mathcal{B}_k \in \Pi$,

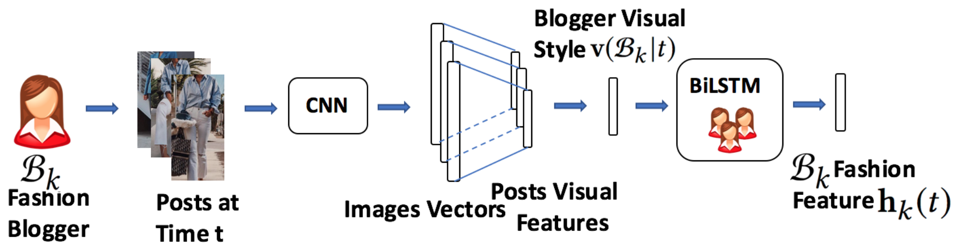

$\mathcal{B}_k$ Fashion Blogger
Posts at Time t
CNN
Images Vectors
Posts Visual Features
Blogger Visual Style $\mathbf{v}(\mathcal{B}_k|t)$
BiLSTM
$\mathcal{B}_k$ Fashion Feature $\mathbf{h}_k(t)$

Figure 6.2: Extracting fashion features $\mathbf{h}_k(t)$ at time t.

we aim to recommend for each user $u \in \mathcal{U}$ a visual influence-aware and time-dependant list of items from the set $\mathcal{P}$ that considers both user visual preference and fashion features. Notations are summarized in Table 6.1.

In the following three sections, we present the design of our proposed visual influence-aware recurrent fashion recommendation FIRN in detail.

6.2.1 Extracting Fashion Features

We begin by extracting *fashion features* $\mathbf{h}_k(t)$ from each blogger $\mathcal{B}_k$. These fashion features represent the blogger's personal preferred fashion style smoothed by the common popular fashion trends in the overall fashion community, as shown in Figure 6.2.

Individual Visual Style. We first represent each blogger's individual visual style by a vector $\mathbf{v}(\mathcal{B}_k|t)$ derived from their visual posts. Since raw image vectors are noisy and low-level representations [11,15], we use an embedding to obtain high-level visual features of each post. Specifically, a post's visual features $\mathbf{v}_i(\mathcal{B}_k|t)$ from blogger $\mathcal{B}_k$ is represented by:

$$\mathbf{v}_i(\mathcal{B}_k|t) = \mathbf{E}_m \mathbf{m}_i(\mathcal{B}_k|t), \tag{6.1}$$

where $\mathbf{E}_m \in R^{K_v \times K_m}$ is the embedding matrix. K_v is the dimension of the embedded visual features $\mathbf{v}_i(\mathcal{B}_k|t)$ and $K_v < K_m$.

Then based on $\mathbf{v}_i(\mathcal{B}_k|t)$, similar to [16, 187], we define the fashion blogger individual-level visual style $\mathbf{v}(\mathcal{B}_k|t)$ at time t as:

$$\mathbf{v}(\mathcal{B}_k|t) = \frac{\sum_{i \in \mathcal{I}(\mathcal{B}_k|t)} \mathbf{v}_i(\mathcal{B}_k|t)}{|\mathcal{I}(\mathcal{B}_k|t)|}. \tag{6.2}$$

Since we observe that posts can be highly visually similar in a short time period (such as one month), we adopt an average here to convey these similar visual features. In this way, the dynamic visual vector $\mathbf{v}(\mathcal{B}_k|t)$ can strengthen the common visual features that the blogger wants to deliver in the time t across her posts. Furthermore, this approach can effectively deal with the distinct different number of posts from bloggers in different time periods.

Incorporating Community Trends. This individual-level blogger visual style vector $\mathbf{v}(\mathcal{B}_k|t)$ is only a partial view of the current fashion features and maybe is noisy for fashion, since the vector is barely based on the current posts of a blogger. In practice, for any time period, a blogger may deliver some visual features that are not closely connected to current fashion trends but only some randomly posts, which could have an influence on the final fashion recommendation. However, the overall fashion community may adopt certain fashion trends that can help re-inforce which aspects of $\mathbf{v}(\mathcal{B}_k|t)$ are representative of fashion features, rather than quirks of this particular collection of posts.

Hence, we propose to smooth the individual blogger style vector $\mathbf{v}(\mathcal{B}_k|t)$ with this community influence to arrive at our goal of *fashion features* $\mathbf{h}_k(t)$. Considering other fashion bloggers can directly (or indirectly) connect to each other by fashion, we model this flow of fashion ideas at time t through a Bidirectional Long Short-Term Memory (BiLSTM) [188] among bloggers.*

Specifically, our blogger BiLSTM is based on the traditional LSTM [51, 188] which has been widely adopted, to capture visual features among bloggers. Formally, we first sort bloggers by the average number of likes of each post to general modulate the flow of fashion information among bloggers. Since the LSTM contains the gating units to bridge very long lags and effectively utilize

*Compared with directly using $\mathbf{v}(\mathcal{B}_k|t)$, experiments in Section 6.3 further show that the BiLSTM can achieve higher accuracy and also has a good efficiency. Furthermore, note that LSTM is not the only choice. For example, the bidirectional gated recurrent unit (BiGRU) can also be used. We use LSTMs here since they are slightly more general as [94] indicated.

input information, it is able to capture different blogger visual features to re-inforce the fashion features for each blogger. Concretely, the fashion features across bloggers at time t is built by:

$$\begin{aligned} [\mathbf{g}_k(t), \mathbf{i}_k(t), \mathbf{o}_k(t)] &= sigmoid(\mathbf{W}[\mathbf{h}'_{k-1}(t), \mathbf{v}(\mathcal{B}_k|t)] + \mathbf{b}), \\ \mathbf{q}_k(t) &= tanh(\mathbf{W}'[\mathbf{h}_{k-1}(t), \mathbf{v}(\mathcal{B}_k|t)] + \mathbf{b}'), \\ \mathbf{c}_k(t) &= \mathbf{g}_k(t) \circ \mathbf{c}_{k-1}(t) + \mathbf{i}_k(t) \circ \mathbf{q}_k(t), \\ \mathbf{h}'_k(t) &= \mathbf{o}_k(t) \circ tanh(\mathbf{c}_k(t)), \end{aligned} \tag{6.3}$$

where the input gate $\mathbf{i}_k$, output gate $\mathbf{o}_k$ and forget gate $\mathbf{g}_k$ are used to control how each fashion blogger influences other bloggers. $\circ$ denotes the element-wise product. For simplicity, we use $\mathbf{h}'_k(t) = LSTM(\mathbf{h}'_{k-1}(t), \mathbf{v}(\mathcal{B}_k|t))$ to denote these operations.

Based on Equation 6.3, the activations of the forward LSTM and backward LSTM for influence $\mathcal{B}_k$ is denoted as $\overrightarrow{\mathbf{h}}_k(t)$ and $\overleftarrow{\mathbf{h}}_k(t)$ which represent the fashion flow from other bloggers to blogger $\mathcal{B}_k$. So the final fashion feature based on blogger $\mathcal{B}_k$ at t is denoted as $\mathbf{h}_k(t) = [\overrightarrow{\mathbf{h}}_k(t), \overleftarrow{\mathbf{h}}_k(t)]$, which considers both a blogger's individual visual posts and their community interactions. This formulation has the benefit of smoothing each blogger's fashion features with the overall trends in the community, so that the extracted fashion features from each blogger are more representative and with more emphasis on common popular fashion trends.

6.2.2 Implicit Personal Visual Funnel

Given these fashion features $h_k(t)$, how can we model the influence from $h_k(t)$ of bloggers to each user u? In practice, it is hard (if not impossible) to get the explicit mapping from fashion bloggers to users and their purchases in many situations considering privacy constraints (e.g. from Instagram posts to Amazon purchases). Recently, Mukherjee *et al.* [83] used review data to build a user social latent influence without requiring explicit social network and showed great improvements in recommendation. Inspired by their success, we aim to leverage the *visual* signals from bloggers to reveal the implicit influence funnel from fashion bloggers to users. We also consider that the influence from extracted fashion features to users may be personalized in both visual as-

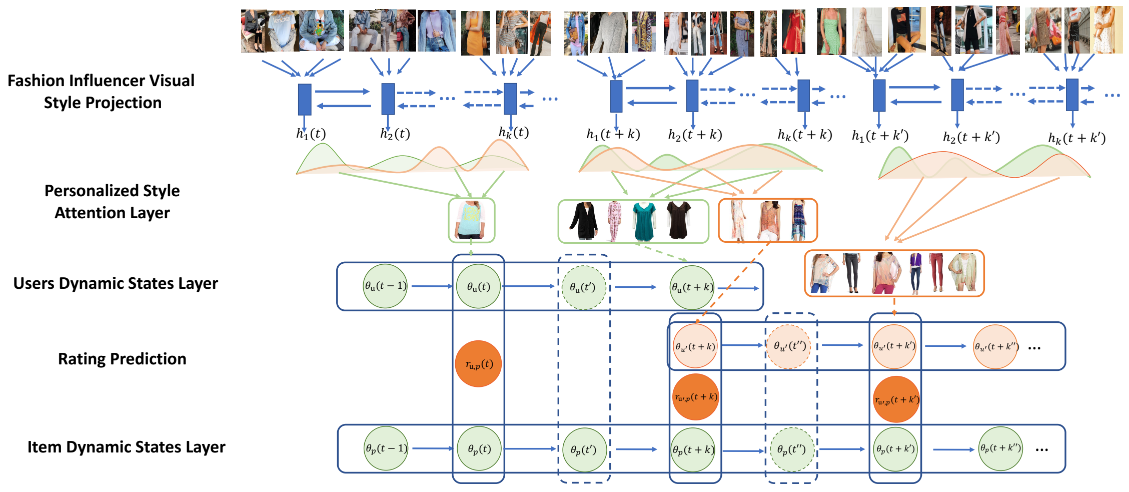


Figure 6.3: Visual influence-aware recurrent fashion recommendation framework. The vertical solid/dashed boxes represent the user dynamic states with/without ratings and the item states at the same time.

pects and degrees. For example, users who like brighter colors would prefer bloggers who post fashionable clothes in similar colors. And this influence could be highly-receptive for some users or not at all for others. Hence, we propose to model this heterogeneity in influence of user preference towards each blogger through a visual personalized attention layer. Concretely, for each user, we hypothesize that if the user's purchased products are visually similar to fashion features across time, the user is more likely to be influenced by the fashion features. Thus, we use *the attention weights [189] capture the fashion aspects that a user prefers and the visual distance models how deep the user is influenced by the extracted fashion features.*

Specifically, given the learned blogger fashion style vectors $(h_1(t), h_2(t), ..., h_K(t))$ at time t, the attention module first transforms each blogger's learned fashion vector through a single perceptron to a lower space:

$$\mathbf{s}_k(t) = sigmoid(\mathbf{E}_s \mathbf{h}_k(t) + \mathbf{b}_s), \tag{6.4}$$

where $\mathbf{E}_s$ and $\mathbf{b}_s$ is the corresponding embedding matrix and bias. Then the attention module compares the similarities between a user u latent influenced visual vector $\mathbf{w}(u)$ and the blogger $\mathcal{B}_k$'s transformed style $\mathbf{s}_k(t)$ by computing the dot products. So the attention weights $\alpha_k(u,t)$ of user u to blogger $\mathcal{B}_k$ is calculated by the softmax of the similarity:

$$\alpha_k(u,t) = \frac{exp(\mathbf{w}(u)^T \mathbf{s}_k(t))}{\sum_{k'}(exp(\mathbf{w}(u)^T \mathbf{s}_{k'}(t)))}. \tag{6.5}$$

Then the attention module computes each user's *influence-aware visual style* vector $\hat{\mathbf{h}}(u,t)$ as the weighted sum of the blogger's fashion style:

$$\hat{\mathbf{h}}(u,t) = \sum_k \alpha_k(u,t)\mathbf{h}_k(t), \tag{6.6}$$

where the user u latent influenced visual vector $\mathbf{w}(u)$ is a trained parameter to minimize the distance between user's influence-aware visual style and user's previously purchased items by:

$$min \sum_t \sum_u ||\mathbf{v}(u,t) - \hat{\mathbf{h}}(u,t)||_F, \tag{6.7}$$

where $\mathbf{v}(u,t)$ is the user visual vector based on the visual features of the user's purchase history. Specifically, for unity and to capture the same visual features, $\mathbf{v}(u,t)$ is calculated by

$$\mathbf{v}(p) = \mathbf{E}_m \mathbf{m}(p), \quad \mathbf{v}(u,t) = \frac{\sum_{p \in P_u(t)} \mathbf{v}(p)}{|P_u(t)|},$$

which is similar to the blogger's visual vector calculation (Equation 6.2). $\mathbf{E}_m$ is the same embedding in Equation 6.1. As a result, each user u has a learned influence-aware visual style $\hat{\mathbf{h}}(u,t)$ that captures the user *personal* preferred visual fashion features at time t. In the next section, we discuss how to integrate this personal influence into time-dependent fashion recommendation.

6.2.3 Visual Influence-aware Recurrent Network

Fashion acts as a strong influence factor for user purchase preference across different time periods. In addition to the fashion influence previously described, other static and dynamic factors may also impact user purchase preferences. Examples include the brand of an item (a static feature), the personal taste drift in clothing materials (a dynamic factor), and so on. In this section, we incorporate these additional factors with the extracted personal fashion influence to build a joint visual influence-aware recurrent network for user clothing recommendation.

To capture these dynamic and stationary states, especially the visual influence from fashion bloggers, we extend recurrent recommendation networks (RRN) [94] with visual influence-aware personalized fashion factors. Specifically, an RRN can capture temporal dependencies for both users and items with a dynamic user state and a dynamic item state. Besides modeling user internal dynamic and stationary states, our proposed FIRN also considers the external fashion drift and its influence for users. Concretely, for the user state, suppose $\mathbf{r}_u(t)$ is the rating vector for user u. That is, $r_{u,p}(t) = r$ is the user rates item p with score r at time t otherwise $r_{u,p}(t) = 0$. We denote τ_t as the wallclock at time step t and $\mathbf{1}_{newbie}$ as the indicator of whether the user is new. So the constructed input for user at time t in FIRN is [94]:

$$\mathbf{x}_u(t) := [\mathbf{r}_u(t), \mathbf{1}_{newbie}, \tau_t, \tau_{t-1}]. \tag{6.8}$$

Then the personalized fashion-aware user vector is:

$$\mathbf{f}_u(t) = \mathbf{E}_u \mathbf{x}_u(t) + \mathbf{E}_f \hat{\mathbf{h}}(u, t), \tag{6.9}$$

where $\mathbf{E}_u$ and $\mathbf{E}_f$ are transformations to be learned to project source and fashion information into the joint user embedding space. Specifically, $\mathbf{E}_u x_u(t)$ represents the latent factor of user personal preference and $\mathbf{E}_f \hat{\mathbf{h}}(u, t)$ is the extracted popular fashion features that the user prefers at time t. The state of u at time t is decided by the user's previous hidden state $\boldsymbol{\theta}_u(t-1)$ and $\mathbf{f}_u(t)$. So:

$$\boldsymbol{\theta}_u(t) := LSTM(\boldsymbol{\theta}_u(t-1), \mathbf{f}_u(t)). \tag{6.10}$$

For an item's time dependent state, similarly, the item vector are calculated by $\mathbf{f}_p(t) = \mathbf{E}_p \mathbf{x}_p(t)$

where $\mathbf{x}_p(t) := [\mathbf{r}_p(t), \mathbf{1}_{newbie}, \tau_t, \tau_{t-1}]$ is the constructed item input. The overall framework for FIRN is shown in Figure 6.3.

Rating Prediction. Besides dynamic states, users/items also contain stationary components across time. So we incorporate the time-dependent user states $u(t)$ and item states $p(t)$ with the stationary state $\tilde{\gamma}_u$ and $\tilde{\gamma}_p$ respectively. Similar to [94], the rating prediction is calculated by:

$$\hat{r}_{u,p}(t) :=< \boldsymbol{\theta}_u(t), \boldsymbol{\theta}_p(t) > + < \tilde{\boldsymbol{\gamma}}_u, \tilde{\boldsymbol{\gamma}}_p > . \tag{6.11}$$

The $\tilde{\gamma}_u$ and $\tilde{\gamma}_p$ are affine functions of γ_u and γ_p which are $\tilde{\boldsymbol{\gamma}}_u = \mathbf{E}'_u \boldsymbol{\gamma}_u + \mathbf{b}_u$, $\tilde{\boldsymbol{\gamma}}_p = \mathbf{E}'_p \boldsymbol{\gamma}_p + \mathbf{b}_p$, where $\mathbf{E}'_u$ ($\mathbf{E}'_p$) is the transformations of user (item) stationary states, and $\mathbf{b}_u$ ($\mathbf{b}_p$) is the corresponding bias term. The stationary part of $\boldsymbol{\gamma}_u$ and $\boldsymbol{\gamma}_p$ is similarly calculated based on the standard factorization. In sum, we build a recurrent recommender FIRN that incorporates both dynamic fashion features from fashion bloggers and user's purchase history to give users a personalized fashion recommendation.

6.2.4 Optimization

In order to predict user ratings that are close to the actual ratings as well as capturing the fashion blogger's visual influence for each user across time, we propose an objective function that jointly learns user's ratings and their visual preference:

$$\underset{\Omega}{\text{minimize}} \sum_{(u,p,t)\in O_{train}} (r_{u,p}(t) - \hat{r}_{u,p}(t|\Omega))^2 + \lambda_1 ||\mathbf{v}(u,t) - \hat{\mathbf{h}}(u,t)||_F^2) + \lambda_2 R(\Omega), \tag{6.12}$$

where $r_{u,p}(t) - \hat{r}_{u,p}(t|\Omega)$ is used to yield predictions that are close to the actual ratings. $||v(u,t) - \hat{h}_k(u,t)||_F^2$ is to ensure we effectively adapt blogger's various visual information for different users in BiLSMT. Here $||\cdot||_F$ is the Frobenius Norm. λ_1/λ_2 is a hyper-parameter which is used to balance the visual/regularizer and rating information. O_{train} are the observed (user, item, timestep) tuples in the training set. Ω is the set of model parameters, and $R(\Omega)$ is the regularization function. Here we use the Frobenius Norm for each model parameter. The optimization method is the same as [94] (subspace gradient descent).

6.3 Experiments

In this section, we conduct experiments on real-world datasets to evaluate the proposed FIRN recommender. Specifically, there are two key research questions: (1) How well does FIRN perform for fashion recommendation; (2) Whether our modeled fashion bloggers implicit visual influence is really helpful for the recommendation, especially compared with other widely used sources of visual information? Besides focusing on answering these two questions, we also conduct an ablation study, as well as explore FIRN performance on different users and their corresponding recommended items to further evaluate the FIRN architecture and the influence of fashion bloggers.

6.3.1 Experimental Setup

Datasets. We require datasets that contain both time-aware visual posts of influential fashion bloggers and user fashion purchases in the same time periods to model the dynamic influence. It is extremely difficult to find available public datasets that satisfy these requirements since few (if any) previous works consider the influence of fashion bloggers in recommendation. So for fashion bloggers, we crawled bloggers and their time-aware visual posts from Instagram which contains many influential and representative fashion bloggers and their rich visual posts [99, 104]. Since other sources of visual fashion could potentially be just as useful as these bloggers, we also consider a collection of images from user purchase histories on Amazon [15] and the AVA dataset of static aesthetic images [91]. For user purchases, we follow previous fashion works that use a public Amazon dataset [15, 91].

- *Instagram.* As an illustration of fashion bloggers and their visual influence in fashion recommendation, we use a list of 100 influential US female fashion bloggers as a seed set of fashion bloggers.[†] While this list is partial and reflects one view on who is influential, it gives us a starting point of many popular fashion bloggers (we further discuss the limitations of this dataset for fashion recommendation in Section 6.4). We crawl each of their Instagram [190] accounts,

[†]https://www.aransweatersdirect.com/blogs/blog/46644481-the-top-100-us-female-fashion-bloggers-to-follow-on-instagram

collecting all posts that overlap in time with the Amazon dataset. Concretely, we crawl the images associated with each post and the posted time, the number of likes for each post, the blogger's comments on the post, and five user comments featured by Instagram (where the comments are from accounts with large followings), resulting in 131,883 *time-aware visual* posts.

- *Amazon.* We use a public large real-world Amazon dataset [11] focusing on the Women's Clothing category that has been widely used for fashion recommendation [15, 91]. The dataset includes both rating history and item images. Concretely, we forcus on women's skirts, dresses, pants and so on (removing Intimates and Socks & Hosiery since they are typically not featured by fashion bloggers). We select items with an image that are rated between Jun. 2011 and Jul. 2014 (overlapping with our Instagram dataset) and their corresponding users. Keeping users with at least two ratings, we finally arrive at a ratings matrix with 22,217 users, 27,244 items and 59,866 ratings.
- *AVA Dataset*: This is a well-known public Aesthetic Visual Analysis (AVA) dataset [92]. It contains over 250,000 images with aesthetic ratings from 1 to 10 and we use the images rated 6-10 as aesthetic visual information for fashion recommendation.

Dataset Inspection. Particularly, we investigate the crawled fashion bloggers from different aspects to confirm its quality. We first examine the user comments towards those bloggers' posts. We find the most frequent unigrams express strong personal affinity – *love* is the most popular followed by *beauty*, *cute*, *like*. Further, unigrams *want*, *get*, and *need* also appear in the top-20 most popular unigrams. Those top frequent words prove bloggers'posts contain user favored visual features and even influence their purchase preference. We then explore the number of users who directly express their likes towards these posts. Figure 6.4 shows the growth in number of posts (blue) and the average number of likes per post (red) from 2011 to 2015. For example, one of our bloggers has a post with 675,000 likes in 2014. The huge amount of likes per post along with the top frequent words in user comments further confirms our intuition of the widely influence of fashion blogger's posts towards user aesthetic preference. Additionally, the fast increases in both

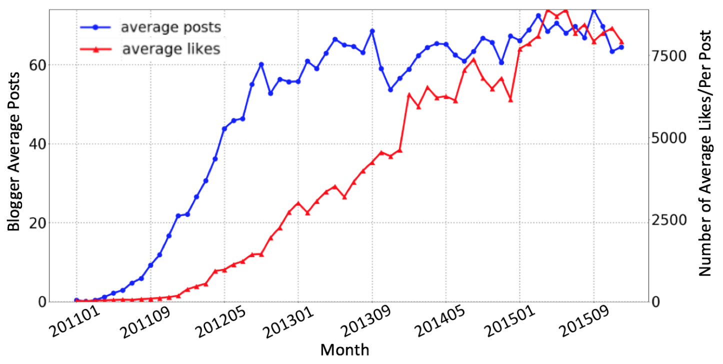


Figure 6.4: Posts and likes for Instagram fashion bloggers in our dataset grew rapidly.

number of posts and average likes per post shows our crawled bloggers are active across time. All those observations ensure the high-quality of our crawled data. In sum, the Instagram dataset naturally contains both *dynamic* and *high aesthetic quality* properties, which makes it potentially valuable for fashion recommendation.

In FIRN, the time variable is used to link Amazon dataset and Instagram dataset, and fashion information is transferred to capture user visual drift. The time intersection for the two datasets is from Jun. 2011 to Jul. 2014. We select the corresponding posts and user records in that time period, and discretize time by month, resulting in 38 time intervals. The choice of granularity, as an important hyper-parameter, is revisited on model performance in Section 6.3.2. We also compare our blogger visual features versus visual images from users time-aware purchased products used in [15] and those derived from the AVA dataset [92] used in [91].

Visual Features. For the visual features in Amazon items ($\mathbf{m}(p)$), Instagram posts ($\mathbf{m}_i(\mathcal{B}_k|t)$) and the AVA images, following previous work [11, 15], we use a convolutional neural network (CNN) proposed by [149] to *unify* the extracted visual features in the three datasets for fair comparison. The CNN is pre-trained by Caffe 1.2 million ImageNet. Particularly, the features that we use are the output of the second fully connected layer in CNN based on their strong performance in previous work [11]. The visual feature vector length is $4,096$.

Baselines. We compare FIRN against the following baselines:

Table 6.2: Model comparison: FIRN is personalized, temporally-aware, visually-aware, and considers the impact of fashion bloggers.

Model	Personalized	Temporally-aware	Visually-aware	Influence-aware
SVD	✓	✗	✗	✗
AutoRecIU	✓	✗	✗	✗
TimeSVD++	✓	✓	✗	✗
SGRU	✓	✓	✗	✗
RRN	✓	✓	✗	✗
NSCR	✓	✗	✓	✓
FSVD	✓	✓	✓	✗
FIRN	✓	✓	✓	✓

SVD [191]: This is a widely used method which achieves robust and strong results in rating prediction. It uses user ratings without considering temporal dynamics. The regularization parameter is 0.01 by cross-validation.

AutoRecIU [192]: This is a recent autoencoder-based method for rating prediction. We use both item-based and user-based AutoRec methods and report the best performing one. The regularization parameter is 1 by cross validation.

TimeSVD++ [93]: One of the most successful models for time-aware recommendation based on matrix factorization, showing strong results across different datasets [94]. It considers temporal dynamics for both users and items. The regularization parameter is 0.01 by cross validation.

SGRU [53]: This method uses a session-based recurrent neural network (RNN) method to capture dynamics in recommendation and has strong results in prediction based on implicit feedback. Here we adapt it to predict ratings for each user. The loss function is mean square error. The drop out rate is 0.5.

RRN [94]: This is a recent state-of-the-art method for time-aware rating prediction. It uses a new RNN method to model long-range dynamics and stationary effects for users and items. The regularization parameter is 16 by cross-validation.

NSCR [193]: This is a recent state-of-the-art method for cross-domain recommendation. It utilizes both user-item attributes and a social network to give an item recommendation. We adapt this method for fashion recommendation, where the social network part is used to model the influence from fashion bloggers to user purchase behavior. Item attributes are represented by item visual vectors and user attributes are denoted by the average visual vectors of their bought items. The social network between users and bloggers is built by their visual similarity between bought items and posts. If the similarity is larger than average, then there is a connect.

FSVD [15]: This is a recent state-of-the-art fashion-specific recommender, where fashion trends are modeled from user purchase history. The method is based on matrix factorization that considers temporal dynamics and item visual information. The regularization parameter is 0.001 by cross-validation.

Ultimately these methods are designed to evaluate the impact of temporal dynamics, visual factors, and FIRN framework for fashion recommendation as shown in Table 6.2.

Table 6.3: FIRN outperforms state-of-the-art methods in terms of RMSE for different time-step granularity. Δ_{RRN} shows the RSME improvement versus the next-best alternative, while Δ_{FSVD} shows the RMSE improvement versus the state-of-the-art fashion recommender FSVD. SVD, AutoRecUI and NSCR have the same performance cross rows since they are not time-aware.

	No time		Time aware			Visual Time & Blogger			Δ	
	SVD	AutoRecUI	TimeSVD++	SGRU	RRN	NSCR	FSVD	FIRN	Δ_{RRN}	Δ_{FSVD}
1 month	2.1392	1.6723	1.0775	1.4146	1.0655	1.1306	1.1146	1.0346	3.09%	7.99%
2 months	2.1392	1.6723	1.0805	1.3371	1.0627	1.1306	1.1126	1.0348	2.79%	7.78%
3 months	2.1392	1.6723	1.0774	1.3125	1.0581	1.1306	1.0983	1.0324	2.56%	6.59%
4 months	2.1392	1.6723	1.0884	1.2263	1.0521	1.1306	1.1187	1.0311	2.10%	8.76%
5 months	2.1392	1.6723	1.0964	1.2108	1.0478	1.1306	1.1197	1.0261	2.17%	9.36%
6 months	2.1392	1.6723	1.1015	1.2708	1.0535	1.1306	1.1293	1.0314	2.21%	9.79%

Metrics. For fashion recommendation, following [94], we split the dataset by time into a training set (Jun. 2011 to Jun. 2013, with 21,112 ratings and 43,870 posts), validation set (Jul. 2013 to Jan. 2014, with 18,089 ratings and 16,452 posts) and test set (Feb. 2014 to Jul. 2014, with 20,665 ratings and 14,390 posts). Our evaluation consists of calculating how bloggers influence each user's ratings for items. So similar to [94], we use the user's average standard root-mean-square error (RMSE) to evaluate rating prediction:

$$RMSE = \frac{1}{|\mathcal{U}_{test}|}(\sum_{u \in \mathcal{U}_{test}} \frac{1}{|P_u(t|\mathcal{U}_{test})|}||r_{u,p}(t) - \hat{r}_{u,p}(t)||_F^2)^{1/2},$$

where $|\mathcal{U}_{test}|$ is the number of users in the test dataset. Here $|P_u(t|\mathcal{U}_{test})|$ represents the number of items that the user u rated at time t in the test dataset.

Reproducibility. All results are reported over the same test set. For a fair comparison, the hidden dimension for all approaches is set to be 100 empirically for a trade-off between performance and computation complexity. Specifically, for RRN and FIRN, the dimension of stationary factors is 90 and the embedding dimension for temporal dynamics is 10. We set the visual dimension to 100 for all methods that use visual information. The hidden dimension of BiLSTM is 100 and $\mathbf{E}_s \in R^{100 \times 10}$. The embedding dimension for FIRN is 10 which is the same as embedding dimension for temporal dynamics. Other hyperparameters are tuned based on the best performance on the same validation dataset. The regularization hyperparameters are tuned by grid search from 0.001 to 30 for different time-step granularity. Specifically, $\lambda_1 = 0.1$ and $\lambda_2 = 12$ for FIRN. Model parameters are first randomly initialized according to truncated normal distribution with mean 0 and standard deviation 0.01. For the optimization, we use mini-batch gradient descent where the max batch size is 100,000, and the corresponding learning rate is determined by grid search in the range of {0.00001, 0.0001, ...,0.1}.

6.3.2 Recommendation Performance of FIRN

We begin by investigating the overall performance of FIRN versus alternatives as shown in Table 6.3. The rows of the table capture different time-step granularaties. Overall, FIRN results in the best RMSE, with a 6% to 10% improvement versus the state-of-the-art fashion recommender FSVD and a 2% to 3% improvement versus the next-best approach (RRN in this case) across rows.

Comparing in different time step granularities, FIRN consistently outperforms other methods, with relatively little change indicating the stability of FIRN in time-step granularaties. The best (lowest) RMSE of FIRN is gained when the time interval is five months, which suggests that fashion trends do not typically change abruptly in a short time period. We also observe that all of the time-aware methods result in significantly lower RMSEs than static methods (SVD and AutoRecUI), verifying the importance of modeling temporal dynamics of fashion preferences. Of the time-aware methods, SGRU does not perform very well, most likely since it is designed for implicit recommendation rather than rating prediction. Furthermore, both RRN and FIRN outperform the other methods, which indicates RRN is effective to capture dynamic changes compared with the other time-aware methods. More importantly, while FIRN and RRN have similar architecture, FIRN consistently performs better than RRN which highlights that incorporation of dynamic visual features from fashion bloggers gives FIRN its edge versus the temporal methods. Specifically, FIRN outperforms TimeSVD++ by 5.52% on average, SGRU by 26.35% on average, and RRN by 2.49% on average. We also observe that FIRN outperforms the state-of-the-art fashion recommendation method FSVD by 8.38% on average. It further highlights the efficacy of our model framework and the importance of incorporating fashion bloggers.

6.3.3 Influence of Fashion Bloggers

An important question is does the modeled blogger's implicit visual influence really help for the recommendation? Or put differently: does this give better performance in fashion recommendation compared with using other sources of visual information? In Section 6.3.2, we showed that FIRN consistently outperforms the corresponding alternative without the blogger's visual fashion information (RRN). This indicates that our modeled bloggers implicit visual influence improves the recommendation performance. However, is the improvement based on the blogger's implicit visual influence or could another visual source achieve similar performance? Here, we compare FIRN versus two alternatives that incorporate two widely used sources of visual information in fashion recommendation:

∘ *FIRN-PH*: The first approach replaces the posts of fashion bloggers with visual features derived from the users' Purchase History. Specifically, we use the average visual features of each user's purchase items in each time period.

∘ *FIRN-AVA*: The second approach uses the AVA dataset [92] as the indicator for user aesthetic

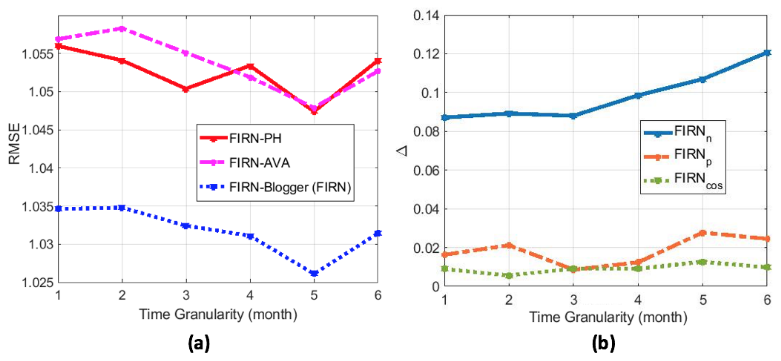


Figure 6.5: (a) RMSE of FIRN with different visual information; (b) Differences between FIRN and its variations.

preference [91]. Since AVA is static, we use the highest rated images (with ratings of 6-10) and cluster those images by their ratings. Each cluster acts as a virtual blogger in our FIRN framework.

Figure 6.5(a) shows RMSEs for the two models and FIRN-Blogger (which is FIRN) – one using purchase history, one using AVA, and our original FIRN approach with fashion bloggers. FIRN consistently results in the best performance, illustrating that our modeled blogger's implicit visual influence brings the largest improvement in fashion recommendation. Particularly, the out-performance of FIRN compared with using purchase history indicates the high aesthetics quality of fashion blogger's posts. FIRN performs better than AVA. One likely reason is that as user's visual interest changes over time with fashion trends, the fashion blogger are able to reflect this evolution versus the static visual information in AVA. Interestingly, though purchase history and AVA outperform the baseline RRN which does not use visual information, they do not perform the best separately, which shows the dynamic and aesthetic visual properties are *mutually correlated* and enhance each other for fashion recommendation. This further shows that the learned implicit visual influence from bloggers, with the unique properties of containing both high-quality and dynamic visual features, indeed captures more visual information for user clothing recommendation than the other two visual datasets.

Table 6.4: Performance of different methods for the most fashion-sensitive users.

Most Fashion-Sensitive Users	No time		Time aware			Visual Time & Blogger			Δ	
	SVD	AutoRecUI	TimeSVD++	SGRU	RRN	NSCR	FSVD	FIRN	Δ_{RRN}	Δ_{FSVD}
100	2.0278	1.6964	1.0078	1.4666	1.0684	1.3828	1.1121	0.9716	9.68%	14.05%
500	2.0558	1.6799	1.0828	1.4666	1.0631	1.2552	1.1198	0.9852	7.79%	13.46%
1,000	2.0763	1.6589	1.0642	1.4398	1.0487	1.3156	1.1289	0.9940	5.47%	13.49%
5,000	2.1291	1.6762	1.0850	1.3279	1.0587	1.0744	1.1661	1.0225	3.63%	14.36%

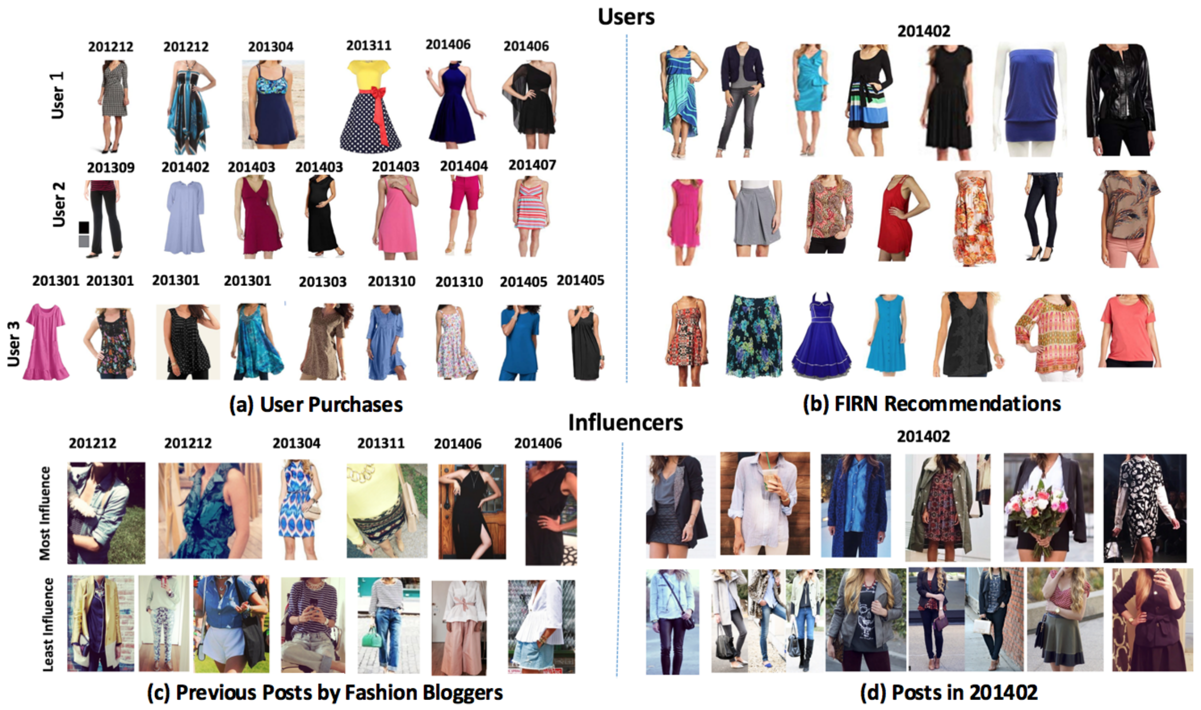


Figure 6.6: (a) A selection of items purchased by three users. FIRN makes the best predictions for User 1 (low RMSE); User 2 and User 3 have higher RMSEs; (b) Corresponding recommendations for those users in Feb. 2014 (201402); (c) The least and most influential bloggers for User 1; (d) Examples of bloggers posts in 201402.

6.3.4 Ablation Study

This section evaluates the key design choices of FIRN: the impact of personalized attention layer, the impacts of visual distance between items and blogger's posts, and the visual distance choice in loss function. Concretely, method $FIRN_n$ uses a non-personalized attention layer (i.e. $\alpha_k(t) = softmax(w_k^T s_k(t))$) and the loss function is to minimize $\sum_{(u,p,t)}((r_{u,p}(t) - \hat{r}_{u,p}(t|\Omega))^2 + \lambda_2 R(\Omega)$ which does not consider visual distance between items and blogger's posts. The second method $FIRN_p$ uses a personalized attention layer but its loss function also does not consider the visual distance. The third method $FIRN_{cos}$ uses a cosine similarity of user and blogger visual vectors rather than Frobenius Norm (i.e. minimize $\sum_{(u,p,t)}((r_{u,p}(t) - \hat{r}_{u,p}(t|\Omega))^2 - \lambda_1 cos(v(u,t), \hat{h}_k(u,t)) + \lambda_2 R(\Omega))$.

Figure 6.5(b) shows the RMSE differences between FIRN and its variations (Δ is RMSE of FIRN variations minus RMSE of FIRN) by different time-step granularity. We observe that FIRN gives an average improvement of 9.8% in RMSE compared with $FIRN_n$. $FIRN_p$ performs better than $FIRN_n$ with a average improvement of 8.0%. This confirms the importance of utilizing personal attention in FIRN for fashion recommendation. Moreover, $FIRN$ offers an 1.8% improvement compared with $FIRN_p$. It illustrates the importance of measuring visual distance between user bought items and blogger's posts. Interestingly, we find FIRN just slightly performs better than $FIRN_{cos}$, which indicates cosine and Frobenius Norm have similar effects for capturing personalized visual preference. It is reasonable since both cosine and Frobenius norm measure linear distance.

6.3.5 Fashion-Sensitive Users

Although FIRN improves the recommendation performance across users, a concern is that users may be differently influenced by fashion and thus a general good prediction can not ensure the recommended items are *fashionable* (e.g., a good prediction for a non-fashion influenced user can not show the recommended items are fashionable). In this section, we examine the FIRN performance for the fashion-sensitive users to further evaluate the fashion recommendation quality. Since it is hard to directly measure the impact of fashion to individual users without knowing user personal information, according to the experimental results from [101], as a proxy, we assume that in a small time-step granularity t, if a user purchased an item which is visually similar to posts that a blogger shares in same time period t, and such similarity is consistent for a long time period, then there is a higher probability that the user is more influenced by fashion bloggers/fashion.

Hence, based on the assumption, we sort users by the visual distance between the user purchased items and bloggers' posts across a long time period T:

$$d_v(u, \cup\mathcal{B}_k|T) = \frac{1}{|T|}\sum_{t\in T}\min_{p\in P_u(t), B_k\in\cup\mathcal{B}_k}(||\mathbf{m}(p) - \mathbf{m}(\mathcal{B}_k|t)||_F), \tag{6.13}$$

where $m(\mathcal{B}_k|t) = \frac{\sum_{i\in\mathcal{I}(\mathcal{B}_k|t)}\mathbf{m}_i(\mathcal{B}_k|t)}{|\mathcal{I}(\mathcal{B}_k|t)|}$ is the blogger's visual features at time t. Since the influenced users could buy one or more items that are not similar to the influenced blogger, here we use min across user bought items $p \in P_u(t)$ to find the smallest distance at time t. Similarly, the min is also calculated across bloggers $B_k \in \Pi$. The sum for the totel time T (38 time intervals in our case) can drop the probability that we include uninfluenced users who coincidentally bought visually similar items.

Table 6.4 reports the performance of FIRN for the most fashion-sensitive users according to Equation 6.13 for different thresholds. For these users who bought items that are most similar to bloggers (top 100 users), we find an RMSE of 0.9716 which is 9.68% better than the next-best alternative and 14.05% better than the fashion-aware FSVD. While as more users are considered, FIRN still maintains its superiority versus the next-best alternative. Furthermore, We observe that most other methods show approximately flat performance for users with different $d_v(u, \cup\mathcal{B}_k|T)$ while FIRN shows approximate monotonous relationship, indicating FIRN could gain a better performance for users with small $d_v(u, \cup\mathcal{B}_k|T)$. One likely reason is that FIRN considers the visual distance between bloggers and user purchased products to capture the visual similarity in Equation 6.13. Those results demonstrate FIRN achieves a significant better performance to these fashion-sensitive users.

6.3.6 Case Study

To further investigate the implicit visual influence for FIRN recommendation, we also look at the FIRN recommended items. In particular, we focus on three specific users as shown in Figure 6.6, for whom FIRN provides varying recommendation quality: user 1 (good, RMSE=0.0778), user 2 (medium, RMSE=0.9767), and user 3 (poor, RMSE=2.7621). Each row of Figure 6.6(a) shows the purchase history for one user, and Figure 6.6(b) shows the recommendations for the corresponding user in the same row made by FIRN for Feb. 2014 (which is the earliest time in the test set). Figure 6.6(c) shows the predicted most influential blogger (first row) and least influential blogger (second row) for user 1 (by the value of the attention weight $\sum_t \alpha_k(u,t)$). Figure 6.6(d) shows the posts by the most influence blogger and a popular blogger ($\sum_u \alpha_k(u,t)$ is the largest) in Feb. 2014 for comparison.

Comparing Figure 6.6(a) and (c), we observe that the most influential blogger for user 1 (as learned by the attention weights $\sum_t \alpha_k(u,t)$) shares similar style/colors posts to user 1's purchased items. For example, the blogger posted a paisley dress (in the second post) and the blue/black color pairing in the second post. It confirms FIRN *does* learn visual features from bloggers that are similar to users. Furthermore, for the recommended items in Figure 6.6(b), we observe FIRN recommends items that reflect both fashion trends revealed by bloggers and the user's purchase history. For example, for user 1, the recommended black suit and black jacket are similar as bloggers, and blue/black color pairing is similar her purchase history. For user 3 whose RMSE by FIRN is poor, we observe the recommended items are visually diverse (stylish according to the fashion bloggers and visually related to the user's purchase history). This shows the potential of FIRN to recommend stylish clothing based on the current posts by fashion bloggers.

6.4 Summary and Discussion

In this chapter, we have focused on the fashion blogger's implicit visual influence towards user's purchase preferences. We propose a novel recurrent neural fashion recommender – FIRN – which utilizes fashion bloggers dynamic visual information to extract fashion features and gives users *personalized visual influence-aware fashion recommendations*. The experimental results show the potential of incorporating dynamic visual fashion trends from fashion bloggers into a recommender, particularly for those users who are most fashion-sensitive, and utilizing fashion bloggers can bring greater improvements in fashion recommendation. We also release a large *time-aware high-quality visual dataset* for reproducibility and further research. In our continuing work, we focus on two major issues:

New Fashion-Related Suggestions. We are interested to explore how to best create new recommenders tailored for different kinds of users – the fashion-sensitive, the fashion-neutral, and the unaffected (such as user 3 in Figure 6.6). For example, are unaffected users interested in seeing trendy recommendations (derived from fashion bloggers) alongside traditional purchase history recommendations (that are more suited to their current preferences)? Can we build recommenders that subtly move users from one group to another, say by tuning the attention weights over time? Furthermore, since fashion as a cultural phenomenon has a huge social influence, we can not declare that the unaffected users are not influenced by fashion at all. The fashion influence for these users could merely by invisible when the time span is relatively short. Since FIRN learns personal influence for each user, FIRN can benefit from the training data which has a relative long time span.

Expanding the Source of Fashion Trends. One key limitation of the current work is the reliance on a single source of fashion trends – Instagram fashion bloggers. We have seen how the fashion features highlighted by these popular and influential bloggers can improve recommendation quality, but we are interested to explore a wider range of fashion bloggers in our future work. As the growth of visual social media continues, we anticipate the influence of these unique fashion personalities will grow even more in comparison to traditional retailers, designers, and celebrities [101], enabling even more powerful recommenders based on their fashion leadership.

7. CONTENT-COLLABORATIVE DISENTANGLEMENT REPRESENTATION LEARNING FOR ENHANCED RECOMMENDATION[‖]

Successfully modeling item and user relations sheds light on the powerful influence of content for user preference towards items. In this chapter, we investigate the situation when multiple sources of content are considered and focus on tackling the duplication between content and collaborative signals. We propose to disentangle representations learned from user behavior data and content information. Specifically, we propose a novel two-level disentanglement generative recommendation model (DICER) that supports both content-collaborative disentanglement and feature disentanglement: for the content-collaborative disentanglement, DICER decomposes the features by their marginal distributions based on content and user-item interactions, to ensure the learned features from each type are statistically independent. For feature disentanglement, by decomposing the Kullback-Leibler divergence, we theoretically show that extracted features within each type are disentangled at a granular level. Furthermore, DICER utilizes a co-decoder that simultaneously decodes the content and user-item interactions to ensure the high-quality of learned features. Through extensive experiments on three real-world datasets, results show that DICER significantly outperforms other state-of-the-art methods by 13.5% in NDCG and 14.4% in hit ratio on average.

7.1 Introduction

One of the fundamental challenges for recommender systems is to learn user and item representations that can uncover user preference towards items. Many recent efforts adopt deep approaches [64, 194] over both *collaborative features* based on user-item interactions (e.g., clicks or likes) and *content information* about the users and items (e.g., user ages or item images) [4, 50, 111–113] to build user and item representations, as shown in Figure 7.1(a). While encouraging, the learned user and item features derived from these collaborative and content-based perspectives can be *entangled*

[‖]Reprinted with permission from "Content-Collaborative Disentanglement Representation Learning for Enhanced Recommendation" by Yin Zhang, Ziwei Zhu, Yun He, James Caverlee, 2020. Proceedings of The Fourteenth ACM Conference on Recommender Systems. Copyright 2020 by ACM. DOI:https://doi.org/10.1145/3383313.3412239 by intermixing the influence from each, harming recommendation quality.

For example, a user, and also many similar users, may prefer a dress because of its visual appearance, price, and high quality. The known content information is the dress images. If the user-item interactions and the dress image are considered separately to learn the features that influence user preference towards items, the collaborative features and content features could be highly correlated. In essence, both the collaborative and content features could redundantly encode the visual characteristics of the dress, meaning there is less capacity to focus on learning other features (like price) that could influence user preference towards items. Hence, learning user preferences based on both content-based features and collaborative features could lead to *feature duplication and high feature correlation*, limiting the representation capability for modeling collections of users with diverse interests. Furthermore, the feature duplication and high correlation among features can also result in overweighting these correlated features in learning user preference, leading to sub-optimal performance and unstable recommendations.

Therefore, we propose to *disentangle representations learned from user behavior data and content information* to improve the integrated user and item representation quality for improved recommendation. Such disentanglement representation learning – which aims to learn disentangled features such that any one feature is relatively not influenced by changes in other features – has recently demonstrated powerful and robust performance in many areas, especially in computer vision [116–119]. However, there is little work on disentangled representation learning in recommendation [124], and none on recommendation when both user behavior data and content information are available. Disentanglement representation learning poses unique challenges in this context:

- First, most existing disentanglement problems target areas where the features are explicitly known (e.g., shape or color features of an image). However, user-item interactions do not necessarily map to specific apriori collaborative features. Hence, this heterogeneity between implicit features in user-item interactions and explicit features in content information makes it extremely difficult to capture the disentanglement *across* content and collaborative features.

- Second, disentanglement across content and collaborative features only ensures that the learned representations from each type are different. However, the specific features *within* the collaborative features could still be highly entangled with each other, and similarly for the content-based features. Hence, we also face the challenge of granular-level disentanglement, to simultaneously learn both disentangled and high-quality features within collaborative and content features for improved recommendation.

With these challenges in mind, we propose a novel two-level disentanglement approach called DICER – DIsentangling Content-aware collaborative filtering for Enhanced Recommendation – that supports both content-collaborative disentanglement and feature disentanglement based on the structure of a variational auto-encoder. For content-collaborative disentanglement, DICER decomposes the features that influence user preference into content features and content disentangled collaborative features, then utilizes their marginal distributions to learn disentangled features for each type. For feature disentanglement, we theoretically show that each extracted feature in DICER is disentangled with other extracted features by decomposing the Kullback-Leibler divergence via statistical independence properties. Furthermore, DICER is characterized by a co-decoder that simultaneously decodes the content and user-item interactions to ensure the high-quality of both learned content and collaborative features. Through extensive experiments on three real-world datasets, results show that DICER significantly outperforms other state-of-the-art methods by 13.5% in NDCG and 14.4% in hit ratio on average. We also find that DICER captures relatively independent features through disentanglement measurement and visualization.

7.2 Method

Problem Statement. Suppose we have a set of users $u \in \{1, 2, ..., U\}$ and items $i \in \{1, 2, , ...I\}$. To learn user preference, we have two types of information: (1) User-item interactions capturing the implicit feedback $x_{u,i}$ from user u towards item i, e.g., based on clicks, likes, or purchases. $x_{u,i} = 1$ indicates positive feedback, whereas $x_{u,i} = 0$ means the corresponding feedback is miss-

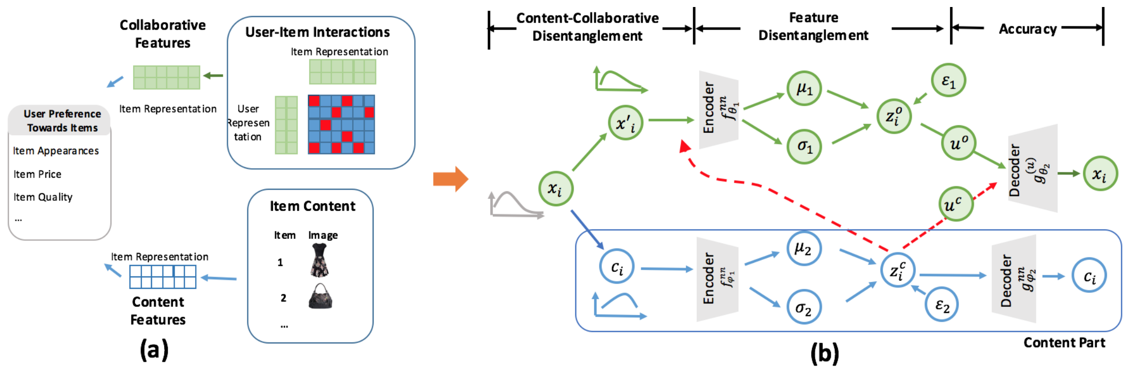


Figure 7.1: Disentanglement motivation in content-aware recommendation and our proposed two-level disentanglement generative recommendation model framework. (a) Both collaborative features learned from user-item interactions and content features learned from content information (like images) can be used to discover user preference. (b) DICER first disentangles the user implicit feedback x_i to content information c_i and content disentangled collaborative information x'_i. Then within each type (either collaborative or content), we further disentangle learned features at a granular level (feature disentanglement) to improve the capacity of user and item representations.

ing; and (2) Item content information $\mathbf{c}_i$ for each item i. This content could correspond to item images, reviews, descriptive text, or other item-specific information. Our task is to consider both user-item interactions and item content information to learn their integrated disentangled representations of users and items for improved recommendation.*

Approach. As we have argued, evidence from both user-item interactions and content information can lead to entangled representations. Hence, we propose a two-level approach called DICER that first disentangles features *between* content and collaborative features – called content-collaborative disentanglement, and then disentangles each feature *within* the collaborative features (and each feature within the content features) – called feature disentanglement. Specifically, as shown in Figure 7.1(b), the proposed DICER approach contains two key variables: *content features* $\mathbf{z}^c_i$ extracted from item content $\mathbf{c}_i$, and *content disentangled collaborative features* $\mathbf{z}^o_i$ extracted from user-item interactions $\mathbf{x}_i$.

*We could also consider user content information alone (instead of item content information), or both user and item information together in addition to user-item interactions. We discuss both of these scenarios in Section 7.2.4.

Table 7.1: Notation of DICER.

Notation	Explanation
$\mathbf{z}_i^o$	representation of learned content disentangled collaborative features, $\mathbf{z}_i^o \in R^{K_1}$
$\mathbf{x}_i$	item i interactions with users
$\theta = \{\theta_1, \theta_2\}$	trainable variables for user-item feedback θ_1/θ_2 is the encoder/decode trainable variables
$\mathbf{z}_i^c$	representation of learned content features, $\mathbf{z}_i^c \in R^{K_2}$
$\mathbf{c}_i$	item i content vector
$\phi = \{\phi_1, \phi_2\}$	trainable variables for item content ϕ_1/ϕ_2 is the encoder/decode trainable variables

7.2.1 Content-Collaborative Disentanglement

We begin by disentangling the information that is learned from content and user-item interactions, to ensure $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ capture different aspects of user preference. For example, suppose we have item images as the content information, and $x_{ui} = 1$ (that user u likes item i) due to the item's visual appearance and price. The content-collaborative disentanglement aims to learn the visual aspect from item images, leaving price-oriented features (that may be hard to precisely learn from images) to user-item interactions. In this way, $\mathbf{z}_i^o$ can *discover* other useful features that are not captured by $\mathbf{z}_i^c$.

To model this content-collaborative disentanglement, we propose to start by modeling the *joint distribution* of all the item influence features $\mathbf{z}_i \in R^K$ (the features that influence the user preference towards items), in order to connect user-item interactions and item content information. Then we decompose the joint distribution to extract disentangled features from content and user-item interactions.

Concretely, we first model the user feedback towards items (e.g., click history) $\mathbf{x}_i$ that is generated from all the features $\mathbf{z}_i$ with the following distribution:

$$p_\theta(\mathbf{x}_i) = \mathbb{E}_{p(\mathbf{z}_i)} p_\theta(\mathbf{x}_i|\mathbf{z}_i), \tag{7.1}$$

where θ is the set of model parameters for modeling user-item interactions.

Then, to utilize both the item content and user-item interaction information, and also disentangle features derived from these two types, we propose to decompose $\mathbf{z}_i$ to be the content features $\mathbf{z}_i^c$ derived from item content $\mathbf{c}_i$ and the content disentangled collaborative features $\mathbf{z}_i^o$ derived from user-item interactions $\mathbf{x}_i$. Therefore, the distribution of $\mathbf{z}_i$ can be expressed as the joint distribution of $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$, i.e. $p(\mathbf{z}_i) = p(\mathbf{z}_i^c, \mathbf{z}_i^o)$. More importantly, to capture the disentangled features from the two types, similar as many disentanglement approaches [121, 124], we set the extracted features from the content and user-item interactions to be statistically independent:

$$p(\mathbf{z}_i) = p(\mathbf{z}_i^c, \mathbf{z}_i^o) = p(\mathbf{z}_i^c)p(\mathbf{z}_i^o). \tag{7.2}$$

That is, based on the disentanglement between $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$, we can further decompose the joint distribution of $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ to be their marginal distributions. Then, with the disentanglement, our model Equation (7.1) can be rewritten as:

$$\begin{aligned} p_\theta(\mathbf{x}_i) &= \mathbb{E}_{p(\mathbf{z}_i)} p_\theta(\mathbf{x}_i|\mathbf{z}_i) = \int p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o) p(\mathbf{z}_i^c, \mathbf{z}_i^o) d(\mathbf{z}_i^c, \mathbf{z}_i^o) \\ &= \int p(\mathbf{z}_i^c) \int p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o) p(\mathbf{z}_i^o) d\mathbf{z}_i^o d\mathbf{z}_i^c = \mathbb{E}_{p(\mathbf{z}_i^c)} \mathbb{E}_{p(\mathbf{z}_i^o)} p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o), \end{aligned} \tag{7.3}$$

where $p(\mathbf{z}_i^o) = p(\mathbf{z}_i^o|\mathbf{z}_i^c)$ is based on their statistical independence. With Equation (7.3), to predict user preference with the known $x_{u,i}$ and $\mathbf{c}_i$, we discuss the two main components: $p(\mathbf{z}_i^c)$ and $p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o)$ in Equation (7.3), respectively.

For content feature distribution $p(\mathbf{z}_i^c)$, it is modeled based on the item content information $\mathbf{c}_i$:

$$p_\phi(\mathbf{c}_i) = \mathbb{E}_{p(\mathbf{z}_i^c)} p_\phi(\mathbf{c}_i|\mathbf{z}_i^c), \tag{7.4}$$

where ϕ is the model parameters for modeling the content information. Thus, we can use this content information to get the $\mathbf{z}_i^c$ distribution, then capture the $\mathbf{z}_i^o$ distribution based on both $\mathbf{x}_i$ and $\mathbf{z}_i^c$.

For $p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o)$, we model it as:

$$p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o) = \prod_{x_{i,u} \in \mathbf{x}_i} p_\theta(x_{i,u}|\mathbf{z}_i^c, \mathbf{z}_i^o).$$

which is similar to the VAE-based method [123, 195]. Then the probability of user u's preference to item i is $p_\theta(x_{i,u}|\mathbf{z}_i^c, \mathbf{z}_i^o)$, which can be derived by a user-aware non-linear transformation, such as a feed-forward neural network. We discuss the details in Section 7.2.3. Thus, we can first learn $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$, and use the decoder to capture the $p_\theta(\mathbf{x}_i|\mathbf{z}_i^c, \mathbf{z}_i^o)$.

Variational Inference. To estimate the model parameters θ in Equation (7.3), we follow the VAE-based paradigm. Note here, different from traditional VAE based methods [195] that can directly estimate the posterior distribution of $p(\mathbf{z}_i|\mathbf{x}_i)$ for each data point, or the conditional VAE [196] that knows part of the data information (i.e., labels of the data), our case is more complex. That is, $\mathbf{x}_i$ is related to the joint distribution of latent factors $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$, where $\mathbf{z}_i^c$ is extracted from item content information. Thus, we need to calculate the evidence lower bound (ELBO) based on both $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$, their independent relations, and their relations to $\mathbf{x}_i$ and $\mathbf{c}_i$.

Concretely, $\mathbf{x}_i$ is generated based on the joint distribution of $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$, where $\mathbf{z}_i^c$ can be estimated by item content (such as images or text-based descriptions). The ELBO of $lnp(\mathbf{x}_i)$ can be written as:

$$lnp_\theta(\mathbf{x}_i) \geq \mathbb{E}_{p(\mathbf{z}_i^c)}[\mathbb{E}_{q_\theta(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c)}(lnp_\theta(\mathbf{x_i}|\mathbf{z}_i^c, \mathbf{z}_i^o)) - D_{KL}(q_\theta(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c)||p(\mathbf{z}_i^o))] \equiv L(\mathbf{x}_i; \theta). \quad (7.5)$$

Since $\mathbb{E}_{q(\mathbf{z}_i^c)}[\cdot]$ and $\mathbb{E}_{q(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c)}[\cdot]$ are intractable, we utilize the variational inference and reparametrization trick [123]. Details are discussed in Section 7.2.3.

To estimate ϕ in Equation (7.4), for consistency, we also use the VAE-based paradigm. The corresponding ELBO is:

$$lnp_\phi(\mathbf{c}_i) \geq \mathbb{E}_{q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)}(lnp_\phi(\mathbf{c}_i|\mathbf{z}_i^c)) - D_{KL}(q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)||p(\mathbf{z}_i^c)) \equiv L(\mathbf{c}_i; \phi). \quad (7.6)$$

Through Equation (7.6), we can get an estimation of the item content features $\mathbf{z}_i^c$. Then we transform it into the user-item space to help learn user preference towards this item through Equation (7.5).

7.2.2 Feature Disentanglement

The content-collaborative disentanglement ensures $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ learn different information from content and user-item interactions. However, the extracted features in $\mathbf{z}_i^c$ (and in $\mathbf{z}_i^o$) at a granular level could also be entangled and confound with each other, making the recommendation unstable and difficult to generalize. Thus, to ensure the quality of learned representations, we also aim to disentangle each extracted feature in $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ at a granular level. For example, we might want to learn features like color and shape from an item image, where changes to each feature (e.g., the color) do not strongly depend on other feature changes (e.g., the shape). To do so, considering each latent dimension represents a single item feature, we disentangle each dimension of the item representation to extract independent features. That is, we make a feature disentanglement that forces $z_{i,k}^c, \forall k$/ $z_{i,k}^o, \forall k$ to be statistically independent. $z_{i,k}^c$ is the k_{th} value in the vector $\mathbf{z}_i^c$. Similar reasoning holds for $z_{i,k}^o$. The user feature disentanglement is jointly modeled with item disentanglement through a decoder of user-item interactions (for details see Section 7.2.3).

More importantly, in DICER, the content-collaborative disentanglement ensures the correlation $Corr_i(\mathbf{z}_i^c, \mathbf{z}_i^o) = \mathbf{0}$ (based on the Equation (7.2)). *Thus, the feature disentanglement inside $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ can further ensure all features in $\mathbf{z}_i$ that are learned in DICER are independent with each other:* $Corr_i(z_{i,k}, z_{i,j}) = 0, \ \forall k \neq j$.

Concretely, to enforce feature independence inside $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$, we have:

$$q_\theta(\mathbf{z}_i^o|\mathbf{z}_i^c) \approx \prod_{k=1}^{K_1} q_\theta(z_{i,k}^o|\mathbf{z}_i^c), \quad q_\phi(\mathbf{z}_i^c) \approx \prod_{k=1}^{K_2} q_\phi(z_{c,k}^c). \tag{7.7}$$

Let's look at each equation separately.

Feature Disentanglement in $\mathbf{z}_i^o$. For the first equation in Equation (7.7), the aggregated posterior distribution of $q_\theta(\mathbf{z}_i^o|\mathbf{z}_i^c) = \int_{x_i} q_\theta(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c) p_{fdata}(\mathbf{x}_i) d\mathbf{x}_i$, where $p_{fdata}(\mathbf{x}_i)$ is the user feedback distribution. The posterior distribution captures the aggregated structure of the latent variables based on the user feedback distribution [197]. Therefore, the KL term in Equation (7.5) can be decomposed as:

$$\mathbb{E}_{p_{fdata}(\mathbf{x}_i)}[D_{KL}(q_\theta(\mathbf{z}_i^o|\mathbf{x}_i,\mathbf{z}_i^c)||p(\mathbf{z}_i^o))] = I_q(\mathbf{x_i};\mathbf{z_i^o}) + D_{KL}(q_\theta(\mathbf{z}_i^o|\mathbf{z}_i^c)||\Pi_k q_\theta(\mathbf{z}_{i,k}^o|\mathbf{z}_i^c)) + \sum_k D_{KL}(q_\theta(\mathbf{z}_{i,k}^o|\mathbf{z}_i^c)||p(\mathbf{z}_{i,k}^o)), \tag{7.8}$$

where $I_q(\mathbf{x_i};\mathbf{z_i^o})$ stands for the mutual information (MI) [198]. A similar decomposition can be found in [121, 198]. For each term in Equation (7.8): (1) the index-code MI $I_q(\mathbf{x_i};\mathbf{z_i^o}) = D_{KL}(q_\theta(\mathbf{z}_i^o,\mathbf{x}_i|\mathbf{z}_i^c)||q_\theta(\mathbf{z}_i^o|\mathbf{z}_i^c))$ is the mutual information between $\mathbf{x}_i$ and $\mathbf{z}_i^o$ based on the empirical user-item feedback distribution $q_\theta(\mathbf{z}_i^o|\mathbf{x}_i,\mathbf{z}_i^c)p_{fdata}(\mathbf{x}_i)$. As indicated by many recent studies [116, 121, 199], penalizing mutual information through the information bottleneck can encourage feature disentanglement; (2) the second term $D_{KL}(q_\theta(\mathbf{z}_i^o|\mathbf{z}_i^c)||\Pi_k q_\theta(\mathbf{z}_{i,k}^o|\mathbf{z}_i^c))$ is the total correlation. This penalty can encourage statistical independence of the learned latent representation in each dimension of $\mathbf{z}_i^o$ under the condition of $\mathbf{z}_i^c$; (3) the third term $\sum_k D_{KL}(q_\theta(\mathbf{z}_{i,k}^o|\mathbf{z}_i^c)||p(\mathbf{z}_{i,k}^o))$ ensures the learned latent representations in each dimension are close to their corresponding priors, which is known as dimension-wise KL [121].

Thus, based on Equation (7.8), if we use an independent prior distribution for each latent dimension of $\mathbf{z}_i^o$, i.e., $p_\theta(\mathbf{z_i^o}) = \Pi_k p_\theta(\mathbf{z}_{i,k}^o)$, the KL penalty term can encourage the disentanglement of the learned features in $\mathbf{z}_i^o$ from the mutual information and total correlation aspects. At the same time, it also ensures the close distribution between the posterior distribution and the priors by $\sum_k D_{KL}(q_\theta(\mathbf{z}_{i,k}^o|\mathbf{z}_i^c)||p(\mathbf{z}_{i,k}^o))$. Hence, the loss function of Equation (7.5) can be refined by adding a KL penalized parameter $\beta_1 > 1$:

$$L_{\beta_1}(\mathbf{x}_i;\theta) \equiv \mathbb{E}_{q_\theta(\mathbf{z}_i^c)}[\mathbb{E}_{q_\theta(\mathbf{z}_i^o|\mathbf{x}_i,\mathbf{z}_i^c)}(ln p_\theta(\mathbf{x_i}|\mathbf{z}_i^c,\mathbf{z}_i^o)) - \beta_1 D_{KL}(q_\theta(\mathbf{z}_i^o|\mathbf{x}_i,\mathbf{z}_i^c)||p(\mathbf{z}_i^o))], \tag{7.9}$$

to encourage each feature disentanglement of $\mathbf{z}_i^o$.

Feature Disentanglement in $\mathbf{z}_i^c$. For the second equation in Equation (7.7), the aggregated posterior distribution of $q_\phi(\mathbf{z}_i^c) = \int_{c_i} q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)p_{cdata}(\mathbf{c}_i)d\mathbf{c}_i$. The $p_{cdata}(\mathbf{c}_i)$ is the item content data distribution. For the disentanglement of the content based representation $\mathbf{z}_i^c$, similarly, we decompose the KL term in Equation (7.6) as:

$$\begin{aligned}\mathbb{E}_{p_{cdata}(\mathbf{c}_i)}[D_{KL}(q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)||p(\mathbf{z}_i^c))] = I_q(\mathbf{c_i};\mathbf{z_i^c}) + D_{KL}(q_\phi(\mathbf{z}_i^c)||\Pi_k q_\phi(\mathbf{z}_{i,k}^c))+ \\ \sum_k D_{KL}(q_\phi(\mathbf{z}_{i,k}^c)||p(\mathbf{z}_{i,k}^c)).\end{aligned} \tag{7.10}$$

Thus, by using an independent prior distribution for each latent dimension of $\mathbf{z}_i^c$ and penalizing the KL term in Equation (7.6), we can encourage the disentanglement of content features. Similarly, the loss function of Equation (7.6) can be refined by adding a KL penalized parameter $\beta_2 > 1$:

$$L_{\beta_2}(\mathbf{c}_i;\phi) \equiv \mathbb{E}_{q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)}(lnp_\phi(\mathbf{c}_i|\mathbf{z}_i^c)) - \beta_2 D_{KL}(q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)||p(\mathbf{z}_i^c)), \tag{7.11}$$

to ensure the feature disentanglement of $\mathbf{z}_i^c$.

In sum, with Equations (7.9) and (7.11), the final loss function for DICER is: $L = L_{\beta_1}(\mathbf{x}_i;\theta) + \lambda L_{\beta_2}(\mathbf{c}_i;\phi)$, where $\beta = \{\beta_1, \beta_2\}$ are parameters used for feature disentanglement. λ is used to balance the two types of information.

7.2.3 Implementation

In this section, we provide details of the implementation of DICER as shown in the Figure 7.1(b): $p(\mathbf{z}_i^c)$, $p(\mathbf{z}_i^o)$ (the prior), $q_{\theta_1}(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c)$ and $q_{\phi_1}(\mathbf{z}_i^c|\mathbf{c}_i)$ (the encoder), $p_{\theta_2}(\mathbf{x}_i|\mathbf{z}_i^o, \mathbf{z}_i^c)$ and $p_{\phi_2}(\mathbf{x}_i|\mathbf{z}_i^o, \mathbf{z}_i^c)$ (the decoder), where the parameters $\theta = \{\theta_1, \theta_2\}$ and $\phi = \{\phi_1, \phi_2\}$. Specifically, the θ_1 and ϕ_1 are parameters for encoders, and the θ_2 and ϕ_2 are parameters for decoders.

Prior. To encourage feature disentanglement, as illustrated in Section 7.2.2, we set the prior of $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$ to be the centered isotropic multivariate Gaussian distribution:

$$\mathbf{z}_i^o \sim N(\mathbf{0}, \mathbf{I}_{K_1}), \mathbf{z}_i^c \sim N(\mathbf{0}, \mathbf{I}_{K_2}).$$

The priors ensure the extracted features from different types of information (be they item content or the user-item interactions) are statistically independent in each dimension.

Encoder. To extract the content features $\mathbf{z}_i^c$ and the content disentangled collaborative features $\mathbf{z}_i^o$, there are two parts in DICER: the encoder for the user-item feedback and the encoder for the item content information. Since both the true posterior distribution $p_\theta(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c)$ and $p_\phi(\mathbf{z}_i^c|\mathbf{c}_i)$ are intractable, here we utilize the variational inference and reparametrization trick [123]. That is, we use a Gaussian distribution form with a diagonal covariance $q_\theta(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c)$ and $q_\phi(\mathbf{z}_i^c|\mathbf{c}_i)$ to approximate the true intractable posterior distributions.

Concretely, for $\mathbf{z}_i^o$, we assume:

$$lnq_{\theta_1}(\mathbf{z}_i^o|\mathbf{x}_i, \mathbf{z}_i^c) = lnN(\mu_u(\mathbf{x}_i, \mathbf{z}_i^c), diag\{\sigma_u^2(\mathbf{x}_i, \mathbf{z}_i^c)\}),$$

where the mean and standard deviation are parameterized by a neural network $f_{\theta_1}^{nn}$:

$$(\boldsymbol{\mu}^o, \boldsymbol{\sigma}^o) = f_{\theta_1}^{nn}(\mathbf{x}_i - \mathbf{U}^c \cdot \frac{\mathbf{z}_i^c}{||\mathbf{z}_i^c||}). \tag{7.12}$$

The $\mathbf{U}^c \in R^{|U|\times K_2}$ here is the user embedding matrix that represents the user preference towards item content features $\mathbf{z}^c$. Each row $\mathbf{u}_u^c \in R^{K_2}$ in $\mathbf{U}^c$ represents user u embedding. Since $\mathbf{x}_i$ is related to the joint contribution of item representation $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$, we extract information in $\mathbf{x}_i$ that are not learned from $\mathbf{z}_i^c$ by $\mathbf{x}_i' = \mathbf{x}_i - \mathbf{U}^c \cdot \frac{\mathbf{z}_i^c}{||\mathbf{z}_i^c||}$. The $\cdot$ is the dot product. Here we first normalize the item representation and project it to the same space as $\mathbf{x}_i$ by multiplying with each user embedding $\mathbf{u}_u^c$. Thus, the learned representation $\mathbf{z}_i^o$ can capture the remaining factors in the user-item feedback. The neural network $f_{\theta_1}^{nn}$ is leveraged here to model the complex and non-linear relationship between $\mathbf{x}_i'$ and $\mathbf{z}_i^o$.

For $\mathbf{z}_i^c$, the encoder of $q_{\phi_1}(\mathbf{z}_i^c|\mathbf{c}_i)$ is similar to other VAE-based methods by using variational inference and $f_{\phi_1}^{nn}(\mathbf{c}_i)$, as shown in Figure 7.1(b).

Decoder. In DICER, we use a co-decoder to ensure the learned latent features in each type of information are appropriately encoded: one decoder $p_{\theta_2}(\mathbf{x}_{u,i}|\mathbf{z}_i^o, \mathbf{z}_i^c)$ is used to predict the user preference towards item i given both $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$; the other one is the content-based decoder $p_{\phi_2}(\mathbf{c}_i|\mathbf{z}_i^c)$

to ensure the quality of the encoded content feature representation $\mathbf{z}_i^c$.

For $p_{\theta_2}(\mathbf{x}_{u,i}|\mathbf{z}_i^o, \mathbf{z}_i^c)$, we assume the distribution is proportional to the nonlinear transformation of both $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$:

$$ln(p_{\theta_2}(\mathbf{x}_{u,i}|\mathbf{z}_i^o, \mathbf{z}_i^c)) \propto ln(g_{\theta_2}^{(u)}(\mathbf{z}_i^o) + g_{\theta_2}^{(u)}(\mathbf{z}_i^c)), \tag{7.13}$$

where $g_{\theta_2}^{(u)}(\mathbf{z}_i^o) = exp(cosine(\mathbf{z}_i^o, \mathbf{u}_u^o))$ and $g_{\theta_2}^{(u)}(\mathbf{z}_i^c) = exp(cosine(\mathbf{z}_i^c, \mathbf{u}_u^c))$. Here, $\mathbf{u}_u^o \in R^{K_1}$ is the user u embedding, which represents the user u preference towards features in z_i^o. The cosine similarity is used to connect user and item embeddings rather than the inner product similarity, since it can prevent mode collapse [124, 198]. Note here since item features are disentangled, the cosine also ensures the disentanglement of user features in each dimension [124].

For $p_{\phi_2}(\mathbf{c}_i|\mathbf{z}_i^c)$, similar to other VAE-based methods, we use the normal distribution with a single hidden fully-connected neural network and we found it has good performance. That is: $lnp_{\phi_2}(\mathbf{c}_i|\mathbf{z}_i^c) = lnN(\boldsymbol{\mu}_i', \boldsymbol{\sigma}_i'^2\mathbf{I})$, where $\boldsymbol{\mu}_i'$ and $\boldsymbol{\sigma}_i'$ are calculated based on $\mathbf{z}_i^c$ by using neural network $g_{\phi_2}^{nn}$ [123].

7.2.4 Variations of DICER

Our presentation so far has focused on a scenario in which we have user-item interactions and content information associated with items. In practice, DICER can also be used when only *user* content information is available (in place of item information) or when *both user and item* information are available. When only user content information is known, we can simply change DICER from item-based to user-based by reversing users and items. That is, instead of encoding each item feedback $\mathbf{x}_i$ across users, we encode each user feedback $\mathbf{x}_u$ towards different items. When both user and item content information are known, we can do both user- and item-based DICER, and combine the results together, similar to many integrated auto-encoder based methods.

7.3 Experiments

In this section, we investigate the following key research questions: (i) What is the recommendation performance of DICER compared with state-of-the-art methods that do not disentangle

Table 7.2: Summary of the three Amazon datasets.

Dataset	# Users	# Items	# Feedback	Sparsity
Clothing	2,872	28,586	43,418	0.053%
Beauty	2,513	39,746	91,044	0.091%
Toys&Games	2,771	58,306	106,131	0.066%

content-collaborative information? How does this performance vary for different numbers of latent dimensions? (ii) What impact do the disentanglement design choices have on DICER? and (iii) Are the learned features in DICER really disentangled? Finally, we visualize the disentanglement learning representation to illustrate how it facilitates recommendation.

7.3.1 Datasets

To evaluate the importance of disentangling content-collaborative information, we require datasets containing interaction data (e.g., clicks, purchases) as well as content information. Hence, we adopt three real-world publicly accessible Amazon datasets [11,45] which cover rich and commonly used content information for items – visual images and text descriptions. Other kinds of content information can be easily incorporated into the proposed model as discussed in Section 7.2.4. For items, we choose three domains that are widely used: *Clothing*, *Toys&Games*, and *Beauty*. We select users with more than 10 reviews in *Clothing*, 20 reviews in *Beauty* and 25 reviews in *Toys&Games* for different levels of feedback sparsity, as shown in Table 7.2. We extract UserID, ItemID, and the rating scores to indicate whether the user purchased the item (1 represents a user purchase, 0 otherwise). For item-based content information, we consider the images associated with each item (which were collected in [11]) and the text-based descriptions provided by the seller. Note here all the items are considered. Thus the datasets have a long-tail distribution, which is challenging in recommendation since many items have very little feedback [7]. We randomly partition the implicit feedback of each user into 80% for training, and the remaining as testing, reserving 20% of the training data as validation.

7.3.2 Setup

Modeling Item Content. For the content information $\mathbf{c}_i$, here we consider the widely used item images and text-based descriptions [4, 8, 114]. For fairness, all content-aware baselines use the same item content as input. Other sources of item content could also be incorporated into DICER, and other pre-processing steps can also be adapted here. Concretely, for item images, we apply the same method as [8, 11, 45] to get the high-level visual feature vector $\mathbf{m}_i \in R^{4096}$. For item descriptions, we use the same word2vec-based method [8] which turns the description paragraph into a fixed-length feature vector $\mathbf{t}_i$. Based on [8], here we set $\mathbf{t}_i \in R^{1000}$. The $\mathbf{c}_i \in R^{5096}$ is the concatenation of $\mathbf{m}_i$ and $\mathbf{t}_i$.

Evaluation Metrics. Following many previous works [159], we adopt NDCG at top-k (N@k) and hit ratio at top-k (HR@k) for evaluating personalized ranking. The HR@k measures the fraction of purchased items that appear in top-k recommendation lists across all users. The N@k takes the position of correctly recommended items into account by assigning higher scores to the top hits.

Baselines. We compare DICER with the following competitive baselines, with particular emphasis on VAE-based methods for comparison.[†] The same content information is used for all content-based methods.

- *POP*. Items are ranked by their popularity based on user's interactions with items.
- *CDAE* [194]. Collaborative Denoising Auto-Encoder uses auto-encoders to find the relationship between users and items based on implicit feedback. The number of negative samples is set to $q = 100$ which is in line with the other negative sampling methods.
- *Multi-VAE* [195]. This is a classic VAE-based latent factor models for recommendation. The implicit feedback from users is treated as being generated from a multinomial likelihood for recommendation.
- *NGCF* [64]. This is a state-of-the-art collaborative filtering approach based on graph neural networks. It treats use-item interactions as a bipartite graph and propagates the user and item

[†]We also experimented with neural machine factorization [200] but the training process is time consuming and its performance is not good here.

embeddings on the graph to explore the high-order connectivity between users and items.

- *CBPR* [4]. This is a classic content-based Bayesian Personalized Ranking [4] method, which is widely used for its robust and strong performance.
- *CVAE* [114]. CVAE is a content-based VAE model that uses a Bayesian generative model to first learn the item content information and then user-item implicit feedback through an inference network. It adds the item latent content variable and collaborative latent variable as the joint representation of each item.
- *CNGCF*. This is an augmented version of NGCF that incorporates item content information [64]. Specifically, we concatenate item content information with latent factors as the joint representation, and then we propagate three layer's embeddings in the user-item bipartite graph for the final recommendation.

Parameter Settings. The dimension of latent factors and hidden dimensions for each method is empirically set to be 40 (for the impact of such choices, see Section 7.3.3.2). Regularization terms are determined by grid search in the range of {0.1, 0.01, 0.001, 0.0001, 0.00001}. β in VAE-based methods are determined from the range {1.0, 0.9, ...,0.1, 0.01, 0.001, 0.0001, 0.00001}. The drop out rate is also chosen by grid search in the range of {0.1, 0.3, 0.5, 0.7, 0.9} and the learning rate is in the range of {0.0001, 0.001, 0.01, 0.1}. Model parameters are first randomly initialized according to truncated normal distributions with mean 0 and standard deviation 0.001.

7.3.3 Top-K Recommendation

We first compare the Top-K recommendation performance of all methods, and then vary the key hyperparameter – the number of latent factor dimensions – to further investigate its effect on recommendation.

7.3.3.1 Overall Comparison

For fair comparison, we set the latent dimension K to be the same for all methods. Notice that CBPR, CVAE and CNGCF all use the same item content information (images and descriptions) as DICER. We report the N@k and HR@k (for k at 5, 10) for the three datasets in Table 7.3. Overall,

Table 7.3: N@K and HR@K of DICER and baselines. Δ is the difference between DICER and the next-best alternative (marked underline).

Dataset	Measure%	User-Item Interaction				User-Item Interaction + Content				
		POP	CDAE	Multi-VAE	NGCF	CBPR	CVAE	CNGCF	DICER	Δ
Clothing	N@5	0.423	0.833	0.693	1.311	0.812	1.228	1.380	**1.661**	20.4%
	N@10	0.616	1.111	1.031	1.735	1.109	1.682	1.809	**2.177**	20.3%
	HR@5	0.731	1.462	1.253	1.950	1.288	2.089	2.089	**2.646**	26.7%
	HR@10	1.358	2.228	2.159	3.203	2.089	3.412	3.377	**4.039**	18.4%
Toys	N@5	0.997	3.139	3.350	3.522	2.612	3.502	4.560	**4.984**	9.3%
	N@10	1.454	4.273	4.490	4.835	3.728	4.991	6.074	**7.017**	15.5 %
	HR@5	1.877	5.485	5.413	5.702	4.186	5.521	7.326	**8.156**	11.3%
	HR@10	3.284	8.697	8.336	9.347	7.434	9.780	11.332	**13.389**	18.2%
Beauty	N@5	2.354	6.338	6.739	6.556	5.922	7.230	7.054	**7.753**	7.2%
	N@10	3.449	8.567	9.005	9.017	7.755	9.464	9.693	**10.477**	8.1 %
	HR@5	4.338	11.421	10.864	10.744	9.391	11.421	11.779	**12.694**	7.8%
	HR@10	7.402	16.514	15.002	16.076	13.490	16.275	17.589	**18.305**	4.1%

we see that DICER consistently outperforms the next-best performing baseline for all datasets and for all metrics. Concretely, we have the following key observations:

First, methods that incorporate item content information generally achieve better performance comparing with corresponding methods that only rely on user-item interactions. For example, the HR@K and N@k of the content-based latent factor model CNGCF is higher than NGCF for the three datasets. This verifies the importance of incorporating additional item content information for improving recommendation performance.

Second, among methods, DICER consistently achieves the best performance over all datasets, as shown in Table 7.3 Δ column. The improvement demonstrates that by carefully disentangling content-collaborative features, DICER is able to enhance the learning of diverse features that influence user preference towards items. Particularly, comparing with CVAE which considers the same content information as DICER, the large improvement of DICER further confirms that the disentangling in DICER can effectively deal with the complicated relationship between item content information and user interactions, which strengthens DICER to discover different features among the two types of information for recommendation improvement.

Third, among datasets, DICER gives a relatively larger improvement for the sparsest dataset

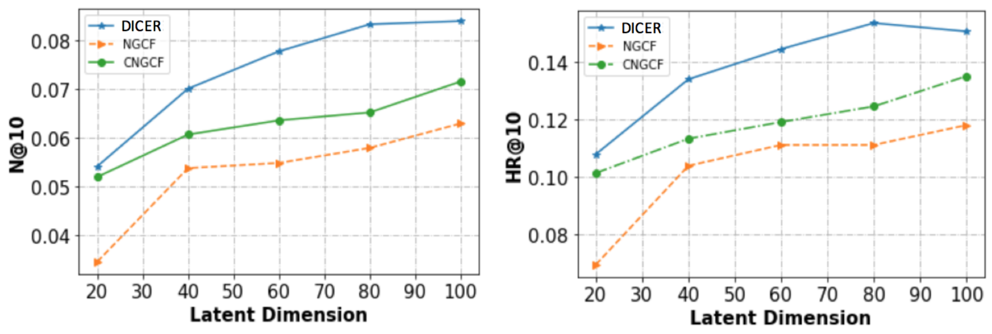


Figure 7.2: Recommendation performance for different latent dimensions in Toys dataset. Similar results hold for the other two datasets.

(Clothing). This may be because DICER not only considers the content information that is helpful for sparse data, but also utilizes disentanglement to discover features that cover a wider space rather than duplicated or related features. This further shows the importance of disentangling content and collaborative information. An interesting finding is that comparing the improvement from NGCF to content-based CNGCF, the improvement of DICER is higher in VAE-based methods. Such high improvement may be attributed to the learning of disentangled features rather than directly concatenating content and collaborative features in CNGCF. This shows the benefit of capturing the disentangled features for recommendation.

7.3.3.2 Influence of Latent Dimension

We also analyze the effect of the key hyperparameter – the number of latent factors – on DICER and representative baselines. Results for the Toys dataset is shown in Figure 7.2. Similar results hold for the other two datasets. We observe that DICER consistently outperforms the other methods for different numbers of latent dimensions.

7.3.4 Ablation Study

Given the good performance of DICER versus baselines, what impact do the design choices have on its performance? Specifically, does the disentanglement approach in DICER effectively incorporate item content and collaborative information to enhance recommendation (encoder in Equation (7.12) and decoder in Equation (7.13))? In this section, we explore several variants to

Table 7.4: Ablation study on Toys dataset. Similar results hold for the other two datasets.

%	**N@5**	**N@10**	**HR@5**	**HR@10**
Default	4.984	7.017	8.156	13.389
Original $\mathbf{x}$	4.071	5.614	6.712	10.682
Nonlinear	4.659	6.206	7.615	11.620
LatentConcat	4.756	6.504	7.470	12.090
LatentAdd	4.828	6.6520	7.795	12.450

incorporate item content information compared with DICER and analyze their effects:

- *Original* $\mathbf{x}$: $\mathbf{z}_i^o$ is encoded based on the original user-item interactions $\mathbf{x}_i$ rather than the content disentangled one. That is, $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$ are separately learned without considering their disentanglement;
- *Nonlinear*: In Equation (7.12), instead of using $\mathbf{z}_i^c$, here we add a $tanh$ layer to project $\mathbf{z}_i^c$ to the same space as $\mathbf{x}_i$, then encode $\mathbf{z}_i^o$. It may influence the disentanglement relation between $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$;
- *LatentConcat*: We directly concatenate $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$ as the joint latent representation to reconstruct $\mathbf{x}_i$. That is, we use $ln(g_{\theta_2}^{(u)}(concate(\mathbf{z}_i^o, \mathbf{z}_i^c))$ in Equation (7.13) to formulate $ln(p_{\theta_2}(\mathbf{x}_{u,i}|\mathbf{z}_i^o, \mathbf{z}_i^c))$;
- *LatentAdd*: $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$ are directly added as the joint latent representation to reconstruct $\mathbf{x}_i$. That is, we use $ln(g_{\theta_2}^{(u)}(\mathbf{z}_i^o + \mathbf{z}_i^c))$ in Equation (7.13).

The results of the ablation study for the Toys dataset are shown in Table 7.4. The *default* row shows the DICER results. We observe that DICER outperforms the other variations that consider the same item content information, which further shows DICER can more effectively model the relationship between item content and user-item interaction information for recommendation improvement.

Concretely, the performance of DICER is superior to *Original* $\mathbf{x}$ method which uses raw user-item feedback. This demonstrates that disentangling evidence learned from item content to encode $\mathbf{z}_i^o$ can bring a large improvement, compared to directly using the raw $\mathbf{x}_i$. This is reasonable since the disentanglement can encourage the learned features in $\mathbf{z}_i^o$ and $\mathbf{z}_i^c$ to cover more diverse

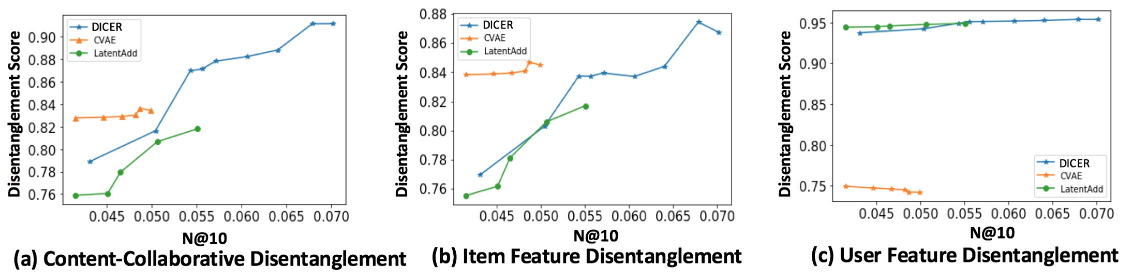


Figure 7.3: Disentanglement results for N@10 on Toys dataset. We vary the disentanglement parameter β and plot the relationship between the disentanglement and recommendation performance. The higher the disentanglement score, the more disentanglement between corresponding features.

independent features in item representations, and thus enhance the learning of user preference towards these items. An interesting finding is that comparing the nonlinear transform (the *Nonlinear* method), DICER achieves a better performance. This is probably because the nonlinear calculation may have difficulties maintaining disentanglement among features and can be more prone to cause overfitting.

Furthermore, DICER outperforms methods that directly concatenate or add $\mathbf{z}_i^c$ to $\mathbf{z}_i^o$ to jointly reconstruct the user-item feedback, such as the *LatentConcat* and *LatentAdd*. This indicates that explicitly decoding item content separately can more precisely capture item content features to further help model user-item interaction.

7.3.5 Disentanglement

Since the disentanglement representation learning plays a pivotal role in DICER, we further explore its influence on the recommendation performance from both the content-collaborative disentanglement and feature disentanglement perspective. Following [124], we measure disentanglement based on statistical independence. We vary β and plot the relationship between the disentanglement and recommendation performance, where β is the parameter of disentanglement for corresponding latent factors.

7.3.5.1 Content-Collaborative Disentanglement

The Content-Collaborative disentanglement is measured by

$$Average_{k,j}(1 - Corr_i(z^c_{i,k}, z^o_{i,j})),$$

where $z^o_{i,k}$ represents the kth dimension of $\mathbf{z}^o_i$. Similar notation holds for $z^c_{i,j}$. $Corr_i(z^c_{i,k}, z^o_{i,j})$ is the correlation between each dimension in the corresponding $\mathbf{z}^c_i$ and $\mathbf{z}^o_i$ across items. The results are shown in Figure 7.3(a). Note here since DICER outperforms the other methods in N@10, the lines for the other methods stop earlier than the line for DICER.

From Figure 7.3(a), we observe that (1) For the content-collaborative disentanglement, DICER achieves a good disentanglement when N@10 is high. This indicates that DICER can effectively capture the relatively statistically independent features between item content and user-item interaction information; (2) The figure also demonstrates that good recommendation performance is related to a relatively high disentanglement, which is consistent with [124]. This makes sense since the disentangled features are more stable [116, 121] and can cover a variety of user preferences.

7.3.5.2 Feature Disentanglement

Similarly, the item and user feature disentanglement are measured based on:

$$Average_{1\le k_1<k_2\le K_1}(1 - Corr_i(z^o_{i,k_1}, z^o_{i,k_2})) + Average_{1\le j_1<j_2\le K_2}(1 - Corr_i(z^c_{i,j_1}, z^c_{i,j_2})) + Average_{k,j}(1 - Corr_i(z^c_{i,k}, z^o_{i,j})),$$

$$Average_{1\le k_1<k_2\le K_1}(1 - Corr_u(u^o_{u,k_1}, u^o_{u,k_2})) + Average_{1\le j_1<j_2\le K_2}(1 - Corr_u(u^c_{u,j_1}, u^c_{u,j_2})) + Average_{k,j}(1 - Corr_u(u^c_{u,k}, u^o_{u,j})),$$

respectively. The calculation of $Corr_u(\cdot,\cdot)$ is similar as $Corr_i(\cdot,\cdot)$ but across users instead of items. Figure 7.3(b)(c) shows the disentanglement results.

We have the following key observations: (1) Similar to the content-collaborative disentanglement finding, when N@10 is high, DICER achieves high feature disentanglement. This indicates

DICER can also capture good statistically independent features at a granular level. (2) For CVAE, though it has a good disentanglement for item and content-collaborative features, the disentanglement for user features is much lower than the other two methods. This may be since DICER and LatentAdd jointly model the disentanglement of user and item features, the user representation can capture the disentangled features based on item features, while CVAE considers them separately. (3) Another finding is that the item feature disentanglement is lower than content-collaborative disentanglement. One possible reason is that content and collaborative representations are calculated from different types of information.

7.3.6 Case Studies of Disentanglement

In this section, we illustrate how the disentangled representation approach facilitates recommendation.

First, we visualize the content-collaborative disentanglement representations $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$, as shown in Figure 7.4(a)(b) with t-SNE [201]. We observe that (1) Comparing Figure 7.4(a) and (b), $\mathbf{z}_i^c$ can nicely separate different categories of items even without knowledge of the ground-truth categories. That is, items are separated mainly based on item content oriented information. For example, as shown in Figure 7.4(a), the accessories, shoes, tops and bottoms are in different clusters. Different from $\mathbf{z}_i^c$, in Figure 7.4(b), $\mathbf{z}_i^o$ separates items by user-oriented information, e.g., items used by male (left bottom) and female (right top). Items that can be bought together (e.g., bottoms and shoes) are also close to each other. This indicates $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ can capture item features in very different aspects (item content and user aspects) by disentanglement. (2) Furthermore, in each cluster of (a), items with similar content information are relatively close to each other. For example, items in similar colors are close to each other, and items with dark colors are far from the light color items, such as for shoes and tops. This demonstrates that $\mathbf{z}_i^c$ can capture reasonable content features. In sum, by using content-collaborative disentanglement, $\mathbf{z}_i^c$ and $\mathbf{z}_i^o$ are complementary in how they discover item features based on different aspects to enhance the prediction of user preference towards items.

Next, we illustrate the features that DICER learns in each dimension at the granular level. We

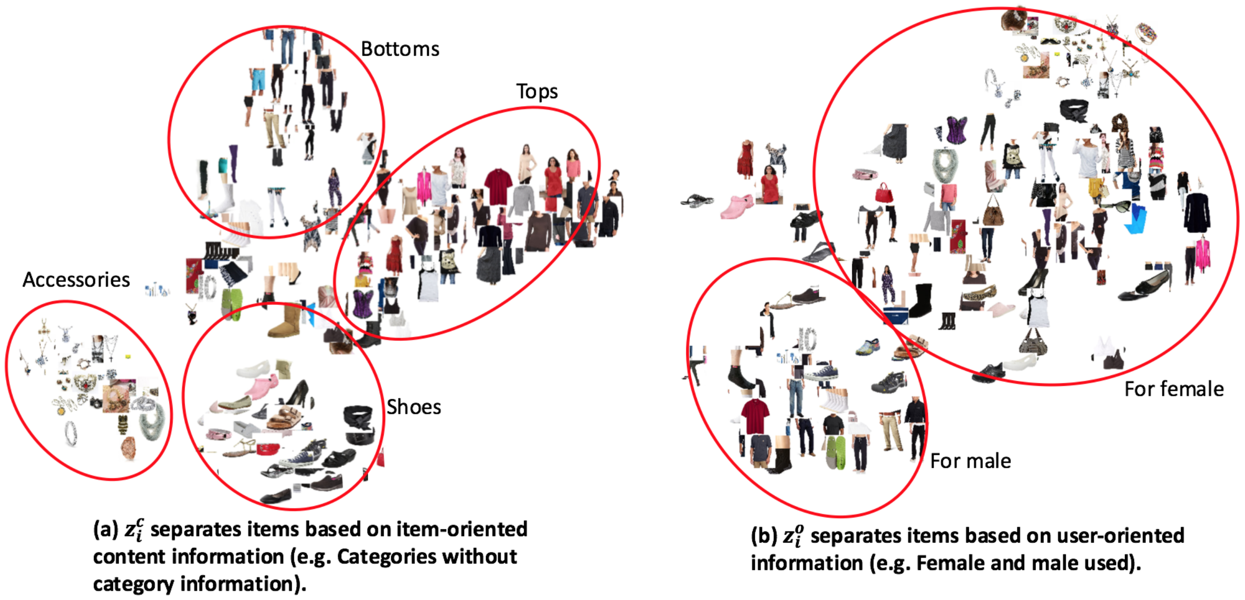


Figure 7.4: The 2-D visualization of items in Clothing dataset by using (a) $\mathbf{z}_i^c$ (b) $\mathbf{z}_i^o$.

vary the target dimension and list the items that have similar values in the target dimension. Some representative examples are shown in Figure 7.5. Each row shows items that have similar values in a target dimension. For each row, we see DICER is able to capture very different features in each dimension without item category information, and some of them may be interpretable. For example, in Figure 7.5, the first target dimension likely captures the circle-based characteristic of items. The second target dimension captures the shoe features. The clothing items in the third row have similar styles. For the fourth row, all the items have colorful patterns. This highlights how DICER can capture independent and interpretable features in each dimension.

7.4 Summary

In this chapter, we focus on disentangled representation learning from both content information and user-item interactions to enhance recommendation. By disentangling the content-collaborative features and each feature at a granular level, the proposed method DICER learns features that are relatively statistically independent and diverse, leading to a more powerful recommender. Through extensive experiments, DICER outperforms the next-best baseline by 13.5% and 14.4% on average in NDCG and hit ratio. In our continuing work, we are interested in extending DICER to other

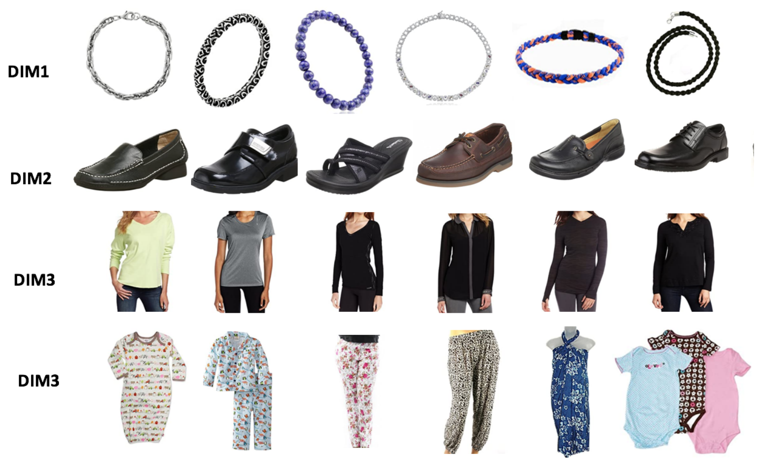


Figure 7.5: Each row shows items that have similar values in one target dimension of item representations.

scenarios, e.g., where a user's social network is available.

8. CROSS DECOUPLING: LEARNING ITEM EMBEDDINGS BASED ON LONG-TAIL ITEM DISTRIBUTION

In the previous chapter, we focused on disentangling representations learned based on the relations between content and collaborative signals. But even within content-based methods alone, we find that content plays different roles for popular versus cold-start items. The skewed long-tail distribution of items – in which some items accumulate lots of attention (e.g., via clicks or views) while most items receive very little attention – can heavily undermine the quality of recommendation. Hence in this chapter, we seek to address this skewed distribution problem to further improve content-aware recommendation performance through a decoupling approach. Concretely, we theoretically analyze and empirically study the skewed distribution problem. We then propose a cross decoupling framework that decouples the learning process of memorization and generalization from prior and conditional knowledge perspective. Extensive experimental results show how the proposed model outperforms state-of-the-art methods.

8.1 Introduction

In recommendation, user feedback usually exhibits long-tail distributions, especially from the item perspective: a small fraction of items are extremely popular and receive most of the user feedback (what we refer to as *head items*), while most items have very little if any user feedback (what we refer to as *tail items*). Recommenders that are trained based on such long-tail data usually suffer from sub-optimal and un-robust performance, especially for tail items. Deploying those recommenders can further lead to popularity bias and a "rich get richer" feedback loop, causing inferior recommendation of tail items.

There are many attempts that aim to address this problem, with most focusing on improving the performance for tail items. In recommendation, a commonly used strategy is to incorporate content information with user or item ID information to enhance the representation learning of tail item. Besides that, there are also methods using re-balancing, e.g., resampling [202], reweighting [142]), transfer learning [144] and meta learning [141] to improve the performance of tail items. Recently, a decoupling method [18] has been proposed in the context of visual recognition, which uses a two-stage training approach. It shows powerful performance in many application areas. However, most of these methods ignore the memorization and generalization discrepancy between head and tail items in the long-tail distribution settings, which may under-represent the head items and lead to inferior performance of overall recommendation. Actually, as shown in Figure 8.1, we find that these kinds of methods only gain improvement for tail items, while leading to worse overall recommendation performance.

In this work, we propose to separately consider memorization and generalization to address the long-tail distribution problem in recommendation. Concretely, we first present a theoretical and experimental analysis of the inferior recommendation performance of decoupling methods that address the long-tail distribution problem. Our key finding is that the popular two-stage training fashion mainly rectifies the prior item distribution in the second stage, which easily suffers from the forgetting issue as the learned knowledge in the first stage is replaced by the knowledge from the second stage. Therefore, when we focus on learning tail items in the second stage, the performance of head items will dramatically decrease.

Considering that, we propose a cross decoupling network CDN to segregate the learning of memorization and generalization from both prior item distribution and conditional distribution aspects. For the prior item distribution, a novel regularized cumulative learning is deployed to simultaneously train a two branches network with different data distributions: the original highly skewed data distribution (trained in the main branch) and a relatively balanced data distribution (the regularized branch). The original long-tail data distribution is used to lean the universal patterns across items, and the relatively balanced data distribution is used to focus on the training of tail items. They are aggregated by a regularized adapter to both maintain the learned knowledge from head items and gain generalization for tail items. From the conditional distribution perspective, we separate item ID and content information through a multi-experts mechanism to explicitly learn the unique trade-off between memorization and generalization for each item based on their frequency in the long-tail distribution. Therefore, the memorization and generalization differences between head and tail items can be smoothly learned in a continuous way without a hard head/tail split threshold.

The contributions of this work are three-fold:

- We theoretically and experimentally analyze existing methods that address the long-tail distribution problem. Those methods mainly modify the prior knowledge of item distributions to gain improvement for tail items, which can cause a forgetting issue that diminishes the recommendation performance of head items.
- We propose a cross decoupling framework that decouples the learning process of memorization and generalization from prior and conditional knowledge perspectives.
- Extensive experiment results show our proposed CDN significantly outperforms the other method, bringing improvement for both head and tail items on hit ratio (HR@K) and NDCG@K. CDN also achieves lower standard error of the mean.

8.2 Long-tail Distribution in Recommendation

Problem settings. We consider a common setup for recommendation problems: Given a set of users $\mathcal{U}$, a set of items $\mathcal{I}$ and their content information. There are m users and n items. Each user $u \in \mathcal{U}$ and item $i \in \mathcal{I}$ can be represented by the feature vectors as $\mathbf{x}_u$ and $\mathbf{y}_i$ using their ID and content information. $d(u, i)$ represents the user u implicit feedback towards item i: $d(u, i) = 1$ when u gives positive feedback to i, such as click, watch or buy, otherwise it is 0. We denote $\sum_{u \in \mathcal{U}} d(u, i)$ as the user feedback towards the item i. It follows a long-tail distribution across items [203]. That is, a few items receive most of the user feedback (head items), while a most items have little user feedback (tail items). In this paper, our goal is to consider the long-tail item distribution to improve the recommendation performance for tail items, and at the same time, relatively keep or improve the overall recommendation performance. Key notations are shown in Table 8.1.

One important question is, how does the long-tail distribution influence the recommendation performance? To address the question, suppose the user preference u towards items i is estimated by $\hat{p}(i|u,\theta)$, which can be represented as:

$$\hat{p}(i|u) = \frac{e^{s(u,k) - log\frac{p(i)}{\hat{p}(i)} - log\frac{p(u|i)}{\hat{p}(u|i)}}}{\sum_{j\in\mathcal{I}} e^{s(u,j) - log\frac{p(j)}{\hat{p}(j)} - log\frac{p(u|j)}{\hat{p}(u|j)}}} \tag{8.1}$$

where $p(i|u;\theta)$ is the true conditional probability of the user preference towards items. θ is the set of model parameters.

Proof. By Bayes' theorem, the user preference towards items can be represented by

$$p(i|u;\theta) = \frac{p(u|i)}{p(u)}p(i), \tag{8.2}$$

then, based on the observed data, the posterior probability is

$$\hat{p}(i|u;\theta) = \frac{\hat{p}(u|i)}{\hat{p}(u)}\hat{p}(i), \tag{8.3}$$

where $\hat{p}(i)$ is usually estimated based on the observed data distribution. That is, $\hat{p}(i) = \sum_{u\in\mathcal{U}} d(u,i)$.

Therefore, the differences between the desired conditional probability and the estimated one is:

$$log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)} = log\frac{p(i)}{\hat{p}(i)} + log\frac{p(u|i)}{\hat{p}(u|i)} + log\frac{\hat{p}(u)}{p(u)} \tag{8.4}$$

Suppose $s(u,i) = log(\frac{p(i|u;\theta)}{p(i|u;\theta)})$, then $s(u,i) - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)} = log\frac{p(i|u;\theta)}{p(j|u;\theta)} - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)} = log\frac{\hat{p}(i|u;\theta)}{p(j|u;\theta)}$. Therefore, $\hat{p}(i|u;\theta) = e^{s(u,i) - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)}} p(j|u;\theta)$. And $1 = \sum_{k\in\mathcal{I}} \hat{p}(i|u;\theta) = \sum_{k\in\mathcal{I}} e^{s(u,k) - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)}}$ $p(j|u;\theta)$. So $p(j|u;\theta) = \frac{1}{\sum_{k\in\mathcal{I}} e^{s(u,k) - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)}}}$. Based on that, we have $\hat{p}(i|u) = \frac{e^{s(u,k) - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)}}}{\sum_{k\in\mathcal{I}} e^{s(u,k) - log\frac{p(i|u;\theta)}{\hat{p}(i|u;\theta)}}}$. Incorporate with equation (8.4), we can get equation (8.1). □

As Equation (8.1) indicates, $p(i|u;\theta)$ is related to the prior differences $\frac{p(i)}{\hat{p}(i)}$ and conditional differences $\frac{p(u|i)}{\hat{p}(u|i)}$. Existing methods for the long-tail distribution problem mainly focus on approximating the item distribution $\hat{p}(i)$ to $p(i)$ (e.g. resampling and reweighting), i.e. $log\frac{p(i)}{\hat{p}(i)} = \epsilon, s.t. \epsilon \to 0$ by assuming the conditional probability $p(u|i) = \hat{p}(u|i)$ for any item, which is referred to as the distribution shift problem.

However, we argue that this assumption does not hold in recommendation, especially for tail items since there are not enough data points to show the user preference when the tail item is given in the training data. For example, many tail items are cold start items, so the $\hat{p}(u|i)$ estimated in training data can not represent the expected $p(u|i)$. In this situation, if we focus on decreasing the discrepancy between $p(u|i)$ and $\hat{p}(u|i)$ in tail items $k \in \mathcal{T}$, the learning of $p(u|i)$ for head item $k \in \mathcal{H}$ will be of poor quality. Furthermore, the user preference towards head and tail could be different, leading to the $p(u|i)$ to be difficult to learn with a unified mechanism. For example, a user may prefer the popular items mainly because of their popularity, while they prefer tail items mainly due to their personal preference. Hence, a unified single model is not enough to uncover this user heterogeneity preference towards head and tail items. This influence would weaken the overall performance in recommendation, where the head items pose a large percentage in the highly skewed long tail item distribution. Considering that, we contend that existing methods that focus on the distribution shift may only play well for tail items leading to a decrease of overall performance in recommendation.

8.3 Data Investigation

To justify our conjecture, in this section, we design an experiment to investigate a state-of-the-art method that addresses the long-tail distribution problem – decoupling [18] which is a two-stage learning mechanism that separately learns representation and ranking. To do so, similar as in [7], we treat the recommendation task as a multi-class classification problem. So the user embedding learning can be treated as the representation learning, and item embedding learning as the classifier [*].

Concretely, in the experiments, we leverage the representative state-of-the-art two-tower model [7, 144] as the backbone model. To apply the two stage decoupling training fashion, in the first stage, we train both the two tower model for representation learning. In the second stage, we fix the parameters of the user tower that is learnt in the first stage, and retrain the item tower as a classifier

[*]There could be different definitions of representation learning and ranking. Here we consider this setting.

training. For each stage, we implement both plain training (e.g., cross-entropy) and the rebalance methods (resampling). Therefore, there are four groups of results based on the training manners in the two-stage training fashion: (1) OR (Original Method): the two-tower model is trained based on the original long tail distribution dataset with the conventional weighted log-likelihood loss; (2) RS (resampling): we use a down-sampling strategy that down-samples the user feedback in head items and keeps all the user feedback for tail items to gain a relatively balanced dataset [144]. The constructed relatively balanced dataset not only ensures tail items are fully trained, but also alleviates the bias among tail items that brings by the new distribution. The benchmark dataset MovienLens1M is used for evaluation. The commonly used hit ratio at top k (HR@K) and NDCG at top K (NDCG@K) are used to measure the recommendation performance of tail slices items and also the overall recommendation performance.

Results are shown in Figure 8.1. Overall, for the performance of tail slice items, when the same representation method is used (OR or RS), RS method can always achieve higher HR@50 and NDCG@50, which is consistent with the findings in [135]. It is reasonable since the relatively balanced dataset in RS focuses more on the training of tail items.

However, we find while the performance of tail items increases, the overall recommendation performance decreases when re-balancing strategies are used. As shown in Figure 8.1 for overall recommendation performance, when the same representation methods are implemented, the re-balanced methods (RS) perform worse (lower HR@50 and NDCG@50). One possible reason is that different from the computer vision area where the imbalanced factor is around 100, the distribution in recommendation is highly skewed (imbalanced factor usually > 10,000). Therefore, sacrificing head item can significantly degrade the overall performance. Furthermore, besides the highly skewed training dataset, the testing dataset is also imbalanced. That is, head items possess the majority part in the testing dataset. A small drop of head item performance could heavily diminish the overall recommendation performance. We refer the phenomenon as the **Forgetting Issue**.

The experimental results confirm our assumption that purely distribution shift methods have difficulty improving both tail and overall recommendation. With those observations, in the next section, we introduce our proposed cross decoupling that can bring improvements for both head and tail items.

8.4 Cross Decoupling

With the observation in Section 8.3, we introduce our proposed cross decoupling learning that decouples the learning process of the long-tail distribution from both prior probability (user side) and conditional probability(item side), as shown in Figure 8.2:

- For the user side, considering the forgetting issue of user representation learning in the two-stage training fashion, we simultaneously train both original highly skewed long-tail distribution dataset and a relatively balanced distributed dataset. A novel regularizer adapter is proposed to automatically balance the learning of head and tail items;
- For the item side, considering the properties of ID and content information, we design a multi-expert mechanism to decoupled the learning of memorization and generalization.

With both user and item side learning, we final cross combine the user side and item side information to comprehensively learn the user various preference for recommendation. In the following, we discuss the cross decoupling design from user and item side respectively.

8.4.1 User Sample Decoupling with Regularizer Adapter

The large distribution gap between the two types of items makes it challenge to learn a reasonable user representation. Traditional recommendation methods that learned the user representation based on the original long-tail item distribution usually under-represented for tail items [125], which easily harms the recommendation of tail items. Furthermore, in recommendation, there are many tail items and their feedback is extremely sparse, which makes it even harder to learn a good user representation learning for tail items. As Section 8.3 shows, if a two-stage learning fashion is applied, it is easy to forget the learned knowledge from the other side information. How could we learn the user representations based on both the head and tail items?

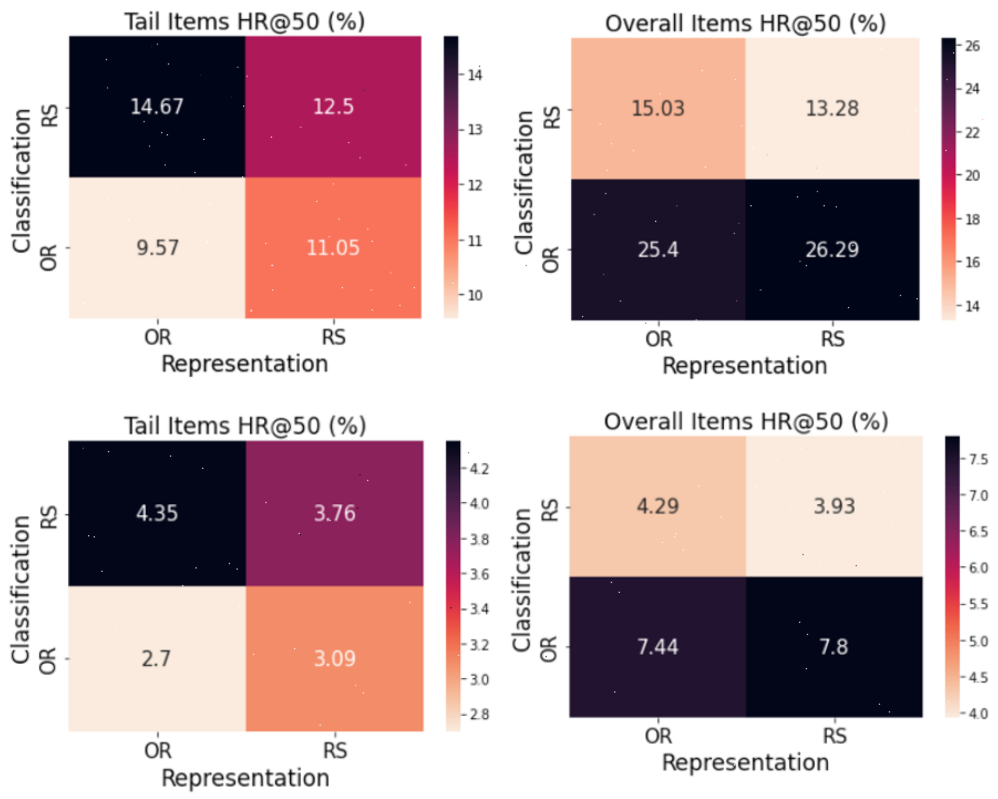


Figure 8.1: Recommendation performance (HR@50 and NDCG@50) of decoupling approach for representation learning and classifier learning in MovieLens1M dataset for tail item and overall performance.

We propose a regularized bilateral branch network that simultaneously trains the model with different data distributions, as shown in Figure 8.2 of user side: one training branch uses the original highly skewed long-tail distribution as the training dataset, the other regularizer branch uses a relatively balanced dataset as the training dataset. We denote the original highly skewed long-tail dataset as Ω_t, and the relatively balanced dataset as Ω_r.

Concretely, in each branch of the regularized bilateral branch, we can get a training sample $(\mathbf{u}_t, \mathbf{i}_t) \in \Omega_t$ from the training branch, and a training sample $(\mathbf{u}_r, \mathbf{i}_r) \in \Omega_r$ from the regularizer branch. Then the user representation vectors are calculated by:

$$\begin{aligned} \mathbf{x}_t &= h_t(f(\mathbf{u}_t)) \\ \mathbf{x}_r &= h_r(f(\mathbf{u}_r)) \end{aligned} \tag{8.5}$$

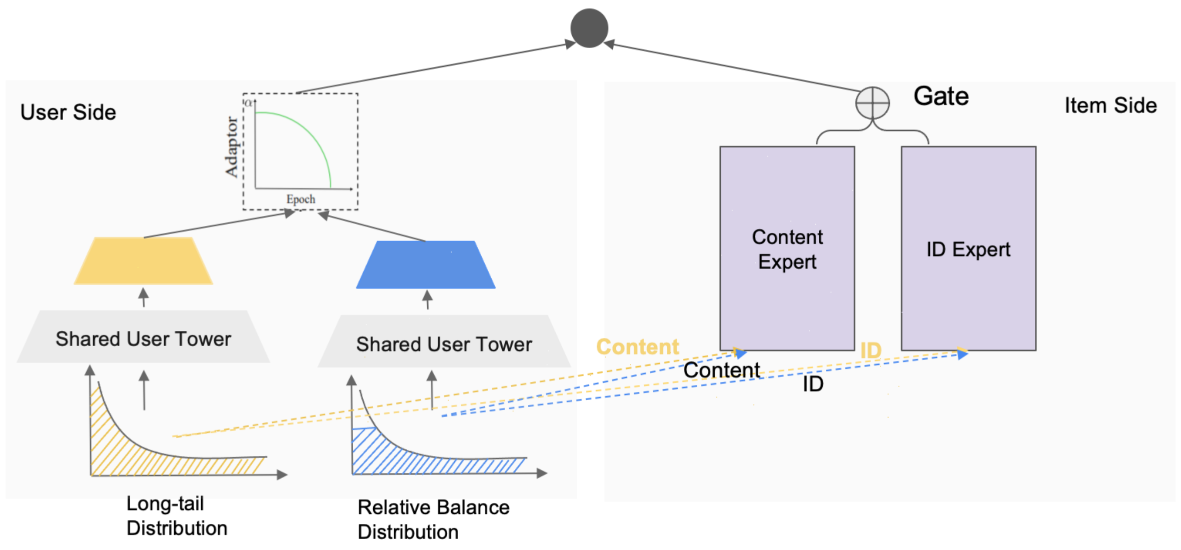


Figure 8.2: CDN framework.

here we first use a shared layer $f(\cdot)$ which can communicate the learned knowledge from the two branches. It can also largely reduce the computational complexity. Then the specific layers are designed $h_t(\cdot)$, $h_r(\cdot)$ in each branch to learn the unique knowledge of each branch.

8.4.2 Item Memorization and Generalization Decoupling with Experts

The user side decoupling ensures the learned user representations are well trained in the long-tail distribution. However, the user preference towards head and tail items is highly heterogeneous. For the retrieval task, since only limited top rated items are selected, it becomes especially important to well train both head and tail items that can ensure head and tail items have equal chance to be selected. Considering that, in this section, we design an item-side decoupling to balance the memorization and generalization between head and tail items.

For head items, since it contains rich user feedback, the ID information plays an important role to memorize the learning information. Different from that, for tail items, since the user feedback is very sparse, the item content information play an important role. In this situation, the generalization is important for tail items. We use a content expert and an ID expert, as shown in Figure 8.2 of item side. Gating is used to balance the memorization and generalization.

Concretely, for a training sample $(\mathbf{u}, \mathbf{i})$, the item embedding is represented as:

$$\mathbf{y} = g(\mathbf{i})_1 f_{id}(\mathbf{i}) + g(\mathbf{i})_2 f_{content}(\mathbf{i}) \tag{8.6}$$

where $f_{id}(\cdot)$ is the ID expert which uses the item ID as the input, and the $f_{content}(\cdot)$ is the Content expert which uses the item content information as the input. $g(\cdot)$ is the gate function . $g(\mathbf{i})_1 + g(\mathbf{i})_2 = 1$.

For the gating $g(\cdot)$, we use item frequency as the input, and build a forward layer:

$$g(\mathbf{i}) = softmax(\mathbf{W}\mathbf{i})$$

where $\mathbf{W}$ is the trainable matrix.

8.4.3 Cross Learning

To fully learn the item content and id information in the long-tail distributions, we finally propose a cross learning framework that crosses the decoupled information from user and item side.

For a training sample $(\mathbf{u}_t, \mathbf{i}_t) \in \Omega_t$ from the training branch, and a training sample $(\mathbf{u}_r, \mathbf{i}_r) \in \Omega_r$ from the regularizer branch, we designed a regularizer adapter to combine the learned feature vector from two branches:

$$s(i_t, i_r) = \alpha_b \mathbf{y}_t^T \mathbf{x}_t + (1 - \alpha_b) \mathbf{y}_r^T \mathbf{x}_r \tag{8.7}$$

where $s(i_t, i_r)$ is the predicted output. And the regularizer adapter α_b is related to the learning epoch b:

$$\alpha_b = 1 - \frac{b}{\gamma * B} \tag{8.8}$$

Here B is the total batch size, and γ is the regularizer rate. That is, it shift the learning focus from the original distribution to the rebalanced data distribution. It is designed to focus on learning the universal pattern and then gradually focus on the tail items.

Table 8.1: Notation of CDN.

Notation	Explanation
$\mathcal{U}, \mathcal{I}$	user set, item set
$\mathbf{u}_t$, $\mathbf{i}_t$	user u_t (item i_t) feature vector
$\mathbf{x}_t$, $\mathbf{y}_t$	user/item representation in the training branch
$\mathbf{x}_r$, $\mathbf{y}_r$	user/item representation in the regulizer branch
α	learning adapter
γ	adapter regulizer

We then use the softmax as the loss function to learn the probabilistic distribution of user u's preference towards different items for recommendation. That is:

$$p(i) = \frac{e^{s(i_t, i_r)}}{\sum_{j \in \mathcal{I}} e^{s(j_t, j_r)}}. \tag{8.9}$$

Therefore, the loss function can be formulated as:

$$L = -\frac{1}{|\Omega_t|} \sum_{(u,i) \in \Omega_t} \alpha_n d_t logp(i) + \alpha_n d_r logp(i), \tag{8.10}$$

where the d_t and d_r is the rewards function and Ω_t is the training dataset. The rewards in the loss function would learn high preference score for items that user engaged with in the training branch and regularizer branch.

For the inference, to predict the user preference towards item, we only use the training branch, since the regularizer branch is used to regularize the learned user and item embeddings. That is, we calculate the preference score as:

$$p(u, i) = \mathbf{y}_t^T \mathbf{x}_t. \tag{8.11}$$

to obtain the logits by softmax.

8.5 Experiment

In this section, we conduct extensive experiments to address the following key research questions:

RQ1: How well does the cross decoupling framework CDN perform compared to the state-of-the-art methods, especially comparing with traditional decoupling methods?

RQ2: How does the cross decoupling work in user and item sides?

RQ3: How does the experts' design influence the CDN performance? How does the model balance between tail/head items by gating mechanisms?

RQ4: Are we learning better representations for tail items?

8.5.1 Datasets

We use two widely used public benchmark datasets which contain rich user and item content information: MovieLens 1M † and BookCrossing ‡ for recommendation evaluation. For each dataset, the <user, item> pairs with explicit ratings are considered as positive examples (1s), and the rest of the pairs are negative examples (0s). Pairs with invalid / missing features are filtered. For feature engineering, we follow [141, 144] for consistent comparison. Items in both datasets follow the highly-skewed long-tail distribution, which shows the two datasets are well-suited for our targeted long-tail problem. Statistical details for the two datasets are shown in Table 8.2.

The MovieLens1M is divided by leave-one-out evaluation [159]: for each user, the most recent interacted item is for testing, the second most recent interacted item is for validation, and the rest are used for training. Since the BookCrossing dataset does not have the timestamp information, for each user, we randomly select one item for testing, one for validation, and use the rest of items for training.

†https://grouplens.org/datasets/movielens/1m/

‡http://www2.informatik.uni-freiburg.de/~cziegler/BX/

Table 8.2: Statistics and used content information in the datasets.

	MovieLens1M	**BookCrossing**
# User	6,040	50,454
# Items	3,706	222,154
# Feedback	1,000,209	1,031,175
Sparsity	95.5316%	99.9908%
User Features	IDs, Gender, Occupation, ZipCode, Age	IDs, Location, Age
Item Features	IDs, Title, Genres, Year	IDs, Title, Author, Year, Publisher

Table 8.3: Evaluation criteria based on different data slices [144]

HR@K/NDCG@K	**Overall**	**Head Items**	**Tail Items**
Good Results	–	↘	↗
Better Results	↗	–	↗
Great Results	↗	↗	↗

8.5.2 Setup

Evaluation Criteria: We consider the widely used metrics [64, 144] – Hit Ratio at top K (HR@K) and NDCG at top K (NDCG@K) – to evaluate the model performance on the long-tail distribution problem. HR@K measures whether the test item is retrieved in the top K ranked items. NDCG@K considers the position of recommended items, which gives higher scores to the top ranked items that are correctly recommended.

With the two metrics, for the long-tail distribution problem, we report those metrics evaluated on the tail, head slices and overall items separately. It shows the recommendation performance on different parts in the long-tail distribution. Similar as [144], based on the Pareto Principle [204], we split the first 20% most frequent items in MovieLens1M and 0.1% items in BookCrossing [§] for head items, and the rest items are tail items.

[§]The split rate of 0.1% can better show the performance differences among methods, especially for tail items.

We aim to build a single model that can improve the recommendation performance on tail slices items, and at the same time, relatively keep or improve the overall performance. So the evaluation criteria is shown in Table 8.3. We claim a model has a better/great performance, when the performance for overall/head items is improved. Note since the item distributions are usually the highly skewed, it is hard to improve the performance for both overall and tail slices items. The high heterogeneity between head and tail items makes it is even harder to improve both head and tail items in a single model. So the ‘better performance‘ and ‘best performance‘ in Table 8.3 is hard to achieve.

Baselines. All the models use the same content information for fair comparison. We use the widely applied state-of-the-art two-tower model as the backbone model.

- *Two-tower Model* [7]: This popular structure is flexible to consider different types of content information. It shows high performance and scalability in many real-word applications. Many models use it as the backbone model [144, 205].

Loss Function Refinement:

- *ClassBalance* [129]: This is a widely used state-of-the-art method for long-tail distribution problem in image classification settings. It calculates an effective number of samples for each imbalanced class and adopts it in a re-weighting term in loss function. We adopt it the method to the recommendation task as [144].
- *LogQ* [130]: This is a re-weighting method that addresses the long tail distribution problem in different areas. It is widely used in different applications. It constructs an item frequency-related term in loss function to reweight head and tail items in the learning process.

Decoupling:

- *NDP* [18]: It refers naive decoupling method, which is adapted from a recent state-of-the-art method for long-tail distribution problem in image recognition. Instead of jointly learning image representation and classification, it separately considers the two learning process and shows great success in image classification. We adopt the method to recommendation, which

Table 8.4: The recommendation performance of CDN versus baselines on MovieLens1M. Numbers after ± indicate the standard error of the mean.

Measure%	Overall		Head		Tail	
	HR@50	NDCG@50	HR@50	NDCG@50	HR@50	NDCG@50
Two-tower	24.68±0.16	7.29±0.06	33.83±0.39	10.06±0.14	10.32±0.50	2.96±0.14
ClassBalance	20.21±0.17	5.91±0.06	30.20±0.20	8.96±0.09	5.19±0.16	1.34±0.03
LogQ	13.37±0.23	3.50±0.07	10.75±0.38	2.62±0.09	17.30±0.15	4.81±0.05
NDP	14.72±0.14	4.28±0.06	14.92±0.22	4.20±0.07	14.41±0.22	4.41±0.07
BBN	24.91±0.09	7.28±0.07	34.19±0.20	10.03±0.0 9	10.37±0.3 6	2.98±0.10
CDN	26.75±0.12	7.93±0.04	36.46±0.14	10.97±0.04	11.51±0.18	3.15±0.08
Improv%	8.39%	8.78%	7.77%	9.05%	11.53%	6.42%

treats items as classes.

- *BBN* [135]: This is another state-of-the-art method for long-tail distribution problem. BBN utilizes the uniform sampler and reversed sampler to create a Bilateral-Branch Network that can simultaneously consider the two learning process. Since it does not handle the embedding learning for both user and item, we use the the user as the representation learning and item as the classifier.

Hyper-parameter settings. The number of user and item embedding dimension for all methods are set to be 64. Methods that utilize the two-tower architecture [7] have the same depth of networks and same hidden unites in each hidden layer for fair comparison. Concretely, the hidden layer size in user and item towers are set to be 1/2 number of neurons of its next lower layer, that is [256, 128, 64]. Relu is used as the activation function. For methods that utilize the experts, we implement each expert as a single-layer network. The number of experts that encode content and id information is set to be 1 for simplicity. The learning rate, dropout rate and regularization parameters are determined by grid search in the range of {0.1, 0.01, 0.001, 0.0001}, {0.1, 0.3, 0.5, 0.7, 0.9} and {0.0, 0.1, 0.01, 0.001} respectively. For methods that utilize 2-stage training, the number of training epochs are set to be 20. For the other methods, the number of epochs are 40 for fair comparison.

Table 8.5: The recommendation performance of CDN versus baselines on BookCrossing. Numbers after ± indicate the standard error of the mean.

Measure%	Overall		Head		Tail	
	HR@50	NDCG@50	HR@50	NDCG@50	HR@50	NDCG@50
Two-tower	2.32±0.45	0.81±0.17	19.76±4.62	7.58±1.81	0.72±0.08	0.19±0.02
ClassBalance	1.47±0.32	0.47±0.10	8.94±2.46	3.36±0.88	0.79±0.13	0.21±0.04
LogQ	1.27±0.25	0.37±0.07	5.39±1.54	1.76±0.51	0.89±0.14	0.24±0.04
NDP	1.58±0.30	0.42±0.08	7.24±1.78	1.94±0.48	1.06±0.19	0.43±0.09
BBN	0.80±0.20	0.20±0.05	2.62±0.96	0.66±0.26	0.63±0.14	0.22±0.01
CDN	2.52±0.3	0.82±0.15	21.18±4.82	7.42±1.86	0.81±0.05	0.22±0.01
Improv%	8.94%	1.16%	7.21%	-2.09%	13.31%	12.90%

8.5.3 Recommendation Performance (RQ1)

Table 8.4 and Table 8.5 show the recommendation performance of CDN and baselines on the MovieLens1M and BookCrossing datasets. The reported numbers are the average from 5 experiments. Numbers after ± are the standard error of the mean. The last row shows the relative improvement over the two-tower model for each metric. We can see CDN outperforms the other methods – it brings significant improvements not only for tail slice items, but also for overall recommendation performance. concretely:

First, among different strategies for the long-tail distribution problem, CDN achieves the best performance. It shows the superior of cross decoupling in dealing with long-tail distribution compared with reweighting and transfer learning. It is also worth to notice that most of those methods only brings improvement for only tail items, but the performance of head items largely decreases. CDN brings improvements for both head and tail items. It further demonstrates the importance to consider both the prior and conditional knowledge differences.

Second, all the decoupling methods – NDP, BBN and CND outperform the backbone two tower model on tail items. Since all the models are based on the two-tower architecture, it confirms the decoupling method can better encode the user preference towards tail items. Furthermore, among decoupling methods, CND further improves the recommendation performance of overall and head items in most situations. It shows the importance of considering the cross decoupling between user and item for long-tail distribution problem in recommendation. It is also worth to notice that, compared with NDP, BBN gains better overall performance. It further shows the forgetting issue in NDP could heavily influence the overall performance, especially for the highly skewed long tail distributions in recommendation. And simultaneously training strategy can alleviate the forgetting issue.

Third, compared with the standard error of the mean, CDN is not only more robust for the recommendation prediction of tail items, and also for overall performance. As shown in Table 8.4 and 8.5, the standard error of the mean in CDN is smaller than the two tower model. One possible reason is that the CDN is jointly trained based on both the original distributed dataset and balanced dataset, which obtains the similar effect as the data augmentation and effectively alleviate the overfitting for tail items. Another reason is that we use the adapter in cumulative learning as a regularization, which can better balanced the learning process between head and tail items in the highly skewed long-tail item distribution.

8.5.4 Cross-Decoupling (RQ2)

One of the key part in CDN is the cross-decoupling design. In this section, we conduct experiment to explore how the cross-decoupling contribute to the recommendation. The ablation study is designed as follows:

- *UDN*: This is the model that only implements the decoupling from the user side. That is, it only uses Bilateral-Branch Network.
- *IDN*: This is the model that only decouples the item side. That is, it utilizes the content and ID multiexperts, and a gate with the item frequency.
- *BDN*: Instead of using the regularizer adapter in the cumulative learning as the regularizer, it considers the two branches as equally important. And the preference score is calculated by the average outputs of the two branches: $s(i_t, i_r) = 0.5\mathbf{y}_t^T\mathbf{x}_t + 0.5\mathbf{y}_r^T\mathbf{x}_r$.

Results are shown in Table 8.6. We can observe that CDN achieves the best performance for both tail items and head items. It further shows the importance of considering cross decoupling.

Table 8.6: Cross decoupling study on MovieLens1M. Numbers after ± indicate the standard error of the mean.

Measure%	Overall		Head		Tail	
	HR@50	NDCG@50	HR@50	NDCG@50	HR@50	NDCG@50
Two-tower	24.68±0.16	7.29±0.06	33.83±0.39	10.06±0.14	10.32±0.50	2.96±0.14
UDN	25.55±0.08	7.53±0.03	34.98±0.11	10.35±0.06	10.77±0.13	3.10±0.03
IDN	25.85±0.15	7.58±0.05	34.82±0.21	10.26±0.08	11.79±0.22	3.37±0.08
CDN	26.75±0.12	7.93±0.04	36.46±0.14	10.97±0.04	11.51±0.18	3.15±0.08

Concretely,

Both UDN and IDN bring improvements on head and tail slice items. The results further demonstrate decoupling can help learning the user preference towards different items in the long-tail distribution, either from user side or item side. However, the improvements of UDN and IDN are not significant. One possible reason is that those methods only decoupled one side (user or item side) information. The learning of the other side information still could be influenced by the long-tail distribution. It confirms our assumption that both prior and conditional knowledge are needed to be considered to improve the recommendation performance.

While BDN outperforms the two tower model on the tail items, the performances of head and tail items largely decrease. Different from that, CDN, which considers the balanced branch as a regularizer, achieves good performance in both tail and head items. It verifies the importance of considering the balanced branch as a regularizer rather than a component in the model for the heavily skewed long-tail distribution dataset in recommendation. The regularizer can efficiently alleviate the overfitting problem of items in the long tail slices.

8.5.5 Comparison of Expert Design (RQ3)

The experts in CDN are designed to separately learn the content and ID information to better handle the high heterogeneity between head and tail items. There are also many other design choices, such as using experts to learn the unbalanced/balanced dataaset in item tower (simiar as the user tower), or directly design experts for head/tail items. In this section, we explore those expert design choices. We first evaluate expert design choices in addressing the long-tail distribution problem in recommendation, and then analyze the differences of memorization and generalization between head and tail items.

8.5.5.1 Recommendation performance

To separately consider the head and tail items, there are some design choices:

- *Unbalanced/Balanced*: First, similar as the user side, we could use two experts: one is to train the original dataset, and the other is used to train the relatively balanced dataset. So the model will be $z_c = u_c i_c$ and $z_r = u_r i_r$ where i_c is the item representation learned from balanced data distribution and i_c is the item representation learned from original data distribution. Then softmax is used to calculate the probability that the user prefer the corresponding item, which is p_r and p_c. The loss function is $L = \alpha E(p_r, y_r) + (1 - \alpha)E(p_c, y_c)$.
- *Head/Tail*: Another way to separately consider the head and tail items is to directly design an expert to learn head items, and an expert to learn tail items. Then a gating network [206] is used to connect the two experts;
- *ID/Content*: the experts in CDN is designed to consider the ID and content separately, to learn the head and tail items. Here the assumption is that head and tail items will consider the ID information differently due to the large gap between user feedback distribution.

Results are shown in Table 8.7. Based on the results, we can see that the method of ID/Content expert achieves the best performance compared with the other methods.

For the Unbalanced/Balanced method, it is consist with the user branch. That is, both user and item uses the balanced/unbalanced datasets to train. However, the method only gains a relative improvement on NDCG for head items. Compared with the ID/Content, the performance of Unbalanced/Balanced method on tail items largely decreases, and the overall performance also decreases. One likely reason is that if both user and item learn the same unbalanced/balanced dataset at the same time, the tail items are easily to get overfitted.

For the head/tail method, we can see there is an obvious drop out in the recommendation performance of tail items. One likely reason is that the tail specific expert easily causes over-fitting

Table 8.7: Expert design study on MovieLens1M. Numbers after ± indicate the standard error of the mean.

Measure%	Overall		Head		Tail	
	HR@50	NDCG@50	HR@50	NDCG@50	HR@50	NDCG@50
Two-tower	24.68±0.16	7.29±0.06	33.83±0.39	10.06±0.14	10.32±0.50	2.96±0.14
Unbalanced/Balanced	22.07±0.12	6.71±0.05	33.77±0.21	10.43±0.07	3.72±0.07	0.88±0.03
Head/Tail	19.21±0.34	5.73±0.11	31.39±0.54	9.36±0.18	0.11±0.05	0.02±0.01
ID/Content	26.75±0.12	7.93±0.04	36.46±0.14	10.97±0.04	11.51±0.18	3.15±0.08

for tail items since the user feedback for tail items are very sparse. Furthermore, for the head and overall performance, though head/tail method achieves similar performance as two-tower model on head items, the overall performance still largely decreases. It is reasonable since there is a large percentage of tail items in the highly skewed long-tail recommendation dataset. It further shows the importance of using ID/Content expert to separately consider head and tail differences in the long-tail distribution dataset.

8.5.5.2 Memorization vs Generalization

Memorization and generalization [146, 148] play a key role in deep neural networks. Properly combining the benefits of memorization and generalization can help improve the performance. One assumption for the CDN model is that the head and tail items pay different attention for the content and ID information. Therefore, in this section, we further investigate the memorization vs generalization for head and tail items. That is, we check the content and ID expert gate value for head/tail items.

Results are shown in Figure 8.3. The blue color shows the gate value for head items, and the orange color shows the gate value for tail items. First, compared the value between id expert and content expert, we can see both head and tail items put more weights to the content experts. It is reasonable since both datasets are very sparse and we utilize rich content information. Second, compared the gate value between head and tail items, we can find head items put more weights to ID experts while tail items put more weights to content experts. It verifies our assumption that head items based more on memorization since they contain relatively rich user feedback information. It

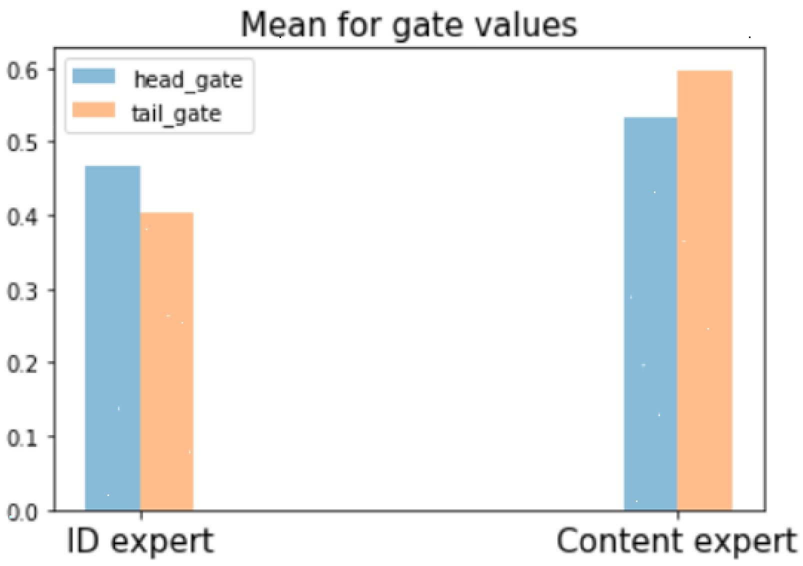


Figure 8.3: Gate value for ID and content experts in CDN.

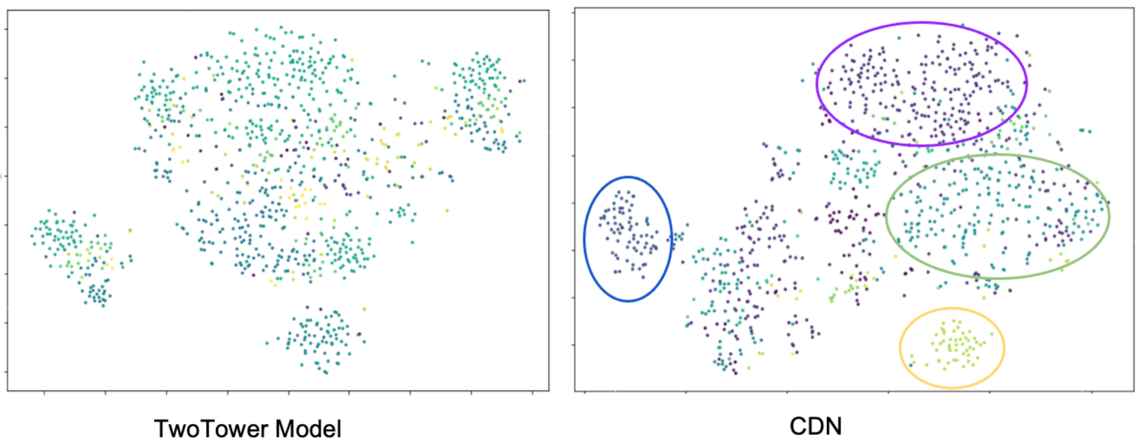


Figure 8.4: Embedding visualization comparison.

further shows the resonability of design ID/content experts to separately consider the head and tail items in the long-tail distribution settings.

8.5.6 Representation Visualization (RQ4)

Last, we visualize the learned embedding of CDN to explore its representation learning ability. Results are shown in Figure 8.4 by using t-SNE [201]. Same color represents the same genera.

As shown in Figure 8.4, compared the learned representation of tail slice items by two tower model, our proposed model CDN could better cluster movies that in the same genre. The visualization shows that CDN is able to well cluster the tail slices items based on their content information. It further shows CDN is able to learn meaningful representations for tail items, which also align with the findings in section 8.5.5.2 where tail slice items put more weights to content gate.

8.6 Summary

We explore the long-tail distribution problem in recommendation. Based on the theoretical analysis and experiment study, we find the long-tail distribution problem could be addressed from both prior and conditional knowledge perspectives. Considering that, we propose a novel cross decoupling framework that decoupled the learning process of user and item from the two perspectives. Experiment results show CDN outperforms the state-of-the-art methods. In the future work, we are interested in considering the dynamic long-tail distribution problem.

9. CONCLUSIONS AND FUTURE WORK

Content-aware recommendations enable us to efficiently leverage different types of information for improved recommendation. They are exceptionally powerful in many applications, especially for cold-start problems where there are only a few user-item interactions, and for new users and items recommendations. With the unprecedented explosion of various information, there are increasing opportunities to explore new methodologies and models for content-aware recommendation. In this chapter, we highlight our conclusions align with challenges, and give a discussion of future research opportunities.

9.1 Conclusions

In this dissertation, we focus on leveraging content information to enhance learning of user preference towards items and further give recommendation, including mining, aggregating and decoupled both user and item side content information. This dissertation mainly focus on the content information of item and user relations and their application to recommendation. While those information is available, there is a significant gap towards properly incorporating the user/item various relations to collaborative signals for enhanced recommendation. Furthermore, when the time factors are considered in recommendation (sequential recommendation), it becomes even harder to efficient handle those complicated relations from both user and item side. To bridge the gap, this dissertation makes three unique contributions:

First, to tackle the **sparsity** challenge of item relations, we leverages the item visual and textual information to uncover both stylistic and functional evidence of item relations. We further aggregate the user ratings through the Bayesian inference to build a quality-aware complementary item recommendation called ENCORE. When the time factor is considered, to tackle the sparsity in both user sequence and item relations, we propose a hierarchical translation-based recommendation method called HierTrans that extends traditional item-level relations to the category-level, to help capture dynamic sequence patterns that can generalize across users and across time. Additionally, HierTrans is build on a hierarchical temporal graph that contains item different relations at category-level and user sequences at the item-level. The graph is capable to capture higher order item relations to alleviate the sparsity issue. The insights are to utilize rich side information and mine the properties of content information (e.g., high order information) which can be utilized to enhance the learning of user preference towards items.

Second, considering the **high heterogeneous** issue between user relations and user-item interactions, we propose to consider the social resonance effect that can naturally connect social network with user interactions towards items. Concretely, social resonance shows that users are more influenced by opinions that have similar vibe. Based on that, we create a item-based user influence network to connect users in user social network, and then learning the social influence towards user with a graph-based mutual learning framework. The framework is able to comprehensively uncover the social resonance from both preference-based and graph-based aspects. Furthermore, when we consider the user social relations in sequential recommendation, we build a bidirectional LSTM to capture the dynamic social influence, and utilize an attention mechanism to connect the social influence with user purchase preference. Both methods enable us to more effectively consider the impacts of user social network to user preference behavior and further improve the recommendation performance.

Third, after mining user and item content information and aggregating them to enhance recommendation, we further analyze the **duplication** problem and **high-skewed** long-tail distribution problem in content-aware recommendation, and further propose to disentangle/decouple those content information that can more effectively consider various content information for different items. Concretely, considering the duplication issue, we propose a two-level disentanglement learning that disentangled collaborative signals and content signals, and each feature in granularity level. With those methods, we successfully address the high-skewed data issue and duplication issue. The proposed methods brings large improvement for recommendation. Furthermore, our proposed decoupling method separately considers the memorization and generalization for popular and cold-start items. In this way, collaborative signals is able to contributes more to popular items and content signals is able to contributes more to cold-start items.

9.2 Future Research Opportunities

While this dissertation has shown the success of properly utilizing the content information to improve the recommendation, there are still many challenges. For future work, we are interested in the following directions:

- **Disentanglement/Decoupling learning in dynamic settings**. In this dissertation, we have discussed the disentanglement/decoupling learning when different types of content information are considered. The proposed models shows improvement in recommendation, and also give us insights that disentanglement/decoupling can help improve the interpretation of learned user and item representations. Besides that, we also find the disentanglement/decoupling learning could be highly related to time. For example, for fashion recommendation, with the visual evolution, the disentangled content would change. Therefore, it becomes challenging to effectively extract disentangled features that are representative in each time period. Actually, in practice, the streaming data is usually considered. Hence, it becomes important to explore the disentanglement learning in sequential recommendation.
- **Content knowledge transfer cross domains**. This dissertation considers user and item content information, and discussed the mining, aggregation and decoupling aspects that can leverage those content to improve the recommendation. In practice, with the increase of various platforms, there are many additional sources of information that can be shared in different platforms. For example, for user side, users in Facebook, Twitter and Instagram can share the same account. For item side, same books (e.g., Harry Potter) can be purchased in different platforms, like Amazon and Ebay. Those connections make it convenient to utilize the shared information to address the sparsity issue in recommendation from either user side or item side. If we consider the sequential recommendation scenarios, this knowledge transfer across different domains becomes more challenging, since the transferred knowledge could dynamically changed over time. We are interested to explore effective methods to extract and transfer the

shared knowledge that can be adapted to different domains for enhanced recommendation.

REFERENCES

[1] J. Xiao, M. Wang, B. Jiang, and J. Li, "A personalized recommendation system with combinational algorithm for online learning," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 3, pp. 667–677, 2018.

[2] T. Chen, "Ubiquitous multicriteria clinic recommendation system," *Journal of medical systems*, vol. 40, no. 5, p. 113, 2016.

[3] N. Mustafa, A. O. Ibrahim, A. Ahmed, and A. Abdullah, "Collaborative filtering: Techniques and applications," in *2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, pp. 1–6, IEEE, 2017.

[4] R. He and J. McAuley, "Vbpr: visual bayesian personalized ranking from implicit feedback," in *AAAI conference on artificial intelligence*, 2016.

[5] H. F. Abdulkarem, G. Y. Abozaid, and M. I. Soliman, "Context-aware recommender system frameworks, techniques, and applications: A survey," in *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, pp. 180–185, IEEE, 2019.

[6] T. Liu, Z. Wang, J. Tang, S. Yang, G. Y. Huang, and Z. Liu, "Recommender systems with heterogeneous side information," in *The World Wide Web Conference*, pp. 3027–3033, 2019.

[7] X. Yi, J. Yang, L. Hong, D. Z. Cheng, L. Heldt, A. Kumthekar, Z. Zhao, L. Wei, and E. Chi, "Sampling-bias-corrected neural modeling for large corpus item recommendations," in *Proceedings of the ACM Conference on Recommender Systems*, pp. 269–277, 2019.

[8] Y. Zhang, H. Lu, W. Niu, and J. Caverlee, "Quality-aware neural complementary item recommendation," in *Proceedings of the 12th ACM Conference on Recommender Systems*, pp. 77–85, 2018.

[9] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206, IEEE, 2018.

[10] J. McAuley, R. Pandey, and J. Leskovec, "Inferring networks of substitutable and complementary products," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 785–794, 2015.

[11] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pp. 43–52, 2015.

[12] Z. Wang, Z. Jiang, Z. Ren, J. Tang, and D. Yin, "A path-constrained framework for discriminating substitutable and complementary products in e-commerce," in *WSDM*, ACM, 2018.

[13] Q. Wu, H. Zhang, X. Gao, P. He, P. Weng, H. Gao, and G. Chen, "Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems," in *WWW*, 2019.

[14] X. Wang, W. Zhu, and C. Liu, "Social recommendation with optimal limited attention," in *SIGKDD*, ACM, 2019.

[15] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *WWW*, International World Wide Web Conferences Steering Committee, 2016.

[16] V. Jagadeesh, R. Piramuthu, A. Bhardwaj, W. Di, and N. Sundaresan, "Large scale visual recommendations from street fashion images," in *SIGKDD*, ACM, 2014.

[17] Y. Zhang, Y. He, J. Wang, and J. Caverlee, "Adaptive hierarchical translation-based sequential recommendation," in *Proceedings of The Web Conference 2020*, pp. 2984–2990, 2020.

[18] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis, "Decoupling representation and classifier for long-tailed recognition," *arXiv preprint arXiv:1910.09217*, 2019.

[19] H. He, D. Chen, A. Balakrishnan, and P. Liang, "Decoupling strategy and generation in negotiation dialogues," *arXiv preprint arXiv:1808.09637*, 2018.

[20] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, 2003.

[21] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, 2001.

[22] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items," *Expert Systems with Applications*, vol. 69, 2017.

[23] P. Pirasteh, J. J. Jung, and D. Hwang, "Item-based collaborative filtering with attribute correlation: a case study on movie recommendation," in *Asian Conference on Intelligent Information and Database Systems*, Springer, 2014.

[24] D. Li, C. Chen, Q. Lv, L. Shang, Y. Zhao, T. Lu, and N. Gu, "An algorithm for efficient privacy-preserving item-based collaborative filtering," *Future Generation Computer Systems*, vol. 55, 2016.

[25] G. D. Linden, J. A. Jacobi, and E. A. Benson, "Collaborative recommendations using item-to-item similarity mappings," 2001. US Patent 6,266,649 (to Amazon.com).

[26] C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," *Recommender systems handbook*, 2011.

[27] I. Cantador, A. Bellogín, and D. Vallet, "Content-based recommendation in social tagging systems," in *RecSys*, ACM, 2010.

[28] P. Melville, R. J. Mooney, and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," in *AAAI*, 2002.

[29] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, 2004.

[30] G. Karypis, “Evaluation of item-based top-n recommendation algorithms,” in *Proceedings of the tenth international conference on Information and knowledge management*, pp. 247–254, 2001.

[31] X. Ning and G. Karypis, “Slim: Sparse linear methods for top-n recommender systems,” in *ICDM*, IEEE, 2011.

[32] S. Kabbur, X. Ning, and G. Karypis, “Fism: factored item similarity models for top-n recommender systems,” in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 659–667, 2013.

[33] Q. Shambour, M. Hourani, and S. Fraihat, “An item-based multi-criteria collaborative filtering algorithm for personalized recommender systems,” *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 8, 2016.

[34] C. Li and K. He, “Cbmr: An optimized mapreduce for item-based collaborative filtering recommendation algorithm with empirical analysis,” *Concurrency and Computation: Practice and Experience*, vol. 29, no. 10, 2017.

[35] Y. Bao, H. Fang, and J. Zhang, “Topicmf: Simultaneously exploiting ratings and reviews for recommendation,” in *AAAI*, 2014.

[36] H.-H. Lee and W.-G. Teng, “Incorporating multi-criteria ratings in recommendation systems,” in *IRI*, IEEE, 2007.

[37] S. Wei, X. Zheng, D. Chen, and C. Chen, “A hybrid approach for movie recommendation via tags and ratings,” *Electronic Commerce Research and Applications*, vol. 18, 2016.

[38] Y. Ma, J. Jia, S. Zhou, J. Fu, Y. Liu, and Z. Tong, “Towards better understanding the clothing fashion styles: A multimodal deep learning approach,” in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[39] X. Chen, Y. Zhang, Q. Ai, H. Xu, J. Yan, and Z. Qin, “Personalized key frame recommendation,” in *SIGIR*, ACM, 2017.

[40] J. Kim, D. Lee, and K.-Y. Chung, “Item recommendation based on context-aware model for personalized u-healthcare service,” *Multimedia Tools and Applications*, vol. 71, no. 2, 2014.

[41] X. Chen, Z. Qin, Y. Zhang, and T. Xu, “Learning to rank features for recommendation over multiple categories,” in *SIGIR*, ACM, 2016.

[42] T. Li, A. Liu, and C. Huang, “A similarity scenario-based recommendation model with small disturbances for unknown items in social networks,” *IEEE Access*, vol. 4, 2016.

[43] J. McAuley and A. Yang, “Addressing complex and subjective product-related queries with customer reviews,” in *WWW*, International World Wide Web Conferences Steering Committee, 2016.

[44] J. Sun, G. Wang, X. Cheng, and Y. Fu, “Mining affective text to improve social media item recommendation,” *Information Processing & Management*, vol. 51, no. 4, 2015.

[45] R. He, C. Packer, and J. McAuley, “Learning compatibility across categories for heterogeneous item recommendation,” in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 937–942, IEEE, 2016.

[46] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He, “A simple convolutional generative network for next item recommendation,” in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 582–590, 2019.

[47] J. You, Y. Wang, A. Pal, P. Eksombatchai, C. Rosenburg, and J. Leskovec, “Hierarchical temporal convolutional networks for dynamic recommender systems,” in *International Conference on World Wide Web*, pp. 2236–2246, 2019.

[48] S. Bai, J. Z. Kolter, and V. Koltun, “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,” *arXiv preprint arXiv:1803.01271*, 2018.

[49] R. He and J. McAuley, “Fusing similarity models with markov chains for sparse sequential recommendation,” in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 191–200, IEEE, 2016.

[50] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of the 19th International Conference on World Wide Web*, pp. 811–820, 2010.

[51] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[52] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[53] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[54] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 565–573, 2018.

[55] R. He, W.-C. Kang, and J. McAuley, "Translation-based recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 161–169, 2017.

[56] R. He, W.-C. Kang, and J. McAuley, "Translation-based recommendation: A scalable method for modeling sequential behavior," in *IJCAI*, pp. 5264–5268, 2018.

[57] R. Pasricha and J. McAuley, "Translation-based factorization machines for sequential recommendation," in *Proceedings of the 12th ACM Conference on Recommender Systems*, pp. 63–71, 2018.

[58] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *International Conference on World Wide Web*, pp. 151–161, 2019.

[59] W.-C. Kang, M. Wan, and J. McAuley, "Recommendation through mixtures of heterogeneous item relationships," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1143–1152, 2018.

[60] X. Xin, X. He, Y. Zhang, Y. Zhang, and J. Jose, “Relational collaborative filtering: Modeling multiple item relations for recommendation,” in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 125–134, 2019.

[61] V. Rakesh, S. Wang, K. Shu, and H. Liu, “Linked variational autoencoders for inferring substitutable and supplementary items,” in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 438–446, 2019.

[62] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, *et al.*, “Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia,” *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[63] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, “Ripplenet: Propagating user preferences on the knowledge graph for recommender systems,” in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 417–426, 2018.

[64] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, “Neural graph collaborative filtering,” in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 165–174, 2019.

[65] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, “Graph convolutional neural networks for web-scale recommender systems,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge discovery and data mining*, pp. 974–983, 2018.

[66] Y. Zhang, Q. Ai, X. Chen, and P. Wang, “Learning over knowledge-base embeddings for recommendation,” *arXiv preprint arXiv:1803.06540*, 2018.

[67] M. Nickel, V. Tresp, and H.-P. Kriegel, “Factorizing yago: scalable machine learning for linked data,” in *Proceedings of the 21st International Conference on World Wide Web*, pp. 271–280, 2012.

[68] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, “A semantic matching energy function for learning with multi-relational data,” *Machine Learning*, vol. 94, no. 2, pp. 233–259, 2014.

[69] W. Ma, M. Zhang, Y. Cao, W. Jin, C. Wang, Y. Liu, S. Ma, and X. Ren, “Jointly learning explainable rules for recommendation with knowledge graph,” in *International Conference on World Wide Web*, pp. 1210–1221, 2019.

[70] Z. Wang and J.-Z. Li, “Text-enhanced representation learning for knowledge graph,” in *IJCAI*, pp. 1293–1299, 2016.

[71] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, “Modeling relation paths for representation learning of knowledge bases,” *arXiv preprint arXiv:1506.00379*, 2015.

[72] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Advances in Neural Information Processing Systems*, pp. 2787–2795, 2013.

[73] Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by translating on hyperplanes,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2014.

[74] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embeddings for knowledge graph completion,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015.

[75] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, “Graph neural networks for social recommendation,” in *WWW*, 2019.

[76] T.-H. Lin, C. Gao, and Y. Li, “Cross: Cross-platform recommendation for social e-commerce,” in *SIGIR*, 2019.

[77] C. Chen, M. Zhang, C. Wang, W. Ma, M. Li, Y. Liu, and S. Ma, “An efficient adaptive transfer neural network for social-aware recommendation,” in *SIGIR*, ACM, 2019.

[78] C. Yang, J. Zhang, H. Wang, S. Li, M. Kim, M. Walker, Y. Xiao, and J. Han, “Relation learning on social networks with multi-modal graph edge variational autoencoders,” in *WSDM*, 2020.

[79] R. Reshma, G. Ambikesh, and P. S. Thilagam, “Alleviating data sparsity and cold start in recommender systems using social behaviour,” in *ICRTIT*, IEEE, 2016.

[80] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, “A neural influence diffusion model for social recommendation,” *arXiv preprint arXiv:1904.10322*, 2019.

[81] G. Guo, J. Zhang, D. Thalmann, A. Basu, and N. Yorke-Smith, “From ratings to trust: an empirical study of implicit trust in recommender systems,” in *SAC*, 2014.

[82] H. Ma, “An experimental study on implicit social recommendation,” in *SIGIR*, ACM, 2013.

[83] S. Mukherjee and S. Guennemann, “Ghostlink: Latent network inference for influence-aware recommendation,” in *WWW*, 2019.

[84] F. Xiong, W. Shen, H. Chen, S. Pan, X. Wang, and Z. Yan, “Exploiting implicit influence from information propagation for social recommendation,” *IEEE transactions on cybernetics*, 2019.

[85] C. Lin, R. Xie, X. Guan, L. Li, and T. Li, “Personalized news recommendation via implicit social experts,” *Information Sciences*, 2014.

[86] Y. Hu, X. Yi, and L. S. Davis, “Collaborative fashion recommendation: A functional tensor factorization approach,” in *MULTIMEDIA*, ACM, 2015.

[87] X. Chao, M. J. Huiskes, T. Gritti, and C. Ciuhu, “A framework for robust feature selection for real-time fashion style recommendation,” in *IMCE*, ACM, 2009.

[88] V. Gabale and A. P. Subramanian, “How to extract fashion trends from social media? a robust object detector with support for unsupervised learning,” *arXiv preprint arXiv:1806.10787*, 2018.

[89] S. C. Hidayati, C.-C. Hsu, Y.-T. Chang, K.-L. Hua, J. Fu, and W.-H. Cheng, “What dress fits me best?: Fashion recommendation on the clothing style for personal body shape,” in *2018 ACM Multimedia Conference on Multimedia Conference*, pp. 438–446, ACM, 2018.

[90] W.-C. Kang, E. Kim, J. Leskovec, C. Rosenberg, and J. McAuley, “Complete the look: Scene-based complementary product recommendation,” in *CVPR*, 2019.

[91] W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, and Z. Qin, “Aesthetic-based clothing recommendation,” in *WWW*, ACM, 2018.

[92] N. Murray, L. Marchesotti, and F. Perronnin, “Ava: A large-scale database for aesthetic visual analysis,” in *CVPR*, IEEE, 2012.

[93] Y. Koren, “Collaborative filtering with temporal dynamics,” in *SIGKDD*, ACM, 2009.

[94] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, “Recurrent recommender networks,” in *WSDM*, ACM, 2017.

[95] P. Sun, L. Wu, and M. Wang, “Attentive recurrent social recommendation,” in *SIGIR*, ACM, 2018.

[96] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, and E. H. Chi, “Latent cross: Making use of context in recurrent recommender systems,” in *WSDM*, ACM, 2018.

[97] Y. J. Ko, L. Maystre, and M. Grossglauser, “Collaborative recurrent neural networks for dynamic recommender systems,” in *Journal of Machine Learning Research: Workshop and Conference Proceedings*, 2016.

[98] E. F. McQuarrie, J. Miller, and B. J. Phillips, “The megaphone effect: Taste and audience in fashion blogging,” *Journal of Consumer Research*, 2012.

[99] Fashion influencer, “Fashion influencer — Wikipedia, the free encyclopedia,” 2018. [Online; accessed 5-November-2018].

[100] A. Marwick, “They are really profound women, they are entrepreneurs: Conceptions of authenticity in fashion blogging,” in *ICWSM*, 2013.

[101] C. L. Vineyard, "The relationship between fashion blogs and intention to purchase and word of mouth behavior," 2014.

[102] M. Z. M. Zain, P. Perry, and L. Quinn, "The influence of fashion bloggers on the pre-purchase decision for online fashion products among generation y female malaysian consumers," *World Academy of Science, Engineering and Technology, International Journal of Social, Behavioral, Educational, Economic, Business and Industrial Engineering*, 2018.

[103] M. De Veirman, V. Cauberghe, and L. Hudders, "Marketing through instagram influencers: the impact of number of followers and product divergence on brand attitude," *International Journal of Advertising*, 2017.

[104] M. Sudha and K. Sheena, "Impact of influencers in consumer decision process: the fashion industry," *SCMS Journal of Indian Management*, 2017.

[105] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, "Aspect-aware latent factor model: Rating prediction with ratings and reviews," in *Proceedings of the World Wide Web conference*, pp. 639–648, 2018.

[106] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[107] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *NIPS*, 2008.

[108] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 17, pp. 3203–3209, Melbourne, Australia, 2017.

[109] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of the International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 549–558, 2016.

[110] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "Geomf: joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proceedings of the ACM*

SIGKDD international conference on Knowledge discovery and data mining, pp. 831–840, 2014.

[111] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proceedings of the ACM International Conference on Information and Knowledge Management*, pp. 261–270, 2014.

[112] Z. Lu, Z. Dou, J. Lian, X. Xie, and Q. Yang, "Content-based collaborative filtering for news topic recommendation," in *AAAI conference on artificial intelligence*, 2015.

[113] S. Rendle, "Factorization machines," in *2010 IEEE International Conference on Data Mining*, pp. 995–1000, IEEE, 2010.

[114] X. Li and J. She, "Collaborative variational autoencoder for recommender systems," in *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 305–314, 2017.

[115] J. Lv, B. Song, J. Guo, X. Du, and M. Guizani, "Interest-related item similarity model based on multimodal data for top-n recommendation," *IEEE Access*, vol. 7, pp. 12809–12821, 2019.

[116] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," *Proceedings of the International Conference on Learning Representations (ICLR)*, vol. 2, no. 5, p. 6, 2017.

[117] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem, "Challenging common assumptions in the unsupervised learning of disentangled representations," in *International Conference on Machine Learning (ICML)*, pp. 4114–4124, 2019.

[118] R. Hamaguchi, K. Sakurada, and R. Nakamura, "Rare event detection using disentangled representation learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9327–9335, 2019.

[119] F. Locatello, G. Abbati, T. Rainforth, S. Bauer, B. Schölkopf, and O. Bachem, "On the fairness of disentangled representations," in *Advances in Neural Information Processing Systems*, pp. 14611–14624, 2019.

[120] C. Burgess and H. Kim, "3d shapes dataset." https://github.com/deepmind/3dshapes-dataset/, 2018.

[121] T. Q. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud, "Isolating sources of disentanglement in variational autoencoders," in *Advances in Neural Information Processing Systems*, pp. 2610–2620, 2018.

[122] Q. Song, S. Chang, and X. Hu, "Coupled variational recurrent collaborative filtering," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 335–343, 2019.

[123] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[124] J. Ma, C. Zhou, P. Cui, H. Yang, and W. Zhu, "Learning disentangled representations for recommendation," in *Advances in Neural Information Processing Systems*, pp. 5711–5722, 2019.

[125] J. Ren, C. Yu, S. Sheng, X. Ma, H. Zhao, S. Yi, and H. Li, "Balanced meta-softmax for long-tailed visual recognition," *arXiv preprint arXiv:2007.10740*, 2020.

[126] X. Wu, C. Li, and Y. Miao, "Discovering topics in long-tailed corpora with causal intervention," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 175–185, 2021.

[127] T. Wei, W.-W. Tu, Y.-F. Li, and G.-P. Yang, "Towards robust prediction on tail labels," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1812–1820, 2021.

[128] K. Tang, J. Huang, and H. Zhang, “Long-tailed classification by keeping the good and removing the bad momentum causal effect,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2020.

[129] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, “Class-balanced loss based on effective number of samples,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[130] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, “Long-tail learning via logit adjustment,” *arXiv preprint arXiv:2007.07314*, 2020.

[131] K. Dahiya, A. Mittal, D. Saini, K. Dave, H. Jain, S. Agarwal, and M. Varma, “Deepxml: Scalable & accurate deep extreme classification for matching user queries to advertiser bid phrases,” 2019.

[132] Y. Cao, J. Kuang, M. Gao, A. Zhou, Y. Wen, and T.-S. Chua, “Learning relation prototype from unlabeled texts for long-tail relation extraction,” *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[133] Y.-X. Wang, D. Ramanan, and M. Hebert, “Learning to model the tail,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 7032–7042, 2017.

[134] M. A. Jamal, M. Brown, M.-H. Yang, L. Wang, and B. Gong, “Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7610–7619, 2020.

[135] B. Zhou, Q. Cui, X.-S. Wei, and Z.-M. Chen, “Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9719–9728, 2020.

[136] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, “Challenging the long tail recommendation,” *arXiv preprint arXiv:1205.6700*, 2012.

[137] Y.-J. Park and A. Tuzhilin, "The long tail of recommender systems and how to leverage it," in *Proceedings of the 2008 ACM conference on Recommender systems*, pp. 11–18, 2008.

[138] X. N. Lam, T. Vu, T. D. Le, and A. D. Duong, "Addressing cold-start problem in recommendation systems," in *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pp. 208–211, 2008.

[139] K. Zhou, S.-H. Yang, and H. Zha, "Functional matrix factorizations for cold-start recommendation," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 315–324, 2011.

[140] J. Li, K. Lu, Z. Huang, and H. T. Shen, "Two birds one stone: on both cold-start and long-tail recommendation," in *Proceedings of the 25th ACM international conference on Multimedia*, pp. 898–906, 2017.

[141] Y. Lu, Y. Fang, and C. Shi, "Meta-learning on heterogeneous information networks for cold-start recommendation," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1563–1573, 2020.

[142] A. Beutel, E. H. Chi, Z. Cheng, H. Pham, and J. Anderson, "Beyond globally optimal: Focused learning for improved recommendations," in *Proceedings of the 26th International Conference on World Wide Web*, pp. 203–212, 2017.

[143] K. Lee and K. Lee, "My head is your tail: applying link analysis on long-tailed music listening behavior for music recommendation," in *Proceedings of the fifth ACM conference on Recommender systems*, pp. 213–220, 2011.

[144] Y. Zhang, D. Z. Cheng, T. Yao, X. Yi, L. Hong, and E. H. Chi, "A model of two tales: Dual transfer learning framework for improved long-tail item recommendation," in *Proceedings of the Web Conference 2021*, pp. 2220–2231, 2021.

[145] J. Yin, C. Liu, W. Wang, J. Sun, and S. C. Hoi, "Learning transferrable parameters for long-tailed sequential user behavior modeling," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 359–367, 2020.

[146] C. Zhang, S. Bengio, M. Hardt, M. C. Mozer, and Y. Singer, "Identity crisis: Memorization and generalization under extreme overparameterization," in *ICLR*, 2020.

[147] G. Brown, M. Bun, V. Feldman, A. Smith, and K. Talwar, "When is memorization of irrelevant training data necessary for high-accuracy learning?," in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 123–132, 2021.

[148] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, *et al.*, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*, pp. 7–10, 2016.

[149] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[150] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, pp. 1188–1196, PMLR, 2014.

[151] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[152] B. Rozenkrants, S. C. Wheeler, and B. Shiv, "Self-expression cues in product rating distributions: When people prefer polarizing products," *Journal of Consumer Research*, 2017.

[153] B. Hooi, N. Shah, A. Beutel, S. Günnemann, L. Akoglu, M. Kumar, D. Makhija, and C. Faloutsos, "Birdnest: Bayesian inference for ratings-fraud detection," in *SDM*, SIAM, 2016.

[154] P. D. Turney, "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in *Proceedings of the 40th annual meeting on association for computational linguistics*, Association for Computational Linguistics, 2002.

[155] B. P. Carlin and T. A. Louis, *Bayes and empirical Bayes methods for data analysis*, vol. 17. Chapman & Hall/CRC Boca Raton, FL, 2000.

[156] W. Niu, J. Caverlee, and H. Lu, “Neural personalized ranking for image recommendation,” in *WSDM*, ACM, 2018.

[157] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical attention networks for document classification,” in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1480–1489, 2016.

[158] Y. He, Y. Zhang, W. Liu, and J. Caverlee, “Consistency-aware recommendation for user-generated item list continuation,” in *Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 250–258, 2020.

[159] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proceedings of the 26th International Conference on World Wide Web*, pp. 173–182, International World Wide Web Conferences Steering Committee, 2017.

[160] Y. Koren, “Factorization meets the neighborhood: a multifaceted collaborative filtering model,” in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 426–434, 2008.

[161] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “Bpr: Bayesian personalized ranking from implicit feedback,” in *UAI*, AUAI Press, 2009.

[162] N. F. Granados, R. J. Kauffman, H. Lai, and H.-C. Lin, “Decommoditization, resonance marketing, and information technology: An empirical study of air travel services amid channel conflict,” *Journal of Management Information Systems*, 2011.

[163] H. Rosa, *Resonanz: Eine Soziologie der Weltbeziehung*. Suhrkamp verlag, 2016.

[164] K. L. Keller, “Building strong brands in a modern marketing communications environment,” *Journal of marketing communications*, 2009.

[165] S. S. Shang, Y.-L. Wu, and Y.-J. Sie, “Generating consumer resonance for purchase intention on social network sites,” *Computers in Human Behavior*, 2017.

[166] E. F. McQuarrie and D. G. Mick, “On resonance: A critical pluralistic inquiry into advertising rhetoric,” *Journal of consumer research*, 1992.

[167] S. Kopp, “Social resonance and embodied coordination in face-to-face conversation with artificial interlocutors,” *Speech Communication*, 2010.

[168] S. Burnett, C. Sebastian, K. C. Kadosh, and S.-J. Blakemore, “The social brain in adolescence: evidence from functional magnetic resonance imaging and behavioural studies,” *Neuroscience & Biobehavioral Reviews*, 2011.

[169] J. Tang, H. Gao, and H. Liu, “mtrust: discerning multi-faceted trust in a connected world,” in *WSDM*, ACM, 2012.

[170] C. Chen, M. Zhang, Y. Liu, and S. Ma, “Social attentional memory network: Modeling aspect-and friend-level differences in recommendation,” in *WSDM*, 2019.

[171] M. McPherson, L. Smith-Lovin, and J. M. Cook, “Birds of a feather: Homophily in social networks,” *Annual review of sociology*, vol. 27, no. 1, 2001.

[172] J. You, R. Ying, and J. Leskovec, “Position-aware graph neural networks,” *arXiv preprint arXiv:1906.04817*, 2019.

[173] P. Pipergias Analytis, D. Barkoczi, P. Lorenz-Spreen, and S. Herzog, “The structure of social influence in recommender networks,” in *Web*, 2020.

[174] W. Fan, Y. Ma, D. Yin, J. Wang, J. Tang, and Q. Li, “Deep social collaborative filtering,” in *RecSys*, ACM, 2019.

[175] C. Cai, R. He, and J. McAuley, “Spmc: socially-aware personalized markov chains for sparse sequential recommendation,” *arXiv preprint arXiv:1708.04497*, 2017.

[176] G. Guo, J. Zhang, and N. Yorke-Smith, “Trustsvd: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings,” in *AAAI*, 2015.

[177] M. Jamali and M. Ester, “A matrix factorization technique with trust propagation for recommendation in social networks,” in *RecSys*, ACM, 2010.

[178] Z.-K. Zhang, C. Liu, Y.-C. Zhang, and T. Zhou, "Solving the cold-start problem in recommender systems with social tags," *EPL (Europhysics Letters)*, 2010.

[179] L. R. Flynn, R. E. Goldsmith, and J. K. Eastman, "Opinion leaders and opinion seekers: Two new measurement scales," *Journal of the academy of marketing science*, 1996.

[180] A. Shoham and A. Ruvio, "Opinion leaders and followers: A replication and extension," *Psychology & Marketing*, 2008.

[181] T. Stefanic, "Outsiders looking in: how everyday bloggers are gaining access to the elite fashion world," *Journal of Digital Research and Publishing*, 2010.

[182] M. Mendoza, "I blog. you buy. how bloggers are creating a new generation of product endorsers," *Digital Research & Publishing*, 2010.

[183] H. L. Schrank and D. Lois Gilmore, "Correlates of fashion leadership: Implications for fashion process theory," *Sociological Quarterly*, 1973.

[184] J. E. Workman and K. K. Johnson, "Fashion opinion leadership, fashion innovativeness, and need for variety," *Clothing and Textiles Research Journal*, 1993.

[185] P. C. Gibson, *Fashion and celebrity culture*. Berg, 2012.

[186] P. Thornley, "Examining the role of bloggers in the fashion industry: A public relations strategy for new designers," 2014.

[187] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: short-term attention/memory priority model for session-based recommendation," in *SIGKDD*, 2018.

[188] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, 2005.

[189] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *SIGIR*, ACM, 2017.

[190] N. J. Evans, J. Phua, J. Lim, and H. Jun, "Disclosing instagram influencer advertising: The effects of disclosure language on advertising recognition, attitudes, and behavioral intent," *Journal of Interactive Advertising*, 2017.

[191] Y. Koren and R. Bell, "Recommender systems handbook," *F. Ricci, L. Rokach, B. Shapira, & BP Kantor (Eds.)*, 2011.

[192] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *WWW*, ACM, 2015.

[193] X. Wang, X. He, L. Nie, and T.-S. Chua, "Item silk road: Recommending items from information domains to social users," in *SIGIR*, ACM, 2017.

[194] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *Proceedings of the ACM International Conference on Web Search and Data Mining*, pp. 153–162, 2016.

[195] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in *Proceedings of the World Wide Web Conference*, pp. 689–698, 2018.

[196] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in Neural Information Processing Systems*, pp. 3483–3491, 2015.

[197] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[198] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in β-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[199] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," *arXiv preprint arXiv:1612.00410*, 2016.

[200] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proceedings of the International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 355–364, 2017.

[201] L. Van Der Maaten, "Accelerating t-sne using tree-based algorithms," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3221–3245, 2014.

[202] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.

[203] W. J. Reed, "The pareto, zipf and other power laws," *Economics letters*, vol. 74, no. 1, pp. 15–19, 2001.

[204] G. E. Box and R. D. Meyer, "An analysis for unreplicated fractional factorials," *Technometrics*, vol. 28, no. 1, 1986.

[205] J. Ma, Z. Zhao, X. Yi, J. Yang, M. Chen, J. Tang, L. Hong, and E. H. Chi, "Off-policy learning in two-stage recommender systems," in *Proceedings of The Web Conference 2020*, pp. 463–473, 2020.

[206] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1930–1939, 2018.