UNIVERSITY OF CALIFORNIA, SAN DIEGO

Building Rich Recommender Systems by Modeling Visual, Sequential, and Relational Signals

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Ruining He

Committee in charge:

Professor Julian McAuley, Chair
Professor Kamalika Chaudhuri
Professor Sanjoy Dasgupta
Professor Virginia de Sa
Professor Lawrence K. Saul

2017

The dissertation of Ruining He is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2017

DEDICATION

To my family.

EPIGRAPH

In God we trust, all others bring data.

–William Edwards Deming

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

A special thanks to my advisor, Prof. Julian McAuley, for his unlimited encouragement, patience, and considerateness throughout these years. There are virtually no words that can express my gratitude to Julian; it would have been impossible for me to go this far without him (though I am still making baby steps toward becoming a decent researcher).

Thanks to my committee members, Prof. Kamalika Chaudhuri, Prof. Sanjoy Dasgupta, Prof. Virginia de Sa, and Prof. Lawrence K. Saul (in alphabetical order of the last name), who were more than generous with their precious time and expertise throughout this process.

I have been lucky to work with some excellent researchers in these years. Dr. Chen Fang and Dr. Zhaowen Wang from *Adobe Research* financially supported part of my research and provided important datasets and kind advice. My labmates and friends Wang-Cheng Kang, Mengting Wan, Jianmo Ni, Xinan Wang, etc. spent much time discussing research ideas with me and even inspired some of my works. Chunbin Lin, Jianguo Wang, and Chenwei Cai worked side by side with me on papers through late nights, which made it possible to meet some tight deadlines. I am very appreciative of the wisdoms, friendships, and helps from all of them.

Chapter 3, in part, contains material as it appears in the *AAAI Conference on Artificial Intelligence*, 2016 ("VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback," Ruining He and Julian McAuley), the *International Conference on World Wide Web*, 2016 ("Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering," Ruining He and Julian McAuley), the *ACM Conference on Recommender Systems*, 2016 ("Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation," Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley), the *IEEE International Conference on Data Min-*

ing series, 2016 ("Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation," Ruining He and Julian McAuley), the *IEEE International Conference on Data Mining series*, 2016 ("Learning Compatibility Across Categories for Heterogeneous Item Recommendation," Ruining He, Charles Packer, and Julian McAuley), and the material that has been submitted for publication in the *ACM Conference on Recommender Systems*, 2017 ("Translation-based Recommendation," Ruining He, Wang-Cheng Kang, and Julian McAuley). The dissertation author was the primary investigator and author of these papers.

Chapter 4, in part, contains material as it appears in the *AAAI Conference on Artificial Intelligence*, 2016 ("VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback," Ruining He and Julian McAuley), and the *International Conference on World Wide Web*, 2016 ("Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering," Ruining He and Julian McAuley). The dissertation author was the primary investigator and author of these papers.

Chapter 5, in part, contains material as it appears in the *ACM Conference on Recommender Systems*, 2016 ("Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation," Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley), and the *IEEE International Conference on Data Mining series*, 2016 ("Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation," Ruining He and Julian McAuley), as well as the material that has been submitted for publication in the *ACM Conference on Recommender Systems*, 2017 ("Translation-based Recommendation," Ruining He, Wang-Cheng Kang, and Julian McAuley). The dissertation author was the primary investigator and author of these papers.

Chapter 6, in part, contains material as it appears in the *IEEE International Conference on Data Mining series*, 2016 ("Learning Compatibility Across Categories for Heterogeneous Item Recommendation," Ruining He, Charles Packer, and Julian

McAuley). The dissertation author was the primary investigator and author of this paper.

In reference to IEEE copyrighted material which is used with permission in this dissertation, the IEEE does not endorse any of University of California, San Diego's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

VITA

2006 - 2010	B.E. in Software Engineering, Hebei University of Technology
2010 - 2013	M.S. in Computer Science, Tsinghua University
2013 - 2017	Ph.D. in Computer Science, University of California, San Diego

PUBLICATIONS

Chenwei Cai, **Ruining He**, and Julian McAuley. SPMC: Socially-Aware Personalized Markov Chains for Sparse Sequential Recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

Jianguo Wang, Chunbin Lin, **Ruining He**, Moojin Chae, Yannis Papakonstantinou, and Steven Swanson. MILC: Inverted List Compression in Memory. In *Proceedings of the VLDB Endowment (VLDB)*, 2017.

Ruining He and Julian McAuley. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *Proceedings of the IEEE International Conference on Data Mining series (ICDM)*, pages 191–200, 2016.

Ruining He, Charles Packer, and Julian McAuley. Learning Compatibility Across Categories for Heterogeneous Item Recommendation. In *Proceedings of the IEEE International Conference on Data Mining series (ICDM)*, pages 937–942, 2016.

Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley. Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 309–316, 2016.

Ruining He, Chunbin Lin, Jianguo Wang, and Julian McAuley. Sherlock: Sparse Hierarchical Embeddings for Visually-aware One-class Collaborative Filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3740–3746, 2016.

Ruining He and Julian McAuley. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 507–517, 2016.

Ruining He, Chunbin Lin, and Julian McAuley. Fashionista: A Fashion-aware Graphical System for Exploring Visually Similar Items. In *Proceedings of the International Conference on World Wide Web (WWW)*, demo paper, pages 199-202, 2016.

Ruining He and Julian McAuley. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 144–150, 2016.

Hui Guo, Zhenjiang Wang, Chenggang Wu, and **Ruining He**. EATBit: Effective automated test for binary translation with high code coverage. In *Proceedings of the Conference on Design, Automation & Test in Europe (DATE)*, pages 1–6, 2014.

Ruining He, Guoqiang Liang, Yuchun Ma, Yu Wang, and Jinian Bian. Unification of PR Region Floorplanning and Fine-grained Placement for Dynamic Partially Reconfigurable FPGAs. *Journal of Circuits, Systems and Computers (JCSC)*, 22(4), 2013.

Ruining He, Yuchun Ma, Kang Zhao, and Jinian Bian. ISBA: An Independent Set-Based Algorithm for Automated Partial Reconfiguration Module Generation. In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 500–507, 2012.

Ruining He, Guoqiang Liang, Yuchun Ma, Yu Wang, and Jinian Bian. PDPR: Fine-Grained Placement for Dynamic Partially Reconfigurable FPGAs. In *Proceedings of the International Symposium on Applied Reconfigurable Computing (ARC)*, pages 350–356, 2012.

ABSTRACT OF THE DISSERTATION

Building Rich Recommender Systems by Modeling Visual, Sequential, and Relational Signals

by

Ruining He

Doctor of Philosophy in Computer Science

University of California, San Diego, 2017

Professor Julian McAuley, Chair

Modeling and predicting user behavior in recommender systems are challenging as there are various types of signals at play. Traditional item recommendation algorithms mainly focus on modeling signals that indicate users' preferences, e.g. purchases, clicks, ratings. Despite their success, they typically ignore other signals that are also important for the recommendation task, e.g. *visual* signals for understanding users' finer-grained preferences toward appearances of items, *sequential* signals for exploiting the recommendation context, or *relational* signals encoding the complex and useful relationships amongst items.

Modeling these signals is non-trivial because it requires one to tackle not only 'standard' recommender systems challenges such as dealing with large, sparse, and long-tailed datasets, but also new challenges from dealing with the signals themselves, such as the need to model content in terms of its visual appearance, highly subjective sequential behavior, and the heterogeneous 'relatedness' amongst items.

In this dissertation, we tackle these challenges by building novel models that are scalable, accurate, and intuitive. Empirical results on a wide spectrum of large, real-world datasets indicate that our methods can not only quantitatively improve recommendation accuracy, but also enrich the system's ability to understand visual interactions (e.g. visual preferences and fashion trends), context (e.g. the impact of recent actions on the future), and complicated relationships amongst items (e.g. complementary or substitutable).

Chapter 1

Introduction

Recommender systems play a key role in surfacing items that match users' personal tastes amongst huge corpora of items. Such systems have been successful in domains from movies and music [7, 50, 62], to research articles, news, books, and many others [19, 31, 113]. Designing a successful recommender system is challenging; one needs to model the interactions between users and items, relationships amongst items themselves, and even the 'third-order' dynamics between users, items, and contexts, while carefully tackling the sheer scale of the system, sparsity, heterogeneity, long-tailed data distribution, and so on.

Traditional methods focused mostly on modeling user-item interactions, e.g. predicting the star-ratings a user would give to unseen items so as to recommend those with the highest ratings. Methodologically, there have been *Content-based Filtering* that seeks to recommend items that are most similar to what a user already consumed, with the similarity measured based on the 'content' (i.e., explicit features, metadata, etc.) of items. However, it is often difficult for *Content-based Filtering* to design appropriate features, recommend diverse items, and exploit quality judgments of other users in the system [59]. In contrast, *Collaborative Filtering* (CF) relies on exploiting large volumes of user feed-

back in a 'collaborative' manner, ignoring explicit features of items or users. It includes earlier neighborhood-based methods (e.g. [51, 103]), Bayesian methods (e.g. [11, 81]), topic models (e.g. [9, 37]), etc. and the state-of-the-art method, *Matrix Factorization* (MF). MF techniques are based on uncovering the underlying *decision factors* that have an influence on users' choices and have been ubiquitous in recent years, from the winning solutions to the *Netflix Prize Competition* [5, 7], to the basis of many state-of-the-art works (e.g. [90, 91, 97, 98, 129]).

In recent years, research focus are shifting, on the one hand, from predicting explicit feedback like star-rating to modeling *implicit* feedback such as purchases, clicks, likes, bookmarks, where only the 'positive' signals of users toward different items have been observed, though similar techniques like MF are still in use (e.g. [39, 90, 91, 97]). Comparatively, implicit feedback usually is easier to obtain and suffers less from sparsity issues.

On the other hand, researchers began to exploit various 'side signals' to build richer recommender systems. For example, one can make use of the content of items such as explicit features, metadata, or even textual reviews written by various users to supplement the CF model being used.[1] Such data are especially useful for recommending *cold-start* items for which we do not have enough collaborative data to make decisions. Likewise, in certain circumstances we have access to the demographics or social group information of users, which can be used to help predict the preferences of *cold-start* users. In addition, contextual signals like user's physical location and mood, time, or even temperature are useful for making context-aware recommendations. Much recent efforts have been made to exploit a particular type of contextual signals—what item(s) a user recently consumed (e.g. [27, 64, 98]). For instance, a user may be interested in cellphone accessories after the purchase of a cellphone. Exploiting such signals requires

[1] The resulting models are technically hybrid methods.

approaching a *sequential prediction* problem where the goal is to predict items that are most likely to be consumed by users in the future given their current sequences. The challenge main lies in how to appropriately model the third-order interactions between users and recent and future items. As we analyze later, existing methods are limited in their ability to tackle such challenges.

Modeling *relationships* between items is another key task of an online recommender system, in order to (e.g.) help users discover items that are functionally complementary or visually compatible. Recently, scalable methods have been developed that address this task by learning *similarity metrics* on top of the content of the items, e.g. [77, 118]. Such methods perform well when less amount of *heterogeneity* is involved, but the human notion of 'relatedness' goes beyond mere similarity: For two items to be compatible—whether jeans and a t-shirt, or a laptop and a charger—they should be similar in some ways, but systematically different in others.

1.1 Dissertation Contributions

We introduce novel methods to model *visual*, *sequential*, and *relational* signals in large-scale recommender systems. Our contributions are summarized as follows.

Chapter 4 — Modeling Visual Signals. We propose to exploit a new type of signals, images of items, for domains like *fashion* where users pay close attention to the visual appearances. Modeling such signals can help recommender systems not only understand users' fine-grained visual preferences and mitigate item *cold-start* issues, but also capture useful and interesting dynamics like fashion trends.

We start by building a novel model to estimate users' *visually-aware* personalized ranking functions based on their past feedback, before we further extend it to incorporate the subtle and non-linear *evolution* of fashion trends. To uncover the complex and

evolving visual factors that people consider when evaluating items, our final method combines high-level visual features extracted from a deep *Convolutional Neural Network*, users' past feedback, as well as evolving trends within the community. Experimentally we evaluate our methods on two large real-world datasets from *Amazon.com*, where we show them to outperform state-of-the-art personalized ranking measures, and also use to visualize the high-level fashion trends across the 11-year span of our dataset.

Chapter 5 — Modeling Sequential Signals. First, we build scalable methods to model the dynamics of a vibrant digital art community, *Behance.net*, consisting of tens of millions of interactions (i.e., clicks and appreciates) of users toward artistic items. Our main contributions here are to jointly model (1) sequential dynamics, in terms of how users prefer 'visually consistent' content within and across sessions; (2) 'social' dynamics, in terms of how users exhibit preferences toward both certain art styles and the artists themselves; and (3) rich content, especially in terms of its visual appearance.

In addition, we introduce two general-purpose algorithms with *unified* predictive components to capture the complex third-order interactions for sequential prediction. Existing methods typically decompose these high-order interactions into a combination of pairwise relationships, by way of which user preferences (user-item interactions) and sequential patterns (item-item interactions) are captured by *separate* components. Our algorithms are advantageous since personalization and sequential dynamics are inherently *correlated* and intertwine with one another. Empirically, our approaches significantly outperform the state of the art on a wide spectrum of large, real-world datasets.

Chapter 6 — Modeling Relational Signals. We propose a new method to learn complicated and heterogeneous relationships amongst items in product recommendation settings. To handle the heterogeneity of real-world data, our method relaxes the *metricity* assumption inherent in previous works and models multiple localized notions of 'relatedness,' so as to uncover ways in which related items should be systematically

similar, and systematically different. Quantitatively, we show that our system achieves state-of-the-art performance on large-scale compatibility prediction tasks, especially in cases where there is substantial heterogeneity between related items. Qualitatively, we demonstrate that richer notions of compatibility can be learned that go beyond similarity, and that our model can make effective recommendations of heterogeneous content.

1.2 Relevant Manuscripts

[1] **Ruining He** and Julian McAuley. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *AAAI*, 2016. **— Proposing to model visual signals.**

[2] **Ruining He** and Julian McAuley. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*, 2016. **— Modeling fashion trends.**

[3] **Ruining He**, Chunbin Lin, and Julian McAuley. Fashionista: A Fashion-aware Graphical System for Exploring Visually Similar Items. In *WWW*, demo paper, 2016. **— Designing a fashion-aware system for browsing clothing items.**

[4] **Ruining He**, Chunbin Lin, Jianguo Wang, and Julian McAuley. Sherlock: Sparse Hierarchical Embeddings for Visually-aware One-class Collaborative Filtering. In *IJCAI*, 2016. **— Proposing to use feature hierarchy to tackle heterogeneity of visual data.**

[5] **Ruining He**, Chen Fang, Zhaowen Wang, and Julian McAuley. Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation. In *RecSys*, 2016. **— Modeling sequential, social, and visual dynamics on *Behance.net*.**

[6] **Ruining He** and Julian McAuley. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *ICDM*, 2016. **— Introducing a new sequential model based on item similarity.**

[7] **Ruining He**, Charles Packer, and Julian McAuley. Learning Compatibility Across Categories for Heterogeneous Item Recommendation. In *ICDM*, 2016. **— Modeling heterogeneous relationships amongst items.**

[8] **Ruining He**, Wang-Cheng Kang, and Julian McAuley. Translation-based Recommendation. Under review at *RecSys*, 2017. **— Introducing a new sequential model based on translation embedding.**

[9] Chenwei Cai, **Ruining He**, and Julian McAuley. SPMC: Socially-Aware Personalized Markov Chains for Sparse Sequential Recommendation. In *IJCAI*, 2017. **— Jointly modeling sequential and social activities.**

Chapter 2

Background

There have been two main types of recommendation approaches in the literature. *Content-based Filtering* (CBF) seeks to recommend items[1] that are *similar* in terms of explicit features, i.e., content, to what a user 'liked' before. For instance, the system can surface a bunch of yellow tees to a user after she bought one such product (see Figure 2.1). Such methods have been widely used for recommending textual items including research papers [35, 69], news [8, 58], webpages [2, 94], emails [89], and so on. Leveraging the content of items enables CBF to handle unpopular or new items in the system; however, it is also limited in that (1) laborious (and often restrictive) feature engineering can be involved to extract useful features from the content, e.g. it is hard to represent the aesthetic quality of a webpage's layout [99]; (2) recommendations it makes suffer from overspecialization and low serendipity; and (3) it can not take advantage of the quality judgments of other users in the system [59].

In contrast to CBF, *Collaborative Filtering* (CF) methods make recommendations based on (only) the historical feedback 'profiles' of users and do not suffer from above issues. Figure 2.1 shows the difference between the two types of approaches.

[1] Objects (e.g. products, movies, music) to be recommended are referred to as *items* in this dissertation.

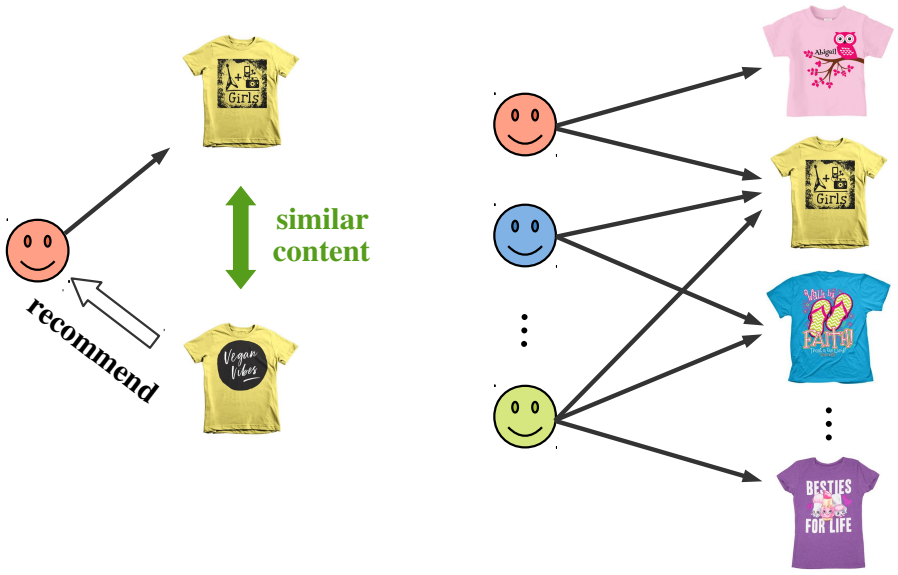


Figure 2.1: Demonstration of the difference between *Content-based Filtering* (left) and *Collaborative Filtering* (right). Black arrows represent positive feedback users expressed toward items.

Neighborhood-based (also known as 'memory-based' [11]) CF methods are amongst the earliest recommendation algorithms. According to the user-item feedback matrix[2], similarity between different users/items are computed by these methods, which can be done in the following two manners:

1. **User-oriented** methods identify each user's peers (or 'like-minded' users) in terms of the similarity of their feedback patterns such that a user's proclivity for an item can be predicted according to the previously observed behaviors of her peers. Representative works in this area include *GroupLens* [51], *Ringo* [103], and *Bellcore* video [36].

2. **Item-oriented** methods predict the preference of a user toward an item based on her previous feedback on those similar to the item being considered. To achieve this, similarity between two items are calculated based on the extent to which users in the system have 'rated' them similarly. When there are less items than users in

[2]i.e., representing the bipartite graph in Figure 2.1 with a matrix.

the system, item-oriented methods are more computationally efficient [21, 62].

More recent CF approaches directly *model* the underlying interactions between users and items, including Bayesian methods (e.g. [11, 81]), topic models (e.g. [9, 37]), clustering methods (e.g. [16, 116]), Restricted Boltzmann Machines [101], and so on. CF has been increasingly popular especially since the *Netflix Prize Competition* [7], when *Matrix Factorization* (MF) methods were proposed to model the interactions between users and items upon a set of *latent* rating dimensions. Due to their superior predictive accuracy, scalability, and flexibility, MF is now usually the first choice for implementing a CF recommender system [54].

MF relates users and items by uncovering a set of *latent* dimensions such that users have similar representations to items they have consumed or rated highly. The basic formulation of MF assumes the following model to predict the preference of a user u towards an item i:

$$\widehat{x}_{u,i} = \alpha + \beta_u + \beta_i + \langle \vec{\gamma}_u, \vec{\gamma}_i \rangle, \tag{2.1}$$

where α is global offset, β_u and β_i are bias terms associated with user/item, and $\vec{\gamma}_u$ and $\vec{\gamma}_i$ are K-dimensional vectors capturing latent factors of u and i respectively. The inner product estimates the extent to which the user's latent 'preferences' are aligned (or compatible) with the item's 'properties'.

MF is the *basis* of many state-of-the-art approaches for tasks from rating prediction (e.g. [5, 52, 53, 93]) to personalized ranking (e.g. [57, 91, 97, 129]) and sequential prediction (e.g. [98, 121]).

Chapter 3

Related Works

The most closely related branches of work to ours are those that (1) recommend items from implicit feedback, (2) extend CF with various side signals in the system, (3) model visual and clothing data, (3) deal with temporal and sequential dynamics, and (4) model relationships amongst items from the content data.

3.1 Item Recommendation from Implicit Feedback

To make personalized recommendations, recommender systems need to uncover preferences of users from their historical feedback, which comes in *explicit* forms like star-ratings or *implicit* forms such as purchase histories, bookmarks, browsing logs, search patterns, and even mouse activities [127]. Previously, recommendation algorithms mainly focused on modeling explicit feedback, i.e., addressing the rating prediction task and thus recommending items with the highest (predicted) ratings (e.g. [52–54, 93]). However, explicit feedback is usually rare and hard to obtain. Recent works have begun to exploit implicit feedback that is often much denser in the system. These works formulate recommendation as a personalized ranking task where the objective is to rank items each user is interested in as high as possible.

The concept of *One-Class Collaborative Filtering* (OCCF) was introduced by Pan *et al.* [90] to allow CF methods to effectively cope with scenarios where implicit feedback is observed. In the same work, they proposed to sample unknown feedback as negative instances and perform matrix factorization. This was further refined by Hu *et al.* in [39], where they assign varying confidence levels to different feedback and factorize the resulting weighted matrix. These two models can be classified as 'point-wise' methods. Following this thread, there are also subsequent works that build probabilistic models (e.g. [92, 110]) to address the same task.

Pairwise methods were later introduced by Rendle *et al.* in [97], where they proposed the framework of *Bayesian Personalized Ranking* (BPR) and empirically demonstrate that MF outperforms competitive baselines when trained with BPR (i.e., BPR-MF). It is a state-of-the-art method for the OCCF setting as well as the basis of a series of recent works, e.g. [57, 91, 98, 129]. In this dissertation, our visually- and fashion-aware recommendation methods (to be introduced in Chapter 4) are directly built on top of BPR-MF while maintaining its scalability.

3.2 Modeling Various Side Signals

There have been some recent works that model various side signals in the system in part to combat the *cold-start* issues suffered by CF methods. Here we briefly introduce *textual*, *social*, and *contextual* signals that have been popular in recent years.

3.2.1 Textual Signals

Textual data can entail rich features of items. This is especially true for *textual* items such as news, research articles, webpages, and so on. In such cases, topic models such as *Latent Dirichlet Allocation* (LDA) are often used to extract useful information

from the textual content in order to facilitate the recommendation task. For example, Wang *et al.* [119] proposed to combine the probabilistic version of MF model and LDA for scientific article recommendation. The resulting model leans a topic distribution from the content of each item, which is then used as a prior of the latent factors of the items. The entire probabilistic model is trained jointly to maximally optimize all parameters.

In recommender systems, another type of textual data is review text, which users use to express their experience, feelings, and opinions about the items they have consumed. Recent works in this area have tried integrating topic models (e.g. [63,74]) and phrase-level sentiment analysis tools (e.g. [29,70,123]) with CF techniques. For example, Zhang *et al.* [128] proposed an explicit factor model that makes use of triplets with the form $(\mathit{Feature}, \mathit{Opinion}, \mathit{Sentiment})$ extracted from user reviews to generate interpretable recommendations.

3.2.2 Social Signals

Social media are connecting and affecting human beings' lives; friendships on Google+ and Facebook, citing relations in research communities, following relations on *Twitter* and *Weibo*[1], etc. are encoding rich information about the social interactions between circles of families, friends, and co-workers. In the recommender systems literature, there has been a large body of work (e.g. [13, 33, 71, 72]) that models social networks for mitigating user *cold-start* issues. Most existing works in this area are based on MF and mainly follow three frameworks [112]: (1) Regularization-based approaches (e.g. [42, 73]) assume that users' preferences should be similar to those of their social circles and methodologically use regularizations to achieve this goal; (2) Joint factorization-based approaches (e.g. [72, 111]) try to find a factorization of the social network matrix so that the resulting user representations can also be useful for explaining

[1] http://www.weibo.com/

users' preferences; and (3) Ensemble-based approaches (e.g. ma2009learning) directly incorporate social network information into the user preference predictor.

3.2.3 Contextual Signals

In recent years, the importance of contextual data such as time, location, temperature, etc. is becoming recognized by the research community [99]. The simplest definition of contextual information is the *environment* a person is in. Many different types of contextual data have proven to be useful for helping generate context-aware recommendations. Here we summarize a few popular types of contextual data as follows.

Location. As mobile devices are gaining popularity in these years, location-aware recommender systems attracted much research attention (e.g. [27, 64]). Location information is often used to recommend items that are geographically close to the user, including restaurants, hotels, cinemas, theaters, tourist sites, and even nearby social activities [95].

Time. Time information like season, day of week, time of day, and holiday can also be useful for recommendation [34, 64]. For instance, one might want to recommend clothes for the current season. Likewise, it is also sensible to recommend items that fit the atmosphere of holidays, e.g. Christmas trees before Christmas.

Modal context. This includes user's mood, state, experience, capabilities, and so on. For example, the music being played can be indicative of the current mood of the user; relevant items like music, movies, and books can be recommended.

3.3 Modeling Visual and Clothing Data

Extensive previous research have emphasized the importance of images in e-commerce scenarios (e.g. [22, 30, 32]). In recent years, there is a growing interest in investigating the visual compatibility between different items. For example, [77] learns a distance metric to classify whether two given items are compatible or not. [118] fine-tunes a *Siamese Convolutional Neural Network* to learn a feature transformation from the image space to a latent space of metric distances. There are also related works that focus more on parsing/retrieving clothing images and capture some notion of 'style' [23, 41, 47, 65, 126]. For instance, the work of [105] can tell a user how to become more fashionable after taking a look at a photograph with the user in it. Another method [47] uses segmentation to detect clothing classes in the query image before it retrieves visually similar products from each of the detected classes.

These works are related to our visually-aware model to be introduced in Chapter 4, but they do not make use of the historical feedback of users to learn their personalized preferences, which is at the core of making *personalized* recommendations. Additionally, our approach takes into account various non-visual factors, which goes beyond the scope of the above methods.

On the other hand, visual features from deep *Convolutional Neural Networks* (CNN) have been widely successful in tasks such as object classification [100], semantic segmentation [67], human action recognition [43], aesthetic quality categorization [68], and photographic style annotations [48], among many others. Furthermore, an increasing number of transfer learning studies have demonstrated that CNNs pre-trained on one large dataset such as ImageNet [100] can provide useful visual features for other (presumably smaller) datasets. Such features can achieve state-of-the-art performance on these new datasets for tasks that may considerably differ from what the CNNs were originally

trained for [26, 96].

Considering the generic and descriptive ability of CNN features, we exploit them to build our fashion-aware recommendation system (in Chapter 4) and model the preferences toward artistic items (in Chapter 5).

3.4 Modeling Temporal Dynamics

Users' preferences are evolving over time, which necessitates the modeling of temporal dynamics in the system. There has been some work in the machine learning community that investigates the notion of *concept drift* in temporally evolving data. Such learning algorithms include decision trees [120], SVMs [49], instance-based learning [1], etc.; see the work of Tsymbal [115] for a comprehensive survey. According to [115], these methods can be classified into three basic approaches: instance selection, instance weighting, and ensemble learning.

There also have been CF models that take temporal dynamics into consideration, though not involved with modeling visual data. For example, to improve similarity-based CF, Ding *et al.* [24] propose a time weighting scheme to assign decaying weights to previously-rated items according to the time difference. More recent efforts are mostly based on MF, where the goal is to model and understand the historical *evolution* of users and items, e.g. Koren *et al.* achieved state-of-the-art rating prediction results on *Netflix* data, largely by exploiting temporal signals [53, 55].

In Chapter 4, we will not only build visually-aware recommendation models, but also handle the particular challenges from modeling temporal dynamics of people's preferences toward visual appearances, by way of which to build *fashion-aware* recommender systems. Our method, in some sense, fits into the instance selection camp, i.e., we use a time-window (or epoch) mechanism to highlight/favor appearance that are

widely accepted by the community in each window.

3.5 Modeling Sequential Dynamics

A user's recent feedback can be seen as a particular type of contextual signal and is useful for predicting their actions in the near future, e.g. the recent purchase of a laptop indicates the possibility of buying a laptop backpack next. To exploit such signals, one needs to model and predict the personalized sequential behavior of users.

Markov Chains (MCs) are a powerful tool to model stochastic transitions between different 'states.' In sequential recommendation domains, MCs have been studied by several earlier works, from investigating their strength at uncovering sequential patterns (e.g. [82, 131]), to directly modeling decision processes with Markov transitions [102]. Scalable sequential models usually rely on MCs to capture sequential patterns (e.g. [27, 98, 121]). Rendle *et al.* proposed to factorize the third-order 'cube' that represents the transitions amongst items made by users. The resulting model, *Factorized Personalized Markov Chains* (FPMC), can be seen as a combination of MF and MC (two *separate* components) and achieves good performance for next-basket recommendation.

There are also works that have adopted metric embeddings for the recommendation task, leading to better generalization ability. For example, Chen *et al.* introduced *Logistic Metric Embeddings* (LME) for music playlist generation [17], where the Markov transitions among different songs are encoded by the distances among them. Recently, Feng *et al.* further extended LME to model personalized sequential behavior and used pairwise ranking for predicting next points-of-interest [27]. However, like FPMC, this method still has to combine two *correlated* components in a separate manner, though using a hyperparameter to balance them.

On the other hand, Wang *et al.* recently introduced a hierarchical representation

model, which extends FPMC by applying aggregation operations (like max/average pooling) to model more complex interactions. Although the predictor can be seen as modeling the third-order interactions between user, previous and next items with a single component, the aggregation is hard to interpret and does not reap the benefits of using metric embeddings.

Our works on modeling sequential signals in Chapter 5 differ from the above in terms of the more complex signals and interactions being modeled (i.e., our visually-, socially-, and sequentially-aware model on *Behance.net*), as well as the types of methods (i.e., our two unified, interpretable, and accurate models based on similarity and translation respectively). We will also include above methods as baselines in our experiments.

3.6 Modeling Relationships Amongst Items

Identifying relationships among items is a fundamental part of many real-world recommender systems, e.g. to generate recommendations of the form 'people who viewed *x* also viewed *y*' on *Amazon.com*. Such methods may be based on CF, e.g. counting the overlap between users who have clicked on / bought both items, as in *Amazon*'s own solution [62]. Of more interest to us are systems that predict item-to-item relationships based on the *content* (e.g. images/text/metadata) of the items themselves. Various systems have been proposed to address specific settings, e.g. to identify relationships between members of 'urban tribes' [84], tweets [109], text [4,14], or music [108]. Several methods have also been used to model visual data [12,25,45,104,125], though typically in settings where the metric assumption is well-founded (e.g. similar image retrieval).

Our work on modeling relational signals in Chapter 6 follows recent examples that aim to model co-purchase and co-browsing relationships, using a recently introduced

dataset from *Amazon.com* [75, 77, 118]. While we extend and compare quantitatively against such work, our main contribution here is that we substantially relax the model assumptions to allow for more complex relationships than mere similarity between items.

Outside of the recommendation scenarios considered here, learning the features that describe relationships between objects is a vast topic. Typically, one is given some collection of putative relationships between items (i.e., a training set), and the goal is then to identify a (parameterized) function that can be tuned to fit these relationships, i.e., to assign observed relationships a higher likelihood or score than non-relationships. State-of-the-art methods identify hidden variables or factors that describe relationships among items [20, 114], e.g. by factorizing the matrix of links between items [78]. Again, the main contributions we hope to make over such approaches are (1) to relax the assumption of metricity, and (2) to allow for multiple notions of relatedness to compete and interact. While a few approaches have recently been proposed to learn non-metric relationships (e.g. [15]), we are unaware of any that allow for the scale of the data (thousands of features, millions of items and relationships) that we consider.

3.7 Acknowledgements

Chapter 3, in part, contains material as it appears in the *AAAI Conference on Artificial Intelligence*, 2016 ("VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback," Ruining He and Julian McAuley), the *International Conference on World Wide Web*, 2016 ("Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering," Ruining He and Julian McAuley), the *ACM Conference on Recommender Systems*, 2016 ("Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation," Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley), the *IEEE International Conference on Data Min-*

ing series, 2016 (“Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation,” Ruining He and Julian McAuley), the *IEEE International Conference on Data Mining series*, 2016 (“Learning Compatibility Across Categories for Heterogeneous Item Recommendation,” Ruining He, Charles Packer, and Julian McAuley), and the material that has been submitted for publication in the *ACM Conference on Recommender Systems*, 2017 (“Translation-based Recommendation,” Ruining He, Wang-Cheng Kang, and Julian McAuley). The dissertation author was the primary investigator and author of these papers.

Chapter 4

Modeling Visual Signals for Fashion-aware Recommendation

4.1 Introduction

In order to surface useful recommendations, it is crucial to be able to learn from user feedback in order to understand and capture the underlying decision factors that have an influence on users' choices. Here we are interested in applications in which *visual* decision factors are at play, such as clothing recommendation. In such settings, visual signals play a key role—naturally one would not buy a t-shirt from *Amazon.com* without being able to see a picture of the product, no matter what ratings or reviews the product had. Likewise then, when building a recommender system, we argue that this important source of information should be accounted for when modeling users' preferences.

In spite of their potential value, there are several issues that make visual decision factors particularly difficult to model. First is simply the complexity and subtlety of the factors involved; to extract any meaningful signal about the role of visual information in users' purchasing decisions shall require large corpora of products (and images)

and purchases. Second is the fact that visual preferences are highly personal, so we require a system that models and accounts for the preferences of and differences between individuals. Third is the fact that complex *temporal dynamics* are at play, since the features considered 'fashionable' change as time progresses. And finally, it is important to account for the considerable amount of *non*-visual factors that are also at play (such as durability and build quality); this latter point is particularly important when trying to interpret the role of visual decision factors, since we need to 'tease apart' the visual from the non-visual components of people's decisions.

Our main goal is to address these four challenges, i.e., to build visually-aware recommender systems that are scalable, personalized, temporally evolving, and interpretable. We see considerable value in solving such problems—in particular we shall be able to build better recommender systems that surface products that more closely match users' and communities' evolving interests. This is especially true for fashion recommendation, where product corpora are particularly 'long-tailed' as new items are continually introduced; in such *cold-start* settings we cannot rely on user feedback but need a rich model of the product's appearance in order to generate useful recommendations.

Beyond generating better recommendations, such a system has the potential to answer high-level questions about how visual features influence people's decisions, and more broadly how fashions have evolved over time (see Figure 4.1 for an example). For instance, we can answer queries such as "what are the key visual features or factors that people consider when evaluating products?" or "what are the main factors differentiating early 2000s vs. late 2000s fashions?", or even "at what point did Hawaiian shirts go out of style?". Thus our main goal is to learn from data how to model users' preferences toward products, and by doing so to make high-level statements about the temporal and visual dynamics at play.

Addressing our goals above requires new models to be developed. Previous

Figure 4.1: Fashion evolution and its impact on users. Above the timeline are the three most fashionable styles (i.e., groups) of women' sneakers during each year/epoch, revealed by our model; while below the timeline is a specific user's purchases (one in each year), which we model as being the result of a combination of fashion and personal factors.

models typically *ignore* visual signals in the system, which means we need to first build a (visually-aware) recommendation model that is able to handle the complexity of visual data. Next, in order to further make the model fashion-aware, the challenges of modeling both visual and temporal aspects simultaneously need to be tackled.

First, as we show quantitatively, the evolution of fashion trends can be abrupt and non-linear, so that existing temporal models such as timeSVD++ [53] are not immediately appropriate to address the challenge of capturing fashion dynamics. Moreover, multiple sources of temporal dynamics can be at play simultaneously, e.g. dynamics at the user or community level; the introduction of new products; or sales promotions that impact the choices people make in the short term. Thus we need a flexible temporal model that is capable of accounting for these varied effects; this is especially true if we want to interpret our findings, which requires that we 'tease-apart' or separate these visual vs. non-visual temporal dynamics.

Secondly, real-world datasets are often highly sparse, especially for clothing data

where new products are constantly emerging and being replaced over time; this means on the one hand that accounting for *content* (i.e., visual information) is critical for new items, but on the other hand that only a modest amount of parameters are affordable per item due to the huge item vocabulary involved. This drives us to avoid using localized structures as much as possible.

Thirdly, scalability can be a potential challenge since the new model needs to be built on top of a large corpus of product image data as well as a huge amount of user feedback. Note that the high dimensionality of the image data also exacerbates the above sparsity issue.

4.1.1 Our Contributions

Specifically, our main contributions include:

1. We propose a novel *visually-aware* model that incorporates visual signals into predictors of people's opinions while scaling to large datasets. Our model is designed for the classical *One-Class Collaborative Filtering* setting [90], where only the implicit feedback of users (i.e., purchase histories, bookmarks, browsing logs, mouse activities, etc.) are available.

2. We further build scalable *fashion-aware* models to capture temporal dynamics in order to make better recommendations. To cope with the non-linearity of fashion trends, we propose to automatically discover the important fashion 'epochs' each of which captures a separate set of prevailing visual decision factors at play.

3. Our method also models non-visual dimensions and non-visual temporal dynamics (in a lightweight manner), which not only helps to account for interference from non-visual sources, but also makes our method a fully-fledged recommendation

system. We develop efficient training procedures based on the *Bayesian Personalized Ranking* framework to learn the epoch segmentation and model parameters simultaneously.

4. Empirical results on two large real-world datasets, *Women's* and *Men's Clothing & Accessories* from *Amazon.com*, demonstrate that our models are able to outperform state-of-the-art methods significantly, both in *warm-* and *cold-start* settings.

5. We provide visualizations of our learned models and qualitatively demonstrate how fashion has shifted in recent years. We find that fashions evolve in complex, non-linear ways, which can not easily be captured by existing methods.

4.2 VBPR: Visually-aware Item Recommendation

In this section, we propose a novel item recommendation model, *Visual Bayesian Personalized Ranking* (VBPR), that is personalized, visually-aware, and scalable. Letting $\mathcal{U}$ and I denote the set of users and items respectively, each user u is associated with an item set I_u^+ about which u has expressed explicit positive feedback. In addition, a single image is available for each item $i \in I$. Using only the above data, our objective is to generate for each user u a personalized ranking of those items about which they haven not yet provided feedback (i.e., $I \setminus I_u^+$).

4.2.1 Modeling Visual Dimensions

Our preference predictor is built on top of *Matrix Factorization* (MF), which is state-of-the-art for rating prediction as well as modeling implicit feedback, whose basic formulation assumes Eq. (2.1) to predict the preference of a user u toward an item i (see Chapter 2).

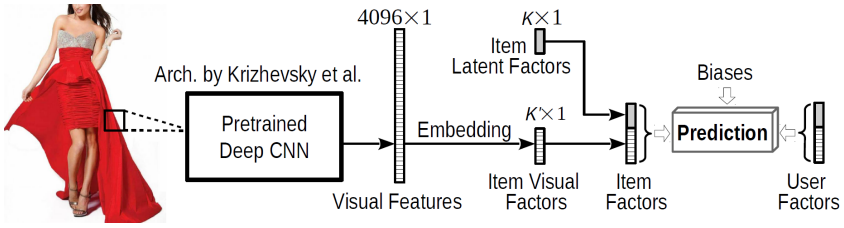


Figure 4.2: Diagram of the proposed visually-aware preference predictor. Rating dimensions consist of visual factors and latent (non-visual) factors. Inner products between users and item factors model the compatibility between users and items.

Although theoretically latent factors are able to uncover any relevant dimensions, one major problem it suffers from is the existence of 'cold' items in the system, about which there are too few associated observations to estimate their latent dimensions. Using explicit features can alleviate this problem by providing an auxiliary signal in such situations. In particular, we propose to partition rating dimensions into visual factors and latent (non-visual) factors, as shown in Figure 4.2. Our extended predictor takes the form

$$\widehat{x}_{u,i} = \alpha + \beta_u + \beta_i + \langle \vec{\gamma}_u, \vec{\gamma}_i \rangle + \langle \vec{\theta}_u, \vec{\theta}_i \rangle, \tag{4.1}$$

where α, β, and $\vec{\gamma}$ are as in Eq. (2.1). $\vec{\theta}_u$ and $\vec{\theta}_i$ are newly introduced K'-dimensional visual factors whose inner product models the visual interaction between u and i, i.e., the extent to which the user u is attracted to each of K' visual dimensions. Note that we still use K to represent the number of latent dimensions of our model.

One naïve way to implement the above model would be to directly use visual features $\vec{f}_i$ from (Deep) *Convolutional Neural Networks* (CNNs), e.g. AlexNet [56], of item i as $\vec{\theta}_i$ in the above equation. However, this would present issues due to the high dimensionality of the features in question, for example the features we use have 4,096 dimensions. Dimensionality reduction techniques like PCA pose a possible solution, with the potential downside that we would lose much of the expressive power of the original features to explain users' behavior. Instead, we propose to learn an embedding kernel

which linearly transforms such high-dimensional features into a much lower-dimensional (say 20 or so) 'visual rating' space:

$$\vec{\theta}_i = \mathbf{E}\vec{f}_i. \tag{4.2}$$

Here $\mathbf{E}$ is a $K' \times F$ matrix embedding CNN feature space (F-dimensional) into visual space (K'-dimensional). The numerical values of the projected dimensions can then be interpreted as the extent to which an item exhibits a particular visual rating facet. This embedding is efficient in the sense that all items share the same embedding matrix which significantly reduces the number of parameters to learn.

Next, we introduce a visual bias term $\vec{\beta}'$ whose inner product with $\vec{f}_i$ models users' overall opinion toward the visual appearance of a given item. In summary, our final prediction model is

$$\widehat{x}_{u,i} = \alpha + \beta_u + \beta_i + \langle \vec{\gamma}_u, \vec{\gamma}_i \rangle + \langle \vec{\theta}_u, \mathbf{E}\vec{f}_i \rangle + \langle \vec{\beta}', \vec{f}_i \rangle. \tag{4.3}$$

4.2.2 Learning the Model

Bayesian Personalized Ranking (BPR) is a state-of-the-art framework for learning parameterized models from implicit feedback [97]. It seeks to optimize the *pairwise* ranking $>_u$ between a positive item i and a negative item j for each user u; $i >_u j$ means i is ranked higher than j for u. Assuming independence of users and items, it maximizes the *maximum a posterior* estimator of the model parameters Θ:

$$\begin{aligned} \widehat{\Theta} &= \arg\max_{\Theta} \ \ln \prod_{u \in \mathcal{U}} \prod_{i \in I_u^+} \prod_{j \notin I_u^+} Prob(i >_u j | \Theta) \, Prob(\Theta) \\ &= \arg\max_{\Theta} \ \sum_{u \in \mathcal{U}} \sum_{i \in I_u^+} \sum_{j \notin I_u^+} \ln \sigma(\widehat{x}_{u,i} - \widehat{x}_{u,j}) - \Omega(\Theta), \end{aligned} \tag{4.4}$$

where the pairwise ranking between a positive (i) and a non-positive (j) item $Prob(i >_u j|\Theta)$ is estimated by a sigmoid function $\sigma(\widehat{x}_{u,i} - \widehat{x}_{u,j})$. $\Omega(\Theta)$ is an $\mathcal{L}_2$ regularizer.

The full set of parameters of VBPR is $\Theta = \{\beta_{i\in I}, \vec{\gamma}_{u\in\mathcal{U}}, \vec{\gamma}_{i\in I}, \vec{\theta}_{u\in\mathcal{U}}, \mathbf{E}, \vec{\beta}'\}$[1]. Using *Stochastic Gradient Ascent*, first a user u and a positive/non-positive pair (i, j) are sampled, and then parameters are updated as follows:

$$\Theta \leftarrow \Theta + \varepsilon \cdot (\sigma(\widehat{x}_{u,j} - \widehat{x}_{u,i}) \cdot \frac{\partial(\widehat{x}_{u,i} - \widehat{x}_{u,j})}{\partial\Theta} - \lambda_\Theta \cdot \Theta), \tag{4.5}$$

where ε represents the learning rate and λ_Θ is a regularization hyperparameter tuned with a held-out validation set.

Compared to BPR-MF (i.e., MF model learned with BPR), we have two sets of parameters to be updated: (1) the non-visual parameters, and (2) the newly-introduced visual parameters. Non-visual parameters can be updated in the same form as BPR-MF (therefore are suppressed for brevity), while visual parameters are updated according to:

$$\begin{aligned}
\vec{\theta}_u &\leftarrow \vec{\theta}_u + \varepsilon \cdot (\sigma(\widehat{x}_{u,j} - \widehat{x}_{u,i}) \cdot \mathbf{E}(\vec{f}_i - \vec{f}_j) - \lambda_\Theta \cdot \vec{\theta}_u), \\
\vec{\beta}' &\leftarrow \vec{\beta}' + \varepsilon \cdot (\sigma(\widehat{x}_{u,j} - \widehat{x}_{u,i}) \cdot (\vec{f}_i - \vec{f}_j) - \lambda_\beta \cdot \vec{\beta}'), \\
\mathbf{E} &\leftarrow \mathbf{E} + \varepsilon \cdot (\sigma(\widehat{x}_{u,j} - \widehat{x}_{u,i}) \cdot \vec{\theta}_u(\vec{f}_i - \vec{f}_j)^T - \lambda_\mathbf{E} \cdot \mathbf{E}).
\end{aligned} \tag{4.6}$$

Note that we sample users uniformly to optimize the ranking metrics across all users. All hyperparameters are tuned using a validation set as we describe in our experimental section later.

[1]Note that α and $\beta_{u\in\mathcal{U}}$ in the preference predictor are canceled out in Eq. (4.5), therefore are removed from the set of parameters.

4.2.3 Scalability Analysis

The efficiency of the underlying MF model makes our method similarly scalable. Specifically, BPR-MF requires $O(K)$ to finish updating the parameters for each sampled triple (u,i,j). In our case we need to update the visual parameters as well. In particular, updating $\vec{\theta}_u$ takes $O(K')$, $\vec{\beta}'$ takes $O(F)$, and $\mathbf{E}$ takes $O(K' \times F) = O(K')$, where F is the dimension of CNN features (fixed to 4096 in our case). Therefore the total time complexity of our model for updating each triple is $O(K + K')$ (i.e. $O(K) + O(K' \times F)$), i.e., linear in the number of dimensions. Note that visual feature vectors ($\vec{f}_i$) from CNNs are sparse, which significantly reduces the above worst-case running time.

4.3 FashionRec: Fashion-aware Item Recommendation

Our proposed method, VBPR, is good at capturing/uncovering visual dimensions as well as the extent to which users are attracted to each of them. Nevertheless, fashions, i.e., the visual elements of items that people are attracted to, evolve gradually over time. This presents challenges when modeling the visual dimensions of opinions because the same appearance may be favored during some time periods while disliked during others. Our goal here is to discover such trends both as a means of making better predictions, but also so that we can draw high-level conclusions about how fashions have evolved over the life of our dataset.

4.3.1 Modeling Fashion Evolution

We aim to enhance the 'static' VBPR model to capture the temporal dynamics of fashions. Considering the sparsity of real-world datasets, it is important to develop models that are expressive enough to capture the relevant dynamics but at the same time are tractable in terms of the number of parameters involved. The resulting model is

termed *Fashion-aware Recommendation Model*, or *FashionRec* in short.

Temporally-evolving Visual Factors

Here we identify three main fashion dynamics from which we can potentially benefit. We propose models to capture each of them with temporally-evolving visual factors; that is we model user/item visual factors as a function of time t, i.e., $\vec{\theta}_u(t)$ and $\vec{\theta}_i(t)$, with their inner products accounting for the temporal user-item visual interactions. This formulation is able to capture different kinds of fashion dynamics as described below.

Temporal Attractiveness Drift. The first notion of temporal dynamics is based on the observation that items gradually gain/lose 'attractiveness' in different visual dimensions as time goes by. To capture such a phenomenon, it is natural to extend our embedding matrix $\mathbf{E}$ to be time-dependent. More specifically, we model our embedding matrix at time t as

$$\mathbf{E}(t) = \mathbf{E} + \Delta_{\mathbf{E}}(t). \tag{4.7}$$

Here the underlying 'stationary' component of the model is captured by $\mathbf{E}$ while the time-dependent 'drifting' component is accounted for by $\Delta_{\mathbf{E}}(t)$. Then item i's visual factors at time t become

$$\vec{\theta}_i(t) = \mathbf{E}(t)\vec{f}_i. \tag{4.8}$$

In this way, we are modeling fashion evolution across entire communities with *global* low-rank structures. Such structures are expressive while introducing only a modest number of parameters.

Temporal Weighting Drift. As fashion evolves over time, it is likely that users *weigh* visual dimensions differently. For example, people may pay less attention to a dimension describing colorfulness as communities become more tolerant of bright colors.

Accordingly, we introduce a K'-dimensional temporal weighting vector $\vec{w}(t)$ to capture users' evolving emphasis on different visual dimensions, namely

$$\vec{\theta}_i(t) = \mathbf{E}\vec{f}_i \odot \vec{w}(t), \tag{4.9}$$

where $\odot$ is the Hadamard product.

Combining the above two dynamics, our formulation for item visual factors becomes

$$\vec{\theta}_i(t) = \underbrace{\mathbf{E}\vec{f}_i \odot \vec{w}(t)}_{\text{base}} + \underbrace{\Delta_{\mathbf{E}}(t)\vec{f}_i}_{\text{deviation}} \tag{4.10}$$

such that (when properly regularized) temporal variances are partly explained by the weighting scheme while the rest are absorbed by the expressive deviation term.

Note that compared to our basic model, so far we have only introduced *global* structures that are shared by all users. This achieves our goal of capturing temporal fashion trends that apply to the entire population. Next, we introduce 'local' dynamics, in order to model the drift of *personal* tastes over time.

Temporal Personal Drift. Apart from the above global temporal dynamics (i.e., fashion evolution), there also exist dynamics at the level of drifts in personal tastes over time. In other words, users' opinions are affected by 'outside' fashion trends as well as their own personal preferences, both of which can evolve gradually. Modeling this kind of drift can borrow ideas from existing works (e.g. timeSVD++ [53]) in order to extend our basic model with time-evolving user visual factors, i.e., by modeling $\vec{\theta}_u$ as a function of time. Here we give one example formulation (see [53] for more details) as follows:

$$\vec{\theta}_u(t) = \vec{\theta}_u + sign(t - t_u) \cdot |t - t_u|^{\kappa} \cdot \vec{\eta}_u, \tag{4.11}$$

which uses a simple parametric form to account for the deviation of user u at time t from

his/her mean feedback date t_u. This method uses two vectors $\vec{\theta}_u$ and $\vec{\eta}_u$ to model each user, with hyperparameter κ learned with a validation set (to be described later).

Temporally-evolving Visual Bias

In addition to temporally evolving factors $\vec{\theta}_i(t)$, we introduce a temporal visual bias term to account for that portion of the variance which is common to all factors. More precisely, we use a time-dependent F-dimensional vector $\vec{\beta}'(t)$ that adopts a formulation resembling that of Eq. (4.10):

$$\vec{\beta}'(t) = \vec{\beta}' \odot \vec{b}(t) + \Delta_{\vec{\beta}'}(t). \tag{4.12}$$

Then the visual bias of item i at time t is computed by taking the inner product $\langle \vec{\beta}'(t), \vec{f}_i \rangle$. The intention is to use low-rank structures to capture the changing 'overall' response to the appearance, so that the rest of the variance (i.e., per-user and per-dimension dynamics) are captured by properly regularized higher-rank structures, namely the inner product of $\vec{\theta}_u(t)$ and $\vec{\theta}_i(t)$. Experimentally, incorporating this term improves the performance to some degree, and is also useful for visualization.

Non-Visual Temporal Dynamics

Up to now, we have described how to extend our basic formulation to model visual dynamics. However, there also exist non-visual temporal dynamics in the datasets, such as sales, promotions, or the emergence of new products. Incorporating such dynamics into our model can not only improve predictive performance, but also helps with interpretability by allowing us to tease apart visual from non-visual decision factors. Here we want to distinguish as much as possible those factors that can be determined by the item's non-visual properties (such as its category) versus those that can only be

determined from the image itself.

To serve this purpose, we propose to incorporate the following two non-fashion dynamics in a lightweight manner, i.e., we guarantee that we are only introducing an affordable amount of additional parameters due to the sparsity of the real-world datasets we consider.

Per-Item Temporal Dynamics. The first dynamics to model are on the per-item level. As said before, various factors can cause an item to be purchased during some periods and not during others. Our choice is to replace the stationary item bias term β_i in VBPR with a temporal counterpart $\beta_i(t)$ [53].

Per-Subcategory Temporal Dynamics. Next, for datasets where the category tree is available (as is the case for the ones we consider), it is also possible to incorporate per-subcategory temporal dynamics. By accounting for category information explicitly as we do here, we discourage the visual component of our model from indirectly trying to predict the subcategory of the product, so that it may instead focus on subtler visual aspects. Letting C_i denote the subcategory the item i belongs to, we add a temporal subcategory bias term $\beta_{C_i}(t)$ to our formulation to account for the drifting of users' opinions toward a subcategory.

Gluing all above components together, we predict $\widehat{x}_{u,i}(t)$, the affinity score of user u and item i at time t, with Eq. (4.13).[2] Experimentally, we found that *global* temporal dynamics (i.e., fashion trends) are particularly useful at addressing personalized ranking tasks. However, modeling user terms, i.e., temporal personal drift, had relatively little effect in our datasets. The reasons are dataset-specific: (1) our datasets span a decade and most users only remain active during a relatively short period of time; (2) our datasets are highly sparse which means that the lack of per-user observations makes it difficult to

[2]Note that when computing personalized rankings for a single user u, α and β_u in Eq. (4.13) can be ignored.

fit the high-dimensional models required (see Eq. (4.11)). Therefore for our experiments we ultimately adopted stationary user visual factors $\vec{\theta}_u$ (note this way users' preferences are still affected by fashion trends).

$$\underbrace{\widehat{x}_{u,i}(t)}_{\substack{\text{preference of user } u \\ \text{toward item } i \text{ at time } t}} = \underbrace{\alpha + \beta_u + \underbrace{\beta_i(t) + \beta_{C_i}(t)}_{\text{temporal non-visual biases}} + \underbrace{\langle \overbrace{\vec{\beta}'(t)}^{\text{defined by Eq. (4.12)}}, \vec{f}_i \rangle}_{\text{temporal visual bias}}}_{\text{bias terms}} + \underbrace{\underbrace{\langle \vec{\gamma}_u, \vec{\gamma}_i \rangle}_{\text{non-visual interaction}} + \underbrace{\langle \overbrace{\vec{\theta}_u(t)}^{\substack{\text{defined by} \\ \text{Eq. (4.11)}}}, \overbrace{\vec{\theta}_i(t)}^{\substack{\text{defined by} \\ \text{Eq. (4.10)}}} \rangle}_{\text{temporal visual interaction}}}_{\text{user-item interactions}}. \tag{4.13}$$

Fashion Epoch Segmentation

So far we have described *what* temporal components to use in the formulation of our time-aware predictor; what remains to be seen is how to model the temporal term, i.e., how $\beta(t), \vec{\theta}(t)$ change as time progresses. One solution is to adopt a fixed schedule to describe the underlying evolution, e.g. to fit some parameterized function of (say) the raw timestamp, as is done by timeSVD++ [53]. However, fashion tends to evolve in a non-linear and somewhat abrupt manner, which goes beyond the expressive power of such methods (we experimentally tried parameterized functions like those in timeSVD++ but without success). Instead, a time-window design which uncovers fashion 'stages' or 'epochs' during the life span of the dataset proved preferable in our case. In other words, we want to *learn* a temporal partition of the timeline of our data into discrete segments during which different visual characteristics predominate to influence users' opinions.

To achieve our goal, we learn a partition of the timeline of our dataset, consisting

of N epochs, and to each epoch ep we attach a set of parameters

$$\Theta_{ep} = \{\Delta_{\mathbf{E}}(ep), \Delta_{\vec{\beta}'}(ep), \vec{w}(ep), \vec{b}(ep), \beta_i(ep), \beta_{C_i}(ep)\}.^3$$

Then we predict the preference of user u toward item i at epoch ep according to $\widehat{x}_{u,i}(ep(t))$, where the function $ep(\cdot)$ returns the epoch index of time t according to the segmentation. Note that while such a model could potentially capture seasonal effects (given fine-grained enough epochs), this is not our goal since we want to uncover long-term temporal drift; this can easily be achieved by tuning the number of epochs such that they tend to span multiple seasons (e.g. we obtained the best performance using 10 epochs in our 11 year dataset).

Finally, there are two components of the model to be estimated: (1) the model parameters $\Theta = \cup_{ep}\Theta_{ep} \cup \{\alpha, \beta_u, \vec{\gamma}_{u\in\mathcal{U}}, \vec{\gamma}_{i\in I}, \vec{\theta}_{u\in\mathcal{U}}, \mathbf{E}, \vec{\beta}'\}$, and (2) the fashion epochs themselves, i.e., a partition Λ of the timeline into segments with different visual rating behavior.

4.3.2 Learning the Model

With the above temporal preference predictor, our objective is for each user u to generate a personalized ranking of the items they haven not interacted with (i.e., $I \setminus I_u^+$) at time t. Here we extend *Bayesian Personalized Ranking* (BPR) to directly optimize the rankings produced by our model. First we derive the likelihood function we are trying to maximize, before we describe the coordinate ascent optimization procedure to learn the fashion epoch segmentation Λ as well as the model parameters Θ.

[3] i.e., discretized $\Delta_{\mathbf{E}}(t), \Delta_{\vec{\beta}'}(t), \vec{w}(t), \vec{b}(t), \beta_i(t), \beta_{C_i}(t)$ (respectively).

Log-Likelihood Maximization

Let $\mathcal{P}_u \subset I_u^+$ be the set of positive (i.e., observed) items for user u in the training set. Then according to BPR, a training tuple set $\mathcal{D}_S$ consists of triples of the form (u,i,j), where $i \in \mathcal{P}_u$ and $j \in I \setminus \mathcal{P}_u$. Given a triple $(u,i,j) \in \mathcal{D}_S$, recall that BPR models the probability that user u prefers item i to item j with $\sigma(\widehat{x}_{u,i} - \widehat{x}_{u,j})$, where σ is the sigmoid function, and learns the parameters by maximizing the regularized log-likelihood function as follows:

$$\sum_{(u,i,j)\in\mathcal{D}_S} \ln \sigma(\widehat{x}_{u,i} - \widehat{x}_{u,j}) - \frac{\lambda_\Theta}{2} \cdot \|\Theta\|_2^2.$$

Building on the above formulation, we want to add a temporal term $t_{u,i}$ encoding the time at which user u expressed positive feedback about $i \in \mathcal{P}_u$. The basic idea is that we want to rank the observed item i higher than all non-observed items at time $t_{u,i}$. More precisely, our training set $\mathcal{D}_{S^+}$ is comprised of quadruples of the form $(u,i,j,t_{u,i})$, where user u expressed positive feedback about item i at time $t_{u,i}$ with j being a non-observed item:

$$\mathcal{D}_{S^+} = \{(u,i,j,t_{u,i}) \mid u \in \mathcal{U} \wedge i \in \mathcal{P}_u \wedge j \in I \setminus \mathcal{P}_u\}. \tag{4.14}$$

To simplify the notion, we introduce the shorthand

$$\widehat{x}_{u,i,j}\big(ep(t_{u,i})\big) \;=\; \widehat{x}_{u,i}\big(ep(t_{u,i})\big) - \widehat{x}_{u,j}\big(ep(t_{u,i})\big),$$

where $ep(t)$ returns the index of the epoch that timestamp t falls into, and $\widehat{x}_{u,i}(ep)$ as well as $\widehat{x}_{u,j}(ep)$ are defined by Eq. (4.13). Then according to the BPR framework, our model is fitted by maximizing the regularized log-likelihood of the corpus (i.e., BPR-OPT in [97]):

$$\widehat{\Theta}, \widehat{\Lambda} \;=\; \arg\max_{\Theta,\Lambda} \sum_{(u,i,j,t_{u,i})\in\mathcal{D}_{S^+}} \ln \sigma\Big(\widehat{x}_{u,i,j}\big(ep(t_{u,i})\big)\Big) - \frac{\lambda_\Theta}{2} \cdot \|\Theta\|_2^2. \tag{4.15}$$

Again, note that there are two components to fit to maximize the above objective function, with one being the parameter set Θ and the other being the segmentation Λ of the timeline comprising N fashion epochs. Next we describe how to derive a coordinate-ascent-style optimization procedure to fit these two components.

Coordinate Ascent Fitting Procedure

We adopt an iterative optimization procedure which alternates between (1) fitting the model parameters Θ (given the segmented timeline Λ), and (2) segmenting the timeline Λ (given the current estimate of the model parameters Θ). This procedure resembles the one used in [76], though the problem setting and data are different.

Fitting the Model Parameters Θ. This step fixes the epoch segmentation Λ and adopts *Stochastic Gradient Ascent* (SGA) to optimize the regularized log-likelihood in Eq. (4.15). Given a randomly sampled training quadruple $(u,i,j,t_{u,i}) \in \mathcal{D}_{\mathcal{S}^+}$, the update rule of Θ is derived as

$$\Theta \leftarrow \Theta + \varepsilon \cdot \left(\sigma\Big(-\widehat{x}_{u,i,j}\big(ep(t_{u,i})\big)\Big) \cdot \frac{\partial \widehat{x}_{u,i,j}\big(ep(t_{u,i})\big)}{\partial \Theta} - \lambda_{\Theta} \cdot \Theta \right), \tag{4.16}$$

where ε is the learning rate. Sampling strategies may affect the performance of the model to some extent. In our implementation, we sample users uniformly to optimize the average AUC metric (to be discussed later).

Fitting the Fashion Epoch Segmentation Λ. Given the model parameters Θ, this step finds the optimal segmentation of the timeline to optimize the objective in Eq. (4.15). To achieve this goal, we first partition the timeline into N continuous bins of equal size. Then the fitting problem is solved with a dynamic programming procedure, which finds the segmentation such that rankings inside all bins are predicted most accurately. This is a canonical instance of a sequence segmentation problem [6],

which admits an $O(|\mathcal{D}_{\mathcal{S}^+}| \times N)$ solution in our case.

Finally, our parameters are randomly initialized between 0 and 1.0. The two fitting steps above are repeated until convergence, or until no further improvement is obtained on the validation set.

Scaling to Large Datasets

Fitting the epoch segmentation in a naïve way would be time-consuming due to the fact that the 'ranking quality' has to be evaluated by enumerating *all* non-observed items for each positive item. Fortunately, it turns out that for this step we can *approximate* the full log-likelihood by sampling a relatively small 'batch' of non-observed items for each positive user-item pair. Experimentally this proved to be effective and allows the dynamic programming procedure to find the optimal solution within around 3 minutes on our largest datasets.

4.3.3 Scalability Analysis

Building on top of BPR-MF and VBPR, FashionRec achieves the goal of scaling up to large real-world datasets. Here we analyze and compare its time complexity with those of BPR-MF and VBPR.

Fitting the Model Parameters

For this step, FashionRec adopts the sampling scheme of BPR-MF implemented in MyMediaLite [28], i.e., during each iteration we sample $|\mathcal{P}|$ training tuples to update the model parameters Θ, which we repeat for 100 iterations. For each training triple (u, i, j), BPR-MF requires $O(K)$ to update the parameters, while VBPR and FashionRec need to update the visual parameters as well. VBPR takes $O(K + K')$ in total to finish updating the parameters for each sampled training triple. Compared to VBPR, although

there are more visual parameters to describe multiple fashion epochs, FashionRec only needs to update the parameters associated with the epoch the timestamp $t_{u,i}$ falls into. This means that FashionRec exhibits the same time complexity as VBPR. Additionally, visual feature vectors ($\vec{f}_i$) from CNNs turn out to be very sparse, which can significantly reduce the above worst-case running time.

Fitting the Epoch Segmentation

In addition to the model parameters, FashionRec has to fit a fashion epoch segmentation term. Compared to the parameter fitting step, training the segmentation (i.e., the 'outer loop') is performed at comparatively much lower frequency and consumes much less time.

Generally speaking, FashionRec takes more iterations to converge than VBPR due to learning the temporal dynamics. Training on our *Women's Clothing* dataset takes around 20 hours (in which epoch fitting accounting for around 45 minutes in total) on our commodity desktop machine.

4.4 Experiments and Analyses

4.4.1 Datasets, Features, and Statistics

To evaluate the strength of our methods at modeling visual signals and capturing fashion dynamics, we are interested in real-world datasets that (1) are broad enough to capture the general tastes of the public, and (2) temporally span a long period so that there are discernibly different visual decision factors at play during different times.

Table 4.1: Dataset Statistics

Dataset	#users	#items	#feedback	Timespan
Women's Clothing	99,748	331,173	854,211	Mar. 2003 - Jul. 2014
Men's Clothing	34,212	100,654	260,352	Mar. 2003 - Jul. 2014
Total	**133,960**	**431,827**	**1,114,563**	**Mar. 2003 - Jul. 2014**

Datasets

The two datasets we use are from *Amazon.com*, as introduced in [77]. We consider two large categories that naturally encode fashion dynamics (within the U.S.) over the past decade, namely Women's and Men's Clothing & Accessories, each consisting of a comprehensive vocabulary of clothing items. The images available from this dataset are of high quality (typically centered on a white background) and have previously been shown to be effective for recommendation tasks (though different from the one we consider here).

We process each dataset by taking users' review histories as implicit feedback and extracting visual features $\vec{f}_i$ from one image of each item i. We discard users u who have performed fewer than 5 actions, i.e., for whom $|I_u^+| < 5$. Statistics of our datasets after this processing are shown in Table 4.1.

Visual Features

To extract a visual feature vector $\vec{f}_i$ for each item i in the above datasets, we employ a pre-trained *Convolutional Neural Network*, namely the Caffe reference model [44], which has previously been demonstrated to be useful at capturing the properties of images of this type [77]. This model implements the architecture proposed by [56] with 5 convolutional layers followed by 3 fully-connected layers and was pre-trained on 1.2 million ImageNet (ILSVRC2010) images. We obtain our $F = 4{,}096$ dimensional visual features by taking the output of the second fully-connected layer (i.e., FC7).

4.4.2 Evaluation Methodology

Given a user-item pair (u,i), the preference of u toward i is a function of time, i.e., the recommended item ranking for u is time-dependent. Therefore for a held-out triple $(u,i,t_{u,i})$, our evaluation consists of calculating how accurately item i is ranked for user u at time $t_{u,i}$.

Each of our datasets is split into training/validation/test sets by uniformly sampling for each user u from I_u^+ an item i (associated with a timestamp $t_{u,i}$) to be used for validation $\mathcal{V}_u$ and another for testing $\mathcal{T}_u$. The rest of the data $\mathcal{P}_u$ is used for training, i.e., $I_u^+ = \mathcal{P}_u \cup \mathcal{V}_u \cup \mathcal{T}_u$.

All methods are then evaluated on $\mathcal{T}_u$ with the widely used AUC (*Area Under the ROC curve*) measure:

$$AUC = \frac{1}{|\mathcal{U}|} \sum_u \frac{1}{|E(u)|} \sum_{(i,j)\in E(u)} \mathbb{I}\big(\widehat{x}_{u,i}(t_{u,i}) > \widehat{x}_{u,j}(t_{u,i})\big), \tag{4.17}$$

where the indicator function $\mathbb{I}(b)$ returns 1 *iff* b is *true*, and the evaluation goes through the pair set of each user u:

$$E(u) = \{(i,j) \mid i \in \mathcal{T}_u \wedge j \in I \setminus I_u^+\}. \tag{4.18}$$

For all methods we select the best hyperparameters using the validation set $\mathcal{V} = \cup_{u\in\mathcal{U}} \mathcal{V}_u$ and report the corresponding performance on the test set $\mathcal{T} = \cup_{u\in\mathcal{U}} \mathcal{T}_u$.

4.4.3 Comparison Methods

Matrix Factorization (MF) based methods are currently state-of-the-art for modeling implicit feedback datasets (e.g. [57, 91, 97]). Therefore we mainly compare against state-of-the-art MF methods in this area, including both point-wise and pairwise MF

models.

1. **Popularity (PopRec):** Items are ranked according to their popularity.

2. **Weighted Matrix Factorization (WR-MF):** A state-of-the-art point-wise MF model for implicit feedback proposed by [39]. It assigns confidence levels to different feedback instances and afterwards factorizes a corresponding weighted matrix.

3. **Bayesian Personalized Ranking (BPR-MF):** Introduced by [97], is a state-of-the-art method for personalized ranking on implicit feedback datasets. It uses standard MF (i.e., Eq. (2.1)) as the underlying predictor.

4. **BPR-TMF:** This model extends BPR-MF by making use of taxonomies and temporal dynamics; that is, it adds a temporal category bias as well as a temporal item bias in the standard MF predictor (using the techniques introduced in Section 4.3.1).

5. **Visual Bayesian Personalized Ranking (VBPR):** Our visually-aware method. It models raw visual signals for recommendation, but does not capture any temporal dynamics.

6. **Temporally-aware VBPR (FashionRec#):** This method models visual dimensions and captures visual temporal dynamics using the temporally-evolving visual factors & bias techniques we introduced earlier, but does *not* account for any *non*-visual dynamics.

7. **Fashion-aware Recommendation (FashionRec):** Compared to FashionRec#, this method further captures *non*-visual temporal dynamics (see Subsection 4.3.1) to improve predictive performance and help with interpretability, i.e., it makes use of all the terms in Eq. (4.13).

Table 4.2: Models. P: Personalized? V: Visually-aware? T: Temporally-aware? X: taXonomy-aware?

Property	PopRec	WR-MF	BPR-MF	BPR-TMF	VBPR	FashionRec#	FashionRec
P	✘	✔	✔	✔	✔	✔	✔
V	✘	✘	✘	✘	✔	✔	✔
T	✘	✘	✘	✔	✘	✔	✔
X	✘	✘	✘	✔	✘	✘	✔

Ultimately these methods are designed to evaluate (1) the performance of the current state-of-the-art non-visual methods (BPR-MF); (2) the value to be gained by using raw visual signals (VBPR); (3) the importance of visual temporal dynamics (FashionRec#); and (4) further performance enhancements from incorporating non-visual temporal dynamics (FashionRec). For clarity, we compare all above models in terms of whether they are 'personalized', 'visually-aware', 'temporally-aware', and 'taxonomy-aware', as shown in Table 4.2. All time-aware methods are trained with our proposed coordinate ascent procedure.

Most of our baselines are from MyMediaLite [28]. To make fair comparisons, our experiments always use the same total number of dimensions for all MF models. Additionally, all visually-aware MF models adopt a fifty-fifty split for visual vs. non-visual dimensions for simplicity. All our experiments were performed on a standard desktop machine with 4 physical cores and 32GB main memory.

4.4.4 Performance and Quantitative Results

Settings: Overall & Cold-Start

We evaluate all methods in two settings: 'Overall' and 'Cold-Start'. 'Overall' measures the overall ranking accuracy, including both *warm-start* and *cold-start* scenarios. However, it is desirable for a system to be able to recommend *cold-start* items effectively,

especially in the domains we consider where new items are constantly added to the system and the data is incredibly long-tailed. Therefore, we also evaluate our model in 'Cold-Start' settings.

To this end, our 'Overall' setting evaluates the average AUC on the full test set $\mathcal{T}$, while 'Cold-Start' is evaluated by only keeping the *cold-start* items in $\mathcal{T}$, i.e., items that had fewer than five positive feedback instances in the training set $\mathcal{P}$. It turns out that such *cold-start* items account for around 60% of the test set. This means that to achieve acceptable performance on sparse real-world datasets, one must be able to deal with their inherent *cold-start* nature.

Results & Analysis

Table 4.3 compares the performance of different models with the total number of dimensions set to 20. Due to the sparsity of our datasets, no MF-based model observed significant performance improvements when increasing the number of dimensions beyond this point. We make a few comparisons to better explain and understand our findings as follows:

1. Being a state-of-the-art method for personalized ranking from implicit feedback, BPR-MF beats the point-wise method WR-MF and the popularity-based baseline PopRec. PopRec is especially ineffective in *cold-start* settings since cold items are inherently 'unpopular'.

2. Further improvement over BPR-MF can be obtained by using taxonomy (i.e., category) information and by modeling temporal dynamics, as we see from the improvement of BPR-TMF over BPR-MF, i.e., on average 1.5% for 'Overall' and 4.3% for 'Cold-Start'.

3. More significant improvements over BPR-MF are obtained by making use of

Table 4.3: AUC on the test set $\mathcal{T}$ (higher is better). 'Overall' evaluates the overall accuracy, while 'Cold-Start' evaluates the ability to recommend/rank *cold-start* items. The best performance for each setting is boldfaced. All temporal methods (d, f, and g) use 10 epochs, though we also report the performance with 5 epochs (g5) for comparison.

ID	Method	Women's Clothing		Men's Clothing	
		Overall	*Cold-Start*	*Overall*	*Cold-Start*
(a)	PopRec	0.5726	0.3214	0.5772	0.3159
(b)	WR-MF	0.6441	0.5195	0.6228	0.5124
(c)	BPR-MF	0.7020	0.5281	0.7100	0.5512
(d)	BPR-TMF	0.7259	0.5749	0.7069	0.5498
(e)	VBPR	0.7834	0.6813	0.7841	0.6898
(f)	FashionRec#	0.8117	0.7325	0.8064	0.7314
(g5)	FashionRec	0.8148	0.7355	0.8074	0.7373
(g)	FashionRec	**0.8210**	**0.7469**	**0.8084**	**0.7459**
Imprv.	(e) vs. (c)	11.6%	29.0%	10.4%	25.1%
Imprv.	(g) vs. (d)	13.1%	29.9%	14.6%	35.7%
Imprv.	(g) vs. (e)	4.8%	9.6%	3.1%	8.1%

additional visual signals, as is done by VBPR. This leads to as high as an 11.6% improvement on Women's Clothing and 10.4% on Men's Clothing. These visual signals are especially helpful in *cold-start* settings where BPR-MF does not have enough observations to learn reliable item factors. In 'Cold-Start' settings, VBPR beats BPR-MF by as much as 29.0% on Women's Clothing and 25.1% on Men's Clothing.

4. Although VBPR can benefit from modeling visual signals, it is limited by its inability to capture temporal dynamics in the system. However in data such as ours (where feedback spans more than a decade) it is necessary to make use of a finer-grained model to capture evolving opinion dynamics. Here FashionRec# captures three types of 'fashion dynamics' (see Section 4.3) and yields significant improvements over VBPR.

5. FashionRec incorporates non-visual dynamics into FashionRec# to further account

for the variety of temporal factors at play. FashionRec outperforms VBPR by 4.8% on Women's Clothing and 3.1% on Men's Clothing for the 'Overall' setting, and even more for the 'Cold-Start' setting (9.6% and 8.1% respectively).

Additionally, all temporal models observed comparably larger improvements on Women's Clothing than Men's Clothing; presumably this is due to the size and variability of the dataset (see Table 4.1) or richer temporal dynamics exhibited by women's clothing.

Reproducibility

In all cases, regularization hyperparameters are tuned to perform the best on the validation set $\mathcal{V}$. The best regularization hyperparameter was $\lambda_\Theta = 100$ for WR-MF, and $\lambda_\Theta = 1$ for other MF-based methods. For visually-aware methods, the embedding matrix $\mathbf{E}$ and visual bias vector $\vec{\beta}'$ are *not* regularized as they introduce only a constant (and small) number of parameters to the model. In FashionRec# and FashionRec, $\Delta_{\mathbf{E}}(ep)$, $\vec{w}(ep)$ and $\vec{b}(ep)$ are regularized with regularization parameter 0.0001. Complete code for all our experiments and baselines is available at https://sites.google.com/a/eng.ucsd.edu/ruining-he/.

4.4.5 Visualization and Qualitative Analysis

Visual Dimensions

Our first visualization consists of demonstrating the visual dimensions uncovered by our method, i.e., what kind of characteristics people consider when evaluating items, as well as the evolution of their weights throughout the years.

A simple visualization of the learned visual dimensions is to find which items

exhibit maximal values for each dimension. That is, we select items according to

$$\arg\max_i \ \mathbf{E}_{k,:}\vec{f}_i,$$

for each row of the embedding matrix $\mathbf{E}$ in Eq. (4.10), corresponding to a visual dimension k. This tells us which items most exhibit, or are 'most representative' of a particular visual aspect discovered by the model.

Figure 4.3 shows such items for our FashionRec model. Two things are notable here. Firstly, the visual dimensions uncovered by our method seem to be meaningful, and capture combinations of color, shape and textural features (e.g. tees in the third row vary in shape but are similar in pattern). Secondly, human notions seem to be revealed by our method, e.g. semi-formal versus casual in rows 1 and 2, graphic designs versus patterns in rows 3 and 5 etc. It is this ability to discover visual characteristics that are correlated with human decision factors that explains the success of our model. Note that at first glance these dimensions may seem to pick up more than just fashion trends (like model poses or photo setups). Considering the size of the dataset we are experimenting on, this may be simply due to the amount of visually similar items available in the corpus. Examining longer ranked-lists for those dimensions helped assure us that they indeed focus on capturing characteristics of the clothes in the pictures.

In addition to the visual dimensions, our formulation of item visual factors (i.e., $\theta_i(t)$ in Eq. (4.10)) also models how the weight of each visual dimension has evolved during these years, with a weighting vector $\vec{w}(t)$. We also show such evolution in Figure 4.3. Due to the sparsity of the data in earlier years, we demonstrate the learned weights of the nine epochs from Aug. 2004 to Jul. 2014. As we can see from this figure, each visual dimension evolves roughly continuously as time progresses, although there do occasionally exist comparatively abrupt changes.

Figure 4.3: Demonstration of ten visual dimensions discovered by FashionRec on *Amazon* Women's Clothing data. Here we focus on a single subcategory, 'tees,' for a clear comparison. Each row shows the top ranked tees for a particular dimension k (i.e., $\arg\max_i \mathbf{E}_{k,:}\vec{f}_i$), as well as the evolution of the weight (i.e., $\vec{w}_k(t)$ in Eq. (4.10)) for this dimension across epochs (x-axis). Note that for many styles the weight evolves non-linearly.

Shifts in Fashion

Next we visualize the distribution of fashionable versus non-fashionable appearances as well as the subtle shifts as time progresses. This enables us to see not only how people weigh each specific dimension/aspect over time (as we did in Figure 4.3), but rather to comprehensively evaluate fashion as a whole by combining the dynamics

from all dimensions. To achieve this goal, we need a metric to qualitatively measure the overall visual popularity of a product image, which we term its 'visual score'.

The visual score of item i in epoch ep, $VisualScore(i, ep)$ is calculated by averaging the 'visual component' of the predictor (i.e., Eq. (4.13)) for all users, which naturally gives us the overall visual popularity of an item during epoch ep:

$$VisualScore(i, ep) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \langle \vec{\theta}_u, \vec{\theta}_i(ep) \rangle + \langle \vec{\beta}'(ep), \vec{f}_i \rangle. \tag{4.19}$$

Then we can visualize how fashion has shifted using a normalized visual score as the metric, i.e., by subtracting the average visual score of all items in each epoch.

By modeling the visual dimensions that best explain users' opinions, our method uncovers a low-dimensional 'visual space' where items that users evaluate similarly (i.e., with similar visual styles) are embedded to nearby positions. By definition, nearby items in the space will have similar visual scores. Then our visualization consists of demonstrating the visual space, as well as the time-dependent visual scores (i.e., popularity) attached to each of those items in the space.

After training our FashionRec model with 10 epochs on Women's Clothing, we take the base portion of the embedding, i.e., $\mathbf{E}\vec{f}_i$ in Eq. (4.10), to map all items into a visual space. The purpose is to help visualize items that have similar visual evaluation characteristics (or styles). Next, we use t-SNE [117] to embed a random sample of 30,000 items from the test set $\mathcal{T}$ into a 2-d space. Figure 4.4 shows the embedding we obtain. As expected, items from the same category tend to be mapped to nearby locations, since they share common features in terms of appearance. What is interesting and useful about the embedding is it can learn (1) a smooth transition across categories, and (2) 'sub-genres' in terms of appearance similarity. This is important since the available taxonomy is limited in its ability to differentiate between items *within* categories and in its ability to

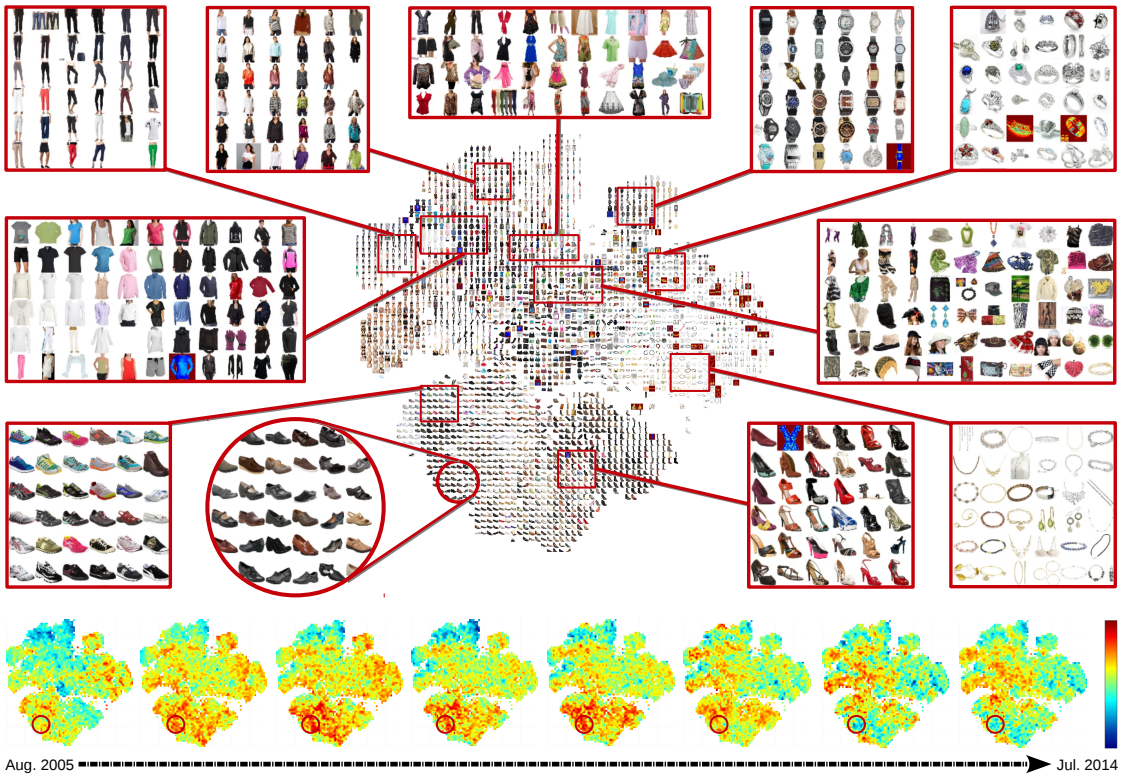


Figure 4.4: Demonstration of the 2-d t-SNE [117] embedding of the visual space learned by FashionRec on *Amazon* Women's Clothing. Images are 30,000 random samples from the test set $\mathcal{T}$. Each cell randomly selects one image to show in case of overlaps. At the bottom we also demonstrate the heat maps describing the normalized visual scores of these images over eight fashion epochs since Aug. 2005. Warmer means more popular, i.e., larger visual score. The circled area shows an example of a certain style which became popular but lost its appeal over time.

discover connections (especially visual ones) among items *across* categories.

To demonstrate how fashion has shifted over the life-span of the dataset, for each item i in the embedding we calculate its normalized visual score during every discovered epoch ep, which can then be used to build a 'heat map' demonstrating which items/styles were considered popular during each epoch.

These heat maps are also presented in Figure 4.4, from which we can observe the gradual evolution of users' tastes. We highlight a particular example where a certain style of shoe gradually gained popularity, which then diminished in recent years (see the circled area in Figure 4.4).

Figure 4.5: On the left we show query images each representing a resurgent style in men's fashion in the late 2000s. According to FashionRec trained on Men's Clothing, nearest neighbors of these images in our style space are shown in the middle and normalized visual scores (i.e., visual popularity) in the past decade on the right. We can see that our model captures such a resurgence especially since 2009.

4.4.6 Case Study: Men's Fashion in the 2000s

To help demonstrate that FashionRec has captured interpretable visual dynamics, we take a review of fashion trends in the 2000s as ground-truth and conduct a case study on men's clothing. The model used for this case study is FashionRec trained on *Amazon* Men's Clothing.

1950s and 1980s fashions resurfaced for men in the late 2000s.[4] Representative items include Ed Hardy T-shirts with low necklines, Hawaiian shirts, ski jackets, straight leg jeans, black leather jackets, windbreakers, and so forth. A simple evaluation then consists of visualizing the *visual popularity* of such items to see if there is any discernible resurgence around the late 2000s, as history tells us there ought to be.

To this end, we randomly selected four query items (from outside of the dataset we trained on, i.e., not from *Amazon.com*) representing each of Ed Hardy T-shirts, Hawaiian

[4]https://en.wikipedia.org/wiki/2000s_in_fashion, retrieved on Oct. 1, 2015.

shirts, black leather jackets, and ski jackets respectively. In Figure 4.5, first we visualize our visual space by retrieving nearest-neighbors for each of the query items (in the middle of the figure), and then compute the normalized visual score of each query image in each fashion epoch.

From Figure 4.5 we can see that, as expected, these styles are indeed predicted by our model to be gaining popularity especially since 2009, no matter how they performed prior to this period. This to some degree confirms that our proposed method can capture real-world fashion dynamics successfully.

4.5 Conclusion

Modeling visual appearance and its evolution is key to gaining a deeper understanding of users' preferences, especially in domains like fashion. In this chapter, we built scalable models on top of large-scale product images and user feedback to capture people's visual preferences and the temporal drifts of fashion. We found that deep CNN features are useful for modeling visual dimensions as well as the associated temporal dynamics. Low-rank structures learned on top of such features are efficient at capturing fashion dynamics and help our method significantly outperform state-of-the-art approaches. Visualization using our trained models helped demonstrate the non-linear characteristics of the evolution of different visual dimensions, as well as how fashion has shifted over the past decade.

4.6 Acknowledgements

Chapter 4, in part, contains material as it appears in the *AAAI Conference on Artificial Intelligence*, 2016 ("VBPR: Visual Bayesian Personalized Ranking from Im-

plicit Feedback," Ruining He and Julian McAuley), and the *International Conference on World Wide Web*, 2016 ("Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering," Ruining He and Julian McAuley). The dissertation author was the primary investigator and author of these papers.

Chapter 5

Modeling Sequential Signals for Context-aware Recommendation

5.1 Introduction

In this chapter, we are interested in modeling sequential signals in implicit feedback data (e.g. purchase histories) to predict personalized sequential behavior, e.g. the next product to purchase, movie to watch, or place to visit. To achieve this goal, it is essential to model the 'third-order' interactions between a user, the item(s) she recently consumed (i.e., the context), and the item to visit next. It is a challenging task as long- and short-term dynamics need to be combined carefully to account for both *personalization* and *sequential transition patterns*. Besides the complexity of the interactions themselves, the model also needs to handle the scale and inherent sparsity of real-world data.

The first part of this chapter introduces our new sequential model, Vista, for a popular social art website *Behance.net*, an online art community with millions of users and artistic images.[1] There are several aspects that make modeling this data interesting

[1] https://www.behance.net/

and challenging. First, on *Behance.net* users are both content creators and content evaluators, meaning that there is a need to model who appreciates what, as well as who appreciates whom ('social' dynamics). Not that here we focus on modeling the *ownership* signal, namely the interactions between a viewer and the creator of an item, which is different from traditional social network data (see Section 3.2). Second is the need to model sequential and visual dynamics, in terms of users' tendency to interact with consistent content within and across sessions, and their preferences toward individual art styles. And third is simply the scale and sparsity of the data involved; with data on the order of millions of users and items we must expend considerable effort developing methods that scale.

The second part of this chapter presents our two new models for the classical (i.e., general) sequential recommendation setting. *Factorized Personalized Markov Chains* (FPMC) models the third-order relationships between user u, previous item i, and next item j by the summation of two separate pairwise relationships: one for the compatibility between u and j, and another for the sequential continuity between item i and j, both captured by means of inner products [98]. Recently, there have been two lines of works that aim to improve FPMC. Personalized metric embedding methods replace the inner products in FPMC with Euclidean distances, where the metricity assumption—especially the triangle inequality—enables the model to generalize better [27, 83, 124]. However, these works still adopt the framework of modeling the user preference component and sequential continuity component *separately*, which may be disadvantageous as the two components are inherently correlated. Another line of work [121] leverages operations like average/max pooling to *aggregate* the representations of the user u and the previous item i, before their compatibility with the next item j is measured. These works partially address the issue of modeling the correlation of the two key components, though are hard to interpret and can not take advantage of the generalization ability of metric embeddings.

To tackle the challenges faced by existing methods, we first propose a new approach, Fossil, that fuses similarity-based methods with *Markov Chains* to naturally learn a personalized weighting scheme over the sequence of items to simultaneously characterize users in terms of both preferences and the strength of sequential behavior. The resulting model is *unified*, accurate and interpretable. To further benefit from metric embeddings, we propose another unified sequential method, TransRec, which embeds items as points in a 'transition space' and represents each user as a 'translation vector' in the same space. The third-order interactions mentioned earlier are captured by a personalized translation operation: the coordinates of previous item $\vec{\gamma}_i$, plus the translation vector $\vec{t}_u$ of user u determine (approximately) the coordinates of the next item $\vec{\gamma}_j$.

5.1.1 Our Contributions

Specifically, our main contributions include:

1. We build new models to capture the large-scale dynamics of artistic preferences. Methodologically we build upon techniques that model short-term, session-level dynamics, and our visually-aware method for modeling item content, to capture the notion of visual consistency between successive actions. Since users of *Behance.net* are themselves content creators, we capture both preferences toward particular art styles, as well as particular artists.

2. We further propose two new methods for the general sequential prediction setting, including one integrating similarity-based methods with *Markov Chains*, and another based on translation embeddings. Both methods naturally capture third-order interactions between users and items with unified components and have good interpretability.

3. Empirical results on a wide spectrum of large, real-world datasets demonstrate that our models are able to outperform state-of-the-art measures significantly. In addition, we visualize our learned models and analyze the sequential and personalized dynamics captured.

5.2 Vista: A Sequential Artistic Recommendation Model

In this section we build a new sequential method to model the large-scale dynamics of a vibrant online community, *Behance.net*, comprising tens of millions of interactions (i.e., clicks and appreciates) of users toward digital art.

5.2.1 Problem Formulation

Formally, let $\mathcal{U}$ represent the set of users and I the set of items (i.e., artistic images). Each item $i \in I$ is created by a certain user $o_i \in \mathcal{U}$ (or in some cases a few users $O_i \subseteq \mathcal{U}$). For each user u, a sequential action history (e.g. items clicked/appreciated by u) $\mathcal{S}^u$ is known: $\mathcal{S}^u = (\mathcal{S}^u_1, \mathcal{S}^u_2, \ldots, \mathcal{S}^u_{|\mathcal{S}^u|})$ where $\mathcal{S}^u_k \in I$. The action history of all users is denoted by $\mathcal{S} = \{\mathcal{S}^{u_1}, \mathcal{S}^{u_2}, \ldots, \mathcal{S}^{u_{|\mathcal{U}|}}\}$. Additionally, each item i is associated with an explicit feature vector $\vec{f_i}$, e.g. in our case visual features extracted from images. Given the above data, our task is to make personalized recommendations based on their predicted future actions.

5.2.2 The Vista Model

According to FPMC, the transition of user u from item i (at $t-1$) to item j (at t) can be explained from two aspects: (1) the interaction between user u and item j, which captures the *long-term* preferences of user u; and (2) the interaction between the previous item i and item j, which captures the temporary interest of user u (i.e., context). FPMC

is essentially the combination of MF and factorized MC:

$$Prob(j \mid u,i) \propto \underbrace{\langle \vec{p}_u, \vec{q}_j \rangle}_{\textit{Matrix Factorization}} + \underbrace{\langle \vec{m}_i, \vec{n}_j \rangle}_{\textit{Markov Chain}}, \tag{5.1}$$

where $Prob(j \mid u,i)$ denotes the probability of user u transitions from item i to j. User embeddings $\vec{p}_u$ and item embeddings $\vec{q}_j$, $\vec{m}_i$, $\vec{n}_j$ are parameters learned from the data.

Following this intuition, we extend FPMC and propose to factorize the personalized *Markov Chain* (MC) on *Behance.net* with the following formulation:

$$Prob(j \mid u,i) \propto \underbrace{\overbrace{\langle \vec{\phi}_u, \vec{\phi}_{o_j} \rangle}^{\substack{\text{artist} \\ \text{appreciation}}} + \overbrace{\langle \vec{\gamma}_u, \vec{\gamma}_j \rangle}^{\substack{\text{item} \\ \text{appreciation}}}}_{\text{personalization}} + w_u \cdot \underbrace{(\overbrace{\langle \vec{\phi}_{o_j}, \vec{\phi}_{o_i} \rangle}^{\text{artist similarity}} + \overbrace{\langle \vec{\psi}_j, \vec{\psi}_i \rangle}^{\text{item similarity}})}_{\text{sequential continuity}}, \tag{5.2}$$

where we use three latent spaces to capture the interactions between users and items, users and users, items and items respectively: (1) **Space Γ:** Vectors $\vec{\gamma}_u \in \Gamma$ and $\vec{\gamma}_i \in \Gamma$ are employed to capture user u's latent preferences and item i's latent properties respectively; (2) **Space Υ:** Each user u is associated with a vector $\vec{\phi}_u \in \Upsilon$ for measuring affinity/similarity with other users; and (3) **Space Ξ:** Each item i is associated with a vector $\vec{\psi}_i \in \Xi$ for measuring affinity/similarity with other items. Inner products of vectors from the three spaces measure the corresponding user-item, user-user, and item-item affinity/similarity respectively.

In Eq. (5.2), the interaction between u and j consists of two parts: how much user u appreciates—or is 'similar' to—the creator/owner o_j of j (term one), and how much u likes the specific item created by o_j (term two). Likewise, the similarity of i and j comprises two components as well: the similarity between their creators (term three) and the two items themselves (term four). w_u is a parameter to capture the statistical short-term consistency of the actions of the specific user u. Intuitively, a bigger value of

w_u will favor short-term interests, which means u tends to interact with items similar to those they just interacted with, instead of those that most fit their long-term preferences.

Modeling ownership data and user interactions can help address both user *cold-start* and item *cold-start* issues in real-world recommender systems:

1. For a cold user u who has very few actions in the system, the representation of u in space Υ (i.e., $\vec{\phi}_u$) can still be learned as long as some users have interacted with items *created by u*. When predicting the actions for u, $\vec{\phi}_u$ will help rank items created by similar users higher.

2. For a cold item i with very few interactions, we can still model it reasonably well as long as we can know about the artist who created it, i.e., $\vec{\phi}_{o_i}$. Fortunately, two sources of signals are available in the system for inferring $\vec{\phi}_{o_i}$: both actions made by o_i (active) and the interactions received by all other items created by o_i (passive). These signals are relatively abundant and unlikely to suffer from sparsity issues.

Handling Multiple Creators

In certain cases an item i may be collaboratively created by multiple users in the system $O_i \subseteq \mathcal{U}$. In such scenarios we take the *average* of their associated vectors as the corresponding vectors (i.e., $\vec{\phi}_{o_i}$ and $\vec{\phi}_{o_j}$) in Eq. (5.2). Note that this will not affect the training efficiency as an owner sampling scheme can be used, which we will detail later.

Modeling Higher-Order Markov Chains

The formulation in Eq. (5.2) only models a personalized MC of order one, with the assumption that a user's next action is independent of any historical actions if given the most recent one. However, this may suffer from noise and can not capture longer-term consistency (e.g. earlier clicks in a session). This inspires us to try modeling MC with

higher orders.

Due to the large scale and sparsity of real-world data, a light-weight yet expressive model is required. To this end, we model high-order personalized MCs by extending our first-order formulation with a personalized *decaying* scheme, as shown below:

$$Prob(j \mid u, \mathcal{S}^u_{t-1}, \mathcal{S}^u_{t-2}, \ldots, \mathcal{S}^u_{t-N}) \propto \underbrace{\langle \vec{\phi}_u, \vec{\phi}_{o_j} \rangle + \langle \vec{\gamma}_u, \vec{\gamma}_j \rangle}_{\text{interaction between } u \text{ and } j} + \sum_{k=1}^{N} \underbrace{w^k_u}_{\substack{\text{personalized} \\ \text{decaying weight}}} \cdot \underbrace{\big(\langle \vec{\phi}_{o_j}, \vec{\phi}_{o_{\mathcal{S}^u_{t-k}}} \rangle + \langle \vec{\Psi}_j, \vec{\Psi}_{\mathcal{S}^u_{t-k}} \rangle \big)}_{\substack{\text{interaction between } j \text{ and} \\ \text{the } k\text{-th previous item}}}. \quad (5.3)$$

The first part still measures the affinity between a user u and the next item j; while the second part computes the weighted sum of the similarities of item j to each of the previous items. There are two main intuitions behind the proposed formulation: (1) recent actions should be more correlated with future actions, which is why we employ a decaying term; and (2) different users may differ in behavior so that personalization should be taken into account. We model the personalized decay scheme as follows:

$$w^k_u = \frac{1}{e^{a_u \cdot (k-1) + b_u}}, \quad k = 1, 2, \ldots, N, \text{ for } u \in \mathcal{U}. \quad (5.4)$$

Note that the above formulation only introduces two additional parameters (i.e., a_u and b_u) for each user u, which allows us to model users' differing behavior in a light-weight manner.

Incorporating Content-Based Features

Up to now, our formulation only makes use of the collaborative data, without being aware of the underlying content of the items themselves.[2] Such a formulation may

[2]Without loss of generality, we take item features as an illustrative example. User features can be handled in a similar manner.

suffer from item *cold-start* issues where there are not enough historical observations to learn accurate representations of each item. Modeling the content of the items can provide auxiliary signals in *cold-start* settings and alleviate such issues.

Intuitively, content-based features of items should be informative of the two latent vectors: $\vec{\gamma}_i$ and $\vec{\psi}_i$. Given the explicit feature vector $\vec{f}_i$ of item i, we have demonstrated in Chapter 4 that embedding techniques are successful at incorporating (high-dimensional) content-based features into the CF framework. In particular, we augment the vector representations of items as follows:

$$\vec{\gamma}_i = \vec{\gamma}'_i + \mathbf{E}_\Gamma \vec{f}_i, \tag{5.5}$$

$$\vec{\psi}_i = \vec{\psi}'_i + \mathbf{E}_\Xi \vec{f}_i. \tag{5.6}$$

Here $\mathbf{E}_\Gamma$ and $\mathbf{E}_\Xi$ are two embedding matrices that project item i into space Γ and Ξ respectively. The new formulation for each vector in the corresponding space (e.g. $\vec{\gamma}_i$) now consists of two parts: the base part from the item content ($\mathbf{E}_\Gamma \vec{f}_i$), and the residue ($\vec{\gamma}'_i$). For *warm-start* items, the residue part is expressive and can represent the item accurately; for *cold-start* items, the residue part will be regularized (toward 0) and the base part will still be able to provide reasonably good approximations of the true representations. By substituting the augmented vector representations Eq. (5.5) and Eq. (5.6) into Eq. (5.3), the new formulation will be able to benefit from the content-based features.

Note that the two matrices are shared by all items which means that only a modest number of parameters are added to the model.

5.3 Fossil: A Similarity-based Sequential Model

In this section, we present our similarity-based model, *Factorized Sequential Prediction with Item Similarity Models* (Fossil), for tacking sequential recommendation at large scale.

The task we are addressing in this section (as well as the following section) are formulated as follows. Let $\mathcal{U}$ denote the set of users and I the set of items. Each user u is associated with a *sequence* of items (or 'actions') $\mathcal{S}_u$, e.g. items purchased by u, or places u has checked in: $\mathcal{S}^u = (\mathcal{S}^u_1, \mathcal{S}^u_2, \ldots, \mathcal{S}^u_{|\mathcal{S}^u|})$, where $\mathcal{S}^u_k \in I$. We use I^+_u to denote the *set* of items in $\mathcal{S}^u$ where the sequential signal is ignored. Using the above data exclusively, our objective is to predict the next action of each user and thus make recommendations accordingly.

5.3.1 Similarity-based Item Recommendation

There is a similarity-based method, called *Sparse Linear Methods* (SLIM), that has been developed for traditional item recommendation task by learning an item-to-item similarity matrix $\mathbf{S}$ from the user action history (e.g. purchase logs) [87]. It predicts the user-item affinity as follows:

$$\widehat{x}_{u,i} = \sum_{j \in I^+_u \setminus \{i\}} \mathbf{S}_{j,i}, \tag{5.7}$$

where I^+_u is the set of items u has interacted with. $\mathbf{S}_{j,i}$ is the element at the j-th row and i-th column of matrix $\mathbf{S}$, denoting the *similarity* of item j to item i. The underlying rationale it follows is that the more j is similar to those items already consumed/liked by u, the more likely j will be a preferable choice for u.

Without parameterizing each user explicitly, SLIM relaxes the low-rank assumption enforced on user representations and has achieved state-of-the-art recommendation accuracy (see [87] for details). The major challenge faced by SLIM comes from the large

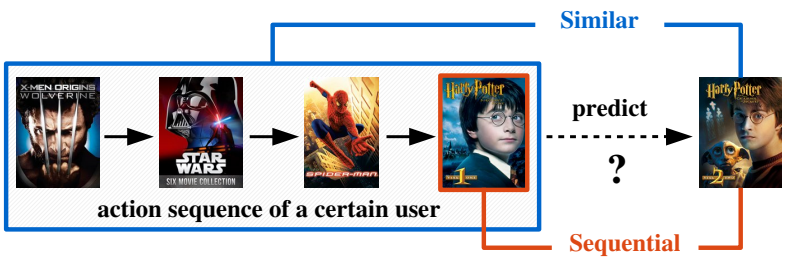


Figure 5.1: An example of how our method, Fossil, makes recommendations. *Harry Potter 2* is recommended to the user because it (1) is similar to content already watched (i.e., fantasy movies), and (2) frequently follows the recently-watched movie *Harry Potter 1*. The former is modeled with a similarity-based method and the latter *Markov Chains*.

amount of parameters ($|I| \times |I|$) to be estimated from the sparse user-item interactions. SLIM approaches this issue by exploring the *sparsity* characteristic of $\mathbf{S}$ using $\mathcal{L}_1$-norm regularization when inferring the parameters. Another direction is to capitalize on the *low-rank* potential of the similarity matrix by decomposing $\mathbf{S}$ into the product of two independent low-rank matrices [46]:

$$\mathbf{S} = \mathbf{P}\mathbf{Q}^T, \tag{5.8}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are both $|I| \times K$ matrices and $K \ll |I|$. This method is called *Factored Item Similarity Models* (FISM) and brings two benefits: (1) It significantly reduces the number of parameters and has been shown to generate state-of-the-art performance on a series of *sparse* datasets; and (2) Compared to SLIM, it is stronger at capturing the *transitive* property of item similarities.[3] When it comes to real-world datasets which are usually highly sparse, the above benefits contribute considerably to recommendation performance.

5.3.2 The Fossil Model

In contrast to FPMC that combines MF and MC, here we take another direction and investigate combining similarity-based methods and MC to approach the sequential prediction task (see Figure 5.1). In particular, we take FISM as our starting point, in light of its ability to handle the *sparsity* issues in real-world datasets. The basic form of our model is as follows:

$$Prob(j \mid u,i) \propto \overbrace{\sum_{j' \in I_u^+ \setminus \{j\}} \langle \vec{p}_{j'}, \vec{q}_j \rangle}^{\text{personalization}} + \underbrace{w_u}_{\substack{\text{personalized} \\ \text{weighting factor}}} \cdot \overbrace{\langle \vec{m}_i, \vec{n}_j \rangle}^{\text{sequential dynamics}}, \tag{5.9}$$

where each user is parameterized with only a single scalar w_u that controls the relative weights of the long- and short-term dynamics.

The above formulation parameterizes each item with four vectors, i.e., $\vec{p}_i$, $\vec{q}_i$, $\vec{m}_i$, and $\vec{n}_i$. Considering the limited number of parameters we can afford in the sparse datasets we are interested in, we reduce the four matrices to two by enforcing $\vec{p} = \vec{m}$ and $\vec{q} = \vec{n}$. This makes sense since ultimately sequentially-related items are also 'similar' to one another. Adding a bias term β_j and normalizing the long-term dynamics component, we arrive at a new formulation as follows:

$$Prob(j \mid u,i) \propto \beta_j + \langle \frac{1}{|I_u^+ \setminus \{j\}|^\alpha} \sum_{j' \in I_u^+ \setminus \{j\}} \vec{p}_{j'} + w_u \cdot \vec{p}_i, \vec{q}_j \rangle. \tag{5.10}$$

Note that in above equation the third-order interactions between u, i, and j are naturally modeled by a single component.

[3] For instance, if items a and b, b and c are two co-purchase pairs in the training data but (a,c) is not, SLIM will erroneously estimate the similarity of a and c to be 0.

Modeling Higher-order Markov Chains

Up to now we have used first-order MC to model short-term temporal dynamics. Next, we extend our formulation to consider high-order MC to capture smoothness across multiple time steps. Given the most recent N items user u has consumed $(\mathcal{S}^u_{t-1}, \mathcal{S}^u_{t-2}, \ldots, \mathcal{S}^u_{t-N})$, the new formulation predicts the probability of item j being the next item (at t) with an N^{th} order MC as shown in Eq. (5.11):

$$Prob(j \mid u, \mathcal{S}^u_{t-1}, \mathcal{S}^u_{t-2}, \ldots, \mathcal{S}^u_{t-N}) \propto \beta_j + \langle \overbrace{\frac{1}{|I_u^+ \setminus \{j\}|^\alpha} \sum_{j' \in I_u^+ \setminus \{j\}} \vec{p}_{j'}}^{\text{personalization}} + \underbrace{\sum_{k=1}^{N} w^u_k \cdot \vec{p}_{\mathcal{S}^u_{t-k}}}_{\text{sequential dynamics}}, \vec{q}_j \rangle. \tag{5.11}$$

Like in Eq. (5.3), each user is associated with a vector $(w^u_1, w^u_2, \ldots, w^u_N)$. The rationale behind this idea is that each of the previous N actions should contribute with different weights to the high-order smoothness.

5.4 TransRec: A Translation-based Sequential Model

In this section, we introduce a novel method called *Translation-based Recommendation* (TransRec) for large-scale sequential recommendation. The key idea behind TransRec is presented in Figure 5.2. The advantages of such an approach are three-fold: (1) TransRec naturally models third-order interactions with a *single* component; (2) As a metric embedding method, TransRec also enjoys the generalization benefits from the implicit metricity assumption; and (3) TransRec can easily handle large sequences (e.g. millions of instances) due to its simple form.

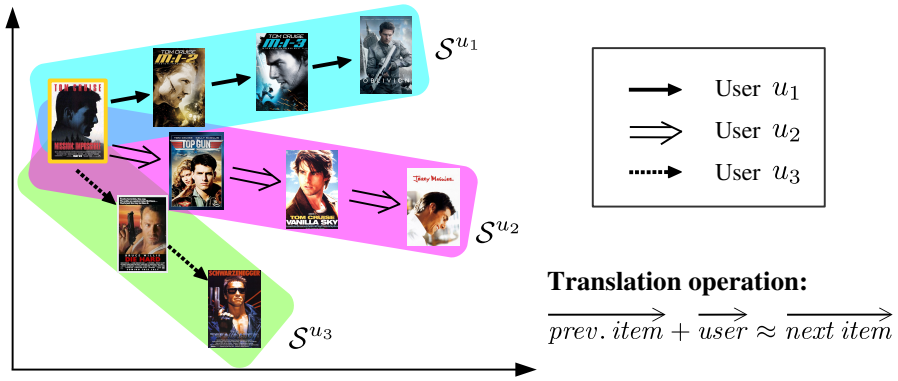


Figure 5.2: The high-level idea of TransRec model: Items are embedded into a 'transition space' where each user is modeled by a *translation* vector. The transition of a user from one item to another is captured by a user-specific translation operation. Here we demonstrate the historical sequences $\mathcal{S}^{u_1}$, $\mathcal{S}^{u_2}$, and $\mathcal{S}^{u_3}$ of three users. Given the same starting point, the movie *Mission: Impossible I*, u_1 went on to watch the whole series, u_2 continued to watch drama movies by Tom Cruise, and u_3 switched to similar action movies.

5.4.1 The TransRec Model

We learn a transition space Φ, where each item i is represented with a point/vector $\vec{\gamma}_i \in \Phi$. $\vec{\gamma}_i$ can be *latent*, or transformed from certain explicit features of item i, e.g. the output of a neural network. Without loss of generality, we take $\vec{\gamma}_i$ as latent vectors in this work. Recall that the historical sequence $\mathcal{S}^u$ of user u is a series of transitions u has made from one item to another. To model the *personalized* sequential behavior, we represent each user u with a *translation* vector $\vec{t}_u \in \Phi$ to capture u's inherent intent or 'long-term preferences' that influenced her to make these decisions. In particular, if u transitioned from item i to item j, what we want is

$$\vec{\gamma}_i + \vec{t}_u \approx \vec{\gamma}_j, \tag{5.12}$$

which means $\vec{\gamma}_j$ should be a nearest neighbor of $\vec{\gamma}_i + \vec{t}_u$ in Φ according to some distance metric $d(x,y)$, e.g. $\mathcal{L}_1$ distance.

Note that we are uncovering a metric space where (1) *neighborhood* captures

the notion of similarity, and (2) *translation* encapsulates various semantically complex transition relationships amongst items. In both cases, the inherent triangle inequality assumption plays an important role in helping the model to generalize well, as it does in canonical metric learning scenarios. For instance, if users tend to transition from item a to two items b and c, then TransRec will also put b close to c. This is a desirable property especially when data sparsity is a major concern. One plausible alternative is to use the inner product of $\vec{\gamma}_i + \vec{t}_u$ and $\vec{\gamma}_j$ to model their 'compatibility.' However, this way item b and c in our above example might be far from each other because inner products do not guarantee the triangle inequality condition.

Due to the sparsity of real-world datasets, it might not be affordable to learn separate translation vectors $\vec{t}_u$ for each user. Therefore we add another translation vector $\vec{t}$ to capture the 'global' transition dynamics across all users, and we let

$$\vec{t}_u = \vec{t} + \vec{t}'_u. \tag{5.13}$$

This way $\vec{t}'_u$ can be seen as an offset vector associated with user u. The advantages are (1) We can still learn personalized sequential behavior as users are being parameterized separately; and (2) vectors of *cold-start* users will be regularized toward 0 and we are essentially using $\vec{t}$—the 'average' behavior—to make predictions for these users.

Finally, the probability that a given user u transitions from the previous item i to the next item j is predicted by

$$\begin{aligned} &Prob(j \mid u, i) \;\propto\; \beta_j - d(\vec{\gamma}_i + \vec{t}_u, \vec{\gamma}_j), \\ &\text{subject to} \quad \gamma_i \in \Psi \subseteq \Phi, \quad \text{for } i \in I. \end{aligned} \tag{5.14}$$

Ψ is a subspace in Φ, e.g. a unit ball, a technique which has been shown to be helpful for mitigating 'curse of dimensionality' issues (e.g. [10, 61, 122]). In the above equation a

bias term β_j is added to capture overall item popularity.

5.4.2 Connection to Knowledge Graph Models

Although different from recommendation, there has been a large body of work in knowledge bases that focuses on modeling multiple, complex relationships between various entities, e.g. Turing was born in England ('was_born_in' is the relation between 'Turing' and 'England'). Recently, partially motivated by the findings made by *word2vec* [79], translation-based methods (e.g. [10,61,122]) have achieved state-of-the-art accuracy and scalability, in contrast to those achieved by traditional embedding methods relying on tensor decomposition or collective matrix factorization (e.g. [85, 86, 107]). Such methods embed entities as points and relations as *translation* vectors such that the relationship between two entities is captured by the corresponding translation operation. In the previous example, if we represent 'Turing', 'England', and 'was_born_in' with vectors $\overrightarrow{head}$, $\overrightarrow{tail}$, and $\overrightarrow{relation}$ respectively, then the following is desired: $\overrightarrow{head} + \overrightarrow{relation} \approx \overrightarrow{tail}$.

Our work is inspired by those findings, and we tackle the challenges from modeling large-scale, personalized, and complicated sequential data. This is the first work that explores this direction to the best of our knowledge.

5.5 Learning the Sequential Models

The ultimate goal of the sequential prediction task is to rank observed (or ground-truth) items as high as possible so that the recommender system can make plausible recommendations. This means it is natural to derive a *personalized* total order $>_{u,t}$ (at each step t) to minimize a ranking loss such as *sequential Bayesian Personalized Ranking* (S-BPR) [98]. Here $j >_{u,t} j'$ means that item j is ranked higher than item j' for user u at step t given the item sequence before t.

For each user u and for each time step t, S-BPR employs a sigmoid function $\sigma(\widehat{x}_{u,t,\mathcal{S}_t^u} - \widehat{x}_{u,t,j'})$ ($\widehat{x}_{u,t,\cdot}$ is a shorthand for the prediction in Eq. (5.3), Eq. (5.11), or Eq. (5.14)) to characterize the probability that ground-truth item $\mathcal{S}_t^u$ is ranked higher than a 'negative' item j' given the model parameters Θ, i.e., $Prob(\mathcal{S}_t^u >_{u,t} j'|\Theta)$. Assuming independence of users and time steps, model parameters Θ are inferred by optimizing the following *maximum a posteriori* estimation:

$$\begin{aligned} \widehat{\Theta} &= \arg\max_{\Theta} \ \ln \prod_{u\in\mathcal{U}} \prod_{t=2}^{|\mathcal{S}^u|} \prod_{j'\in I\setminus I_u^+} Prob(\mathcal{S}_t^u >_{u,t} j'|\Theta)\ Prob(\Theta) \\ &= \arg\max_{\Theta} \ \sum_{u\in\mathcal{U}} \sum_{t=2}^{|\mathcal{S}^u|} \sum_{j'\in I\setminus I_u^+} \ln \sigma(\widehat{x}_{u,t,\mathcal{S}_t^u} - \widehat{x}_{u,t,j'}) - \Omega(\Theta), \end{aligned} \tag{5.15}$$

where the *pairwise* ranking between the ground-truth and all negative items goes through all users and all time steps.[4]

We adopt *Stochastic Gradient Ascent* (SGA) to learn our sequential models. First, we uniformly sample a user u from $\mathcal{U}$ as well as a time step t from $\{2,3,\ldots,|\mathcal{S}^u|\}$. Next, a negative item j' is uniformly sampled, which forms a training triple (u,t,j'). Finally, the optimization procedure updates parameters in the following fashion:

$$\Theta \leftarrow \Theta + \varepsilon \cdot \Big(\sigma(\widehat{x}_{u,t,j'} - \widehat{x}_{u,t,\mathcal{S}_t^u}) \cdot \frac{\partial(\widehat{x}_{u,t,\mathcal{S}_t^u} - \widehat{x}_{u,t,j'})}{\partial\Theta} - \lambda_\Theta \cdot \Theta\Big), \tag{5.16}$$

where ε is the learning rate and λ_Θ is a regularization hyperparameter.

Due to the space constraint, TransRec needs one additional step after the updating of parameters; that is we re-normalize $\vec{\gamma}_i$, $\vec{\gamma}_j$, and $\vec{\gamma}_{j'}$ to be vectors in Ψ. For example, if we let Ψ be the unit $\mathcal{L}_2$-ball, then $\vec{\gamma} \leftarrow \vec{\gamma}/\max(1, \|\vec{\gamma}\|)$.

[4]Note that due to the 'additive' characteristic, our formulation can allow t to run from 2 (instead of $N+1$) to the last item in $\mathcal{S}^u$. $Prob(\Theta)$ is a Gaussian prior over the model parameters.

5.6 Experiments and Analyses

In this section, we first introduce baselines and evaluation methodology, and then we evaluate our sequential artistic model, Vista, on *Behance.net*, as well as the two general models, Fossil and TransRec, on a variety of real-world datasets.

5.6.1 Comparison Methods

1. **Popularity (PopRec):** This is a naïve baseline that ranks items according to their popularity, i.e., it recommends the most popular items to users and is not personalized.

2. **Bayesian Personalized Ranking (BPR-MF) [97]:** is a state-of-the-art item recommendation model which takes *Matrix Factorization* as the underlying predictor. It ignores the sequential signals in the system.

3. **Factored Item Similarity Models (FISM) [46]:** is a similarity-based algorithm for personalized item recommendation. Our Fossil model is built on top of it to tackle the sequential prediction task.

4. **Factorized Markov Chain (FMC):** captures the 'global' sequential dynamics by factorizing the item-to-item transition matrix (shared by all users), but does not capture personalized behavior.

5. **Factorized Personalized Markov Chain (FPMC) [98]:** uses a predictor that combines *Matrix Factorization* and factorized *Markov Chains* so that personalized Markov behavior can be captured (see Eq. (5.1)).

6. **Hierarchical Representation Model (HRM) [121]:** extends FPMC by using aggregation operations like pooling to blend users' preferences ($\vec{p}_u$) and their

recent activities ($\vec{q}_i$):

$$Prob(j \mid u,i) \propto \langle aggregation(\vec{p}_u, \vec{q}_i),\ \vec{q}_j \rangle. \tag{5.17}$$

We compare against HRM with both max pooling and average pooling, denoted by HRM_{max} and HRM_{avg} respectively.

7. **Personalized Ranking Metric Embedding (PRME) [27]:** models personalized Markov behavior by the summation of two Euclidean distances: one for measuring user-item affinity and another for sequential continuity. A hyperparameter (ζ) is used to balance the two components:

$$Prob(j \mid u,i) \propto -\left(\zeta \cdot \|\vec{p}_u - \vec{p}_j\|_2^2 + (1-\zeta) \cdot \|\vec{q}_i - \vec{q}_j\|_2^2\right). \tag{5.18}$$

8. **Visually, Socially, and Sequential-aware Artistic recommendation (Vista):** Our method for sequential artistic recommendation on *Behance.net* (see Section 5.2). *Markov Chains* of different orders will be experimented with and compared against other methods.

9. **Factorized Sequential Prediction with Item Similarity Models (FISM):** Our similarity-based method for classical sequential recommendation (see Section 5.3). *Markov Chains* of different orders will be experimented with and compared against other methods.

10. **Translation-based Recommendation (TransRec):** Our metric embedding method for classical sequential recommendation task (see Section 5.4). In experiments we try both $\mathcal{L}_1$ and squared $\mathcal{L}_2$ distance[5] for the predictor.

[5]Note that this can be seen as optimizing an $\mathcal{L}_2$ distance space, similar to the approach used by PRME [27].

Table 5.1: Models. P: Personalized? S: Sequentially-aware? M: Metric-based? U: Unified model of third-order interactions?

Property	PopRec	BPR-MF	FISM	FMC	FPMC	HRM	PRME	Vista	Fossil	TransRec
P	✘	✔	✔	✘	✔	✔	✔	✔	✔	✔
S	✘	✘	✘	✔	✔	✔	✔	✔	✔	✔
M	✘	✘	✘	✘	✘	✘	✔	✘	✘	✔
U	✘	✘	✘	✘	✘	✔	✘	✘	✔	✔

Table 5.1 examines the properties of different methods. The ultimate goal of the baselines is to demonstrate (1) the performance achieved by state-of-the-art sequentially-unaware item recommendation models (BPR-MF and FISM) and purely sequential models without modeling personalization (FMC); (2) the benefits of combining personalization and sequential dynamics in a 'linear' (FPMC) and non-linear way (HRM), or using metric embeddings (PRME); (3) the strength of our sequential artistic recommendation model by modeling ownership as well as content-based features (Vista); and (4) the effectiveness of using a single component to model the third-order interactions with similarity-based methods (FISM) and translations (TransRec).

5.6.2 Evaluation Methodology

For each dataset, we partition the historical sequence $\mathcal{S}^u$ for each user u into three parts: (1) the most recent one $\mathcal{S}^u_{|\mathcal{S}^u|}$ for test, (2) the second most recent one $\mathcal{S}^u_{|\mathcal{S}^u|-1}$ for validation, and (3) all the rest for training. Hyperparameters in all cases are tuned by grid search with the validation set. Finally, we report the performance of each method on the test set in terms of the following ranking metrics:

Area Under the ROC Curve (AUC):

$$AUC = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|I \setminus I_u^+|} \sum_{j' \in I \setminus I_u^+} \mathbb{I}(R_{u,g_u} < R_{u,j'}),$$

Hit Rate at position 50 (Hit@50):

$$Hit@50 = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{I}(R_{u,g_u} \leq 50),$$

where g_u is the 'ground-truth' item associated with user u at the most recent time step, $R_{u,i}$ is the rank of item i for user u (smaller is better), and $\mathbb{I}(b)$ is the indicator function that returns 1 if the argument b is *true*; 0 otherwise.

5.6.3 Experiments on Sequential Artistic Recommendation

Behance Dataset

Behance.net is a popular online community where millions of professional photographers, designers and artists share their work, organized as projects, with others. The contents of these projects vary significantly, ranging from photographs to animation, fashion, interior design, paintings, sculpting, calligraphy, cartoons, culinary arts, etc. Each project is created by a user or a few users and consists of a collection of images (as well as videos in certain cases). The creator/owner of the project selects the most representative image which the website presents to all users as the cover image. On the website, users browse through large numbers of cover images, click through attractive projects, and 'appreciate' those they like.

From *Behance.net* we collected two large corpora of timestamped user actions: (1) *clicks* comprising 381,376 users, 972,181 items (i.e., projects), and 48,118,748 clicks; and (2) *appreciates* consisting of 373,771 users, 982,002 items, and 11,807,103 appreciates. 52.7% users have created their own projects, and 2.3% items are created by multiple users. In our experiments, appreciates and clicks are used as two *separate* datasets to evaluate the efficacy of all methods on different types of feedback.

For each item, we extract a 4,096-dimensional feature vector from its cover image

with a pre-trained VGG neural network [106] as the content-based features $\vec{f}_i$. Such features have been shown to generate state-of-the-art results on visually-aware item recommendation tasks (see Chapter 4).

Speeding Up

Training with large volumes of content-based features can be time-consuming due to the need to update the embedding matrices. To speed up the training procedure, we make the following two observations and employ two modifications accordingly.

Sampling. Matrices $\mathbf{E}_\Gamma$ and $\mathbf{E}_\Psi$ are global parameters and only account for a tiny fraction of the parameter set (e.g. 0.29% for *Behance.net*). This means that less training data is needed to accurately estimate $\mathbf{E}_\Gamma$ and $\mathbf{E}_\Psi$. In other words, they are updated more often than needed if we update them for each training instance. As such, we lower their updating frequency by flipping a biased coin for each training triple to decide whether or not to update $\mathbf{E}_\Gamma$ and $\mathbf{E}_\Psi$. Likewise, we can also sample a single owner and only update their associated parameters in multiple-owner cases due to the rich user interactions available.

Asynchronous SGA. Notably, only a tiny fraction of parameters will be updated for each training triple (u, t, j), i.e., representations of u and a few *relevant* items, and they are unlikely to overlap.[6] It has been pointed out that in such cases lock-free parallelization of SGA could be employed to achieve fast convergence [88].

Experimentally, this naïve sampling and asynchronous SGA procedure can help finish training on huge datasets within reasonable time on commodity machines without losing prediction accuracy.

[6]The sampling scheme also helps reduce collisions when updating the embedding matrices.

Performance and Analysis

We compare all methods in terms of overall accuracy (denoted by 'Overall') in terms of AUC and Hit@50. Overall accuracy is evaluated with the full test set as introduced in Section 5.6.2. In addition, we also decompose the full test set into *subsets* according to whether the previous item (at $t-1$) of the action being tested (at t) is created by a different artist (i.e., artist transition), or whether the action is the start of a new session[7] (i.e., session transition). This gives us four settings: 'Artist Trans.' vs. 'Same Artist,' and 'Session Trans.' vs. 'Same Session' to thoroughly evaluate the ability of all models under various transitioning circumstances.

For all methods, we use 10 dimensions for all representations, e.g. Vista uses 10 dimensions for $\vec{\gamma}_u$, $\vec{\phi}_u$, $\vec{\gamma}_i$, and $\vec{\psi}_i$. Using additional dimensions yielded marginal performance improvement for all methods. Results on the two datasets are shown in Tables 5.2 and 5.3. We make a few comparisons and analyze our findings as follows:

BPR-MF vs. FMC vs. FPMC. Ultimately, BPR-MF and FMC focus on modeling user preferences and sequential dynamics respectively. BPR-MF ranks items according to what the given user likes from a long-term perspective, which makes it relatively strong when a user's action differs significantly from the previous one ('Artist Trans.' and 'Session Trans.'). In contrast, FMC ranks items based on the transition matrix, i.e., 'similarity' of the next item to the previous item. Such short-term awareness makes FMC strong in cases where action consistency is maximally demonstrated, i.e., 'Same Artist' and 'Same Session.' FPMC is inherently a combination of BPR-MF and FMC, which makes it the strongest among the three, though it is not necessarily the best in all settings.

Vista vs. Other Models and Cold-Start. By modeling ownership signals and

[7] Since no session metadata is available, sessions are obtained by temporally partitioning each user's clicks/appreciates with gaps larger than 1 hour.

Table 5.2: Accuracy for next-**click** prediction on *Behance.net* (higher is better). 'Overall' evaluates the overall accuracy, and the rest are more detailed settings for evaluation, i.e., whether it is a transition across different artists or sessions. The best performance for each setting is boldfaced. We test different orders of the *Markov Chain* (i.e., 1, 2, and 3).

Setting	Metric	PopRec	BPR-MF	FMC	FPMC	HRM_{max}	HRM_{avg}	PRME	Vista-1	Vista-2	Vista-3	% Imprv.
Overall	*AUC*	0.8282	0.9024	0.8840	0.9076	0.9056	0.9119	0.9130	0.9329	**0.9356**	0.9355	2.48%
	Hit@50	1.25%	2.10%	2.73%	5.32%	4.02%	4.24%	6.20%	6.96%	**8.28%**	7.47%	33.55%
Artist Trans.	*AUC*	0.8470	0.9061	0.8709	0.9019	0.8976	0.8983	0.9033	0.9172	0.9217	**0.9233**	1.90%
	Hit@50	1.51%	2.36%	2.84%	4.40%	3.42%	3.70%	4.34%	3.97%	**5.08%**	5.05%	15.45%
Same Artist	*AUC*	0.7677	0.8903	0.9261	0.9259	0.9312	0.9558	0.9440	**0.9834**	0.9805	0.9746	2.89%
	Hit@50	0.43%	1.25%	2.39%	8.27%	5.96%	5.96%	12.17%	16.55%	**18.56%**	15.27%	52.51%
Session Trans.	*AUC*	0.7876	0.8726	0.8243	0.8658	0.8602	0.8600	0.8690	0.8992	0.8998	**0.9013**	3.29%
	Hit@50	1.29%	1.90%	1.84%	2.54%	1.97%	2.15%	2.45%	3.48%	**4.04%**	3.90%	59.05%
Same Session	*AUC*	0.8545	0.9217	0.9226	0.9346	0.9350	0.9455	0.9414	0.9548	**0.9588**	0.9576	1.41%
	Hit@50	1.23%	2.23%	3.31%	7.12%	5.34%	5.59%	8.63%	9.21%	**11.03%**	9.78%	27.81%

Table 5.3: Accuracy for next-**appreciate** prediction on *Behance.net* (higher is better). 'Overall' evaluates the overall accuracy, and the rest are more detailed settings for evaluation, i.e., whether it is a transition across different artists or sessions. The best performance for each setting is boldfaced. We test different orders of the *Markov Chain* (i.e., 1, 2, and 3).

Setting	Metric	PopRec	BPR-MF	FMC	FPMC	HRM_{max}	HRM_{avg}	PRME	Vista-1	Vista-2	Vista-3	% Imprv.
Overall	*AUC*	0.8116	0.8776	0.8559	0.8727	0.8786	0.8844	0.8714	**0.9255**	0.9226	0.9207	4.65%
	Hit@50	2.41%	2.15%	1.12%	4.23%	3.07%	3.28%	2.91%	6.54%	**7.16%**	5.45%	69.27%
Artist Trans.	*AUC*	0.8522	0.8934	0.8600	0.8952	0.8917	0.8884	0.8889	**0.9147**	0.9135	0.9120	2.18%
	Hit@50	2.83%	2.15%	1.08%	4.13%	2.85%	2.87%	2.69%	3.96%	**4.82%**	3.43%	16.71%
Same Artist	*AUC*	0.6251	0.8051	0.7152	0.7697	0.8187	0.8658	0.7911	**0.9748**	0.9643	0.9608	12.59%
	Hit@50	4.90%	2.14%	1.28%	4.67%	4.08%	5.18%	3.94%	**18.42%**	17.94%	14.71%	246.33%
Session Trans.	*AUC*	0.8031	0.8670	0.8174	0.8627	0.8629	0.8626	0.8597	**0.9098**	0.9055	0.9058	4.94%
	Hit@50	2.65%	1.83%	0.99%	3.37%	2.09%	2.17%	2.25%	4.68%	**5.27%**	4.32%	56.38%
Same Session	*AUC*	0.8234	0.8924	0.8493	0.8867	0.9005	0.9147	0.8877	**0.9473**	0.9464	0.9416	3.56%
	Hit@50	2.08%	2.59%	1.29%	5.42%	4.44%	4.83%	3.83%	9.14%	**9.80%**	7.01%	80.81%

content-based features, Vista beats all other methods in all settings significantly (see the last column of Table 5.2 and 5.3). Note that appreciate data can be seen as a 'cold-start' version of the click data, i.e., people appreciate a subset of items they have clicked—about a quarter from the statistics. From the two tables we can see that the improvements of Vista on appreciate data over other methods are comparatively larger in almost all settings.

Order of Markov Chains. Vista benefits from using higher-order MCs considerably on both datasets, especially in terms of Hit@50. Using a small order seems to be good enough, presumably since the few most recent actions capture enough information to predict future actions.

Convergence. Next, we demonstrate the accuracy on the full test set (i.e., 'Overall' setting) of all comparison methods as the number of training iterations increases. Each iteration consists of processing training samples with the size of the whole training corpus. All baselines, except PopRec, are trained with multiple iterations until convergence or observance of overfitting on the validation set. As shown by Figure 5.3 and 5.4, Vista can converge in a few iterations due to the rich interactions being modeled.

Complexity Analysis

We analyze the time complexity of the N^{th}-order Vista as follows. Recall that we use a sampling scheme to handle multiple creators. Let K be the dimensionality of the user/item representations. For a given user u, time step t, and item j, the prediction $\widehat{x}_{u,t,j}$ takes $O(K \times F + K)$ to compute the interactions between u and j, and $O(N \times (K + K \times F))$ for the interactions between j and N previous items. Therefore for each sampled training triple (u,t,j), the calculation of $\widehat{x}_{u,t,j}$ will take $O(K \times F + K + N \times (K + K \times F)) = O(N \times K \times F)$. According to Eq. (5.16), computing the partial derivatives and updating relevant parameters also have the time complexity of $O(N \times K \times F)$ (when the two embedding

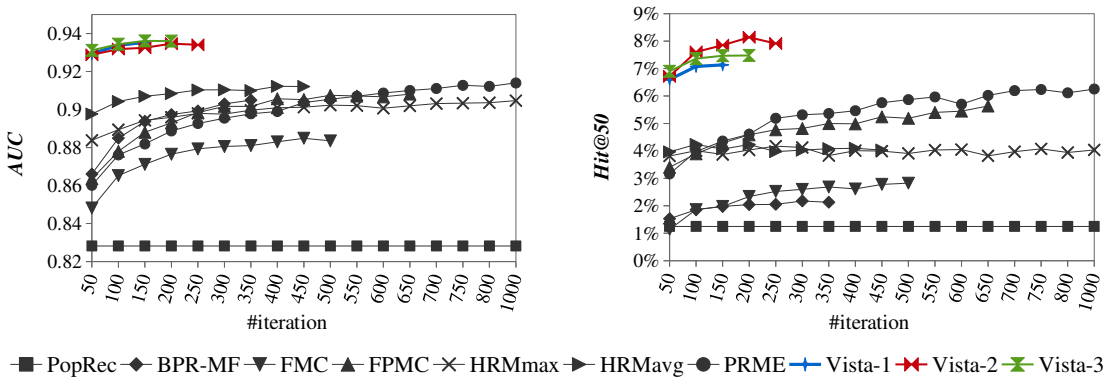


Figure 5.3: Performance comparison of different methods with number of training iterations on *Behance* click data.

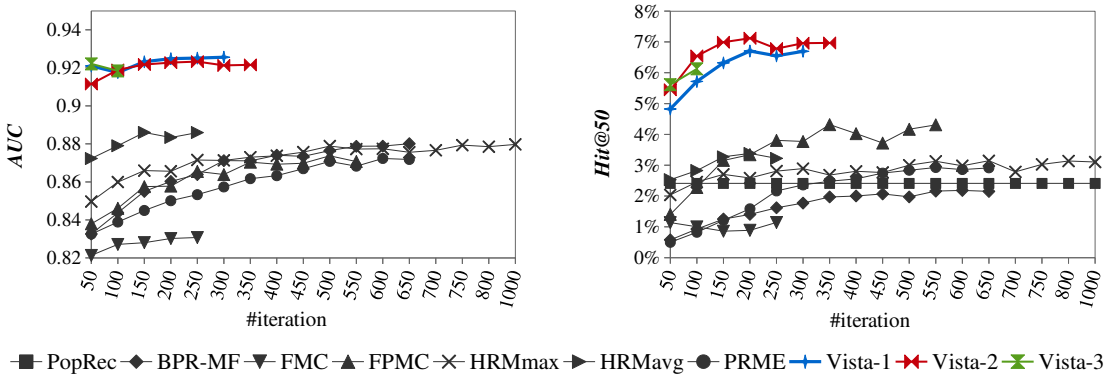


Figure 5.4: Performance comparison of different methods with number of training iterations on *Behance* appreciate data.

matrices are updated). Summing up all above components, it takes $O(N \times K \times F)$ to process each training triple. Since F is fixed to 4,096 in our case, the final time complexity becomes $O(N \times K)$. Using our lock-free asynchronous SGA training procedure with sampling, multiple triples are trained simultaneously. In our experiments, we used a sampling probability of 5% for Vista to update the two embedding matrices. Benefiting from multi-threading, each training iteration takes around 20 minutes to train our third-order Vista model on click data.

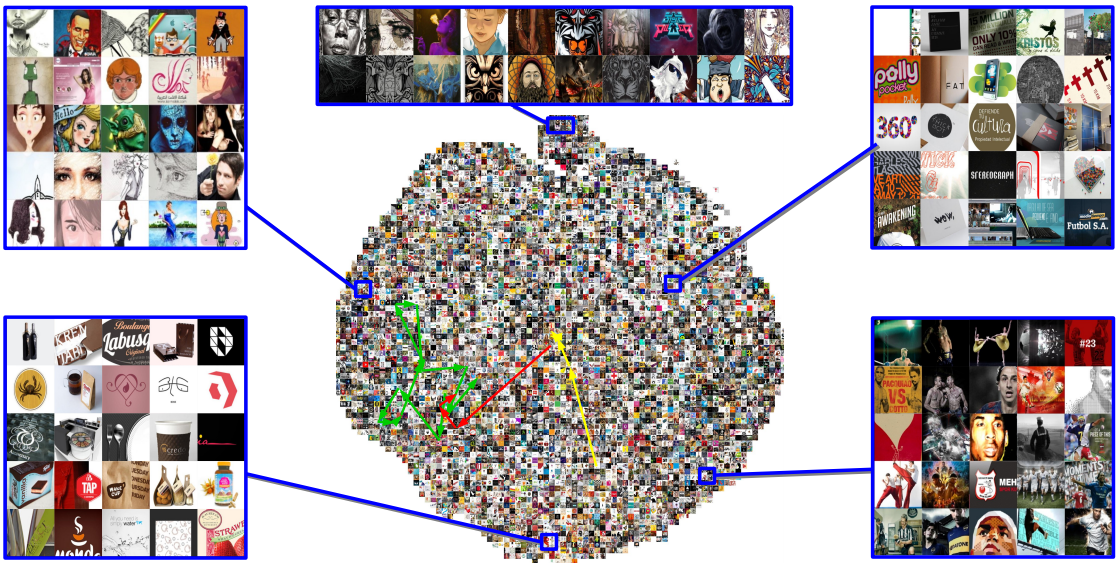


Figure 5.5: A t-SNE [117] visualization of space Ξ learned by Vista on *Behance* click data. We randomly sample a user and visualize three of their click sessions in the space, denoted by three paths with different colors on the grid view. There is a tendency that clicks are around a specific region in the space (i.e., near the green arrows), which reflects the long-term preference of the user. And each click sequence shows a certain amount of consistency, encoding the transition of short-term interests.

Latent Space Visualization

We proceed by visualizing the latent space Ξ learned by Vista, which is used to measure the similarity between different items. To this end, we take our model trained on *Behance* click data and further use t-SNE [117] to embed the 10-d space into 2-d. Figure 5.5 demonstrates a grid view of 972,181 items in the 2-d space. From this figure we can see that items with similar contents and styles tend to be neighbors in the latent space, e.g. culinary arts on the lower-left patch, and sports on the lower-right. Notably, items on the same patch may visually deviate from each other significantly.

On the grid view, we also demonstrate a few click sessions of a randomly selected user. The click sequence of each session is represented by a directed path with a unique color (red/yellow/green). We make a few observations as follows.

1. Clicks tend to occur around a specific region in the space, near the green and red

arrows. This reflects the long-term preferences of the user as people ultimately tend to explore items that they like.

2. Each click sequence demonstrates consistency to a certain degree (e.g. red and yellow) and encodes the transition of short-term interests (e.g. green).

3. Finally, the choice made at each click is a combination of long- and short-term preferences, due to which there are both long jumps and short jumps. Therefore, it is essential to capture both long- and short-term dynamics simultaneously in order to be successful at addressing our prediction task.

Visualizing Sessions

Users differ in habitual patterns especially in a dataset as large as ours. For example, for some users their short-term dynamics are more important. This means they tend to click/appreciate items similar to those they just interacted with. In contrast, there are also users whose long-term preferences are more emphasized, demonstrating less short-term consistency during a session. This characteristic is captured by the personalized weighting factor w_u in Eq. (5.2) (and w_u^k in Eq. (5.3)).

In Figure 5.6, we show a few sample sessions of the above two types of users, with different session lengths. Sampled sessions of users with the largest w_u (i.e., $\arg\max_u w_u$) are shown in Figure 5.6a, with each row demonstrating the list of items clicked during the corresponding session. We can see some consistency from these sessions: logo designs, a certain style of cartoon characters, interior designs, and fashion models respectively. On the right is the corresponding recommendation with the highest score predicted by the Vista model.

In contrast, in Figure 5.6b we also demonstrate a few sessions from users with the least w_u, i.e., for whom long-term dynamics are more important. As expected, contents in

each session demonstrate comparatively larger variance, though the long-term preference toward object designs, logos, cartoons, and characters (respectively) are captured by the Vista model.

5.6.4 Experiments on General Sequential Recommendation

In this subsection, we aim to fully evaluate the capability and applicability of Fossil and TransRec on the classical sequential recommendation task, e.g. predicting the next movie to watch, product to consume, or place to visit.

Datasets and Statistics

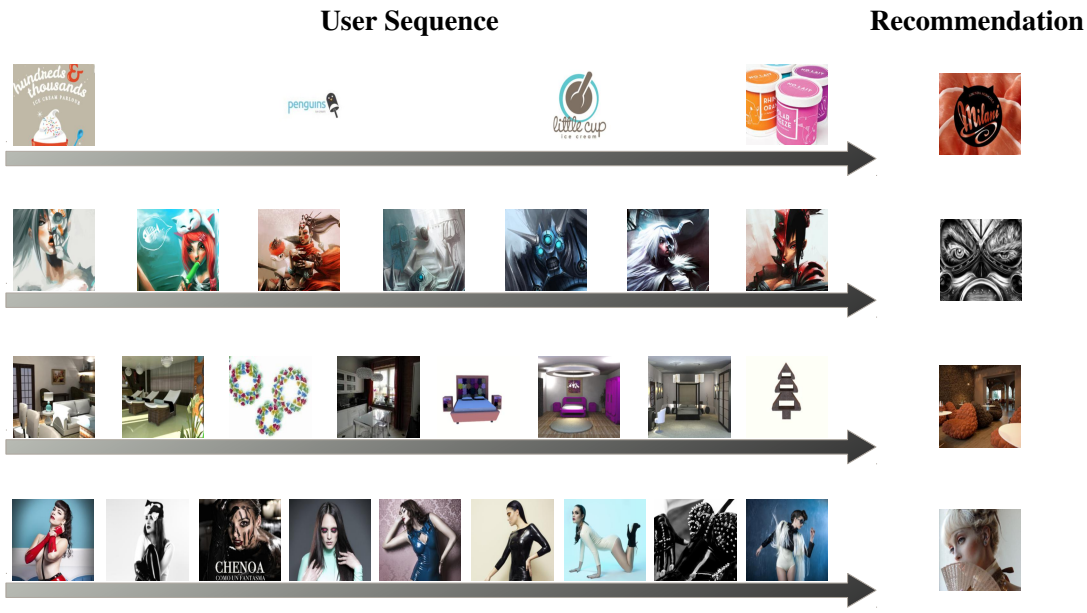
We include a wide range of publicly available datasets varying significantly in domain, size, data sparsity, and variability/complexity.

Amazon.[8] The first group of datasets, comprising large corpora of reviews and timestamps on various products, were recently introduced by [77]. These data are originally from *Amazon.com* and span May 1996 to July 2014. Top-level product categories on *Amazon* were constructed as separate datasets. Here we take a series of large categories including 'Automotive,' 'Cell Phones and Accessories,' 'Clothing, Shoes, and Jewelry,' 'Electronics,' 'Office Products,' 'Toys and Games,' and 'Video Games.' This set of data is notable for its high sparsity and variability.

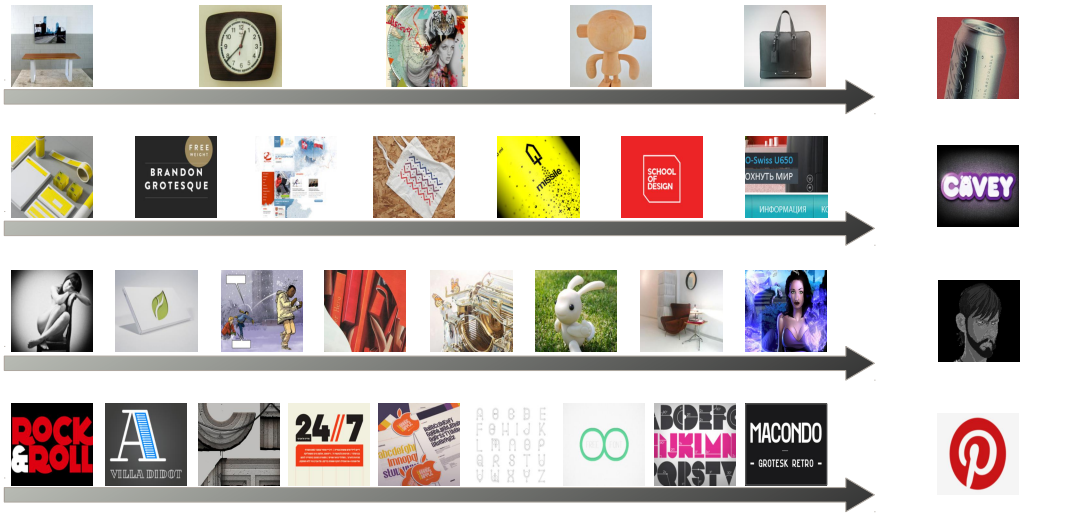
Foursquare.[9] Is originally from *Foursquare.com*, containing a large number of check-ins of users at different venues from December 2011 to April 2012. This dataset was collected by [60] and is widely used for evaluating next point-of-interest prediction methods. Note that the setting here is to compare all methods using collaborative data exclusively; making use of side signals like geographical data is beyond the scope here.

[8] http://jmcauley.ucsd.edu/data/amazon/

[9] https://archive.org/details/201309_foursquare_dataset_umn



(a) Sample sessions of users with the largest w_u (i.e., strong short-term consistency)



(b) Sample sessions of users with the least w_u (i.e., weak short-term consistency)

Figure 5.6: Demonstration of click sessions with different lengths (on the left) and the corresponding recommendations made by Vista (on the right).

Table 5.4: Dataset Statistics (in ascending order of item density).

Dataset	#users	#items	#actions	avg. #actions /user	avg. #actions /item
Automotive	34,316	40,287	183,573	5.35	4.56
Google	350,811	505,516	2,591,026	7.39	5.13
Office	16,716	22,357	128,070	7.66	5.73
Toys	57,617	69,147	410,920	7.13	5.94
Clothing	184,050	174,484	1,068,972	5.81	6.13
Cellphone	68,330	60,083	429,231	6.28	7.14
Games	31,013	23,715	287,107	9.26	12.11
Electronics	253,996	145,199	2,109,879	8.31	14.53
Foursquare	43,110	13,335	306,553	7.11	22.99
Total	**1,039,959**	**1,054,123**	**7,515,331**	-	-

Google Local. The last dataset is from *Google.com* which contains 11,453,845 reviews and ratings from 4,567,431 users on 3,116,785 local businesses. There are as many as 48,013 categories of local businesses distributed over five continents, ranging from restaurants, hotels, parks, shopping malls, movie theaters, schools, military recruiting offices, bird control, mediation services (etc.). The vast vocabulary of items, variability, and data sparsity make it a challenging dataset to examine the effectiveness of a model.

For each of the above datasets, we discard users and items with fewer than 5 associated actions in the system. In cases where star-ratings are available, we take all of them as users' positive feedback, since we are dealing with implicit feedback settings and care about purchases/check-in actions (etc.) rather than the specific ratings. Statistics of our datasets (after pre-processing) are shown in Table 5.4.

Performance and Quantitative Analysis

Results are collated in Table 5.5. Due to the sparsity of the datasets in consideration, the number of dimensions K of all latent vectors in all cases is set to 10 for

Table 5.5: Ranking results on different datasets in the classical sequential prediction setting (higher is better). The number of latent dimensions K for all comparison methods is set to 10. We test different orders of the *Markov Chain* (i.e., 1, 2, and 3) for Fossil. For HRM, we take the best results achieved by HRM_{max} and HRM_{avg}. The best performance in each case is boldfaced.

Dataset	Metric	PopRec	BPR-MF	FISM	FMC	FPMC	HRM	PRME	Fossil-1	Fossil-2	Fossil-3	$\text{TransRec}_{\mathcal{L}_1}$	$\text{TransRec}_{\mathcal{L}_2}$
Automotive	*AUC*	0.5870	0.6342	0.6695	0.6438	0.6427	0.6704	0.6469	0.6854	0.6867	0.6865	0.6779	**0.6868**
	Hit@50	3.84%	3.80%	3.20%	2.32%	3.11%	4.47%	3.42%	**5.59%**	5.28%	5.13%	5.07%	5.37%
Google	*AUC*	0.5391	0.8188	0.8376	0.7619	0.7740	0.8640	0.8252	0.8425	0.8447	0.8451	0.8359	**0.8691**
	Hit@50	0.32%	4.27%	2.30%	3.54%	3.99%	4.59%	5.07%	4.38%	3.91%	3.72%	6.37%	**6.84%**
Office	*AUC*	0.6427	0.6979	0.7100	0.6867	0.6866	0.7005	0.7020	0.7173	0.7218	0.7205	0.7186	**0.7302**
	Hit@50	1.66%	4.09%	4.36%	2.66%	2.97%	5.50%	6.20%	5.63%	5.70%	5.61%	**6.86%**	6.51%
Toys	*AUC*	0.6240	0.7232	0.7446	0.6645	0.7194	0.7579	0.7261	0.7622	0.7642	**0.7653**	0.7442	0.7590
	Hit@50	1.69%	3.60%	2.31%	1.55%	4.41%	5.25%	4.80%	**5.61%**	5.29%	5.11%	5.46%	5.44%
Clothing	*AUC*	0.6189	0.6508	0.7069	0.6640	0.6646	0.7057	0.6886	0.7198	0.7231	0.7242	0.7047	**0.7243**
	Hit@50	1.11%	1.05%	1.00%	0.57%	0.51%	1.70%	1.00%	2.02%	1.87%	1.70%	1.76%	**2.12%**
Cellphone	*AUC*	0.6959	0.7569	0.7735	0.7347	0.7375	0.7892	0.7860	0.7982	0.8011	0.8007	0.7988	**0.8104**
	Hit@50	4.43%	5.15%	2.51%	3.23%	2.81%	8.77%	6.95%	**9.73%**	9.14%	9.10%	9.46%	9.54%
Games	*AUC*	0.7495	0.8517	0.8635	0.8407	0.8523	0.8776	0.8597	0.8794	0.8806	**0.8816**	0.8711	0.8815
	Hit@50	5.17%	10.93%	9.11%	13.93%	12.29%	14.44%	14.22%	15.99%	15.91%	15.92%	**16.61%**	16.44%
Electronics	*AUC*	0.7837	0.8096	0.8184	0.8158	0.8082	0.8212	0.8337	0.8412	0.8433	0.8445	0.8457	**0.8484**
	Hit@50	4.62%	2.98%	1.27%	4.15%	2.82%	4.09%	3.07%	**5.81%**	5.36%	5.47%	4.89%	5.19%
Foursquare	*AUC*	0.9168	0.9511	0.9269	0.9463	0.9479	0.9559	0.9565	0.9603	0.9606	0.9606	0.9631	**0.9651**
	Hit@50	55.60%	60.03%	45.67%	63.00%	64.53%	60.75%	65.32%	65.13%	65.03%	65.42%	66.12%	**67.09%**

simplicity. The main findings from this table are summarized as follows:

1. BPR-MF, FISM and FMC achieve considerably better results than the popularity-based baseline in most cases, in spite of modeling personalization and sequential patterns in isolation. This means that uncovering the underlying user-item and item-item relationships is key to making meaningful recommendations.

2. BPR-MF and FISM are two powerful methods to model users' personalized preferences, i.e., long-term dynamics. Although ultimately they all factorize a matrix at their core, they differ significantly both in terms of the rationales they follow and performance they achieve. According to our experimental results, FISM exhibits significant improvements over BPR-MF mainly on sparse datasets in terms of AUC, which makes it a reasonable building-block for our sequential model.

3. FPMC and HRM are essentially combinations of MF and FMC. FPMC beats BPR-MF and FMC mainly on relatively dense datasets like Toys and Foursquare, and loses on sparse datasets—possibly due to the large number of parameters it introduces. From Table 5.5 we can see that HRM achieves strong results amongst all baselines in most cases, presumably from the aggregation operations.

4. PRME replaces the inner products in FPMC by distance functions. It beats FPMC in most cases, though loses to HRM due to different modeling strategies. Note that like FPMC, PRME turns out to be quite strong at handling dense datasets like Foursquare. We speculate that the two models could benefit from the considerable amount of additional parameters they use when data is dense.

5. By fusing FISM, which is strong at modeling long-term dynamics on sparse data, and FMC, Fossil enhances the performance of FISM by as much as 2.5% and 133.9% in terms of AUC and Hit@50 respectively. Comparing Fossil with

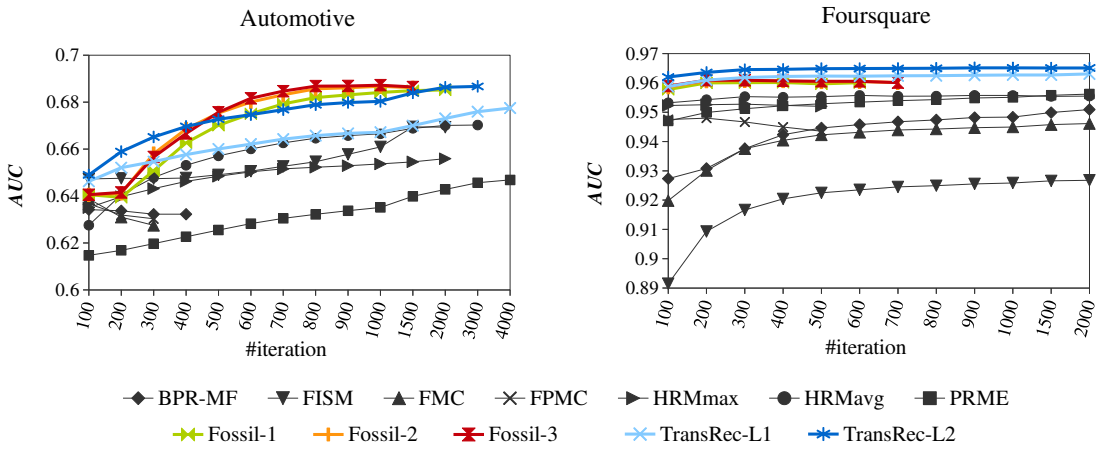


Figure 5.7: Performance comparison of different methods with number of training iterations on Automotive and Foursquare, representing sparse and dense data respectively.

FPMC/HRM/PRME, we found that Fossil beats them all on almost all datasets, with the average improvement of 1.2% (AUC) and 10.3% (Hit@50) over the best baseline performances.

6. TransRec outperforms all baselines in all cases (on average 1.6% in terms of AUC and 14.8% in terms of Hit@50). The improvements seem to be correlated with variability. TransRec achieves large improvements (32.54% and 24.71% in terms of Hit@50) on Google Local and Clothing, two datasets with the largest vocabularies of items in our collection. Taking Google Local as an example, it includes all kinds of restaurants, bars, shops (etc.) as well as a global user base, which requires the ability to handle the vast variability. On the other hand, TransRec beats all baselines especially on comparatively sparser datasets like Automotive and Google, which shows its ability to tackle *cold-start* data. In addition, we find that (squared) $\mathcal{L}_2$ distance typically outperforms $\mathcal{L}_1$ distance, though the latter also beats baselines in most cases.

Convergence

In Figure 5.7 we demonstrate (test) AUCs with increasing training iterations on two datasets—Automotive and Foursquare, representing sparse and dense data respectively. Simple baselines like FMC and BPR-MF converge faster than other methods on sparse datasets, presumably due to the relatively simpler dynamics they capture. FPMC also converges fast on such datasets as a result of its tendency to overfit (recall that we terminate once no further improvements are achieved on the validation set). On dense datasets like Foursquare, all methods tend to converge at comparable speeds due to the need to unravel denser relationships amongst different entities.

Visualizing Fossil

Here we visualize the learned Fossil model and qualitatively analyze our findings. We choose to visualize the results achieved on 'Clothing, Shoes and Jewelry' dataset from *Amazon.com* due to its large size, significant variability, and the convenience to demonstrate user actions. The model we use for visualization is the first-order Fossil model trained on the dataset with K set to 10.

First, we visualize the transition among items to answer questions like 'What kind of outfits are compatible with this outdoor cap?'. Fossil encodes this dynamic by the inner product of $\vec{p}_i$ and $\vec{q}_j$ where i is the item already consumed and j the item considered for recommendation. Quantitatively, given a 'query' item i at the current time step, items that are most likely to appear at next step are computed according to

$$\arg\max_{j \in I} \ \langle \vec{p}_i, \vec{q}_j \rangle. \tag{5.19}$$

For demonstration, we take a few samples from the above dataset, as shown on the left of the separator in Figure 5.8. Next, we use them as queries to get corresponding

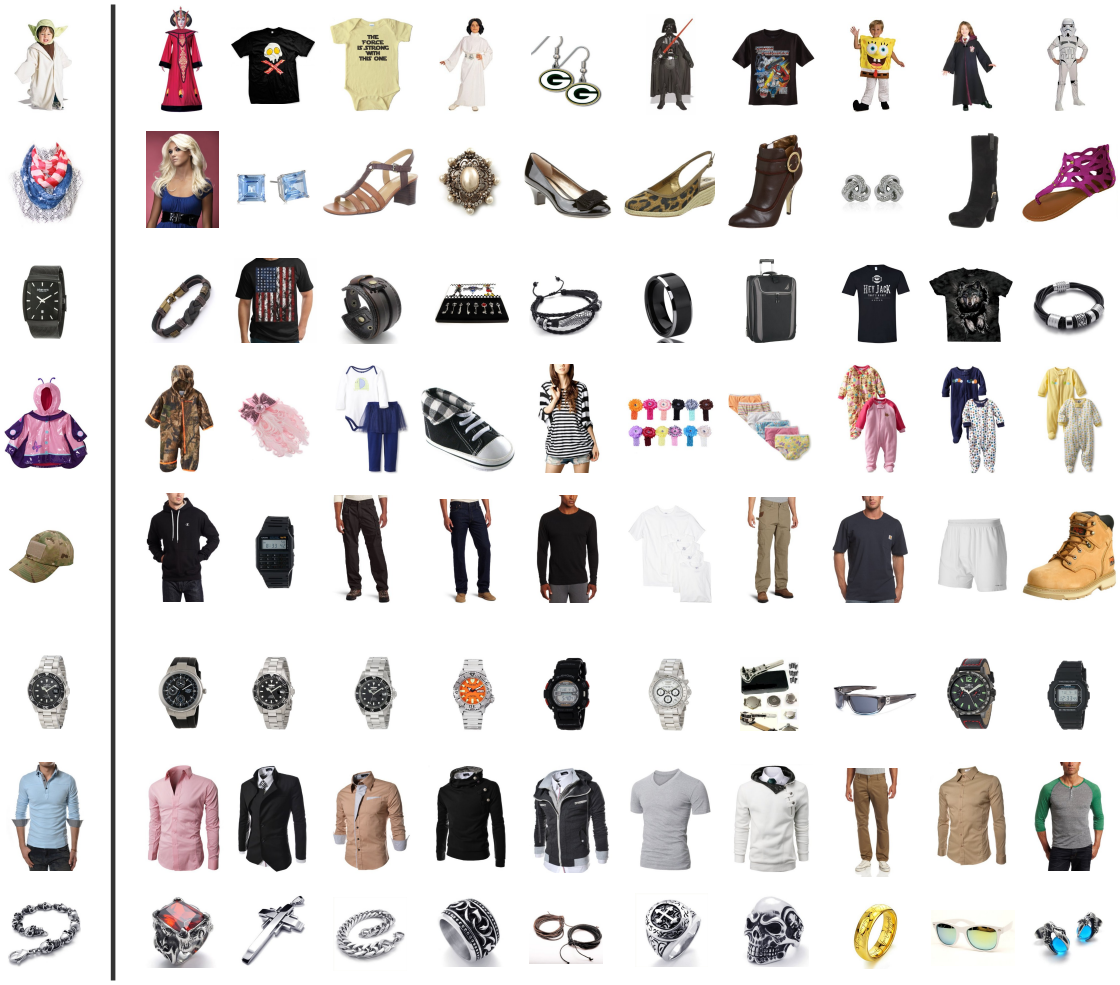


Figure 5.8: Demonstration of item transitions learned by Fossil. On the left are a few sampled items (or queries) from the *Amazon* Clothing dataset. On the right are the top-ranked items (from the same dataset) that each query is likely to transition to, according to $\arg\max_{j \in I} \langle \vec{p}_{query}, \vec{q}_j \rangle$.

recommendations according to Eq. (5.19). Items retrieved for each query are shown on the right. We make two observations from this figure. On the one hand, although the model is unaware of the identity of items, it learns the underlying homogeneity correctly, as we see from the first row (i.e., the *Star Wars* theme) and last three rows (i.e., watches, shirts and jewelry respectively). On the other hand, items from different subcategories are surfaced to generate compatible outfits, e.g. rows 2 and 5.

In Figure 5.9 we demonstrate recommendations made for a few users. Here the

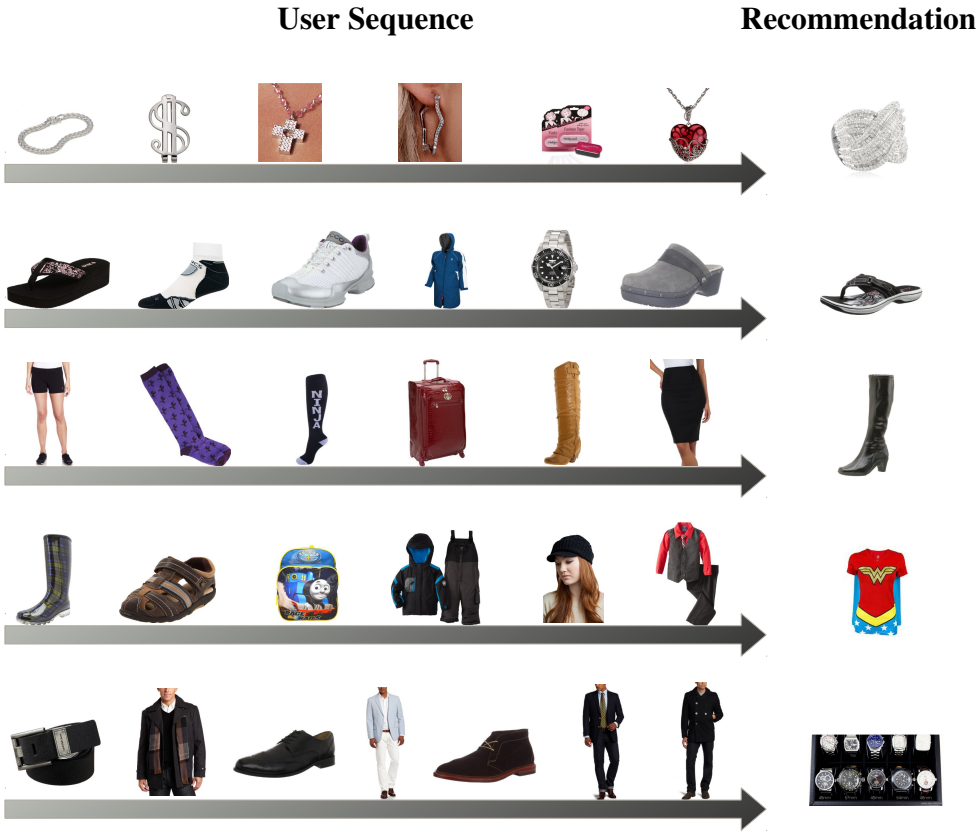


Figure 5.9: Demonstration of recommendations made by Fossil to users with large w_u^0 values, indicating strong 'sequential consistency.'

users are sampled from those with the largest w_0^u and at least 5 actions in the training set. The threshold is used so that w_0^u is forced to capture the sequential consistency of user u to some degree, instead of merely the user sparsity involved. From Figure 5.9 we can observe a certain amount of such consistency within each sequence. E.g., jewelry (row 1), wearables for boys (row 4) and business men (row 5).

In conclusion, it is precisely the ability to carefully accommodate multiple types of dynamics that makes Fossil a successful method to address the sequential recommendation task.

Figure 5.10: Demonstration of recommendations made by TransRec to a random sample of users on *Amazon* Electronics data.

Visualizing TransRec

In Figure 5.10 we demonstrate some recommendations made by TransRec ($K = 10$) on Electronics data from *Amazon.com*. We randomly sample a few users from the datasets and show their historical sequences on the left, and demonstrate the top-1 recommendation on the right. As we can see from these examples, TransRec can capture long-term dynamics successfully. For example, TransRec recommends a tripod to the first user who appears to be a photographer. The last user bought multiple headphones and similar items in history; TransRec recommends new headphones after the purchase of an iPod accessory. In addition, TransRec also captures short-term dynamics. For instance, it recommends a desktop case to the fifth user after the purchase of a motherboard. Similarly, the sixth user is recommended a HDTV after recently purchasing a home theatre receiver/speaker.

5.7 Conclusion

In this chapter, we built three new algorithms for large-scale sequential recommendation. The first algorithm, Vista, is designed for predicting personalized sequential behavior toward artistic items on a popular social art website, *Behance.net*. We found that our visually- and socially-aware *Markov Chains* are particularly effective at tackling multiple types of signals on a large scale, on both click and appreciate data. Additionally, our techniques for speeding up the training procedure as well as modeling high-order *Markov Chains* are successful and lead to improved performance.

The other two algorithms, Fossil and TransRec, are designed for the standard/general sequential recommendation setting. We found that by modeling user's transitioning behavior in a 'unified' manner, our two algorithms both achieved state-of-the-art results on a series of large, real-world datasets.

In addition, we visualized our learned models and observed that they capture sequential and personalized dynamics in a reasonable way, along with the favorable quantitative results achieved.

5.8 Acknowledgements

Chapter 5, in part, contains material as it appears in the *ACM Conference on Recommender Systems*, 2016 ("Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation," Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley), and the *IEEE International Conference on Data Mining series*, 2016 ("Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation," Ruining He and Julian McAuley), as well as the material that has been submitted for publication in the *ACM Conference on Recommender Systems*, 2017 ("Translation-based Recommendation," Ruining He, Wang-Cheng Kang, and Julian McAuley). The dissertation author was the primary investigator and author of these papers.

Chapter 6

Modeling Relational Signals for Item-to-item Recommendation

6.1 Introduction

Identifying and understanding relationships between items is a key component of any modern recommender system. Knowing which items are 'similar,' or which otherwise may be substitutable or complementary, is key to building systems that can understand a user's context, recommend alternative items from the same style [38], or generate bundles of items that are compatible [62, 77, 130].

Typically, identifying these relationships means defining (or otherwise learning from training data) an appropriate distance or similarity measure between items. This is appropriate when the goal is to learn some notion of 'equivalence' between items, e.g. in order to recommend an item that may be a natural alternative to the one currently being considered. However, identifying such a similarity measure may be insufficient when there is substantial *heterogeneity* between the items being considered. For example, the characteristics that make clothing items, electronic components, or even romantic partners

compatible exhibit substantial heterogeneity: for a pair of such items to be compatible they should be systematically similar in some ways, but systematically different in others.

Recently, a line of work has aimed to model such heterogeneous relationships, e.g. to model co-purchasing behavior between products based on their visual appearance or textual descriptions [75, 77, 118]. In spite of the substantial heterogeneity in the data used for training (a large dataset of co-purchase 'dyads' from *Amazon.com*) and the complexity of the models used, these works ultimately follow an established metric-learning paradigm: (1) Collect a large dataset of related (and unrelated) items; (2) Propose a parameterized similarity function; and (3) Train the parameterized function such that related items are more similar than non-related items. Such metric-learning approaches can be incredibly flexible and powerful, and have been used to identify similarities between items ranging from music [108] to members of the same tribe [20]. Such methods work to some extent even in the presence of heterogeneity, since they learn to 'ignore' dimensions where similarity should not be preserved. But we argue that ignoring such dimensions discards valuable information that ought to be used for prediction and recommendation.

In this chapter, we propose new models and algorithms to identify relationships amongst items for item-to-item recommendation. In particular, we relax the metricity assumption present in recent work, by proposing more flexible notions of 'relatedness' while maintaining the same levels of speed and scalability. Specifically, we hope to overcome the following limitations of previous work:

1. The similarity measures learned by previous approaches ultimately project categories as clusters into a metric space (albeit potentially via a complex embedding), since an item is inherently more similar to those from the same category than others (as we show later in Figure 6.5). This means that cross-category recommendations can only be made by exploiting an explicit category tree (e.g. 'find the shoes

nearest to these jeans'). Not only do such approaches require explicit category labels, but they are also subject to any noise or deficiencies in the category data. Our method can make cross-category recommendations without any dependence on the presence (or quality) of explicit category information.

2. Other assumptions made by metric learning approaches are also too strict for recommendation: an item is not necessarily compatible with itself (identity), nor are the types of relationships we want to learn necessarily symmetric (e.g. a spare battery is a good add-on item for a laptop, but not vice-versa). Other assumptions hidden in previous approaches (such as transitivity) may also be too strict, e.g. an iPhone is dissimilar from a Surface, though both are related to an iPad. Our approach is flexible enough to capture such complex and non-metric relationships.

3. Previous approaches learned a single 'global' (albeit complex) notion of relatedness, neglecting any 'local' notions that could be equally important. In contrast, we capture multiple (and possibly competing) notions of relatedness simultaneously. This is also key to generating *diverse sets* of recommendations. E.g. a shirt may be compatible with (1) a similar shirt from a different brand, (2) a similar shirt with a different color, (3) a complementary pair of pants, or (4) a complementary pair of shoes. By learning relatedness as a mixture of multiple competing notions, we can handle diverse sets of recommendations naturally.

We propose a novel method, *Mixtures of Non-Metric Embeddings for Recommendation*, or Monomer for short, that addresses the above limitations. We demonstrate our idea in Figure 6.1 (we later show an example on real data in Figure 6.2). Here we embed the first item x (the query) into one space (the 'anchor space'), and embed its potential match y into a series of N additional spaces. Now, the relatedness between x and y is measured in terms of multiple notions, each captured by one of the N spaces

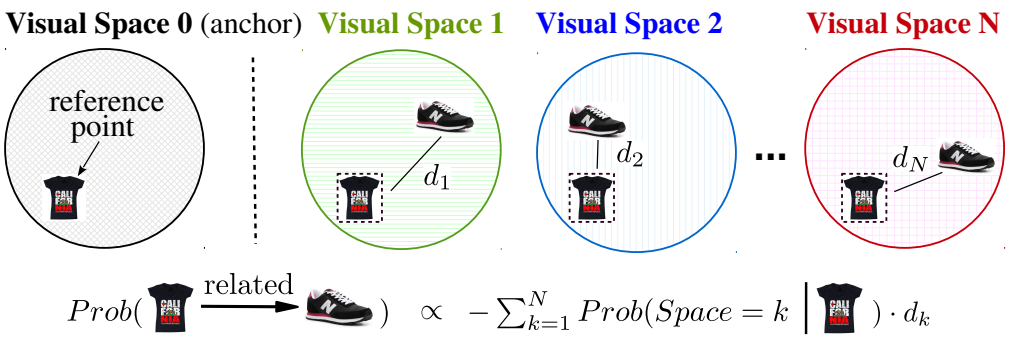


Figure 6.1: Illustration of the high-level ideas of Monomer. The query item (a t-shirt) is embedded into visual space 0 (the anchor space) whose position we superimpose into the other spaces. The potential match (a shoe) is embedded to N visual spaces and within each of them Euclidean distance between the pair is computed. Finally, the *mixtures-of-experts* framework is adopted to model the relative importance of the different components w.r.t. the given query. We show this on a real example in Figure 6.2.

involved. Furthermore, the N spaces are *weighted* according to a *mixtures-of-experts* type framework, determining to what extent each of the N embeddings is 'relevant' to a particular query.

Note in particular that the method described in Figure 6.1 can learn *non-metric* relationships since we are measuring the distance between *two* different embeddings. The learned relationships are not necessarily symmetric, nor do identity and transitivity necessarily hold; on the other hand the model is flexible enough such that a metric embedding could be learned if that was what the data supported.

6.1.1 Our Contributions

For clarity, our contributions are summarized as follows:

1. We propose a new scalable method, Monomer, for heterogeneous item-to-item recommendation. The presented mixtures-of-embeddings framework allows it to learn *non-metric* relationships, thereby overcoming multiple limitations present in existing work.

2. We demonstrate quantitatively that Monomer is effective at learning notions of 'relatedness' from heterogeneous dyads of co-purchases from *Amazon.com*, and in particular that it does so more accurately than recent approaches based on metric/similarity learning.

3. We qualitatively show that Monomer can effectively learn multiple, semantically complex notions of 'relatedness,' and that these can be useful to generate rich, heterogeneous, and diverse sets of recommendations.

6.2 Preliminaries

6.2.1 Visual Features

In this chapter, we mainly consider the case of using high-level visual features for relationship prediction. This is particularly useful for clothing recommendation (for example), a natural domain in which learning heterogeneous relationships between items across categories is particularly important. Our visual features are extracted from a *Convolutional Neural Network* pre-trained on 1.2 million ImageNet (ILSVRC2010) images, the same one we previously used in Chapter 4. In particular, we used the Caffe reference model [44], which has 5 convolutional layers followed by 3 fully-connected layers, to extract $F = 4,096$ dimensional visual features from the second fully-connected layer (i.e., FC7).

Note however that our proposed method is agnostic to the type of features used, and as we show later can handle other types of features (e.g. text) in order to address more general settings.

6.2.2 Mahalanobis Transform

In order to model subtle notions like 'compatibility' upon the raw visual features, we need expressive transformations that are capable of relating feature dimensions to explain the relationships between pairs of items. To this end, we follow the approach from [77]: there, a *Mahalanobis Distance* is used to measure the distance (or 'dissimilarity') between items within the feature space according to the knowledge of how different feature dimensions relate to each other. Let $\mathbf{M}$ denote the matrix that parameterizes the *Mahalanobis Distance*, then the distance between an item pair (x, y) is defined by

$$d_{\mathbf{M}}(x,y) = (\vec{f}_x - \vec{f}_y)^T \mathbf{M} (\vec{f}_x - \vec{f}_y), \tag{6.1}$$

where $\vec{f}_x$ and $\vec{f}_y$ are the features vectors of x and y respectively. Although such an approach defines a distance function (and therefore suffers from the issues we are hoping to address), we use this method as a building block and ultimately relax its limitations.

6.2.3 Mixtures-of-Experts

Mixtures of Experts (MoEs) are a classical machine learning method to aggregate the predictions of a set of (weak) learners, known as experts [40]. What is particularly elegant about this approach is that it allows each learner to focus on classifying instances about which it is relevant (i.e., expert), without being penalized for making misclassifications elsewhere.

For regression tasks such as the one we consider, each learner (denoted by l) outputs a prediction value $Pred_l(X)$ for the given input X. These predictions are then aggregated to generate the final prediction by associating weighted 'confidence' scores with each learner. Here we are interested in probabilistically modeling such confidences

to be proportional to the expertise of the learners:

$$\underbrace{Pred(X)}_{\text{final prediction}} = \sum_{l} \overbrace{Prob(l|X)}^{\text{confidence in } l\text{'s expertise}} \cdot \underbrace{Pred_l(X)}_{l\text{'s prediction}}. \tag{6.2}$$

In our model, each expert shall correspond to a single notion of relatedness between items. Thus, for a given pair of items that are potentially related, we can determine (1) which notions of relatedness are relevant for these items ($Prob(l|X)$); and (2) whether or not they are related according to that notion ($Pred_l(X)$). These two functions are learned jointly, such that the model automatically uncovers multiple notions of relatedness simultaneously.

6.3 Problem Formulation

Formally, we are given a dataset $\mathcal{D}$ comprising a large corpus of items and the pairwise relationships $\mathcal{R}$ between items from different subcategories, i.e., if $(x,y) \in \mathcal{R}$ then (1) item x and y are related, and (2) x and y are not from the same subcategory (e.g. a shirt and a matching pair of pants). We choose such cross-category recommendations to highlight the ability of our model to generate recommendations between heterogeneous pairs of items. This matches the training instance selection approach from [118]. Additionally, a high-dimensional feature vector $\vec{f}_x$ associated with each item x is also provided (encoding e.g. its image or the text of its reviews). We seek a scalable method to model such relationships with a set of parameterized transform functions $d(x,y)$ such that related items ($(x,y) \in \mathcal{R}$) are assigned higher probabilities than non-related ones ($(x,y) \notin \mathcal{R}$).

6.4 The Monomer Model

First we describe how Mahalanobis transforms have previously been applied to this task, and can be used as a building block for this task, before describing our proposed non-metric method.

6.4.1 Low-rank Mahalanobis Metric

Considering the high dimensionality of the visual features we are modeling (feature dimension F = 4,096 in our case), learning a full rank positive semi-definite matrix $\mathbf{M}$ as in Eq. (6.1) is neither computationally tractable for existing solvers nor practical given the size of the dataset.

Recently it was shown in [77] that a low-rank approximation of a Mahalanobis matrix works very well on visual datasets for the tasks considered here. Specifically, the $F \times F$ Mahalanobis matrix is approximated by $\mathbf{M} \approx \mathbf{E}^T\mathbf{E}$, where $\mathbf{E}$ is an $K \times F$ matrix and $K \ll F$. Then the distance between a pair (x, y) is calculated by

$$d_{\mathbf{E}}(x,y) = (\vec{f}_x - \vec{f}_y)^T\mathbf{E}^T\mathbf{E}(\vec{f}_x - \vec{f}_y) = ||\mathbf{E}\vec{f}_x - \mathbf{E}\vec{f}_y||_2^2. \tag{6.3}$$

This can be viewed as embedding the high-dimensional feature space (F-d) into a much lower-dimensional one (K-d) within which the squared Euclidean distance is measured. Note that the low rank property reduces the number of model parameters and increases the training efficiency significantly.

6.4.2 Multiple, Non-Metric Embeddings

There are two key limitations from using a low-rank Mahalanobis embedding approach like the one above. First, it can capture only a single set of dimensions (or

the 'statistically dominant reason') that determines whether two given items are related or not. However, there might be multiple reasons relevant to the link discrimination task in question. For example, a shirt and a pair of pants might go well together due to complementary colors, compatible textures, or simply some common characteristics they share (such as both having pockets/buttons, etc.). This drives us to use a group of embeddings, parameterized by N matrices $\mathbf{E}_1, \ldots, \mathbf{E}_N$ each with dimensionality $K \times F$ for the prediction task, with each capturing a different set of factors or 'reasons' that items may be related.

Another limitation of the single Mahalanobis embedding method, or more generally any metric-based method, is that it assumes that the closest neighbor of a given item is always itself, which is inappropriate for our task of placing many different categories of items close to the target. To overcome this shortcoming, we propose to use an anchor embedding (denoted by $\mathbf{E}_0$, again, with dimensionality $K \times F$) to learn the feature mappings in a non-metric manner.

In our model, $\mathbf{E}_0$ projects item x to a *reference* point $\mathbf{E}_0\vec{f}_x$ in the corresponding space, referred to as the anchor space as it will be used as the basis for further comparisons. Next, embeddings $\mathbf{E}_k$ (for $k = 1, 2, \ldots, N$) map the potential match y and correspond to a particular notion of relatedness, such that $\mathbf{E}_0\vec{f}_x$ will be close to $\mathbf{E}_k\vec{f}_y$ (for some k) if x and y are related.

That is, the predicted distance $d_k(x,y)$ by the k-th learner is

$$d_k(x,y) = ||\overbrace{\mathbf{E}_0\vec{f}_x}^{x\text{'s position in the anchor space}} - \underbrace{\mathbf{E}_k\vec{f}_y}_{y\text{'s position in the k-th 'pseudo' space}}||_2^2. \tag{6.4}$$

For clarity, we call the N spaces defined by $\mathbf{E}_k$ ($k > 0$) 'pseudo spaces' as all distance calculations are still performed within one actual space, i.e., the anchor space.

The above definition supports learning directed relationships as the model is not

required to be symmetric; but, it is flexible enough to learn symmetric (or even metric) embeddings if such structures are exhibited by the data.

6.4.3 Probabilistic Mixtures of Embeddings

Now we introduce how we aggregate the predictions from different embeddings. Given an item pair (x,y), we build our model upon the MoEs framework to learn a probabilistic gating function to 'switch' among different embeddings. Considering our asymmetric setting where the query item x in the pair is used as the reference point, we model the probability that the k-th embedding is used for the given pair (x,y) with a softmax formulation:

$$Prob\big(k|\underbrace{(x,y)}_{\text{the given item pair}}\big) = \overbrace{Prob(k|x)}^{\text{only depends on } x} = \frac{\exp(\mathbf{U}_{k,:}\vec{f}_x)}{\sum_i \exp(\mathbf{U}_{i,:}\vec{f}_x)}, \tag{6.5}$$

where $\mathbf{U}$ is a newly-introduced $N \times F$ parameter matrix with $\mathbf{U}_{k,:}$ being its k-th row. Briefly, the idea is to compute the probability distribution over the N learners given the characteristics of the 'pivot' item x. Note that our formulation is efficient as it only introduces a small number of parameters given that N is usually a small number (e.g. on the order of 4 or 5 in our experiments).

Finally, our model calculates the distance of an item pair (x,y) by the probabilistic expectation:

$$d(x,y) = \sum_{k=1}^{N} Prob\big(k|(x,y)\big) \cdot d_k(x,y). \tag{6.6}$$

Note that our 'distance' definition is a *non-metric* method as it only preserves the non-negativity and is relaxing the symmetry, identity, and triangle inequality properties.

6.4.4 Learning the Model

With the distance function defined above, we model the probability that a pair is related by a shifted sigmoid function (in a way similar to [77]):

$$Prob\big((x,y) \in \mathcal{R}\big) = \sigma_c\big(-d(x,y)\big) = \frac{1}{1+\exp\big(d(x,y)-c\big)}. \tag{6.7}$$

Next, we need to randomly select a negative set of relationships $\bar{\mathcal{R}}$. To this end, we use a procedure from [80] which randomly rewires the positive set in such a way that (1) the degree sequence of items is preserved and (2) each negative pair consists of items from two categories.

Then we proceed by fitting the parameters by maximizing the log-likelihood of the training corpus:

$$\begin{aligned}\widehat{\Theta} \;=\; \arg\max_{\Theta} \mathcal{L}(\mathcal{R},\bar{\mathcal{R}}|\Theta) = &\sum_{(x,y)\in\mathcal{R}} \log\Big(Prob\big((x,y)\in\mathcal{R}\big)\Big) \\ &+ \sum_{(x,y)\in\bar{\mathcal{R}}} \log\Big(1-Prob\big((x,y)\in\mathcal{R}\big)\Big) - \Omega(\Theta),\end{aligned} \tag{6.8}$$

where Θ is the full parameter set $\{\mathbf{E}_0,\mathbf{E}_1,\ldots,\mathbf{E}_N,\mathbf{U},c\}$, and $\Omega(\Theta)$ is an $\mathcal{L}_2$ regularizer. The total number of parameters is $F\times(N\times K+K+N)+1$. Since N and K are small numbers (see Section 6.5), the log-likelihood as well as the derivatives can be computed efficiently.

Monomer is learned with L-BFGS [66], a quasi-Newton method for non-linear optimization of problems with a large number of variables. Our log-likelihood and the full derivative computations can be naïvely parallelized over all training pairs $(x,y)\in\mathcal{R}\cup\bar{\mathcal{R}}$. This means the optimization can easily benefit from multi-threading and even parallelization across multiple machines (e.g. [18]).

6.4.5 Scalability Analysis

In Monomer, each embedding matrix has $K \times F$ parameters, which means there are in total $F \times K \times (N+1)$ embedding parameters. Since Monomer does not need to use more embedding parameters to outperform the *Low-rank Mahalanobis Transform* (LMT) method that uses only a single embedding matrix (as we show later in Section 6.5), we focus on comparing Monomer and LMT under the same total number of embedding parameters, in terms of the amount of multiplications involved.

For complete clarity, we denote the embedding dimension of LMT and Monomer by K' and K respectively ($F \times K' = F \times K \times (N+1)$). For each training pair (x, y), LMT takes $O(F \times K')$ to compute the distance between them and the corresponding derivatives. While for Monomer, it takes $O(F \times K')$ to project x and y to the multiple spaces. Afterwards the distance will be calculated in $O(N \times K) + O(N \times F)$, where the former is for computing N distance components and the latter is spent on the probabilistic weights. In total, it takes $O(F \times K') + O(N \times K) + O(N \times F) = O(F \times K')$ for Monomer to finish distance computation. Likewise, it's easy to verify that the corresponding derivatives can also be computed in $O(F \times K')$ time. To sum up, training Monomer and LMT will have the same time complexity when using the same amount of embedding parameters.

6.5 Experiments and Analyses

6.5.1 Datasets and Statistics

To fully evaluate the ability of Monomer to handle real-world tasks, we want to experiment on the largest dataset available. To this end, we adopt the dataset from *Amazon.com* recently introduced by [77]. We focus on five large top-level categories

Table 6.1: Statistics of a few representative categories from the *Amazon* 'Clothing, Shoes, and Jewelry' dataset (using visual features).

Dataset	#subcategories	#items	Relationship (#edges)	
			co-purchase	*co-browsing*
Men's Clothing	56	306,215	1,075,547	635,610
Women's Clothing	116	659,566	1,923,952	1,691,121
Boys's Clothing	41	42,156	169,503	75,689
Girls's Clothing	44	56,593	191,964	97,881
Baby's Clothing	6	36,588	96,253	95,784
Total	**263**	**1,101,118**	**3,457,219**	**2,596,085**

under the category tree rooted with 'Clothing, Shoes, and Jewelry,' i.e., Men's, Women's, Boys', Girls', and Baby's Clothing & Accessories. Statistics are shown in Table 6.1.

For each of the above categories, we experiment with two important types of relationships: 'users who bought x also bought y,' and 'users who viewed x also viewed y,' denoted by 'co-purchase' and 'co-browsing' respectively for brevity. Such relationships are a key source of data to learn from in order to recommend items of potential interests to customers. Ground-truth for these relationships is also introduced in [77], and are originally derived from co-purchase and co-browsing data from *Amazon.com*.

6.5.2 Evaluation Methodology

Recall that our objective is to learn heterogeneous relationships so as to support cross-category recommendation. Across the entire Clothing, Shoes, and Jewelry dataset, such relationships are noisy, sparse, and not always meaningful. To address issues of noise and sparsity to some extent, it is sensible to focus on the relationships within the scope of a particular top-level category, e.g. Women's Clothing, Men's Clothing etc. We then consider relationships between '2$^{\text{nd}}$-level' categories, e.g. women's shirts, women's shoes, etc.

In summary, our evaluation protocol is as follows:

1. A single experiment consists of a specific category (e.g. Men's Clothing) and a graph type (e.g. co-purchase).

2. For each experiment, the relationships ($\mathcal{R}$) and a random sample of non-relationships ($\bar{\mathcal{R}}$, see Section 6.4.4) are pairs of items connecting different subcategories of the category we are experimenting on. Note that $|\mathcal{R}| = |\bar{\mathcal{R}}|$ and they share the same distribution over the items.

3. For each experiment, we use an 80/10/10 random split of the dataset ($\mathcal{R} \cup \bar{\mathcal{R}}$) with the training set being at most two million pairs. Our goal is then to predict the relationships and non-relationships correctly, i.e., link prediction.

4. For all methods, the validation set is used for tuning the regularization hyperparameters, and finally the learned models are evaluated on the test set in terms of error/misclassification rate.

For example, one experiment is to predict co-purchase relationships for Men's Clothing. There are 56 subcategories under Men's Clothing (see Table 6.1), so our goal is to distinguish edges from non-edges connecting items from among these subcategories.

All experiments were performed on a single machine with 64GB memory and 8 cores. Our largest experiment required around 40 hours to train, though most were completed in a few hours.

6.5.3 Comparison Methods

1. **Weighted Nearest Neighbor (WNN):** uses a weighted Euclidean distance in the raw feature space to measure similarity between items:

$$d_{\vec{w}}(x,y) = \|\vec{w} \circ (\vec{f}_x - \vec{f}_y)\|_2^2.$$

Here $\circ$ is the Hadamard product and $\vec{w}$ is a weighting vector learned from the data.

2. **Category Tree (CT):** computes a matrix of co-occurrences between subcategories from the training data. Then a pair (x, y) is predicted to be positive if the subcategory of y is one of the top 50% most commonly connected subcategories to the subcategory of x.

3. **Low-rank Mahalanobis Transform (LMT):** is a state-of-the-art method for learning visual similarities among different items (possibly between categories) on large-scale datasets [77]. LMT learns a *single* low-rank Mahalanobis embedding matrix to embed all items into a low-dimensional space (see Section 6.4.1). Then it predicts the links between a given pair based on the Euclidean distance within the embedded space (i.e., Eq. (6.3)).

4. **Mixtures of Non-metric Embeddings (Monomer):** Our method. It learns a mixture of low-rank transforms/embeddings to uncover groups of underlying reasons that explain the relationships between items. It measures the distance (or dissimilarity) between items in a non-metric manner (i.e., Eq. (6.6)).

Ultimately, our baselines are designed to demonstrate that (1) the raw feature space is not directly suitable for learning the notions of relationships (i.e., WNN); (2) using category metadata directly and not using other features (i.e., CT) results in relatively poor performance; and that (3) our proposed model is an improvement over the state-of-the-art method on our task (i.e., LMT).

6.5.4 Performance and Quantitative Analysis

Error rates on the test set for all experiments are reported in Table 6.2. To perform a fair comparison between LMT and Monomer, the following setting is used for all experiments:

Table 6.2: Test errors of the link prediction task (i.e., predicting co-purchase and co-browsing relationships between items) using visual features (4,096-d) on clothing categories of the *Amazon* dataset. The best performing method in each case is boldfaced. Lower is better.

Dataset	Relationship	(a) WNN	(b) CT	(c) LMT	(d) Monomer	% Imprv. (d) vs. (c)
Men's Clothing	*co-purchase*	34.95%	47.71%	9.20%	**6.48%**	30%
	co-browsing	18.98%	47.40%	6.78%	**6.58%**	3%
Women's Clothing	*co-purchase*	30.50%	49.73%	11.52%	**7.87%**	32%
	co-browsing	20.50%	49.48%	7.90%	**7.34%**	7%
Boys's Clothing	*co-purchase*	31.16%	46.02%	8.80%	**5.71%**	35%
	co-browsing	21.52%	46.22%	6.72%	**5.35%**	20%
Girls's Clothing	*co-purchase*	31.10%	47.63%	8.33%	**5.78%**	31%
	co-browsing	22.36%	46.43%	6.46%	**5.62%**	13%
Baby's Clothing	*co-purchase*	37.26%	48.01%	12.48%	**7.94%**	36%
	co-browsing	30.89%	47.72%	11.88%	**9.25%**	22%
Avg.		27.92%	47.64%	9.00%	**6.79%**	22.9%

1. It has been shown by [77] that LMT can achieve better accuracy when using a reasonably large number of embedding dimensions (K). Therefore in all cases we choose K large enough such that LMT obtains the best possible (validation) performance.

2. In all cases we try to compare LMT and Monomer under the same total number of model parameters. For example, if we set the number of dimensions K to 100 for LMT, then a fair setting for Monomer would be $K = 20$ and $N = 4$. This way both of them are using $100F$ embedding parameters.[1]

For experiments on co-purchase relationships, LMT uses $K = 100$ dimensions and Monomer uses $K = 20$ and $N = 4$. While for experiments on co-browsing relationships, K is set to 50 for LMT and $K = 10$ and $N = 4$ for Monomer. Note that co-browsing relationships are almost twice as sparse as co-purchase relationships and thus a model

[1] Recall that N is the number of embeddings (excluding the anchor embedding).

with fewer parameters performed better at validation time (as shown in Table 6.1). We make a few observations to explain and understand our findings as follows:

1. WNN is particularly inaccurate for our task. We also observed relatively high training errors with this method for most experiments. This confirms our conjecture that raw similarity is inappropriate for our task, and that in order to learn the relationships across (sub)categories, some sort of expressive transforms are needed for manipulating the raw features.

2. The counting method (CT) performs considerably worse than other methods. This reveals that the predictive information used by the other models goes beyond the categories of the products, i.e., that the image-based models are learning relationships between finer-grained attributes.

3. Note that all models perform better at predicting co-browsing than co-purchase relationships. This is reasonable since intuitively items that are "also viewed" indeed tend to share more common characteristics compared to the "also bought" scenario. The greater heterogeneity between training pairs in the latter task makes it comparatively harder to address.

4. Monomer outperforms LMT significantly for all experiments, especially for the harder task of predicting co-purchase dyads.

6.5.5 Visualization and Qualitative Analysis

Visual Embeddings

We proceed by demonstrating the embeddings learned from our largest dataset, Women's Clothing, by Monomer. We take the same model trained on co-purchase relationships from the previous subsection and visualize it in Figure 6.2. In this figure,

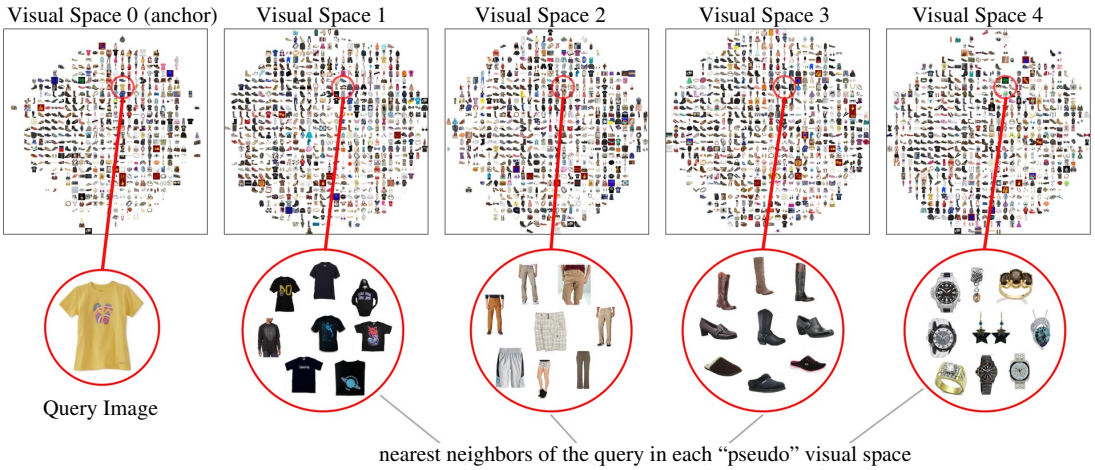


Figure 6.2: Visualization of Monomer trained on Women's Clothing for co-purchase prediction. Each visual space is demonstrated by a 2-d t-SNE grid view [117] (each cell randomly selects one image in overlapping cases). According to our distance function (i.e., Eq. (6.6)), Monomer recommends the nearest neighbors of the query within each visual space, based on the associated reasons learned from data. Note that each visual space exhibits different category clusters at the query image's location, allowing us to recommend *diverse* sets of items from the most closely-related categories.

we show each of the 5 visual spaces by a 2-d visualization with t-SNE [117]. Images are a random sample of size 50,000 from the Women's Clothing dataset and projected (using the learned embedding matrices) to each visual space to demonstrate the underlying structure.

As analyzed in Section 6.4, each embedding (i.e., learner) is capturing a specific notion of relatedness that explains the relationships of pairs of items in the corpus. In other words, it means that the nearest neighbors in each of the N pseudo spaces should be related to the query according to the specific notion captured. Therefore those neighbors should be recommended as potential matches to the query item, as shown by the example in Figure 6.2. For the query image (a t-shirt) in this example, Monomer recommends bundles of similar t-shirts, pants, shoes, and accessories (watches etc.) that resemble the query in terms of patterns (e.g. space 1), colors (e.g. space 2), and more generally 'styles' (e.g. space 3 and 4).[2] Such matching between a query image and nearby items in

[2]The second patch actually contains a few men's clothing items due to data deficiency—an intrinsic

alternate spaces directly facilitates the task of recommending visually consistent outfits, where modeling and understanding the visual compatibility across categories is essential.

Visual Dimensions

Next we demonstrate the visual dimensions learned by Monomer, i.e., what kind of characteristics the model is capturing to explain the relationships among items. Here we visualize these dimensions by showing items that exhibit maximal values for each dimension. In other words, we select items according to

$$\arg\max_{x} \mathbf{E}_{k,:}\vec{f}_x,$$

where $\mathbf{E}_{k,:}$ is the k-th row of the embedding matrix $\mathbf{E}$, corresponding to a visual dimension. Intuitively, this informs us of items that are most representative of a particular visual aspect discovered by the model.

We trained Monomer on Men's Clothing ($K = 10$, $N = 4$), predicting co-purchase relationships. Due to limited space, we randomly select one embedding and demonstrate its 10 visual dimensions in Figure 6.3. From the figure we can see that (1) Monomer seems to uncover meaningful visual dimensions, each of which highlights certain *fine-grained* item types (e.g. plaid tees and jeans in row 1 and 5); (2) human notions seem to have been captured, e.g. casual versus formal in rows 2 and 9; and (3) subtle differences between different characteristics can be distinguished (e.g. tees in rows 2 and 6). Monomer's ability to discover and model the correlations among visual characteristics explains its success.

---

problem suffered by *Amazon.com*.

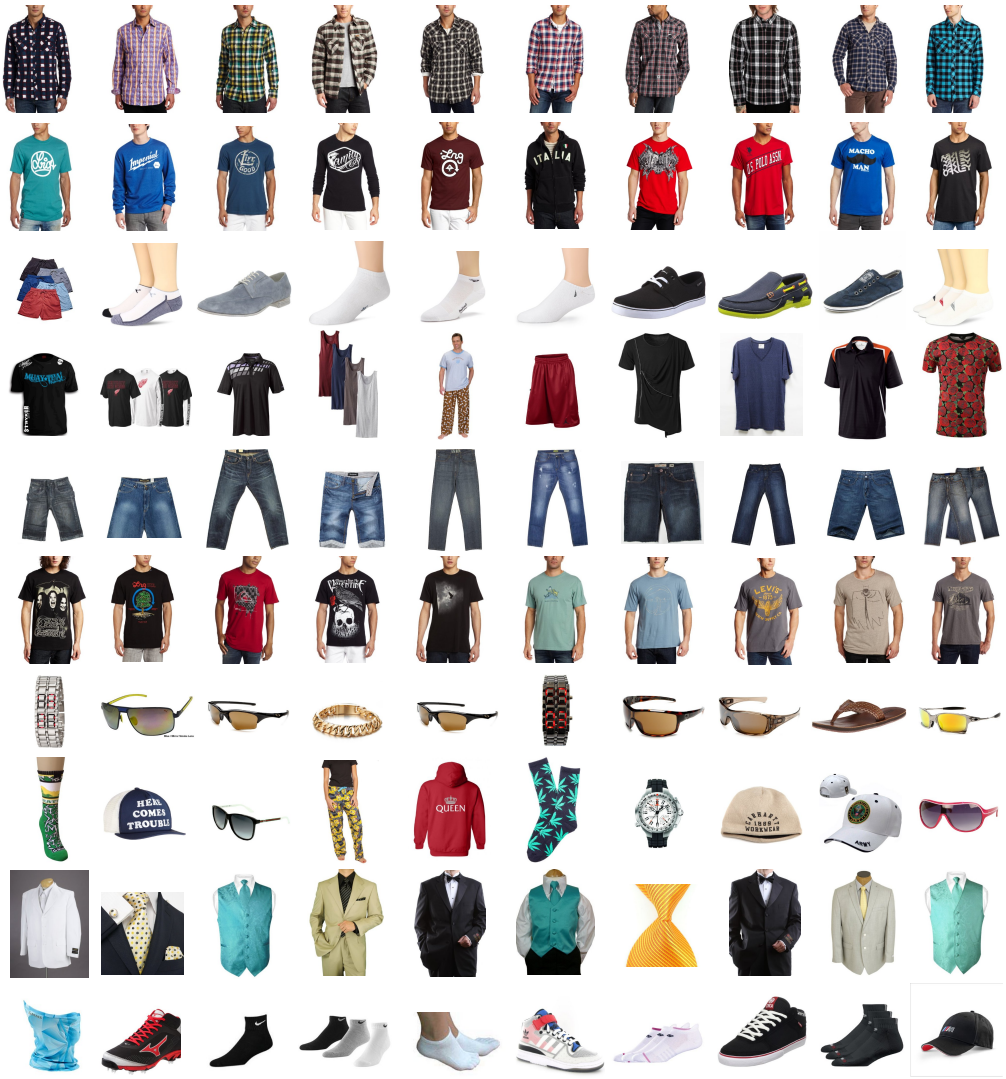


Figure 6.3: Demonstration of the 10 visual dimensions of one (randomly selected) visual space learned by Monomer on Men's Clothing for co-purchase prediction ($K = 10$, $N = 4$). Each row shows the top ranked items for a particular dimension i, i.e., $\arg\max_x \mathbf{E}_{k,:}\vec{f}_x$ where $\mathbf{E}_{k,:}$ is the k-th row of the corresponding embedding matrix $\mathbf{E}$.

Visual Recommendation and Analysis

Beyond achieving high prediction accuracy, we want to test the ability of Monomer to generate useful recommendations. Again, we mainly compare to the state-of-the-art metric-based method, LMT. Both methods are able to learn relationships from the data, so one common setting is to retrieve 'similar' items (i.e., maximum probability of being

related) to a given query.

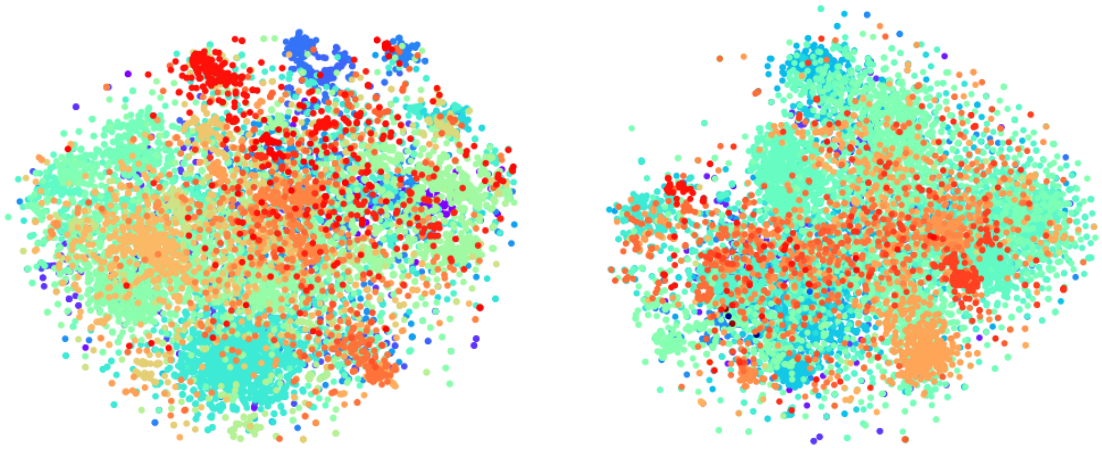
First we train LMT and Monomer on Women's Clothing to predict co-purchase relationships, under the same setting as in Table 6.2. This way the two models will learn their own similarity (or distance) functions from the data. Next, from Women's Clothing we randomly select a few query items, for each of which LMT and Monomer will retrieve its highest-probability links according to their own similarity functions. Figure 6.4 demonstrates such queries and the retrieved connections (in all cases ranked in decreasing order in terms of the probability of the link) by the two models.

As shown in Figure 6.4, the metric-based method (LMT) tends to recommend items that are very similar to the query, even though for this task it is trained to predict complementary relationships (i.e., co-purchase). Indeed it is very difficult for a metric-based method to project items from different subcategories to be nearer than items from the same category; presumably such methods are limited by their underlying assumption that the most similar item to a given query is always itself. In [77] this was addressed to some extent by making explicit use of the category information at test time (e.g. 'find the shirt closest to this pair of shoes'), though our model is able to make diverse sets of recommendations without such a dependence on explicit category information.

Recall that LMT learns an embedding within which the Euclidean distance is used to distinguish relationships from non-relationships. Visualizing such spaces can help understand the behavior of LMT. Again we uniformly sample 10,000 items from the dataset and use t-SNE [117] to visualize their positions in the embedded space. We are particularly interested in the distribution of different subcategories of items over the space. Therefore we assign a unique color to each subcategory in the dataset. Figure 6.5 shows results on two representative datasets, Men's and Women's Clothing.

From Figure 6.5 we find that subcategories of items tend to become 'clusters' in the embedded space. This can be problematic especially for recommending related items

Figure 6.4: Comparison between the state-of-the-art metric-based method, LMT, and our non-metric method Monomer. On the left are a few query images, for each of which we show its nearest neighbors retrieved by LMT (above the horizontal line) and Monomer (below the horizontal line) respectively. Both models were trained on Women's Clothing for co-purchase prediction (using the same setting as in Table 6.2). All query images and all neighbors are from Women's Clothing.



(a) *Amazon* Men's Clothing (b) *Amazon* Women's Clothing

Figure 6.5: Demonstration of the distribution of different subcategories in the 100-d space learned by LMT [77]. For visualization, we use t-SNE [117] to further embed this space into 2-d. In each subgraph, a color represents a specific subcategory within the corresponding dataset. The main finding is that LMT tends to project subcategories to be clusters in the embedded space, which may cause a 'limited coverage' problem for the recommendation task.

across subcategories:

1. From a recommendation perspective, there will be a *limited coverage* issue because given a query LMT tends to recommend only those items on the *boundaries* of the clusters. There is no way that items located near the center of a cluster will ever be recommended, since the closest item outside the query's own category cluster will be on the fringe of a different category cluster.

2. Recommendations will suffer from *mislabeling* issues. Note that LMT relies heavily on the taxonomy metadata at test time to filter out items from the same subcategory as the query. However, even a tiny number of mislabeled items in the dataset can poison recommendations, and certainly some such examples exist in the *Amazon* dataset [118]. Unexpected items may appear on the recommendation list when there are mislabeled items that actually come from the same subcategory as the query.

Table 6.3: Statistics of a variety of top-level categories from *Amazon.com*.

Dataset	#subcategories	#items	Relationship (#edges)	
			co-purchase	*co-browsing*
Electronics	306	412,082	1,654,552	718,361
Automotive	178	312,642	959,353	1,298,774
Video Games	16	49,801	314,124	54,559
Movies & TV	2	199,737	648,256	49,924
Office Products	245	127,054	448,720	370,630
Home & Kitchen	81	393,781	560,574	960,925
Cell Phones	28	317,965	867,418	225,785
Total	**856**	**1,813,062**	**5,452,997**	**3,678,958**

Note that Monomer does not suffer from either of above issues since it has already successfully blended different subcategories, as shown by the nearest neighbors in Figure 6.4.

6.6 Learning Compatibility from Textual Features

In previous sections, we have shown that Monomer not only performs very well on link prediction tasks but also that it recommends highly diverse sets of items. However, above we only considered scenarios in which relationships like co-purchasing can be predicted from visual features. Following this, a natural question would be "Is Monomer able to learn relationships from *non*-visual features and achieve similarly competitive performance?"

To answer the above question, we perform further experiments on Bag-of-Words (BoW) features extracted from the text of product reviews, which are also available in the *Amazon* dataset. In particular, we experiment with a variety of top-level *Amazon* categories, i.e., 'Electronics,' 'Automotive,' 'Video Games,' 'Movies & TV,' 'Office Products,' 'Home & Kitchen,' and 'Cell Phones & Accessories.' Statistics of these datasets are shown in Table 6.3.

For each category (e.g. Electronics) we use the following procedure to generate BoW features for all its items: (1) Remove stop-words and construct a dictionary. Our dictionary consists of 5,000 nouns or adjectives or adjective-noun bigrams that appear most frequently in the review corpus being considered. (2) For each item i, a document doc_i is generated by bagging all the reviews it has received. (3) The 5,000-d BoW feature vector $\vec{f_i}$ of each item i is computed by normalizing the raw word counts of document doc_i to sum to 1. (4) Items without any reviews attached are seen as invalid items and are dropped from the dataset. In the following experiments, we use the same evaluation protocol as in the previous visual feature experiments.

Latent Dirichlet Allocation + WNN (LDA): Here we add another baseline for further comparison. This method first obtains 100 topics with LDA with a vocabulary of size 5,000,[3] and then uses WNN to distinguish relationships from non-relationships within the 100-d topic space.

Table 6.4 summarizes the error rates on the test sets for all experiments. We observe that (1) basic methods like WNN and LDA are not particularly accurate for the task; (2) Monomer outperforms LMT considerably especially on the harder tasks, which demonstrates its ability to handle textual features; and (3) the comparative hardness of co-purchase over co-browsing prediction now seems to be dependent on the dataset in question, presumably due to different semantics of the two link types, or different patterns of customer behavior, among different categories.

6.7 Conclusion

In this chapter, we introduced Monomer, a method to model heterogeneous relationships for item-to-item recommendation. We noted that existing methods for item-

[3] We adopted the implementation in Gensim (default parameters): https://radimrehurek.com/gensim/

Table 6.4: Test errors of the link prediction task using BoW features (5,000-d) on a variety of top-level categories of the *Amazon* dataset. For LMT, $K = 100$, while for Monomer, $K = 20$ and $N = 4$. Lower is better.

Dataset	Relationship	(a) WNN	(b) LDA	(c) LMT	(d) Monomer	% Imprv. (d) vs. (c)
Electronics	*co-purchase*	37.58%	36.67%	13.73%	**10.09%**	26%
	co-browsing	39.37%	26.60%	16.41%	**9.44%**	42%
Automotive	*co-purchase*	42.63%	38.57%	17.94%	**14.09%**	21%
	co-browsing	42.44%	34.37%	21.15%	**14.74%**	30%
Video Games	*co-purchase*	44.31%	42.55%	14.43%	**12.03%**	17%
	co-browsing	40.08%	33.22%	16.18%	**11.29%**	30%
Movies & TV	*co-purchase*	40.00%	23.51%	11.36%	**9.47%**	17%
	co-browsing	42.01%	26.63%	15.12%	**14.98%**	1%
Office Products	*co-purchase*	41.35%	39.05%	18.53%	**14.30%**	23%
	co-browsing	37.33%	28.13%	13.52%	**9.72%**	28%
Home & Kitchen	*co-purchase*	39.74%	31.91%	13.96%	**12.40%**	11%
	co-browsing	36.49%	23.09%	13.97%	**9.95%**	29%
Cell Phones	*co-purchase*	43.35%	42.73%	29.44%	**22.68%**	23%
	co-browsing	43.70%	34.30%	24.65%	**16.04%**	35%
Avg.		40.74%	32.95%	17.19%	**12.94%**	23.8%

to-item recommendation suffer from a few limitations when dealing with heterogeneous data, due mainly to their reliance on metricity or 'nearest-neighbor' type assumptions. To overcome these limitations, our method made use of *mixtures* of non-metric embeddings, which allows us to relax the identity and symmetry assumptions of existing metric-based methods. The proposed scalable approach generates diverse and cross-category recommendations effectively that capture more complex relationships than mere visual similarity. We showed quantitatively that Monomer is accurate at link prediction tasks using co-purchase and co-browsing dyads from *Amazon.com*, and qualitatively that it is able to generate diverse recommendations that are consistent with a particular visual style.

6.8 Acknowledgements

Chapter 6, in part, contains material as it appears in the *IEEE International Conference on Data Mining series*, 2016 ("Learning Compatibility Across Categories for Heterogeneous Item Recommendation," Ruining He, Charles Packer, and Julian McAuley). The dissertation author was the primary investigator and author of this paper.

Chapter 7

Conclusion

7.1 Summary of Contributions

In this dissertation, we presented our novel algorithms for modeling visual, sequential, and relational signals in recommender systems at large scale.

In terms of visual signals, we modeled visual appearance and its evolution as a key to gaining a deeper understanding of users' preferences. In particular, our models learn people's visual preferences and the temporal drifts of fashion from large corpus of product images, user feedback, and timestamps from *Amazon.com*. We found that low-rank structures on top of visual features extracted by *Convolutional Neural Networks* are particularly effective for modeling visual dimensions and fashion dynamics, quantitatively outperforming existing methods significantly. Qualitatively, we found that our model uncovers meaningful visual dimensions as well as captures their complex, non-linear evolution over the past decade.

In terms of sequential signals, we built three scalable algorithms for sequential recommendation. Our visually- and socially-aware *Markov Chains* are effective at modeling people's artistic preferences and predicting their sequential behavior on *Behance.net*,

in terms of both next-click prediction and next-appreciate prediction. We noted the ineffectiveness of existing sequential methods at modeling the third-order interactions associated with personalized transitioning behavior. As a result, we proposed two new methods, both of which achieved state-of-the-art performance on a wide range of large, real-world datasets. We qualitatively visualized our learned models and found that they capture reasonable dynamics.

In terms of relational signals, we presented new methods for modeling large-scale heterogeneous relationships for the item-to-item recommendation task. We demonstrated that existing methods are ineffective when dealing with heterogeneous data, mainly due to their reliance on metricity assumptions. Our mixtures of non-metric embeddings tackled this limitation by relaxing the identity and symmetry assumptions of existing metric-based methods. In addition to their superior quantitative performance at link prediction tasks, our methods generated diverse and cross-category recommendations effectively.

7.2 Future Directions

Besides the directions we have explored in this dissertation, we also see potential research opportunities that are challenging and could be explored in the future.

1. Handling *heterogeneous* user feedback. There are often multiple types of feedback in a system; items clicked, liked, bookmarked, shared, rated, purchased etc. by users could indicate different levels of preferences. For instance, items purchased by a user may suggest a higher level of preference than items clicked by her. Leveraging such heterogeneity could lead to enhanced prediction accuracy.

2. *Personalizing* recommender systems. As people are getting more and more familiar with, as well as more dependent on, recommender systems, there is an increasing

need from users to customize the system to make it behave in desired manners. A simple example would be that users provide rules to prevent certain types of items from being recommended. Many recommender systems are using parameters to balance diversity, freshness, popularity, etc. of the recommendations they make [3]; proper techniques shall be developed to take into account users' feedback toward both items and the recommender system itself.

3. Designing *new* recommender systems. Smart personal assistants like *Alexa* are gaining popularity in recent years, indicating a need of developing appropriate recommendation approaches that are suitable for making recommendations in interactive environments. In addition, technologies like *Virtual Reality* provide the opportunity of designing novel recommendation interfaces that are more interesting, engaging, and/or efficient for exploring large corpora of items.

In addition, we also see the importance of establishing standard datasets for evaluating different models and helping guide the healthy growth of the recommender systems community.

Bibliography

[1] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine learning*, 6:37–66, 1991.

[2] Yahya AlMurtadha, Md Nasir Sulaiman, Norwati Mustapha, and Nur Izura Udzir. Improved web page recommender system based on web usage mining. In *Proceedings of the International Conference on Computing and Informatics (ICOCI)*, pages 8–9, 2011.

[3] Xavier Amatriain and Justin Basilico. Past, present, and future of recommender systems: An industry perspective. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 211–214, 2016.

[4] Ramnath Balasubramanyan and William W. Cohen. Block-LDA: Jointly modeling entity-annotated text and entity-entity links. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pages 450–461, 2011.

[5] Robert M. Bell, Yehuda Koren, and Chris Volinsky. The bellkor solution to the netflix prize, 2007.

[6] Richard Bellman. On the approximation of curves by line segments using dynamic programming. *Communications of the ACM*, 4(6):284, 1961.

[7] James Bennett and Stan Lanning. The netflix prize. In *KDDCup*, pages 3–6, 2007.

[8] Daniel Billsus and Michael J. Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10(2):147–180, 2000.

[9] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 3(Jan):993–1022, 2003.

[10] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 2787–2795, 2013.

[11] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 43–52, 1998.

[12] Gustavo Carneiro, Antoni B. Chan, Pedro J. Moreno, and Nuno Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29(3):394–410, 2007.

[13] Allison JB Chaney, David M. Blei, and Tina Eliassi-Rad. A probabilistic model for using social networks in personalized item recommendation. In *Proceedings of the ACM conference on Recommender systems (RecSys)*, pages 43–50, 2015.

[14] Jonathan Chang and David M. Blei. Relational topic models for document networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 81–88, 2009.

[15] Soravit Changpinyo, Kuan Liu, and Fei Sha. Similarity component analysis. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 1511–1519, 2013.

[16] Sonny Han Seng Chee, Jiawei Han, and Ke Wang. Rectree: An efficient collaborative filtering method. In *Proceedings of the International Conference on Data Warehousing and Knowledge Discovery (DaWaK)*, pages 141–151, 2001.

[17] Shuo Chen, Josh L. Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 714–722, 2012.

[18] Weizhu Chen, Zhenghao Wang, and Jingren Zhou. Large-scale l-bfgs using mapreduce. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 1332–1340, 2014.

[19] Abhinandan Das, Mayur Datar, Ashutosh Garg, and ShyamSundar Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 271–280, 2007.

[20] Matthew Der and Lawrence Saul. Latent coincidence analysis: A hidden variable model for distance metric learning. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 3230–3238, 2012.

[21] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[22] Wei Di, Neel Sundaresan, Robinson Piramuthu, and Anurag Bhardwaj. Is a picture really worth a thousand words?-on the role of images in e-commerce. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, pages 633–642, 2014.

[23] Wei Di, Catherine Wah, Anurag Bhardwaj, Robinson Piramuthu, and Neel Sundaresan. Style finder: Fine-grained clothing style detection and retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 8–13, 2013.

[24] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 485–492, 2005.

[25] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei A. Efros. What makes paris look like paris? *Communications of the ACM*, 58(12):103–110, 2015.

[26] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 647–655, 2014.

[27] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. Personalized ranking metric embedding for next new poi recommendation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2069–2075, 2015.

[28] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Mymedialite: A free recommender system library. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 305–308, 2011.

[29] Gayatree Ganu, Noemie Elhadad, and Amélie Marian. Beyond the stars: Improving rating predictions using review text content. In *Proceedings of the International Workshop on the Web and Databases (WebDB)*, pages 1–6, 2009.

[30] James H. Gilkeson and Kristy Reynolds. Determinants of internet auction success and closing price: An exploratory study. *Psychology & Marketing*, 20(6):537–566, 2003.

[31] Sharon Givon and Victor Lavrenko. Predicting social-tags for cold start book recommendations. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 333–336, 2009.

[32] Anjan Goswami, Naren Chittar, and Chung H. Sung. A study on the impact of product images on user clicks for online shopping. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 45–46, 2011.

[33] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. Trustsvd: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 123–129, 2015.

[34] Jing He, Xin Li, Lejian Liao, Dandan Song, and William K. Cheung. Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 137–143, 2016.

[35] Qi He, Jian Pei, Daniel Kifer, Prasenjit Mitra, and Lee Giles. Context-aware citation recommendation. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 421–430, 2010.

[36] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (SIGCHI)*, pages 194–201, 1995.

[37] Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):89–115, 2004.

[38] Diane J. Hu, Rob Hall, and Josh Attenberg. Style in the long tail: Discovering unique interests with latent variable models in large scale social e-commerce. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 1640–1649, 2014.

[39] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 263–272, 2008.

[40] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[41] Vignesh Jagadeesh, Robinson Piramuthu, Anurag Bhardwaj, Wei Di, and Neel Sundaresan. Large scale visual recommendations from street fashion images. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 1925–1934, 2014.

[42] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 135–142, 2010.

[43] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(1):221–231, 2013.

[44] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia (MM)*, pages 675–678, 2014.

[45] Xin Jin, Jiebo Luo, Jie Yu, Gang Wang, Dhiraj Joshi, and Jiawei Han. Reinforced similarity integration in image-rich information networks. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 25(2):448–460, 2013.

[46] Santosh Kabbur, Xia Ning, and George Karypis. FISM: factored item similarity models for top-n recommender systems. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 659–667, 2013.

[47] Yannis Kalantidis, Lyndon Kennedy, and Li-Jia Li. Getting the look: clothing recognition and segmentation for automatic product suggestions in everyday photos. In *Proceedings of the ACM Conference on International Conference on Multimedia Retrieval (ICMR)*, pages 105–112, 2013.

[48] Sergey Karayev, Matthew Trentacoste, Helen Han, Aseem Agarwala, Trevor Darrell, Aaron Hertzmann, and Holger Winnemoeller. Recognizing image style. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 1–11, 2014.

[49] Ralf Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300, 2004.

[50] Noam Koenigstein, Gideon Dror, and Yehuda Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 165–172, 2011.

[51] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R Gordon, and John Riedl. Grouplens: applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[52] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 426–434, 2008.

[53] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[54] Yehuda Koren and Robert Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 77–118. 2011.

[55] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

[57] Artus Krohn-Grimberghe, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, pages 173–182, 2012.

[58] Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 331–339, 1995.

[59] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.

[60] Justin J. Levandoski, Mohamed Sarwat, Ahmed Eldawy, and Mohamed F. Mokbel. LARS: A location-aware recommender system. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pages 450–461, 2012.

[61] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 2181–2187, 2015.

[62] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[63] Guang Ling, Michael R. Lyu, and Irwin King. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 105–112, 2014.

[64] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 194–200, 2016.

[65] Si Liu, Jiashi Feng, Zheng Song, Tianzhu Zhang, Hanqing Lu, Changsheng Xu, and Shuicheng Yan. Hi, magic closet, tell me what to wear! In *Proceedings of the ACM International Conference on Multimedia (MM)*, pages 619–628, 2012.

[66] Yang Liu, Wenping Wang, Bruno Lévy, Feng Sun, Dong-Ming Yan, Lin Lu, and Chenglei Yang. On centroidal voronoi tessellation—energy smoothness and fast computation. *ACM Transactions on Graphics (ToG)*, 28(4):101, 2009.

[67] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[68] Xin Lu, Zhe Lin, Hailin Jin, Jianchao Yang, and James Zijun Wang. RAPID: rating pictorial aesthetics using deep learning. In *Proceedings of the ACM International Conference on Multimedia (MM)*, pages 457–466, 2014.

[69] Yang Lu, Jing He, Dongdong Shan, and Hongfei Yan. Recommending citations with translation model. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2017–2020, 2011.

[70] Yue Lu, Malu Castellanos, Umeshwar Dayal, and ChengXiang Zhai. Automatic construction of a context-aware sentiment lexicon: an optimization approach. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 347–356, 2011.

[71] Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 203–210, 2009.

[72] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. SoRec: social recommendation using probabilistic matrix factorization. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 931–940, 2008.

[73] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, pages 287–296, 2011.

[74] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 165–172, 2013.

[75] Julian McAuley, Rahul Pandey, and Jure Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 785–794, 2015.

[76] Julian J. McAuley and Jure Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 897–908, 2013.

[77] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 43–52, 2015.

[78] Aditya Krishna Menon and Charles Elkan. Link prediction via matrix factorization. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery (ECML PKDD)*, pages 437–452, 2011.

[79] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119, 2013.

[80] Ron Milo, Nadav Kashtan, Shalev Itzkovitz, Mark EJ Newman, and Uri Alon. On the uniform generation of random graphs with prescribed degree sequences. *Arxiv preprint cond-mat/0312028*, 2003.

[81] Koji Miyahara and Michael J. Pazzani. Collaborative filtering with the simple bayesian classifier. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, pages 679–689. 2000.

[82] Bamshad Mobasher, Honghua Dai, Tao Luo, and Miki Nakagawa. Using sequential and non-sequential patterns in predictive web usage mining tasks. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 669–672, 2002.

[83] Joshua L. Moore, Shuo Chen, Douglas Turnbull, and Thorsten Joachims. Taste over time: The temporal dynamics of user preferences. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 401–406, 2013.

[84] Ana C. Murillo, Iljung S. Kwak, Lubomir Bourdev, David Kriegman, and Serge Belongie. Urban tribes: Analyzing group photos from a social perspective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 28–35, 2012.

[85] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 809–816, 2011.

[86] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 271–280, 2012.

[87] Xia Ning and George Karypis. SLIM: Sparse linear methods for top-n recommender systems. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 497–506, 2011.

[88] Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J. Wright. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Proceedings of*

the Advances in Neural Information Processing Systems (NIPS), pages 693–701, 2011.

[89] Woojin Paik, Sibel Yilmazel, Eric Brown, Maryjane Poulin, Stephane Dubon, and Christophe Amice. Applying natural language processing (nlp) based metadata extraction to automatically acquire user preferences. In *Proceedings of the International Conference on Knowledge Capture (ICKC)*, pages 116–122, 2001.

[90] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 502–511, 2008.

[91] Weike Pan and Li Chen. GBPR: Group preference based bayesian personalized ranking for one-class collaborative filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2691–2697, 2013.

[92] Ulrich Paquet and Noam Koenigstein. One-class collaborative filtering with random graphs. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 999–1008, 2013.

[93] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *KDDCup*, pages 5–8, 2007.

[94] Michael Pazzani and Daniel Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine learning*, 27(3):313–331, 1997.

[95] Zhi Qiao, Peng Zhang, Yanan Cao, Chuan Zhou, Li Guo, and Binxing Fang. Combining heterogenous social and geographical information for event recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 145–151, 2014.

[96] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 806–813, 2014.

[97] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 452–461, 2009.

[98] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 811–820, 2010.

[99] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul Kantor. *Recommender systems handbook*. Springer US, 2011.

[100] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2014.

[101] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 791–798, 2007.

[102] Guy Shani, David Heckerman, and Ronen I. Brafman. An MDP-based recommender system. *Journal of Machine Learning Research (JMLR)*, 6(9):1265–1295, 2005.

[103] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating word of mouth. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (SIGCHI)*, pages 210–217, 1995.

[104] Abhinav Shrivastava, Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. Data-driven visual similarity for cross-domain image matching. *ACM Transactions on Graphics (TOG)*, 30(6):154, 2011.

[105] Edgar Simo-Serra, Sanja Fidler, Francesc Moreno-Noguer, and Raquel Urtasun. Neuroaesthetics in fashion: Modeling the perception of fashionability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 869–877, 2015.

[106] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[107] Ajit P. Singh and Geoffrey J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 650–658, 2008.

[108] Malcolm Slaney, Kilian Weinberger, and William White. Learning a metric for music similarity. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 313–318, 2008.

[109] Damiano Spina and Julio Gonzalo. Learning similarity functions for topic detection in online reputation monitoring. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 527–536, 2014.

[110] David Stern, Ralf Herbrich, and Thore Graepel. Matchbox: large scale online bayesian recommendations. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 111–120, 2009.

[111] Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. Exploiting homophily effect for trust prediction. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, pages 53–62, 2013.

[112] Jiliang Tang, Xia Hu, and Huan Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, 2013.

[113] Xuewei Tang, Xiaojun Wan, and Xun Zhang. Cross-language context-aware citation recommendation in scientific articles. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 817–826, 2014.

[114] Lorenzo Torresani and Kuang chih Lee. Large margin component analysis. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1385–1392, 2007.

[115] Alexey Tsymbal. The problem of concept drift: definitions and related work. *Technical Report*, 2004.

[116] Lyle H. Ungar and Dean P. Foster. Clustering methods for collaborative filtering. In *Proceedings of the AAAI Conference on Artificial Intelligence Workshops (AAAI Workshops)*, pages 114–129, 1998.

[117] Laurens van der Maaten. Accelerating t-sne using tree-based algorithms. *Journal of Machine Learning Research (JMLR)*, 15(1):3221–3245, 2014.

[118] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4642–4650, 2015.

[119] Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 448–456, 2011.

[120] Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ACM SIGKDD Conferences on Knowledge Discovery and Data Mining (SIGKDD)*, pages 226–235, 2003.

[121] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 403–412, 2015.

[122] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 1112–1119, 2014.

[123] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 347–354, 2005.

[124] Xiang Wu, Qi Liu, Enhong Chen, Liang He, Jingsong Lv, Can Cao, and Guoping Hu. Personalized next-song recommendation in online karaokes. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 137–140, 2013.

[125] Hao Xia, Pengcheng Wu, Steven Hoi, and Rong Jin. Boosting multi-kernel locality-sensitive hashing for scalable image retrieval. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 55–64, 2012.

[126] Kota Yamaguchi, M. Hadi Kiapour, and Tamara L. Berg. Paper doll parsing: Retrieving similar styles to parse clothing items. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3519–3526, 2013.

[127] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. Beyond clicks: dwell time for personalization. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, pages 113–120, 2014.

[128] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 83–92, 2014.

[129] Tong Zhao, Julian McAuley, and Irwin King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 261–270, 2014.

[130] Jiaqian Zheng, Xiaoyuan Wu, Junyu Niu, and Alvaro Bolivar. Substitutes or complements: another step forward in recommendations. In *Proceedings of the ACM conference on Electronic Commerce (EC)*, pages 139–146, 2009.

[131] Andrew Zimdars, David Maxwell Chickering, and Christopher Meek. Using temporal data for making recommendations. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 580–588, 2001.