6-2018

Machine Learning Models for Context-Aware Recommender Systems

Yogesh Jhamb
Santa Clara University, yjhamb@scu.edu

Santa Clara University
Department of Computer Engineering
500 El Camino Real
Santa Clara, CA 95030

Machine Learning Models for Context-Aware Recommender Systems

Yogesh Jhamb

June 2018

Thesis Advisor

Prof. Yi Fang

Thesis Committee

Prof. Silvia Figueira

Prof. Weijia Shang

Prof. Nicholas Tran

Prof. Yuhong Liu

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Engineering of the Santa Clara University.

Abstract

The mass adoption of the internet has resulted in the exponential growth of products and services on the world wide web. An individual consumer, faced with this data deluge, is expected to make reasonable choices saving time and money. Organizations are facing increased competition, and they are looking for innovative ways to increase revenue and customer loyalty. A business wants to target the right product or service to an individual consumer, and this drives personalized recommendation. Recommender systems, designed to provide personalized recommendations, initially focused only on the user-item interaction. However, these systems evolved to provide a context-aware recommendations. Context-aware recommender systems utilize additional context, such as *genre* for movie recommendation, while recommending items to users. Latent factor methods have been a popular choice for recommender systems. With the resurgence of neural networks, there has also been a trend towards applying deep learning methods to recommender systems.

This study proposes a novel contextual latent factor model that is capable of utilizing the context from a dual-perspective of both users and items. The proposed model, known as the *Group-Aware Latent Factor Model (GLFM)*, is applied to the event recommendation task. The *GLFM* model is extensible, and it allows other contextual attributes to be easily be incorporated into the model. While latent-factor models have been extremely popular for recommender systems, they are unable to model the complex non-linear user-item relationships. This has resulted in the interest in applying deep learning methods to recommender systems. This study also proposes another novel method based on the denoising autoencoder architecture, which is referred to as the *Attentive Contextual Denoising Autoencoder (ACDA)*. The *ACDA* model augments the basic denoising autoencoder with a context-driven attention mechanism to provide personalized recommendation. The *ACDA* model is applied to the event and movie recommendation tasks.

The effectiveness of the proposed models is demonstrated against real-world datasets from *Meetup* and *Movielens*, and the results are compared against the current state-of-the-art baseline methods.

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my advisor and mentor, Prof. Yi Fang. I stand on the verge of a significant accomplishment in my life only because of Prof. Fang's guidance, support and encouragement. Prof. Fang's in-depth knowledge of the machine learning field has been instrumental in our successful publications, and he has been a great inspiration to me. It was a pleasure to work with Prof. Fang, and it was wonderful to have him as my thesis advisor.

My journey towards the successful completion of the Ph.D. program would not have been possible without the initial support from Prof. JoAnne Holliday and Prof. Ahmed Amer. I worked with both of them initially to find a viable research topic. They both were patient and extremely helpful in focusing my research efforts in the right direction. In fact, it was Prof. Holliday who suggested that I collaborate with Prof. Fang, and I am so glad that I did so.

I am also grateful to my committee members for their constructive feedback and guidance. I have been fortunate to have Prof. Figueira, Prof. Shang, Prof. Tran and Prof. Liu, who in spite of their busy schedules have been readily available to participate in the reviews and discussions.

I had the opportunity to work with a wonderful group of graduate students. I really enjoyed our group meetings, presentations and discussions. Working with my thesis advisor and the other students in our group has been a pleasure, and something that I will always cherish.

Finally, I would like to thank my wife, Hema, and my children, Nitya and Dhruv, for their unconditional love, encouragement, and most importantly for believing in me. As I embarked on this journey with a full-time job and other family commitments, it was my wife who motivated me to work hard and never quit. My children have always been loving, affectionate and understanding. I hope that they do bigger and better things than what I accomplish in my life time.

*To my wonderful children, **Nitya** and **Dhruv**.*

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The mass adoption of the Internet has resulted in the dawn of the big data era. There is abundance of data in every domain in the world today, and organizations are looking for ways to draw intelligence from this data. Consumers, inundated with this data deluge, are looking to make smart decisions based on the numerous choices that are available. Whether it is a movie a person would like to watch, a book that the individual wishes to read, or a place the person desires to visit, the consumer wants to make the best possible decision with a view towards making the most effective use of time and money. Organizations are looking to have an edge in the competitive marketplace, and therefore, they are looking to market their services and products in an intelligent way by recognizing the needs of the consumers.

Recommender systems aim to provide the intelligence that benefits both organizations and consumers. A business benefits in the form of increased revenue and customer loyalty, whereas consumers benefit by attaining the satisfaction of consuming the right product and services. First-generation recommender systems were rudimentary as they provided generic recommendations targeted towards all users. However, recommenders evolved with time to provide personalized recommendation targeted to a specific user by utilizing the known preferences of the user to predict the unknown preferences of that user on other items. Recommender systems are designed to perform either rating prediction, or top-N recommendation. While the goal of both methods is to provide personalized recommendations, they vary in the approach. Rating prediction works by considering explicit feedback data, such as users providing ratings for the Netflix movies. Top-N recommendation, on the other hand, is suitable for both explicit and implicit feedback systems. An implicit feedback system is one that attempts to gather the user's preference via implicit actions, such as viewing history, or mouse-clicks while navigating a website.

While earlier recommender systems focused solely on the user-item interaction, there is a growing trend of utilizing contextual data to provide a more meaningful personalized recommendation. A comprehensive study on contextual recommendations [ASST05] highlights the fact that the decision making by consumers is contingent on certain context, instead of being invariant of it. The same consumers may prefer different products or services under a different context. Therefore, accurate prediction of consumer preferences depends upon the relevant contextual information being incorporated into a recommendation method. The context augments the basic user-item interaction to achieve a higher quality of recommendation. The need to utilize the context for recommendation resulted from the observation that the user-item interaction never occurs in isolation, and there are additional factors that can be used to explain the interaction. For example, the *genre* is an important context for movie recommendation, and *time-of-day* and *geographic distance* are essential contextual attributes for point-of-interest recommendation.

Recommender systems primarily deal with two entities, *users* and *items*, with the preference or rating of the user u on item i being denoted by $r(u, i)$. The goal of the recommender system is then to use the observed ratings or preferences, which may be provided via explicit feedback or learnt via implicit actions, to predict the user's preference on other items. The predicted rating of the user u on an item is denoted by $\hat{r}(u, i)$. The recommender system learns to make predictions by learning the parameters that minimize the loss between the actual preference $[r(u, i)]$ and the predicted prefrence $[\hat{r}(u, i)]$.

Recommender systems can broadly be categorized into the following types:

- Collaborative Filtering Recommender Systems

- Content-Based Recommender Systems
- Hybrid Recommender Systems

Collaborative filtering (CF) has been been the most popular method for recommender systems. The CF method can further be classified into *Neighborhood-based* methods and *Model-based* methods. Neighborhood-based methods, such as *UserKNN* and *ItemKNN*, rely on a similarity measure to find the nearest neighbors. *UserKNN* predicts the preference of a user based on the similarity of that user with its k nearest users. *ItemKNN*, on the other hand, performs the prediction based on the similarity of the user's past item preference with the k nearest items. While the neighborhood methods are easier to implement, their performance degrades with sparse datasets when the similarity is hard to determine based on limited historical preferences of a user or item. This problem is particularly severe in cold-start conditions, which is caused by new users and items with no prior history of preferences. Model-based methods utilize machine learning methods to model the user-item interaction. There are many model-based techniques that have been used for recommendation, such as the bayesian models, clustering, regression, matrix factorization based latent factor models, and in recent times, deep learning models. Since model-based methods don't rely on the nearest neighbors, they are better at handling sparse datasets and the cold-start problem. Latent factor based models have been the most popular method for recommendation, and they are typically realized using matrix factorization. While latent factor models and other model-based methods have been prevalent in the past, they are limited to modeling the linear relationships between users and items. The resurgence in neural network architecture has resulted in deep learning methods being applied to recommender systems. The deep learning methods, which are also considered as model-based methods, are capable of modeling the complex non-linear relationships between users and items. While other machine learning methods require the model features to be identified explicitly, deep learning methods are capable of learning complex data representations without any prior feature knowledge.

Content-based recommenders utilize the content of the information describing the user or item, such as the user profile or item description, to perform the recommendation. Since the user profile and item description consist of textual data, the content-based methods utilize a measure such as *term frequency / inverse document frequency* (TF-IDF) to find the similarity between the content. While recommending an item to a user, the content-based recommender finds the similarity between the item's description and the description of other items that the user has preferred in the past based on the TF-IDF score. The recommender may also recommend based on the TF-IDF score of the user's profile and the item's description. For example, a content-based recommender may recommend a web page to the user based on the similarity of the content of that page with other pages that the user has visited. The similarity of the page content may be computed using TF-IDF. Content-based recommenders work well with sparse datasets and do not suffer from the cold-start problem. However, they are limited by the amount of content available for providing a meaningful recommendation.

Hybrid algorithms combine the collaborative filtering and content-based approaches. These methods attempt to use the best of both worlds by incorporating the user-item interaction model, and including the content-based method to overcome the data sparsity and cold-start problems. The hybrid methods are difficult to implement, however, they have shown good performance on specific tasks and datasets.

1.2 Overview

The objective of this study is to propose two novel model-based methods for personalized recommendation:

- The first approach, known as *Group-Aware Latent Factor Model (GLFM)*, is a latent factor model realized using matrix factorization.
- The second method is a model based on the neural network architecture, which is referred to as the *Attentive Contextual Denoising Autoencoder (ACDA)*.

The proposed methods differ from existing work in the literature as they incorporate context in an innovative and novel way, which has not been previously done. This study expains both the models in detail, and applies the proposed models to recommendation tasks against real-world datasets. Both the proposed models are observed to perform better than the current state-of-the-art methods.

The *Group-Aware Latent Factor Model (GLFM)* is a context-aware *supervised* learning method that utilizes certain contextual attributes from a dual-perspective of users and items. The *GLFM* model is extensible as it allows additional contextual attributes to be incorporated to the basic implementation. This study applies the *GLFM* model to the task of event recommendation. Event recommendation has become popular with the advent of *Event-Based Social Networks (EBSNs)*, such as *Meetup*. EBSNs allows like-minded individuals to socialize and collaborate of topics of mutual interest by organizing real-world events. Users in an EBSN organize themselves into groups, and events are held at physical venues. Therefore, the *group* and *venue* are contextual attributes that can be considered for event recommendation. The *group* contextual attribute is considered from a dual-perspective–the user's perspective signifies the *group* as a topic of interest, whereas the event perspective signifies the *group* in terms of the organization style and logistics. The *GLFM* models incorporates the *group* from a dual-perspective by modeling it as two different latent representations, one from the user's perspective, and the other from the event's perspective. The *GLFM* model is also extended by incorporating other contextual attributes such as *venue*, *event popularity*, *time-of-day* and *geographical distance*. The contextual attributes are either represented by their respective latent representation, or modeled as a bias.

Users in a EBSN express their interest in attending an event by providing a RSVP [1] for an event. Users typically respond to events that they are interested in attending, and ignore the others. Therefore, the dataset contains more positive preferences as compared to negative/unobserved preferences. The *GLFM* model uses pairwise learning to account for unobserved preferences. The objective function is setup to maximize the loss based on the diffrence in score of a positive-negative preference pair. The parameters of the model are estimated using *Stochastic Gradient Descent (SGD)*. The *GLFM* effectively handles the cold-start condition that arises from new users and new events. There are always new users that join the EBSN, and events are always

[1] RSVP is a French expression, which means "please respond"

short-lived. In the absence of historical preferences of the user or event, the recommendation task becomes challenging. The *GLFM* model utilizes the contextual parameters in the absence of the historical user / event data to perform the recommendation. Extensive experiments are performed on the GLFM model and its contextually-aware variants, and the results are compared with other state-of-the-art recommenders. The results demonstrate that the *GLFM* model performs better than the other methods on both the regular and cold-start experimental settings. The *GLFM* model is explained in detail in chapter 3.

The second method proposed in this study is the *Attentive Contextual Denoising Autoencoder (ACDA)* model. The *ACDA* is an *unsupervised* learning method based on a neural network architecture. Neural networks have experienced a resurgence with surge in big data and distributed computing. Deep learning methods based on neural networks have been primarily applied to tasks related to computer vision and natural language processing. However, there is recent interest in applying deep learning models to recommender systems. The *ACDA* is based on the denoising autoencoder neural network architecture, which is augmented with a context-driven attention mechanism.

Autoencoders [GBC16] are unsupervised feed-forward neural networks that learn a representation of the data that is of much lower dimensionality than the input. The network then attempts to recover the original input from this lower dimensionality representation at the output. These two steps are referred to as *encoding* and *decoding* respectively. Denoising autoencoders [GBC16] are a variant of the basic autoencoder that corrupt the input and attempt to recover it at the output with the objective of learning a more robust representation. The mapping of the input to a lower dimension representation is considered as projecting the data into a latent space. It has been shown in the literature that the denoising autoencoder architecture is a nonlinear generalization of latent factor models [KBV09, MS07]. The attention mechanism is utilized in neural network architectures to focus on certain parts of the input data in terms of the relevance. The *ACDA* model is based on the denoising autoencoder, and it applies the contextual attributes to the lower dimensional data representation via the attention mechanism. The partially corrupted user's preference on items is input into the *ACDA* model, which is then mapped to a lower dimensional representation in the hidden layer. The contextual attributes are applied to the lower dimensional representation via the attention mechanism, and the model then reconstructs the lower dimensional representation back to its original form at the output layer. The model is trained to minimize the loss between the corrupted input and its reconstructed form at the output layer.

The *ACDA* model is generic and it can be applied to both rating prediction and top-N recommendation tasks. This study applies the *ACDA* model to the top-N recommendation tasks of event and movie recommendation. Datasets from *Meetup* and *Movielens* are used for the event and movie recommendation tasks respectively. The performance of the proposed *ACDA* model is compared on these datasets against the state-of-the-art recommenders. The experimental results demonstrate the effectiveness of the *ACDA* model as compared to the other methods. The *ACDA* model is explained in chapter 4.

The main contributions of this study can be summarized as follows.

- This study proposes a novel extensible latent factor model, referred to as the *Group-Aware Latent Factor Model (GLFM)*. *GLFM* is realized using matrix factorization, and it incorporates contextual attributes from a dual-perspective for personalized recommendation. The model can be extended to add other contextual attributes that are related to only the user or item.

- The *GLFM* model is applied to the task of event recommendation, where it considers the *group* contextual attribute, which is associated with both the user and event, from a dual perspective.

- The *GLFM* model incorporates other context attributes, such as venue, event popularity, temporal influence and geographical distance. These contextual attributes are considered individually and grouped together to form variations of the *GLFM* model. The performance of the different *GLFM* variants is studied via extensive experiments.

- The *GLFM* model is demonstrated to be suitable for addressing the cold-start problem. Experimental results validate the suitability of the model for both regular and cold-start conditions.

- This study also proposes a generic model based on the neural network architecture, which is referred to as the *Attentive Contextual Denoising Autoencoder (ACDA)* model. The *ACDA* model is based on the denoising autoencoder architecture, which is augmented with a context-driven attention mechanism.

- The *ACDA* model is presented as a generic method for both rating prediction and top-N recommendation.

- The *ACDA* model is applied to the task of event recommendation by utilizing the *group* and *venue* as contextual attributes. The results, based on extensive experiments against the *Meetup* dataset, demonstrate the effectiveness of the approach against the other state-of-the-art recommenders.

- The *ACDA* model is also applied to the task of movie recommendation by utilizing the *genre* as a contextual attribute. Extensive experiments against the *Movielens* dataset demonstrate the superior performance of the proposed *ACDA* method as compared to the other state-of-the-art baseline methods.

1.3 Outline

The introduction section provides the motivation for this work, and it also provides a brief overview of the chapters that follow. The *GLFM* model is presented in chapter 3, and the *ACDA* model is presented in chapter 4. Chapters 3 and 4 are self-contained and may be reviewed independently. Chapter 2 outlines the current work in the literature related to this study, and chapter 5 concludes this study by providing future direction.

Chapter 2

Related Work

The objective of this study is to apply machine learning models to context-aware recommender systems with a focus towards event and movie recommendation. This section outlines the existing work in the literature related to this study.

2.1 Context-Aware Recommendation

There is a growing trend towards context-aware recommendations. A comprehensive study on contextual recommendations [ASST05] proposes a multidimensional recommendation model that extends the user-item interaction with contextual data. The proposed multidimensional model is similar to the OLAP-based models widely used in data warehousing applications related to databases. The proposed model is applied to movie recommendation considering contextual data such as, when the movie was seen, with whom and where. Karatzoglou et al. propose *Multiverse* recommendation model [KABO10], which is based on tensor factorization, a generalization of matrix factorization framework. The *Multiverse* model includes contextual data as additional dimensions of the data in the form of tensors. Contextual video recommendation is also addressed by a study [MYHL11], which proposes a contextual model based on multi-modal content relevance and user feedback.

Chen et al. propose a model for tweet recommendation that incorporates contextual attributes to improve the recommendation quality [CCZ+12]. The proposed method peforms the recommentation by modeling contextual attributes, such as the tweet topic level, user social relations, authority of the tweet author and the quality of the tweet. A study on contextual movie recommendation [SLH13] proposes a context-aware recommendation model that performs joint matrix factorization, combining the mood-specific movie similarity measure with the similarity measure that takes into account the movie plot keywords. Context is also applied while recommending services. Li et al. [LCLS10] model personalized recommendation of news articles as a contextual-bandit problem, where the model recommends articles to users based on the accompanying contextual information, while simultaneously adapting the article selection strategy based on the user click feedback. Contextual recommendation is also prevalent in music recommendation. A study [CZW+07] performs emotional allocation modeling by characterizing the mood of the user based on the web pages the user visited. This emotional context is used to recommend music to the user.

Contextual information is also predominant in event recommendation due to the availability of many contextual attributes, such as venue, time-of-day, event popularity and geographic distance. Du et al. [DYM+14] considered spatial and temporal context to predict event attendance. Macedo and Marinho [dMM14] conducted a large-scale analysis of several factors that impact user preferences on events. They observed that users tend to provide RSVPs close to the occurrence of the events. Macedo et al. [MMS15] further proposed a context-aware approach by exploiting various contextual information including social signals based on group memberships, location signals based on the users' geographical preferences, and temporal signals derived from the users' time preferences. Chen and Sun [CS16] proposed a social event recommendation method that exploits a user's social interaction relations and collaborative friendships. Zhang et al. [ZWF13] perform group recommendations for events by exploiting matrix factorization to model interactions between users and groups. By considering both explicit features (e.g., location and social features) and implicit patterns, the proposed approach demonstrated improved performance for group recommendations. A group recommender for movies is proposed based on content similarity and popularity [PN13]. A recent study [PLCZ15] proposed a general graph-based model to solve three recommendation tasks for event-based social networks in one framework, namely recommending groups to users, recommending tags to groups, and recommending events to users. The work models the rich information with a heterogeneous graph and considers the recommendation problem as a query-dependent node proximity problem. Another study [JL16] on event-based social networks, such as *Meetup*, investigates how social network, user profiles and geo-locations affect user participation when the social event is held by a single organizer. Lu et al. [LVT$^+$16] presented a system that extracts events from multiple data modalities and recommends events related to the user's ongoing search based on previously selected attribute values and dimensions of events being viewed.

The existing work in the literature signifies the importance of context-aware recommendation in every aspect of life. Context is utilized while recommending products, movies, services, social media and social events.

2.2 Cold-Start Problem

Cold-start is a prevalent problem in recommender systems as it is generally difficult for a model to handle new users and items. The cold-start problem is often alleviated by utilizing content information [FFCC13]. Word-based similarity methods [PB07] recommend items based on textual content similarity in word vector space. Collaborative Topic Regression (CTR) couples a matrix factorization model with probabilistic topic modeling to generalize to unseen items [WB11]. Collective matrix factorization (CMF) [SG08] simultaneously factorizes both rating matrix and content matrix with shared item latent factors. SVDFeature [CZL$^+$12] combines content features with collaborative filtering. The latent factors are integrated with user, item, and global features.

In [DYM$^+$14], topic modeling [BNJ03] is utilized to learn topics of users based on the content of their attended events, and then the similarity between topic factor of user and events is calculated, which is an important component of their method. Recently, Zhang and Wang [ZW15] explicitly addressed the cold-start problem in event recommendation by modeling the event content text. Liao and Chang [LC16] proposed a rough set based association rule approach. Sun et al. [SWCF15] integrated sentiment information from affective texts into recommendation models. The cold-start problem in tag recommendation is studied in [MBAG16].

The existing work in the literature signifies the importance of handling the cold-start condition. Both the models proposed in this study are suitable for handling the cold-start condition, and the *GLFM* model in particular is shown to effectively handle the cold-start problem.

2.3 Latent Factor Modeling

Latent factor models are among the most popular methods for recommender systems. Latent factor models, such as as matrix factorization [KBV09], probabilistic matrix factorization [MS07], and other variants [AC09, BKV07, Kor10, ZMK15, Cao15] demonstrated effectiveness in various recommendation tasks [SZW+12, HSL14, YSQ+15]. Among the various MF models proposed, SVD++ [Kor08] is one of the most widely used models. SVD++ integrates the implicit feedback information from a user to items, and the user latent factors are complemented by the latent factors of the items to which the user has provided explicit feedback.

Matrix factorization has been adapted to learn from relative pairwise preferences rather than absolute ones. One of the most effective techniques is based on Bayesian Personalized Ranking (BPR) [RFGST09], which has been shown to provide strong results in many recommendation tasks. There are several extensions of BPR, which include pairwise interaction tensor factorization [RST10], multi-relational matrix factorization[KGDFST12], richer interactions among users [PC13], and non-uniformly sampled items [GDFST12]. Other pairwise learning based collaborative filtering models include EigenRank [LY08] and probabilistic latent preference analysis [LZY09]. A pairwise ranking based geographical factorization method was recently proposed [LCL+15] for point-of-interest recommendation. This study also utilizes the pairwise ranking approach in the *GLFM* model for the event recommendation task.

2.4 Deep Learning for Recommender Systems

Recently, a surge of interest in applying deep learning to recommendation systems has emerged. Neural Matrix Factorization [HLZ+17a] addresses implicit feedback by jointly learning a matrix factorization and a feedforward neural network. [WYZ+17] unify the generative and discriminative methodologies under the generative adversarial network [GBC16] framework for item recommendation, and question answering. A recent survey [SK] provides a comprehensive overview of deep learning for recommender systems.

Autoencoders [GBC16] have been a popular choice of deep learning architecture for recommender systems. Specifically, denoising autoencoders [GBC16] are based on an unsupervised learning technique for learned representations that is robust to partial corruption of the input pattern [VLBM08]. This eventually led to denoising autoencoders being used for collaborative personalized recommenders. One of the early works that applied deep learning to recommender systems is based on the Restricted Boltzmann Machines (RBM) [SMH07]. The authors of the RBM study propose a method for rating prediction that uses *Contrastive Divergence* as the objective function to approximate the gradients. Wu et al. [WDZE16] propose a collaborative denoising autoencoder model that utilizes an additional input encoding for the user latent factor for recommendation based on implicit feedback. Chen et al. [CWSB14] introduced the marginalized denoising autoencoder model that offer a better performance by reducing the training time. The AutoRec model [SMSX15] for collaborative filtering proposes two variants: user-based (U-AutoRec) and item-based (I-AutoRec) denoising autoencoders that respectively take the partially observed user vector or item vector as input. The study evaluates both models on the Netflix dataset and concludes that the I-AutoRec performs better than the U-AutoRec model due to the high variance in the number of user ratings. An existing study proposes a hierarchical bayesian model called collaborative deep learning model (CDL) [WWY15], which is based on stacked denoising autoencoders. The CDL model tightly couples the deep representation learning of the content information and collaborative filtering for the ratings matrix in a unified model. Neural Collaborative Filtering [HLZ+17b] is another hybrid technique that combines matrix factorization and multi-layer perceptron to learn the user-item interactions.

Other forms of autoencoders have also been used for recommendation tasks. Li et al.[LS17] propose a variational autoencoder that learns the deep latent representation and the implicit relationalship between users and items from ratings and content data. AutoSVD++ [ZYX17] is a recent study that combines contrastive autoencoders and matrix factorization to provide recommendations based on content data and implicit user feedback. This study proposes the *ACDA* recommender model that incorporates the contextual information via the attention mechanism into the denoising autoencoder architecture.

2.5 Attention Mechanism

The attention mechanism has been widely adopted in deep learning for tasks related to image recognition and natural language processing [XBK+15, BCB14]. The significance of the attention mechanism has been highlighted in a study [CCB15], where the mechanism has been applied to structured output problems that involve multimedia content. However, there has been minimal work in the literature that applies the attention mechanism to recommender systems. The existing works utilize the attention mechanism for recommender systems are based primarily on the *Convolutional Neural Network (CNN)* and the *Recurrent Neural Network(RNN)* architectures. Sungyong et al. [SHYL17a] integrate a local and global attention mechanism with a *CNN* to model review text in the hopes of producing more interpretable recommendations. Likewise, [CZH+17] introduce item- and component-level attention to address multimedia collaborative filtering on implicit datasets. Factorization machines [Ren10] combine higher order pairwise interactions between features, but treat each feature with equal weight. Motivated by this idea, [XYH+17] learns to weigh the importance of each feature with an attention mechanism. The attention mechanism has also be used for hashtag recommendation using a *CNN* based model [GZ16]. Yet another study [SHYL17b] utilizes an attention-based *CNN* for personalized recommendation based on review text. The *ACDA* model integrates the attention mechanism with the denoising autoencoder architecture to process contextual data for personalized recommendation.

Chapter 3

Group-Aware Latent Factor Model (GLFM)

3.1 Background

Event recommender systems have gained prevalence with the advent of *Event-Based Social Networks (EBSNs)*. EBSNs, which allow like-minded people to gather together and socialize on a wide range of topics, have experienced increased popularity and rapid growth. Due to the huge volume of events available in EBSNs, event recommendation becomes essential for users to find suitable events to attend. *Meetup*[1], one of the largest EBSNs today, has over 24 million members, with approximately 200,000 groups in 181 countries. There are approximately 500,000 events organized every month on *Meetup*. The sheer volume of available events, especially in large cities, often undermines the users' ability to find the ones that best match their interests. Consequently, personalized event recommendation is essential for overcoming such an information overload.

Users of an EBSN indicate their interest to attend an event by responding to a RSVP[2] for the event. *Meetup* generates over 3 million RSVPs every month. The RSVP indicates a user's

[1] `http://www.meetup.com`

[2] RSVP is a French expression, which means "please respond"

preference on an event, and it allows future events to be recommended to the user. The events are hosted by groups at venues that are often in the vicinity of the local community. Such group structures provide additional context for event recommendation, and the context provided is unique as it can be used from a dual perspective: user-oriented and event-oriented. The user-oriented perspective regards a group as a topic of interest so that users associated with a group are interested in the same topic with the group. On the other hand, the event-oriented perspective views a group as an organizing entity. The events organized by the same group have the same organizing style such as logistics, event planning, structure, quality of talks, etc. These two perspectives complement each other and together they form a complete view of a group.

This study proposes a dual-perspective latent factor model for group-aware event recommendation by using two kinds of latent factors to model the dual effect of groups: one from the user-oriented perspective (e.g., topics of interest), and another from the event-oriented perspective (e.g., event planning and organization). Pairwise learning is used to exploit unobserved RSVPs by modeling the individual probability of preference via the *logistic* and *Probit sigmoid* functions. These latent group factors alleviate the cold-start problems, which are pervasive in event recommendation because events published in EBSNs are always in the future and many of them have little or no trace of historical attendance. The proposed model is flexible, and it can incorporate additional contextual information such as event venue, event popularity, temporal influence and geographical distance. A comprehensive set of experiments are conducted on four datasets from *Meetup* in both regular and cold-start settings. The results demonstrate that the proposed approach yields substantial improvement over the state-of-the-art baselines by utilizing the dual latent factors of groups. The proposed model utilizes pairwise ranking by taking unobserved RSVPs into account. In addition to the typical user and item latent factors, two novel latent factors are used to model a group: one for its user-oriented characteristics and another for its event-oriented characteristics. The influences of the groups on the user is then modeled as the linear combination of the latent factors for the user-oriented characteristics of its groups. The results also indicate that the performance can be further improved when incorporating factors associated with event venue, event popularity, temporal influence and geographical distance. It is worth noting that while adding more features helps, the group influence drives the most performance gain and it is the focus of this work.

Moreover, optimal use of group information can largely alleviate the cold-start problems, which are pervasive in the setting of event recommendation. New events and new users are constantly emerging in EBSNs. Many events published in EBSNs have little or no trace of prior attendance because the events are always in the future and they are often short-lived. Also, as EBSNs grow rapidly, there are many new users joining without record of historical attendance. By knowing the group that organizes the new event, one can expect the organizing style of the event based on the event-oriented perspective of groups. Similarly, by looking at the groups that the new user is associated with, one can infer the interests of the user based on the user-oriented perspective of groups. Therefore, this dual perspective of groups can help address both new item and new user cold-start problems.

In EBSNs, a user may RSVP for an event in the affirmative by a positive response (*"yes"*), or the user may provide a negative response to an event with a RSVP as (*"no"*). The numbers of positive responses and negative responses are largely disproportional. Many users just ignore RSVPs if they are not interested in attending the events. Therefore, it is more desirable to treat event recommendation as the top-N ranking task [KKB16] than a binary rating prediction problem. On the other hand, the absence of a response does not necessarily mean that the user is not interested in the event. It may be that the user is not aware of the event, or that the user is unable to attend this event due to other conflicts. Thus, the event recommendation model needs to take into account not only the positive and negative RSVPs, but also the missing/unobserved RSVPs. Event recommendation is much less studied in the literature than traditional recommendation tasks, such as movie and book recommendations. The proposed dual-perspective group-aware latent factor model addresses the unique characteristics of event recommendation.

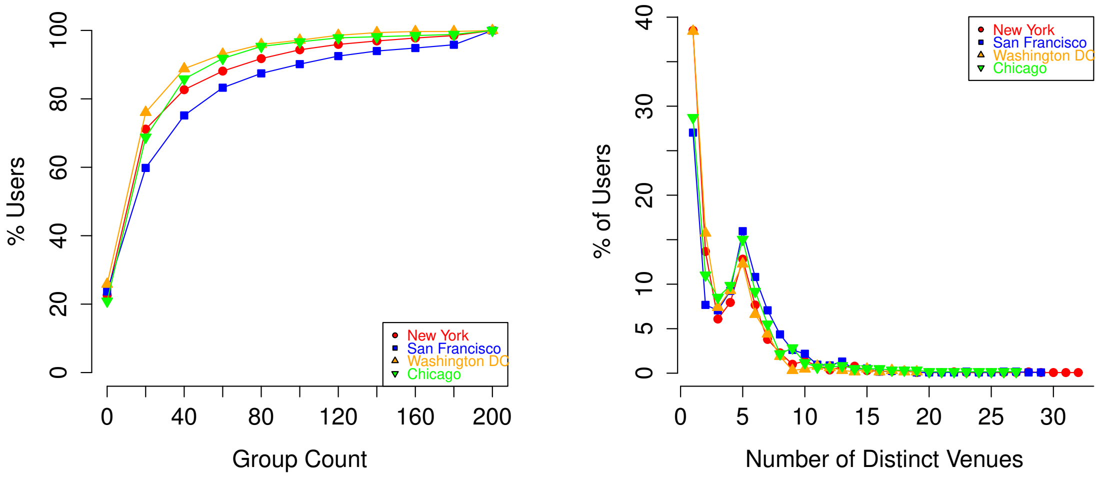

Figure 3.1: User-Group and User-Venue Relation

3.2 Data Analysis

This section analyzes the real-world datasets collected from *Meetup* for four cities in U.S.: New York, San Francisco, Washington DC and Chicago, which are among the most active cities for the *Meetup* community[3]. The detailed statistics and information of the datasets are presented in Section 3.4.1.

Figure 3.1 shows the cumulative percentage of users who join a given number of groups, i.e., number of groups (n) vs. the percentage of users joining less or equal than n groups. This information is depicted for all four cities and a consistent pattern emerges from the data. In all four cities, approximately 80% of the users have joined at least one group. Around 5% of the users join one group, and a significant majority join between 1 and 30 groups. There are certain users who have joined more than 30 groups, but the percentage of such users is very small. The group membership in the user-base is an important indication that group is an essential feature for event recommendation. Table 3.1 provides the average number of groups per user in the four cities. It is also observed from the dataset that in all four cities, there were a couple of users who had joined 200 groups, which is the maximum allowed by *Meetup*. There

[3] http://priceonomics.com/what-meetups-tell-us-about-america/

Table 3.1: GLFM - Average Number of Groups Per User in the Four Cities

New York	San Francisco	Washington DC	Chicago
9.54	9.88	8.63	7.78

is also a remarkable consistency in the average number of groups per user in the four cities, which is in the range of 7 - 10. It is also important to note that *all* events in the dataset are organized by a group, i.e. the group information is always present in the RSVP. Therefore, a group is a critical contextual parameter that is associated with the majority of users and *all* events.

The data collected for the four cities is also analyzed with respect to relationship between users and venues. As indicated in Figure 3.1, most users usually attend events at a limited number of venues, which means that there is an implicit relationship between the user and venue. Further insight into the user-venue relationship can be obtained by looking at a random user in the New York dataset. The user has three RSVPs that are for three different venues. The venues associated with the user's RSVPs are: *230 Fifth* (bar and lounge), *Madison Square Tavern* (restaurant and event center with a full-scale bar) and *Croton Reservoir Tavern* (upscale restaurant and bar that hosts private parties). The three venues associated with the user are similar based on the fact that all of them are upscale restaurants with bars, and two of them host private events. This observation indicates that the characteristics of the venue may affect the user's RSVP for the event, which is the motivatation behind including the event venue as one of the parameters into the proposed model in Section 3.3.3.

3.3 Event Recommendation Models

As discussed earlier, the task of event recommendation is treated as top-N item recommendation by providing a user with a ranked list of events. The proposed approach is based on the latent factor model with pairwise ranking. In the following subsections, the proposed model is presented by considering the dual role of group influence. Then additional contextual information is incorporated into the model, such as venue, event popularity, temporal influence and geographical distance. Table 3.2 lists the notations used in this paper.

Table 3.2: GLFM - Notations

m, n, f	Total number of users, events, and latent factors, respectively
G_u	The set of groups that user u belongs to
(u, i, j)	A preference triplet indicating user u prefers event i over event j
D_s	The set that contains all the preference triplets
K	The set of (u, i) pairs with known ratings
$s_{u,i}$	Ranking score of event i for user u
$x_{u,i,j}$	Difference of ranking scores between event i and j for user u
$\boldsymbol{p_u}$	Latent factors for user u
$\boldsymbol{q_i}$	Latent factors for event i
$\boldsymbol{r_g}$	User-oriented latent factor for group g
$\boldsymbol{t_g}$	Event-oriented latent factor for group g that organizes event
$\boldsymbol{v_i}$	Latent factor for venue that host event i
$\boldsymbol{y_i}$	Event-oriented latent factor related to the day of the week for event i
$\boldsymbol{z_i}$	Event-oriented latent factor related to the period of the day for event i
c_i	Popularity for event i
$d_{u,i}$	Normalized geo-distance between user u and event i
β_c, β_d	Event popularity bias and Geo-distance bias
λ, γ	Regularization parameters for latent factors and bias respectively

3.3.1 Pairwise Ranking

In EBSNs, most users only respond to a small portion of events since they may only be aware of a few events. In addition, there exist many more positive examples than negative ones. Many users just ignore RSVPs if they are not interested in attending the events. Consequently, there are many unobserved user-event pairs, which are a mixture of real negative feedback (the user is not interested in attending) and missing values (the user might attend if she is aware of the event). Therefore, instead of performing a point-wise RSVP prediction, the proposed model utilizes a pairwise ranking approach to learn the preferences of users on events.

Formally, given a user u, if item i is preferred over item j, then the preference instance $(u, i, j) \in D_S$, where D_S is the whole set of preference instances. In EBSNs, the preference instances can be derived from three types of relations between items given user u: 1) RSVP "Yes" is preferred over RSVP "No"; 2) RSVP "Yes" is preferred over unobserved RSVP; 3) unobserved RSVP of an event organized by the user's group is preferred over RSVP "No". Let $s(u, i)$ represent the ranking score of item i for user u and denote:

$$x(u,i,j) = s(u,i) - s(u,j)$$

The pairwise ranking optimization criterion is the log likelihood of the observed preferences, which can then be defined as

$$\max_{\Theta} \quad \mathcal{L}(\Theta) = \sum_{(u,i,j)\in D_S} \log \sigma\Big(x(u,i,j)\Big) - Reg(\Theta) \tag{3.1}$$

where $\sigma(x)$ defines the probability of pairwise preference, i.e., the probability of item i being preferred over j given their ranking score difference $x(u,i,j)$. $\sigma(x)$ is a monotonically increasing function with respective to the argument $x(u,i,j)$. The intuitive explanation of Eqn.(3.1) is that if item i is preferred over j for user u, the difference between their ranking scores $s(u,i)$ and $s(u,j)$ is maximized since $\log \sigma(x)$ is a monotonically increasing function. As a result, item i is more preferable than item j. In the above equation, Θ is the set of all model parameters and $Reg(\Theta)$ is a regularization term to prevent overfitting. The proposed model uses L2 regularization, since the L2-regularization terms are differentiable, allowing us to apply gradient-based optimization methods.

Since $\sigma(x)$ is a probability function while being monotonically increasing, the Logistic function defined as follows is a natural choice:

$$\sigma(x) = \frac{1}{1+\exp(-x)}$$

In fact, the choice of the Logistic function in Eqn.(3.1) would lead to the widely used Bayesian Personalized Ranking (BPR) optimization criterion [RFGST09] in recommender systems. The objective function of BPR is shown as Eqn.(3.3) in Table 3.3. Theoretically, optimizing for the above BPR is a smoothed version of optimizing for the well-known ranking measure Area under the ROC Curve (AUC) by approximating the non-differentiable Heaviside function by the differentiable Logistic function $\sigma(x)$. See [RFGST09] for a more detailed explanation. On the other hand, the use of logistic function to model pairwise preference probability is a type of Bradley-Terry models [AK14], where exponential score functions are used.

Alternatively, the pairwise preference probability $\sigma(x)$ can be modeled by the Probit function, which is a popular specification for an ordinal or a binary response model in Statistics [MN89]. The Logistic and Probit are both sigmoid functions with a domain between 0 and 1, which makes them both quantile functions - i.e., inverses of the cumulative distribution function (CDF) of a probability distribution. In fact, the Logistic is the quantile function of the Logistic distribution, while the Probit is the quantile function of the Gaussian distribution defined as follows:

$$\sigma(x) = \Phi(x) = \int_{-\infty}^{x} \mathcal{N}(x)\, dx = \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) dx \tag{3.2}$$

where $\Phi(x)$ is the cumulative distribution function of Gaussian distribution. $\mathcal{N}(x)$ is the probability density function of the Gaussian distribution. For the purpose of this study, the parameters are set as: $\mu = 0$ and $\sigma^2 = 1$, yielding the standard Gaussian distribution. Both Logistic and Probit functions have a similar 'S' shape. The Logistic has a slightly flatter tail, while the Probit curve approaches the axes more quickly. Increasing the variance in the probit function results in the curve becoming flatter and elongated. The experiments in Section 4.3 compare the performance of the two variants of the proposed model.

3.3.2 Group-Aware Latent Factor Model

The latent factor model is one of the most successful collaborative filtering models, which jointly maps the users and items into a shared latent space of a much lower dimensionality. This study utilizes the latent factor model to characterize the ranking scores $s(u, i)$ and $s(u, j)$ in Eqn.(3.1). Formally, users and events are projected into a shared f*-dimensional* latent space, where $f \ll min(m, n)$: m is the number of users and n is the number of events. In the most basic form, user u is mapped to a latent factor vector $\boldsymbol{p_u} \in \mathbb{R}^f$, and event i is mapped to a latent factor vector $\boldsymbol{q_i} \in \mathbb{R}^f$. The inner product of $\boldsymbol{p_u}$ and $\boldsymbol{q_i}$ is used to compute the predicted ranking score of user u on event i such as $s_{u,i} = \boldsymbol{p_u^T q_i}$. Similarly, we have $s_{u,j} = \boldsymbol{p_u^T q_j}$ for event j where $\boldsymbol{q_j}$ is the latent factor for j.

Table 3.3: GLFM- Objective functions $\mathcal{L}(\Theta)$ for BPR, GLFM, GLFM-V, GLFM-VPD, and GLFM-VPDT, respectively

$$\max_{P,Q} \sum_{(u,i,j)\in D_S} \log\sigma\Big(x(u,i,j)\Big) - \lambda\left(\sum_u \|\boldsymbol{p_u}\|_2^2 + \sum_i \|\boldsymbol{q_i}\|_2^2\right) \tag{3.3}$$

$$\max_{P,Q,R,T} \sum_{(u,i,j)\in D_S} \log\sigma\Big(x(u,i,j)\Big) - \lambda\left(\sum_u \|\boldsymbol{p_u}\|_2^2 + \sum_i \|\boldsymbol{q_i}\|_2^2 + \sum_g \|\boldsymbol{r_g}\|_2^2 + \sum_i \|\boldsymbol{t_g}\|_2^2\right) \tag{3.4}$$

$$\max_{P,Q,R,T} \sum_{(u,i,j)\in D_S} \log\sigma\Big(x(u,i,j)\Big) - \lambda\left(\sum_u \|\boldsymbol{p_u}\|_2^2 + \sum_i \|\boldsymbol{q_i}\|_2^2 + \sum_g \|\boldsymbol{r_g}\|_2^2 + \sum_i \|\boldsymbol{t_g}\|_2^2 + \sum_i \|\boldsymbol{v_i}\|_2^2\right) \tag{3.5}$$

$$\max_{P,Q,R,T} \sum_{(u,i,j)\in D_S} \log\sigma\Big(x(u,i,j)\Big) - \lambda\left(\sum_u \|\boldsymbol{p_u}\|_2^2 + \sum_i \|\boldsymbol{q_i}\|_2^2 + \sum_g \|\boldsymbol{r_g}\|_2^2 + \sum_i \|\boldsymbol{t_g}\|_2^2 + \sum_i \|\boldsymbol{v_i}\|_2^2\right) - \gamma\left({\beta_c}^2 + {\beta_d}^2\right) \tag{3.6}$$

$$\begin{aligned}\max_{P,Q,R,T,Y,Z} \sum_{(u,i,j)\in D_S} \log\sigma\Big(x(u,i,j)\Big) - \lambda\Big(&\sum_u \|\boldsymbol{p_u}\|_2^2 + \sum_i \|\boldsymbol{q_i}\|_2^2 + \sum_g \|\boldsymbol{r_g}\|_2^2 + \sum_i \|\boldsymbol{t_g}\|_2^2 + \sum_i \|\boldsymbol{v_i}\|_2^2 \\ &+ \sum_i \|\boldsymbol{y_i}\|_2^2 + \sum_i \|\boldsymbol{z_i}\|_2^2\Big) - \gamma\left({\beta_c}^2 + {\beta_d}^2\right)\end{aligned} \tag{3.7}$$

Based on the data analysis in Section 3, a large majority of users are associated with at least one group and each event is organized by one group. These observations suggest that considering the group influence may improve the accuracy of event recommendation. The group influence can be viewed from a dual perspective: user-oriented and event-oriented. The user-oriented perspective regards a group as a topic of interest, so that users associated with a group are interested in the same topic with the group. On the other hand, the event-oriented perspective views a group as an organizing entity. The events organized by the same group have the same organizing style such as logistics, event planning, structure, quality of talks, etc. These two perspectives complement each other and together they form a complete view of a group.

This study proposes a group-aware latent factor model (GLFM) to model user preference by encoding the dual perspective of group influence. Mathematically, for group g, $\boldsymbol{r_g}$ is used to model its user-oriented characteristics, and $\boldsymbol{t_g}$ is used to model its event-oriented characteristics. Since a user could be a member of multiple groups, an average all the user-oriented latent vectors

$\boldsymbol{r_g}$ of these groups is used to influence the user latent factor. Similarly, the event-oriented latent factor $\boldsymbol{t_g}$ of the group that organizes the event is used to influence the event latent factor. Let G_u be the set of groups that user u belongs to. Let $g \in G_u$ be a specific group that includes user u. Let $\boldsymbol{t_g}$ denote the latent factor of the group that organizes event i. Incorporated with influence from groups, the predicted ranking score for event i given user u is now computed with both $\boldsymbol{r_g}$ and $\boldsymbol{t_g}$, shown as follows.

$$s_{u,i} = \Big(\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}\Big)^T \Big(\boldsymbol{q_i} + \boldsymbol{t_g}\Big) \tag{3.8}$$

The ranking score $s_{u,j}$ for event j given user u can be similarly calculated. The objective function is shown as Eqn.(3.4) in Table 3.3.

It is worth noting that by considering the group information, GLFM addresses the cold-start problems for both new events and new users that do not appear in training data. When a new event i is just released in an EBSN, there is no information about $\boldsymbol{q_i}$, but the event-oriented group latent factor $\boldsymbol{t_g}$ is not empty since the group that organized the event is known. Intuitively, if the group has an excellent track record of organizing events like having great talks and good event planning, users may prefer the events organized by this group. Similarly, when a new user u has not responded to any RSVPs, the latent factor $\boldsymbol{p_u}$ is not present, but we may know what groups she is associated with and thus can use $\frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}$ for prediction/ranking. These latent factors capture the user-oriented characteristics of the groups such as topics of interest. It is important to view $\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}$ as a kind of smoothed version of user latent factor smoothed by the groups that the user belongs to. The group influence serves as the background model and is crucial when $\boldsymbol{p_u}$ is empty. Similarly, $\boldsymbol{q_i} + \boldsymbol{t_g}$ can be viewed as the smoothed version of event latent factor. With these latent group factors, we can tackle new users and new items.

3.3.3 Event Venue

Each event is held at a venue. Some groups often organize events at the same or a similar venue, indicating a correlation between the event group and the event venue. The event venue may affect the attendance of the event. For example, some venues have a great facility, or they are at a convenient location, which can attract more people in general. Some venues can accommodate special needs of certain users such as being pets or kids friendly. Some venues are specialized for certain types of events such as ballroom dance or tennis games.

This study introduces the venue latent factor to exploit event venues for more accurate event recommendation. The venue is treated as an attribute of the event, and the model is augmented with a latent factor $\boldsymbol{v_i}$ for the venue that hosts event i. The model that includes the influence of venue is GLFM-V. By incorporating the venue influence, the predicted ranking score of event i given user u is now defined as

$$s_{u,i} = \Big(\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}\Big)^T \Big(\boldsymbol{q_i} + \boldsymbol{t_g} + \boldsymbol{v_i}\Big) \tag{3.9}$$

The objective function is shown as Eqn.(3.5) in Table 3.3.

3.3.4 Event Popularity and Geographical Distance

In EBSNs, some events have general themes such as entrepreneurship, while others have niche topics such as *Minecraft*. A hypothesis is users may be more likely to RSVP on popular/mainstream events than on unpopular/niche events. The event popularity may be measured by the number of people who RSVP for the event. An event that has a higher number of RSVPs is considered to be more popular. The model includes event popularity as a ranking bias and perform feature scaling while considering this in conjunction with the other features. By incorporating the popularity bias, the predicted ranking score of event i given user u is as follows:

$$s_{u,i} = \Big(\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}\Big)^T \Big(\boldsymbol{q_i} + \boldsymbol{t_g} + \boldsymbol{v_i}\Big) + \beta_c c_i \tag{3.10}$$

where c_i is the popularity bias for event i and β_c is the weight of the bias, which is learned from the training data.

Geographical distance is another important consideration while recommending products and services that require the user to travel to the location. The geographical distance is incorporated into the model by computing the Haversine distance[4] from the user latitude-longitude and venue latitude-longitude data. The logarithm of this distance is computed, and it is modeled as a ranking bias.

$$s_{u,i} = \Big(\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}\Big)^T \Big(\boldsymbol{q_i} + \boldsymbol{t_g} + \boldsymbol{v_i}\Big) + \beta_c c_i + \beta_u d_{ui} \tag{3.11}$$

where $d_{u,i}$ is the normalized logarithm geo-distance between user u and venue that hosts event i, and β_u is the geo-distance bias parameter associated with the user that is learned from the training data. The objective function is provided in Eqn.(3.6) in Table 3.3. In the equation, γ is the regularization parameter used to prevent overfitting. The model that augments the group and venue latent factors with the event popularity and geo-distance bias is called GLFM-VPD.

3.3.5 Temporal Influence

Events are organized during certain days of the week and at certain times of the day. Some events are organized in the day between 9 am and 5 pm, whereas others are organized in the evening after 5 pm, so people can attend after work. Events that are targeted towards working individuals are generally organized on the weekends. The temporal influence is added into the model using two types of latent time factors: one for the day of the week, $\boldsymbol{y_i}$, which is associated with the event i, and another for the period of the day, $\boldsymbol{z_i}$, for event i. The day of the week is derived from weekday that the event is scheduled on, whereas the period of the day is mapped to one of two time-slots: "Day" if the event time is between 9 am and 5 pm, and "Evening" is the event is scheduled after 5 pm.

With the inclusion of the temporal influence parameters, the predicted ranking score of event

[4] `https://en.wikipedia.org/wiki/Haversine_formula`

i given user u is now defined as

$$s_{u,i} = \Big(\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}\Big)^T \Big(\boldsymbol{q_i} + \boldsymbol{t_g} + \boldsymbol{v_i} + \boldsymbol{y_i} + \boldsymbol{z_i}\Big) + \beta_c c_i + \beta_u d_{ui} \tag{3.12}$$

where the y_i parameter that models the influence of the day of the week, and the z_i models the influence of the period of the day. The objective function is provided in Eqn.(3.7) in Table 3.3 with the model denoted by GLFM-VPDT.

3.3.6 Parameter Estimation

The parameters of the proposed models are estimated using stochastic gradient descent (SGD) algorithm [Bot10]. In this case, an update is performed for each preference instance $(u, i, j) \in D_s$. Since this is a maximization problem, the parameters are learned by moving in the direction of the gradient with a learning rate α in an iterative manner as follows.

$$\Theta \leftarrow \Theta - \alpha \frac{\partial \mathcal{L}}{\partial \Theta} \tag{3.13}$$

By plugging our pairwise ranking optimization criterion in Eqn.(3.1) into Eqn.(3.13), we obtain

$$\Theta \leftarrow \Theta - \alpha \Bigg(\frac{1}{\sigma\Big(x(u,i,j)\Big)} \frac{\partial \sigma\Big(x(u,i,j)\Big)}{\partial \Theta} - \frac{\partial Reg(\Theta)}{\partial \Theta} \Bigg) \tag{3.14}$$

The algorithm repeatedly iterates over the training data and updates the model parameters in each iteration until convergence. Based on Eqn.(3.14), the derived SGD updates for GLFM-VPDT are shown in Table 3.4. The updates for other proposed latent factor models (e.g., GLFM, GLFM-V, GLFM-VPD) are similar. In the table, $\omega_{u,i,j}$ is defined in order to simplify the notation. For the model based on the Logistic function,

$$\omega_{u,i,j} = \frac{e^{-x(u,i,j)}}{1 + e^{-x(u,i,j)}}$$

Table 3.4: GLFM - Stochastic gradient descent updates for GLFM-VPDT

$$\boldsymbol{p_u} \leftarrow \boldsymbol{p_u} + \alpha \cdot \Big(\omega_{u,i,j} \cdot \big((\boldsymbol{q_i} + \boldsymbol{t_{g(i)}} + \boldsymbol{v_i} + \boldsymbol{y_i} + \boldsymbol{z_i}) - (\boldsymbol{q_j} + \boldsymbol{t_{g(j)}} + \boldsymbol{v_j} + \boldsymbol{y_j} + \boldsymbol{z_j})\big) - \lambda \cdot \boldsymbol{p_u}\Big)$$

$$\forall g \in G_u : \boldsymbol{r_g} \leftarrow \boldsymbol{r_g} + \alpha \cdot \Big(\omega_{u,i,j} \cdot \big((\boldsymbol{q_i} + \boldsymbol{t_{g(i)}} + \boldsymbol{v_i} + \boldsymbol{y_i} + \boldsymbol{z_i}) - (\boldsymbol{q_j} + \boldsymbol{t_{g(j)}} + \boldsymbol{v_j} + \boldsymbol{y_j} + \boldsymbol{z_j})\big) - \lambda \cdot \boldsymbol{r_g}\Big)$$

$$\boldsymbol{q_i} \leftarrow \boldsymbol{q_i} + \alpha \cdot \left(\omega_{u,i,j} \cdot (\boldsymbol{p_u} + \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{q_i}\right)$$

$$\boldsymbol{q_j} \leftarrow \boldsymbol{q_j} + \alpha \cdot \left(\omega_{u,i,j} \cdot (-\boldsymbol{p_u} - \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{q_j}\right)$$

$$\boldsymbol{t_{g(i)}} \leftarrow \boldsymbol{t_{g(i)}} + \alpha \cdot \left(\omega_{u,i,j} \cdot (\boldsymbol{p_u} + \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{t_{g(i)}}\right)$$

$$\boldsymbol{t_{g(j)}} \leftarrow \boldsymbol{t_{g(j)}} + \alpha \cdot \left(\omega_{u,i,j} \cdot (-\boldsymbol{p_u} - \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{t_{g(j)}}\right)$$

$$\boldsymbol{v_i} \leftarrow \boldsymbol{v_i} + \alpha \cdot \left(\omega_{u,i,j} \cdot (\boldsymbol{p_u} + \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{v_i}\right)$$

$$\boldsymbol{v_j} \leftarrow \boldsymbol{v_j} + \alpha \cdot \left(\omega_{u,i,j} \cdot (-\boldsymbol{p_u} - \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{v_j}\right)$$

$$\boldsymbol{y_i} \leftarrow \boldsymbol{y_i} + \alpha \cdot \left(\omega_{u,i,j} \cdot (\boldsymbol{p_u} + \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{y_i}\right)$$

$$\boldsymbol{y_j} \leftarrow \boldsymbol{y_j} + \alpha \cdot \left(\omega_{u,i,j} \cdot (-\boldsymbol{p_u} - \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{y_j}\right)$$

$$\boldsymbol{z_i} \leftarrow \boldsymbol{z_i} + \alpha \cdot \left(\omega_{u,i,j} \cdot (\boldsymbol{p_u} + \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{z_i}\right)$$

$$\boldsymbol{z_j} \leftarrow \boldsymbol{z_j} + \alpha \cdot \left(\omega_{u,i,j} \cdot (-\boldsymbol{p_u} - \sum_{ug} \boldsymbol{r_g}) - \lambda \cdot \boldsymbol{z_j}\right)$$

for i, $\beta_c \leftarrow \beta_c + \alpha \cdot (\omega_{u,i,j} \cdot (c_i + c_j) - \gamma \cdot \beta_c)$
$\beta_d \leftarrow \beta_d + \alpha \cdot (\omega_{u,i,j} \cdot (d_{ui} + d_{uj}) - \gamma \cdot \beta_d)$
for j, $\beta_c \leftarrow \beta_c + \alpha \cdot (\omega_{u,i,j} \cdot (-c_i - c_j) - \gamma \cdot \beta_c)$
$\beta_d \leftarrow \beta_d + \alpha \cdot (\omega_{u,i,j} \cdot (-d_{ui} - d_{uj}) - \gamma \cdot \beta_d)$

For the model based on the Probit function:

$$\omega_{u,i,j} = \frac{\mathcal{N}\big(x(u,i,j)\big)}{\Phi\big(x(u,i,j)\big)}$$

where $\mathcal{N}(\cdot)$ and $\Phi(\cdot)$ are defined in Eqn.(3.2).

As introduced in Section 3.3.1, the preference instances can be derived from three types of relations between items given a user based on RSVP "Yes", RSVP "No", and missing RSVP. Thus, the preference instances (u, i, j) is generated from the training data based on the following strategy:

- If the user has positive RSVPs, then a "Yes" RSVP is randomly sampled along with a randomly sampled "No" RSVP from the same user to form the preference pair.
- If there is no negative RSVP for the user, then a missing RSVP from the user is randomly sampled. This pairing is based on the assumption that a RSVP with unknown preference is negative when paired with a true positive example.
- If the user has no positive RSVP, then a random unknown RSVP of an event organized by one of the user's groups is paired with a random negative RSVP from the same user. This pairing is based on the assumption that an unknown preference for a RSVP of an event organized by a group that the user belongs to, is preferred over a true negative example.

Section 3.4.3 investigates an alternative preference generation strategy without assuming that the unobserved RSVPs are preferred over the observed RSVP "No". Once a sufficient number of preference instances are sampled, the data is randomly shuffled to avoid bias for certain users. The model is then trained on these permuted instances by SGD. The learned model parameters are then applied to the test users for the top-N event recommendation based on descending order of the ranking score $s_{u,i}$.

3.4 Experiments

The proposed dual perspective group-aware model and its variants are evaluated on real-world datasets collected from *Meetup*. The results of the proposed models are compared against the state-of-the-art recommendation techniques. In addition to performing a comparison in regular settings, a comparison is also made with the baseline methods in cold-start scenarios. The results are presented in this section, and the findings are discussed in detail.

3.4.1 Data Collection

As introduced in Section 3.2, the RSVP data is collected from *Meetup* for events organized in four cities in the U.S.: New York, San Francisco, Washington DC and Chicago. The RSVP data was collected by using the Meetup API[5] between the time periods January 2016 and May 2016. The dataset was filtered for each city to retain only RSVPs associated with users having greater than 5 RSVPs. The statistics of the data are given in Table 4.1. These four cities represent different geographic regions of the U.S. and they have varied statistics as shown in the table.

Table 3.5: GLFM - Data Statistics

City	RSVPs	Sparsity	Positive	Negative	Users	Events	Groups	Venues
New York	50,150	0.9989	49,163	987	1,397	35,179	1,326	1,696
San Francisco	24,923	0.9984	23,848	1,075	1,147	13,938	748	1,075
DC	23,688	0.9968	23,205	483	635	11,906	503	845
Chicago	12,598	0.9976	11,782	816	599	8,819	433	853

Table 4.1 also provides statistics of the RSVPs, including the breakup of the RSVPs into the positive and negative ones. It can be observed that for all four cities, the positive RSVPs far exceed the negative ones. This indicates that users generally respond when they are interested in attending an event. Users who intend to respond with a negative or *no* RSVP for an event generally ignore the event and do not provide a response. This observation justifies the pairwise learning approach, which utilizes both negative and unobserved RSVPs by forming preference pairs instead of performing pointwise prediction. Section 3.4.3 includes the comparison of the experimental results of different methods.

[5] `http://www.meetup.com/meetup_api/`

3.4.2 Experimental Setup

The data is sorted in chronological order of event time so that the model is trained on past events, and it recommends future ones. The sorted datasets are then split to use 80% as the training set and 20% as the test set for each city. The sampling strategy introduced in Section 3.3.6 is applied to generate preference instances for model training. The learned model parameters are applied to the test users to generate a ranking score for the events for each user based on $s_{u,i}$. The evaluation metrics include $P@5$, $P@10$, $R@5$, $R@10$, $NDCG@5$, $NDCG@10$, and $MAP@10$ [MRS+08]. These are common metrics for top-N recommendations.

The proposed models are compared with the following baseline methods. *Librec*[6], a widely used recommender library, is used to obtain results for the baseline methods. The regularization parameters λ and γ are set to 0.025, and the learning rate in SGD is $\alpha = 0.05$. The same parameter values are used with both the existing and proposed methods (to the extent possible).

- *Item Mean*: The ranking score of an event is predicted on the basis of the mean of the event RSVPs in the training set.
- *User KNN [BHK98]*: User-based K-Nearest Neighborhood collaborative filtering method that predicts the user preference based on the similarity with the K nearest users calculated using Pearson's correlation.
- *Item KNN [SKKR01]*: Item-based K-Nearest Neighborhood collaborative filtering method that predicts the user preference based on the similarity with the K nearest items calculated using Pearson's correlation.
- *Group-Membership*: This is a naive method that utilizes the user group membership data to recommend events organized by the groups that the users belong to. If the user does not belong to any group, then this model recommends the most popular events (based on the RSVP count) to the user. The user group membership data is obtained using the Meetup API.

[6] `http://www.librec.net`

- *Biased-MF [KBV09]*: Basic matrix factorization that includes global mean, user bias and event bias.
- *BPR-MF [RFGST09]*: Bayesian Personalized Ranking method that utilizes pairwise loss to provide top-N item recommendation using matrix factorization (MF).
- *SVD++ [Kor08]*: A state-of-the-art matrix factorization method that incorporates implicit feedback from the user for a superior accuracy.
- *SVDFeature*[7]: State-of-the-art model that incorporates domain-specific features to SVD++. The toolkit is configured to utilize the group and venue information from the dataset. The group is indicated as both a user and event feature, and the venue as only an event feature.

The following proposed models are evaluated by integrating influences from multiple factors: group-aware (G), venue influence (V), popularity influence (P), distance influence (D) and temporal influence (T). The models are also varied with the choice of the pairwise probability functions: Logistic (Logit) or Probit.

- *LFM-V-Logit*: This model incorporates the influence of only the event venue (V) into the BPR with matrix factorization (BPR-MF). The logistic function is used for the pairwise ranking.
- *LFM-T-Logit*: This model incorporates the temporal influence into the BPR with matrix factorization (BPR-MF), which utilizes the logistic function.
- *LFM-P-Logit*: This model incorporates the influence of the event popularity (P) into the BPR with matrix factorization (BPR-MF), which utilizes the logistic function.
- *LFM-D-Logit*: This model incorporates the influence of the geographical distance (D) into the BPR with matrix factorization (BPR-MF), which utilizes the logistic function.

[7] `http://svdfeature.apexlab.org/`

- *GLFM-Logit*: This model considers the influence of the dual perspective of groups (G) and uses the logistic function for the pairwise ranking.

- *GLFM-V-Logit*: This model considers the dual-perspective groups (G) and the influence of event venue (V) with the logistic function.

- *GLFM-VPD-Logit*: This model includes the dual perspective of groups (G), event venue (V), event popularity (P) and geographical distance (D) with the logistic pairwise function.

- *GLFM-VPDT-Logit*: This model includes all the information–dual perspective of groups (G), event venue (V), event popularity (P), geographical distance (D), and temporal influence (T)–with the logistic pairwise function.

- *LFM-V-Probit*: This model is similar to *LFM-V-Logit*, but with the Probit pairwise probability function.

- *LFM-T-Probit*: This model is similar to *LFM-T-Logit*, but with the Probit pairwise probability function.

- *LFM-P-Probit*: This model is similar to *LFM-P-Logit*, but with the Probit pairwise probability function.

- *LFM-D-Probit*: This model is similar to *LFM-D-Logit*, but with the Probit pairwise probability function.

- *GLFM-Probit*: This model is similar to *GLFM-Logit*, but with the Probit pairwise probability function.

- *GLFM-V-Probit*: This model is similar to *GLFM-V-Logit*, but with the Probit pairwise probability function.

- *GLFM-VPD-Probit*: This model is similar to *GLFM-VPD-Logit*, but with the Probit pairwise probability function.

- *GLFM-VPDT-Probit*: This model is similar to *GLFM-VPDT-Logit*, but with the Probit pairwise probability function.

- *GLFM-VPDT-Pointwise*: This model includes the dual perspective of group (G), event venue (V), event popularity (P), geographical distance (D), and temporal influence (T), utilizing a point-wise loss function. The ranking score for the user u on the event i is as follows. The objective function considers the actual RSVP of the user $a_{u,i}$, with the value 1 if the user provided an affirmative RSVP, and 0 for a negative or missing RSVP.

$$s_{u,i} = \Big(\boldsymbol{p_u} + \frac{1}{|G_u|}\sum_{g \in G_u} \boldsymbol{r_g}\Big)^T \Big(\boldsymbol{q_i} + \boldsymbol{t_g} + \boldsymbol{v_i} + \boldsymbol{y_i} + \boldsymbol{z_i}\Big) + \beta_c c_i + \beta_u d_{ui} \tag{3.15}$$

$$\begin{aligned} \min_{P,Q,R,T,Y,Z} \quad & \textstyle\sum_{(u,i)\in K} (s_{u,i} - a_{u,i})^2 + \lambda\Big(\sum_u \|\boldsymbol{p_u}\|_2^2 + \sum_i \|\boldsymbol{q_i}\|_2^2 + \sum_g \|\boldsymbol{r_g}\|_2^2 + \sum_i \|\boldsymbol{t_g}\|_2^2 \\ & + \textstyle\sum_i \|\boldsymbol{v_i}\|_2^2 + \sum_i \|\boldsymbol{y_i}\|_2^2 + \sum_i \|\boldsymbol{z_i}\|_2^2\Big) + \gamma\Big({\beta_c}^2 + {\beta_d}^2\Big) \end{aligned} \tag{3.16}$$

3.4.3 Results

The following subsections first compare the proposed models with the baseline methods, and then investigate the effect of dimensionality of the latent factor space. Finally, the proposed models are evaluated in cold-start settings.

Baseline Comparisons

The results of baseline comparisons are presented in Table 4.2, Table 4.3, Table 4.4 and Table 4.5 for the four cities, respectively. The best results in each evaluation metric are highlighted in boldface. The following observations can be made from the results.

- The proposed dual-perspective group-aware models (i.e., *GLFM-Logit*, *GLFM-Probit*, *GLFM-V-Logit*, *GLFM-V-Probit*, *GLFM-VPD-Logit*, *GLFM-VPD-Probit*, *GLFM-VPDT-*

Table 3.6: GLFM - Experimental Results of Baseline Comparisons for New York

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
Item Mean	0.0062	0.0033	0.0158	0.0164	0.0142	0.0163	0.0138
UserKNN	0.0187	0.0102	0.0403	0.0428	0.0372	0.0419	0.0336
ItemKNN	0.0445	0.0257	0.0731	0.0785	0.0679	0.0728	0.0558
Biased-MF	0.0002	0.0002	0.0009	0.0017	0.0007	0.0008	0.0004
BPR-MF	0.2340	0.1743	0.2559	0.3242	0.2819	0.3004	0.2594
SVD++	0.0003	0.0003	0.0015	0.0024	0.0011	0.0012	0.0008
SVDFeature	0.1606	0.1597	0.1877	0.2030	0.2223	0.2478	0.1505
Group-Membership	0.4063	0.3806	0.2155	0.3581	0.3791	0.4247	0.4029
GLFM-VPDT-Pointwise	0.0036	0.0030	0.0022	0.0050	0.0011	0.0013	0.0023
LFM-V-Logit	0.4491	0.4456	0.16641	0.3170	0.2580	0.2849	0.4481
LFM-T-Logit	0.2540	0.2177	0.1805	0.2130	0.2565	0.2682	0.2253
LFM-P-Logit	0.2020	0.1865	0.1437	0.1943	0.2005	0.2116	0.1919
LFM-D-Logit	0.1870	0.1634	0.1582	0.1698	0.1905	0.2145	0.1554
GLFM-Logit	0.7153	0.6623	**0.3463**	0.5615	0.4284	0.4731	0.7107
GLFM-V-Logit	0.7180	0.6760	0.3327	0.5628	0.4272	0.4717	0.7143
GLFM-VPD-Logit	0.7193	0.6707	0.3242	0.5413	0.4277	0.4724	0.7135
GLFM-VPDT-Logit	0.7177	0.6777	0.3292	0.5631	0.4218	0.4658	0.7124
LFM-V-Probit	0.4284	0.4202	0.1687	0.3070	0.2478	0.2736	0.4271
LFM-T-Probit	0.2446	0.2357	0.1881	0.1917	0.1938	0.2133	0.2349
LFM-P-Probit	0.2166	0.2070	0.1549	0.1706	0.1725	0.1816	0.2054
LFM-D-Probit	0.1871	0.1760	0.1410	0.1476	0.1522	0.1643	0.1606
GLFM-Probit	0.6886	0.6555	0.3074	0.5282	0.4045	0.4467	0.6854
GLFM-V-Probit	0.7217	0.6794	0.3364	0.5749	0.4248	0.4691	0.7172
GLFM-VPD-Probit	0.7257	0.6856	0.3287	**0.5767**	0.4268	0.4713	0.7207
GLFM-VPDT-Probit	**0.7397**	**0.6908**	0.3353	0.5589	**0.4418**	**0.4879**	**0.7353**

Logit, *GLFM-VPDT-Probit*) substantially outperform the methods that do not consider group information. The best results in all the four cities are achieved by the proposed latent factor models with a large margin of improvement.

- Excluding the proposed group-aware models, the venue-aware latent factor models (*LFM-V*) perform the best. These results validate the observation in Section 3.2 that there may exist a correlation between the user and venue. Event venue is an important consideration while deciding to attend an event, as groups generally host events at the same or similar venues.

- The latent factor models that only consider the temporal influence, event popularity and geographical distance (*LFM-T*, *LFM-P*, *LFM-D*) yield mediocre results for all the four cities. Out of the three models, the model with the temporal influence yields the best results, followed by the model that considered the event popularity and geographical distance. The geographical distance has the least impact due to the fact that the RSVPs were considered local to a city, so the distance is not a major consideration while deciding

Table 3.7: GLFM - Experimental Results of Baseline Comparisons for San Francisco

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
Item Mean	0.0055	0.0031	0.0108	0.0118	0.0097	0.0036	0.0021
UserKNN	0.0321	0.0253	0.0732	0.1165	0.0592	0.0704	0.0486
ItemKNN	0.0548	0.0370	0.1267	0.1776	0.1219	0.1310	0.0950
Biased-MF	0.0004	0.0003	0.0010	0.0024	0.0004	0.0011	0.0006
BPR-MF	0.2124	0.1700	0.2552	0.3661	0.2991	0.3097	0.2588
SVD++	0.0003	0.0003	0.0013	0.0027	0.0007	0.0008	0.0004
SVDFeature	0.1499	0.1343	0.1926	0.2180	0.1903	0.1979	0.1366
Group-Membership	0.3019	0.2746	0.2068	0.3317	0.4063	0.4513	0.2987
GLFM-VPDT-Pointwise	0.0043	0.0034	0.0151	0.0236	0.0019	0.0021	0.0041
LFM-V-Logit	0.3295	0.2143	0.1848	0.2591	0.2208	0.2412	0.4364
LFM-T-Logit	0.2277	0.2001	0.2385	0.2405	0.2164	0.2182	0.2266
LFM-P-Logit	0.1973	0.1821	0.1547	0.1714	0.2412	0.2589	0.1902
LFM-D-Logit	0.1888	0.1709	0.1787	0.2088	0.1612	0.1842	0.1793
GLFM-Logit	0.5843	0.5110	0.3657	**0.5573**	0.3529	0.3897	0.5745
GLFM-V-Logit	0.6355	0.5542	**0.3721**	0.5506	0.3861	0.4264	0.6248
GLFM-VPD-Logit	**0.6403**	**0.5629**	0.3697	0.5545	**0.3890**	**0.4296**	**0.6317**
GLFM-VPDT-Logit	0.6186	0.5457	0.3402	0.5085	0.3774	0.4168	0.6108
LFM-V-Probit	0.3757	0.3372	0.1914	0.3549	0.2137	0.2361	0.3764
LFM-T-Probit	0.2390	0.2280	0.2090	0.2254	0.2272	0.2490	0.2307
LFM-P-Probit	0.2134	0.1933	0.1689	0.1815	0.2017	0.2288	0.2033
LFM-D-Probit	0.1799	0.1633	0.1500	0.1645	0.1911	0.1939	0.1771
GLFM-Probit	0.6112	0.5392	0.3666	0.5539	0.3651	0.4032	0.5985
GLFM-V-Probit	0.6026	0.5353	0.3573	0.5444	0.3637	0.4017	0.5948
GLFM-VPD-Probit	0.6026	0.5284	0.3612	0.5414	0.3646	0.4026	0.5935
GLFM-VPDT-Probit	0.5826	0.5366	0.2989	0.4848	0.3449	0.3808	0.5772

to attend an event. In conjunction with group and venue information, the event popularity, temporal influence and geographical distance may slightly improve the performance, as shown on the New York and San Francisco datasets.

- The Logistic and Probit pairwise probability functions yield very competitive results, while the Logistic function generates slightly better performance than the Probit function does. All the best results on San Francisco and Washington DC are attained by the Logistic function. For New York and Chicago, the results are mixed.

- Among the baselines, the Group-Membership method obtains the best performance on all the four cities, which further validates the assumption that group is an important factor for event recommendation. On the other hand, there still exists a large gap between the results of Group-Membership and the proposed group-aware models, which demonstrates the effectiveness of the dual perspective of group information.

- Both User-KNN and Item-KNN performed better than Item Mean, Biased-MF and

Table 3.8: GLFM - Experimental Results of Baseline Comparisons for Washington DC

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
Item Mean	0.0029	0.0021	0.0053	0.0112	0.0058	0.0061	0.0034
UserKNN	0.0043	0.0027	0.0078	0.0132	0.0068	0.0074	0.0050
ItemKNN	0.0312	0.0161	0.0731	0.0773	0.0661	0.0698	0.0566
Biased-MF	0.0013	0.0012	0.0039	0.0072	0.0021	0.0023	0.0009
BPR-MF	0.3322	0.2633	0.3692	0.4843	0.3915	0.4270	0.3664
SVD++	0.0014	0.0019	0.0043	0.0086	0.0030	0.0031	0.0016
SVDFeature	0.2281	0.2104	0.2332	0.2556	0.2789	0.2831	0.2175
Group-Membership	0.3120	0.2690	0.2482	0.3656	0.3878	0.4321	0.3110
GLFM-VPDT-Pointwise	0.0043	0.0040	0.0052	0.0069	0.0020	0.0023	0.0047
LFM-V-Logit	0.2880	0.2538	0.1425	0.2099	0.1929	0.2131	0.2959
LFM-T-Logit	0.2243	0.2112	0.1629	0.1641	0.2458	0.2506	0.2146
LFM-P-Logit	0.1861	0.1854	0.1336	0.1572	0.1974	0.2114	0.1860
LFM-D-Logit	0.1402	0.1356	0.1222	0.1279	0.1833	0.1914	0.1390
GLFM-Logit	0.6909	0.5825	0.4766	0.6605	0.4414	0.4875	0.6820
GLFM-V-Logit	**0.7410**	**0.6167**	**0.5111**	**0.7003**	**0.4753**	**0.5249**	**0.7315**
GLFM-VPD-Logit	0.7301	0.6123	0.5026	0.6873	0.4631	0.5031	0.7100
GLFM-VPDT-Logit	0.7032	0.6153	0.4603	0.6748	0.4373	0.4829	0.6950
LFM-V-Probit	0.3469	0.3134	0.1741	0.2632	0.2214	0.2445	0.3514
LFM-T-Probit	0.2061	0.2043	0.1956	0.2183	0.2611	0.2675	0.2060
LFM-P-Probit	0.1750	0.1646	0.1342	0.1491	0.2361	0.2399	0.1677
LFM-D-Probit	0.1651	0.1642	0.1342	0.1507	0.2112	0.2349	0.1640
GLFM-Probit	0.6756	0.5832	0.4540	0.6580	0.4262	0.4707	0.6690
GLFM-V-Probit	0.6865	0.6080	0.4407	0.6469	0.4357	0.4811	0.6848
GLFM-VPD-Probit	0.6516	0.5720	0.4213	0.6026	0.4274	0.4719	0.6587
GLFM-VPDT-Probit	0.6843	0.6043	0.4367	0.6634	0.4181	0.4617	0.6744

SVD++ models. These results are consistent with [MMS15] which found that state-of-the-art matrix factorization algorithms did not perform better than neighborhood-based methods in event recommendation. In fact, the pointwise variation of the proposed model, GLFM-VPDT-Pointwise, also yields poor results that are similar to Biased-MF and SVD++. The BPR-MF model, on the other hand, yields the best results among the matrix factorization based baselines, validating the decision to utilize the pairwise ranking approach for event recommendation. SVDFeature also generates good results, which are second only to the BPR-MF model.

In sum, the experimental results consistently demonstrate the effectiveness of the proposed latent factor models by exploiting the dual-perceptive of group information with pairwise learning.

Table 3.9: GLFM - Experimental Results of Baseline Comparisons for Chicago

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
Item Mean	0.0005	0.0002	0.0002	0.0002	0.0002	0.0002	0.0001
UserKNN	0.0252	0.0171	0.0583	0.0757	0.0592	0.0623	0.0476
ItemKNN	0.0516	0.0306	0.1166	0.1348	0.1025	0.1150	0.0870
Biased-MF	0.0012	0.0010	0.0036	0.0066	0.0034	0.0033	0.0019
BPR-MF	0.1653	0.1178	0.3011	0.3867	0.2917	0.3167	0.2672
SVD++	0.0015	0.0014	0.0047	0.0087	0.0038	0.0043	0.0023
SVDFeature	0.1441	0.1402	0.2754	0.2819	0.3043	0.3109	0.1414
Group-Membership	0.4189	0.3536	0.3322	0.4930	0.3907	0.4360	0.4070
GLFM-VPDT-Pointwise	0.0073	0.0036	0.0124	0.0124	0.0041	0.0046	0.0064
LFM-V-Logit	0.4842	0.4105	0.2585	0.4163	0.2830	0.3126	0.4565
LFM-T-Logit	0.2684	0.2621	0.2274	0.2965	0.2375	0.2415	0.2599
LFM-P-Logit	0.1977	0.1805	0.2030	0.2161	0.2018	0.2086	0.1931
LFM-D-Logit	0.1505	0.1499	0.1906	0.2044	0.2110	0.2286	0.1470
GLFM-Logit	0.7073	0.5673	0.5200	0.6797	0.4290	0.4738	0.6801
GLFM-V-Logit	**0.7431**	0.6026	0.5242	**0.6828**	**0.4550**	**0.5026**	**0.7176**
GLFM-VPD-Logit	0.7200	0.5894	0.5081	0.6763	0.4388	0.4847	0.6963
GLFM-VPDT-Logit	0.6463	0.5742	0.3882	0.5870	0.3849	0.4251	0.6344
LFM-V-Probit	0.4313	0.3708	0.2287	0.3391	0.2531	0.2824	0.4011
LFM-T-Probit	0.2836	0.2773	0.2289	0.2315	0.2410	0.2454	0.2745
LFM-P-Probit	0.2018	0.1991	0.1742	0.1855	0.1752	0.1863	0.1984
LFM-D-Probit	0.1712	0.1616	0.1542	0.1747	0.2015	0.2044	0.1675
GLFM-Probit	0.7136	0.5678	**0.5328**	0.6748	0.4378	0.4835	0.6879
GLFM-V-Probit	0.7221	**0.6047**	0.4978	0.6801	0.4396	0.4855	0.7021
GLFM-VPD-Probit	0.7294	0.6042	0.4968	0.6685	0.4464	0.4930	0.7082
GLFM-VPDT-Probit	0.6842	0.5842	0.4368	0.6083	0.4146	0.4578	0.6683

Effect of Dimensionality of Latent Factor Space

This section investigates the effect of the number of latent factors f (i.e., dimensionality of the latent factor space) on the proposed models (*GLFM-VPD-Logit* and *GLFM-VPD-Probit*) and other state-of-the-art latent factor models (BPR-MF and SVD++) in event recommendation. The number of latent factors is varied from 10 to 100 in increments of 10. Figure 3.2 plots the results in $P@10$ for the four cities, respectively. The following observations can be derived from the figure:

- The proposed group-aware latent factor models (*GLFM-VPD-Logit* and *GLFM-VPD-Probit*) perform significantly better than the other latent factor methods at most values of f for all the cities. These results demonstrate consistent improvement of the proposed models over the baselines across different numbers of latent factors.
- The results of the group-aware latent factor models gradually improve until $f = 40$ and plateau after that with no significant improvement. This pattern is observed for all four

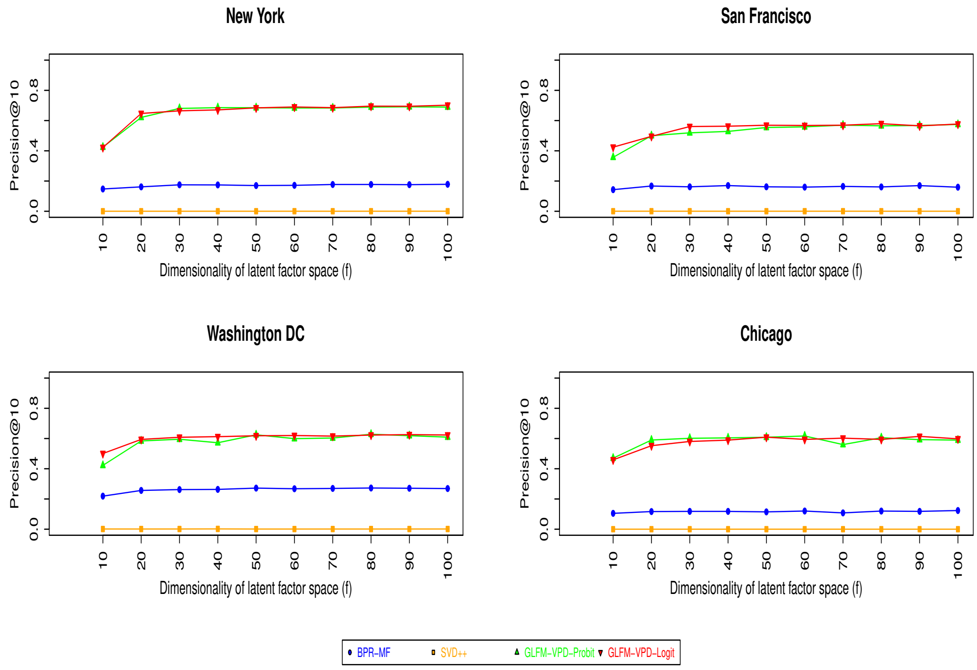

Figure 3.2: Effect of Dimensionality of Latent Factor Space for P@10

cities. On the other hand, the results of *GLFM-VPD-Logit* and *GLFM-VPD-Probit* are quite similar at different f while *GLFM-VPD-Logit* yields slightly better performance than *GLFM-VPD-Probit* in the majority of the cases of f.

- The results of the BPR-MF method are significantly better than SVD++ across all values of f, while there is not much variation in the results of the two methods as the value of f increases from 10 to 100. The same pattern is observed for all four cities. These results validate the advantage of using the pairwise ranking approach. A noticeable observation is that the recall results of BPR-MF for all four cities are almost similar to the results for the proposed group-aware latent factor model (*GLFM-VPD-Logit* and *GLFM-VPD-Probit*) for $f = 10$ to 20, but the group-aware latent factor models perform significantly better on recall for values of $f > 30$.

As observed, the relative performances of all the latent-factor based methods seem quite stable. These results indicate the dimensionality of the latent factor space may not be very sensitive. Therefore, $f = 40$ is set as the default value in other experiments.

Preference Instance Generation

Section 3.3.6 introduces how preference instances are derived based on three relations. In this section, two strategies for generating the preference instances are compared. The first strategy is the default one shown in Section 3.3.6, which is denoted as WUP (With Unknown RSVPs). It forms a preference pair with an unknown RSVP as the positive instance and a random negative RSVP as the negative instance. The second strategy, which is denoted as WOUP (WithOut Unknown RSVPs), does not assume that the unknown RSVPs are preferred over the observed RSVP "No". The results of both the strategies are provided in Table 3.10 (only the GLFM-VPD-Logit model for Chicago is shown to avoid clutter since GLFM-VPD-Logit usually gives the best results across different cities as shown in the previous sections). As the results show, the default strategy WUP yields much superior performance than WOUP does, which validates the advantage of utilizing the unobserved RSVPs.

Table 3.10: GLFM - Comparison between the default preference instance generation strategy WUP and the alternative strategy WOUP with the GLFM-VPD-Logit model for Chicago

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
GLFM-VPD-Logit (WOUP)	0.1989	0.1947	0.1260	0.2086	0.1093	0.1208	0.1965
GLFM-VPD-Logit (WUP)	**0.7200**	**0.5894**	**0.5081**	**0.6763**	**0.4388**	**0.4847**	**0.6963**

Cold-Start Event Recommendation

Table 3.11: GLFM - Experimental Results in the Cold-Start Setting on New York

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
New Users							
Item Mean	0.0031	0.0021	0.0139	0.0194	0.0088	0.0096	0.0029
Group-Membership	0.0003	0.0003	0.0018	0.0036	0.0033	0.0036	0.0003
Biased-MF	0.0001	0.0001	0.0005	0.0011	0.0004	0.0005	0.0003
BPR-MF	0.0163	0.0124	0.0769	0.1178	0.0219	0.0266	0.0202
SVD++	0.0001	0.0001	0.0005	0.0010	0.0005	0.0005	0.0003
SVDFeature	0.0124	0.0118	0.0547	0.0919	0.0183	0.0191	0.0116
GLFM-VPD-Logit	0.0477	**0.0293**	0.2107	**0.2618**	**0.0424**	**0.0468**	**0.0579**
GLFM-VPD-Probit	**0.0486**	0.0280	**0.2163**	0.2507	0.0407	0.0450	0.0562
New Events							
Group-Membership	0.0025	0.0025	0.0027	0.0053	0.0039	0.0044	0.0025
Biased-MF	0.0001	0.0001	0.0002	0.0009	0.0003	0.0003	0.0002
BPR-MF	0.0156	0.0121	0.0721	0.1118	0.0317	0.0331	0.0275
SVD++	0.0001	0.0001	0.0005	0.0011	0.0005	0.0005	0.0003
SVDFeature	0.0108	0.0100	0.0523	0.0667	0.0204	0.0202	0.0102
GLFM-VPD-Logit	**0.0509**	**0.0292**	**0.2263**	0.2619	**0.0432**	**0.0477**	**0.0592**
GLFM-VPD-Probit	0.0463	0.0291	0.2053	**0.2619**	0.0394	0.0435	0.0551

Table 3.12: GLFM - Experimental Results in the Cold-Start Setting on San Francisco

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
			New Users				
Item Mean	0.0072	0.0074	0.0320	0.0688	0.0091	0.0108	0.0084
Group-Membership	0.0003	0.0003	0.0016	0.0033	0.0029	0.0030	0.0003
Biased-MF	0.0001	0.0001	0.0002	0.0005	0.0002	0.0002	0.0001
BPR-MF	0.0270	0.0180	0.1291	0.1526	0.0177	0.0189	0.0238
SVD++	0.0001	0.0001	0.0003	0.0011	0.0003	0.0004	0.0002
SVDFeature	0.0194	0.0187	0.1011	0.1276	0.0163	0.0171	0.0190
GLFM-VPD-Logit	0.0201	0.0122	0.0765	0.0947	0.0180	0.0199	0.0235
GLFM-VPD-Probit	**0.0308**	**0.0186**	**0.1299**	**0.1590**	**0.0242**	**0.0267**	**0.0335**
			New Events				
Group-Membership	0.0075	0.0071	0.0031	0.0059	0.0042	0.0049	0.0075
Biased-MF	0.0001	0.0001	0.0003	0.0009	0.0002	0.0003	0.0001
BPR-MF	0.0270	0.0181	0.1096	0.1171	0.0093	0.0105	0.0826
SVD++	0.0001	0.0001	0.0003	0.0010	0.0003	0.0004	0.0002
SVDFeature	0.0186	0.0173	0.1144	0.1261	0.0179	0.0185	0.0164
GLFM-VPD-Logit	0.0171	0.0114	0.0647	0.0887	0.0145	0.0160	0.0191
GLFM-VPD-Probit	**0.0281**	**0.0185**	**0.1183**	**0.1549**	**0.0200**	**0.0221**	**0.0297**

Table 3.13: GLFM - Experimental Results in the Cold-Start Setting on Washington DC

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
			New Users				
Item Mean	0.0013	0.0012	0.0068	0.0101	0.0084	0.0097	0.0011
Group-Membership	0.0004	0.0004	0.0019	0.0037	0.0024	0.0026	0.0004
Biased-MF	0.0001	0.0001	0.0006	0.0012	0.0005	0.0006	0.0004
BPR-MF	0.0150	0.0094	0.0143	0.0186	0.0107	0.0119	0.0147
SVD++	0.0001	0.0001	0.0008	0.0020	0.0007	0.0009	0.0005
SVDFeature	0.0124	0.0119	0.0167	0.0186	0.0144	0.0151	0.0111
GLFM-VPD-Logit	0.0165	0.0108	0.0671	0.0870	0.0127	0.0129	0.0184
GLFM-VPD-Probit	**0.0174**	**0.0126**	**0.0733**	**0.1054**	**0.0133**	**0.0147**	**0.0197**
			New Events				
Group-Membership	0.0004	0.0004	0.0015	0.0032	0.0026	0.0029	0.0004
Biased-MF	0.0002	0.0001	0.0009	0.0015	0.0006	0.0008	0.0005
BPR-MF	0.0144	0.0101	0.0207	0.0261	0.0094	0.0107	0.0188
SVD++	0.0002	0.0001	0.0009	0.0017	0.0009	0.0009	0.0006
SVDFeature	0.0131	0.0118	0.0219	0.0221	0.0171	0.0188	0.0126
GLFM-VPD-Logit	**0.0180**	0.0109	**0.0784**	**0.0935**	**0.0122**	**0.0134**	**0.0194**
GLFM-VPD-Probit	0.0173	**0.0111**	0.0688	0.0903	0.0096	0.0106	0.0161

Cold-start is a challenging problem in any recommendation system when there is no information about users or items. Cold-start is especially prevalent in event recommendation, because events are always in the future and short-lived. In this section, the performance of the proposed models is tested in the cold-start settings, and compared with the relevant baseline methods. The models are evaluated under two scenarios: *new users* and *new items*.

For the *new users* scenario, the dataset is split to ensure the same user is not present in both the training and test sets. The time order is reserved so that the events in the test data always

Table 3.14: GLFM - Experimental Results in the Cold-Start Setting on Chicago

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@10
New Users							
Item Mean	0.0011	0.0019	0.0045	0.0164	0.0041	0.0057	0.0014
Group-Membership	0.0005	0.0005	0.0019	0.0032	0.0037	0.0039	0.0005
Biased-MF	0.0002	0.0002	0.0010	0.0023	0.0008	0.0010	0.0006
BPR-MF	0.0234	0.0181	0.1073	0.1662	0.0120	0.0153	0.0296
SVD++	0.0003	0.0003	0.0015	0.0036	0.0016	0.0017	0.0008
SVDFeature	0.0194	0.0182	0.0997	0.1055	0.0193	0.0217	0.0187
GLFM-VPD-Logit	**0.0276**	**0.0203**	**0.1105**	**0.1730**	**0.0204**	**0.0225**	**0.0318**
GLFM-VPD-Probit	0.0190	0.0146	0.0779	0.1200	0.0158	0.0175	0.0234
New Events							
Group-Membership	0.0005	0.0005	0.0017	0.0034	0.0042	0.0061	0.0005
Biased-MF	0.0003	0.0003	0.0015	0.0034	0.0012	0.0015	0.0009
BPR-MF	0.0176	0.0158	0.0822	0.1492	0.0132	0.0146	0.0199
SVD++	0.0005	0.0003	0.0022	0.0034	0.0011	0.0016	0.0007
SVDFeature	0.0167	0.0158	0.0822	0.0945	0.0171	0.0183	0.0159
GLFM-VPD-Logit	**0.0311**	**0.0199**	**0.1308**	**0.1703**	**0.0199**	**0.0220**	**0.0312**
GLFM-VPD-Probit	0.0186	0.0173 e	0.07403	0.1470	0.0132	0.0146	0.0220

occur after those in the training data. Similar to the experiments in the regular setting, a training-test data split of 80% vs 20% is performed. The dataset is split similarly for the *new events* scenario with a 80% vs 20% ratio for training and testing sets, making sure that the same event is not present in both sets.

The experiments are performed on the group-aware models that consider the group, venue, popularity and distance factors (*GLFM-VPD-Logit* and *GLFM-VPD-Probit*), as they provide the best results for a majority of the regular experiments. The performance of the group-aware latent factor models is compared with the baseline methods: Item Mean, Group-Membership, Biased-MF, BPR-MF, SVD++, and SVDFeature. Item Mean is not applicable to the *new events* scenario due to no historical information available for the new events. UserKNN and ItemKNN cannot handle the cold-start settings, and hence they are omitted from this experiment. Since a user in the test set is not present in the training set, the proposed models are unable to learn the user-specific parameters. In other words, the user latent factor $\boldsymbol{p_u}$ is not present, however, the dual-perspective group factors $\boldsymbol{r_g}$ and $\boldsymbol{t_g}$ enable the calculation of the ranking score for the *new users* scenario. The prediction of the proposed models is made by removing p_u from Eqn.(3.11). Similarly, for the *new items* scenario, the event latent factor $\boldsymbol{q_i}$ and the popularity count c_i are not present, but the event-oriented factor $\boldsymbol{t_g}$ exists. The prediction is made by removing $\boldsymbol{q_i}$ and c_i from Eqn.(3.11).

Table 3.11, Table 3.12, Table 3.13 and Table 3.14 contain the results for the cold-start experiments. As observed, the best results in both new-user and new-event scenarios are achieved by the proposed methods, *GLFM-VPD-Logit* or *GLFM-VPD-Probit*. This pattern holds true across different cities and different evaluation metrics. The improvement is especially substantial on New York and Washington DC. Furthermore, *GLFM-VPD-Logit* and *GLFM-VPD-Probit* yield competitive results. *GLFM-VPD-Probit* achieves the best results on San Francisco, while *GLFM-VPD-Logit* obtains the best on Chicago for both scenarios. On the other two cities, the results are mixed. Among the baseline methods, BPR-MF generates the best results across four cities in all the metrics for both cold-start scenarios. These results validate the advantage of pairwise training for the event recommendation task. In sum, the experimental results demonstrate the advantage of the proposed models in dealing with the cold-start problems for event recommendation. In the cold-start experiments, we considered *only* new users and events, which is atypical of a real-world scenario.

3.5 Summary

This chapter systematically investigates the effect of group information on event recommendation. A latent factor model is proposed based on the dual-perspective of groups. Logistic and Probit functions are used to model the probability of pairwise preferences that consist of observed and unobserved user feedback. Additional contextual information such as event venue, popularity, temporal influence, and geographical distance can be readily incorporated into the model. The experiments on the Meetup data of four cities demonstrate the importance of group information, and show much improved performance over the state-of-the-art baselines. Moreover, the proposed approach demonstrates advantages of tackling the cold-start problems by utilizing the dual role of groups.

Chapter 4

Attentive Contextual Denoising Autoencoder (ACDA)

4.1 Background

Machine learning model-based techniques have been effective for recommender systems in the past, and they have offered a reasonable level of performance. However, these methods lack the ability to model complex nonlinear relationships that usually accompany the user-item interaction. With the recent success in the adoption of deep learning in computer vision and speech recognition [GBC16], there has been a surge of interest in applying deep learning methods to recommendation tasks [ZYS17]. The existing work in this domain is still quite limited, and furthermore, it does not utilize contextual information, which is largely present in the real-world scenarios. Context provides additional information to the user-item interaction, which in turn improves the quality of the recommendation [DYM+14]. The attention mechanism [BCB14] provides an intuitive way to incorporate context into the user-item interaction. Motivated by these factors, this study proposes a novel model for personalized recommendation based on the denoising autoencoder augmented with a context-driven attention mechanism. The proposed model is called the *Attentive Contextual Denoising Autoencoder (ACDA).*

Autoencoders [GBC16] are unsupervised feed-forward neural networks capable of learning a representation of the input data, also known as codings. The codings typically learnt by an autoencoder are of much lower dimensionality than the original input. Denoising autoencoders [GBC16] are a variant of the basic autoencoder that add noise to the input and train the network to recover the original uncorrupted input at the output layer. This forces the network to discover robust features in the data representation, and prevents the model from learning the trivial identity function. The autoencoder architecture makes it suitable for use in recommender systems as the hidden layer captures the latent representation of the data, allowing the model to learn the latent factors associated with the user-item interaction. It has been shown [WDZE16] that the denoising autoencoder architecture is a nonlinear generalization of latent factor models [KBV09, MS07], which have been widely used in recommender systems. Therefore, the denoising autoencoder is utilized as the main building block for the proposed *ACDA* model.

Context provides an added dimension to real-world applications. Recommender systems for movies, products, point-of-interests and services utilize context to provide a meaningful personalized recommendation [DYM+14]. For example, genre such as horror, drama, thriller, comedy etc., is an important context for movie recommendation as people generally like the same type of movies. Location and time-of-day are useful context to consider while recommending point-of-interests. There is existing work in the literature that provides contextual recommendations [ZWF13, PN13, KT13]. The *ACDA* model incorporates contextual information via the attention mechanism for personalized recommendation. The contextual information may be related to the user or the item. The *ACDA* model is applied to two real-world problems of event recommendation and movie recommendation. For the event recommendation task, the model utilizes the user *group* and event *venue* as the contextual attributes, whereas the movie *genre* is used as the contextual attribute for the movie recommendation task.

The attention mechanism has been instrumental while dealing with structured problems, such as machine translation and caption generation [VTBE15, BCB14, HKG+15]. The objective of the mechanism is to highlight, or focus attention on, a certain subset of the data. The attention mechanism accepts a certain input and a context that accompanies that input. The output of the attention mechanism is considered as a summary of the input focusing on the information linked to the provided context. The attention mechanism is generally applied for two reasons–first, to provide for efficient processing of a high-dimensional input by processing only subsets of the input, and second, to focus on specific parts of the input in terms of its relevance. A classical example of the use of the attention mechanism is image captioning, where the mechanism focuses on certain subsets of the image to generate the suitable caption. The *ACDA* model utilizes the attention mechanism to apply the contextual attributes to the hidden representation of the user's preference. This helps the model to associate personalized context with each user's preference to provide recommendation targeted to that specific user.

The *ADCA* model accepts the user's preference on existing items as input, which includes both positive and negative instances. The input is partially corrupted to learn a robust representation of the data. The input is mapped to an internal representation of lower dimensionality by the hidden layer, where the contextual parameters are applied via the attention mechanism to focus on the user-specific relevant context. The output of the model is the reconstructed user input, which is the predicted preference of the user. The model is trained to minimize the loss between the original corrupted input and the reconstructed input generated at the output layer. Real-world datasets are used to conduct comprehensive experiments for the proposed *ACDA* model. The datasets for the event recommendation task is obtained from *Meetup*[1], a popular Event-Based Social Network (EBSN). The publicly available *Movielens* 100K dataset is used for the movie recommendation task. The experimental results show that the proposed model performs better than the state-of-the-art baselines.

4.2 Attentive Contextual Denoising Autoencoder

The architecture of the *Attentive Contextual Denoising Autoencoder (ACDA)* model is explained first, which is followed by an explanation of how this model is applied for the event recommendation and movie recommendation tasks.

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

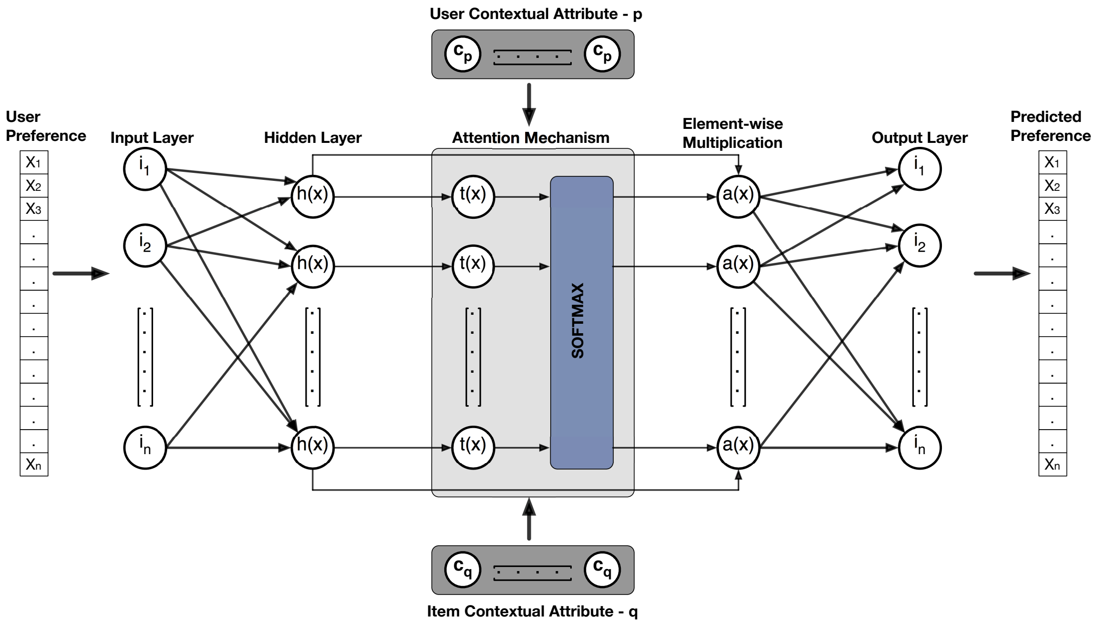

Figure 4.1: Attentive Contextual Denoising Autoencoder

4.2.1 The Architecture

The proposed model, as illustrated in Figure 4.1, is based on the denoising autoencoder neural network architecture. The model takes as input a vector indicating the preference of a user u on all the items i in the dataset. Assuming that there are m users and n items, the autoencoder takes as input a vector $\mathbf{x} \in \mathbb{R}^n$, which is the known preference of the user u on the n items. The input vector $\mathbf{x}$ is corrupted using mask-out/drop-out corruption to obtain $\tilde{\mathbf{x}}$. The corruption method randomly overwrites some of the dimensions of $\mathbf{x}$ with 0 using the probability ρ. To offset the effect of the corruption on certain dimensions, the remaining dimensions are scaled by applying a factor δ to the original value. This corruption method is similar to the one used in [WDZE16].

$$P(\tilde{\mathbf{x}}_\theta = 0) = \rho$$

$$P(\tilde{\mathbf{x}}_{\bar{\theta}} = \delta\tilde{\mathbf{x}}) = 1 - \rho$$

where $\delta = 1/(1-\rho)$. The symbol θ denotes the dimensions that are set to 0, whereas $\bar{\theta}$ denotes the dimensions that are scaled.

The corrupted vector $\tilde{\mathbf{x}}$ is fed into the model to generate the latent hidden representation $h \in \mathbb{R}^k$ using the encoding function $e(\cdot)$. The dimensionality of the hidden representation is represented by $k \ll n$, which is the number of hidden units in the model. The input user preference is corrupted only while training the model, and not during cross-validation and test.

$$h(\tilde{\mathbf{x}}) = e\big(\mathbf{W} \cdot \tilde{\mathbf{x}} + \mathbf{b}\big) \tag{4.1}$$

where $\mathbf{W} \in \mathbb{R}^{k \times n}$ is the weight matrix and $\mathbf{b} \in \mathbb{R}^k$ is a bias vector.

The encoding function $e(\cdot)$ is set to the *ReLU* function [NH10] as it performs well due to its suitability for sparse data.

$$ReLU(x) = \max(0, x)$$

The hidden representation $h(\tilde{\mathbf{x}})$ is input into the attention mechanism layer, where the contextual attributes are applied. The objective of the attention mechanism is to summarize the input representation based on the provided context. The context may be associated with the user or item. The model is flexible enough to accommodate as many contextual attributes as desired. However, two contextual attributes (p and q) are specified in our model for ease of presentation. The attention mechanism applies a weighted user-context $\mathbf{c}_p$ and item-based context $\mathbf{c}_q$ for any given contextual parameters p and q to the output of the hidden layer with a nonlinear activation function $f(\cdot)$. Mathematically, this is denoted as:

$$t(\tilde{\mathbf{x}}) = f\big(\mathbf{W}_h \cdot h(\tilde{\mathbf{x}}) + \mathbf{W}_p \cdot \mathbf{c}_p + \mathbf{W}_q \cdot \mathbf{c}_q\big) \tag{4.2}$$

where $\mathbf{W}_h$ is a $\mathbb{R}^{k \times k}$ weight matrix, with k being the number of units in the attention layer, which is the same as the number of units in the hidden layer. $\mathbf{W}_p$ and $\mathbf{W}_q$ are weight matrices of dimensions $\mathbb{R}^{k \times |p|}$ and $\mathbb{R}^{k \times |q|}$ respectively, with $|p|$ and $|q|$ being the number of contextual parameters. $h(\tilde{\mathbf{x}})$ is the output of the hidden layer.

The *tanh* function was selected as the attention mechanism activation function ($f(\cdot)$) as it gave us the best results.

$$tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

The output of the attention activation function, $t(\tilde{\mathbf{x}})$, is then fed into a *softmax* layer. Finally, the *softmax* output is combined with the hidden layer output via element-wise multiplication ($\otimes$) to generate the final output of the attention mechanism, which is denoted as $a(\tilde{\mathbf{x}})$.

$$a(\tilde{\mathbf{x}}) = softmax\big(t(\tilde{\mathbf{x}})\big) \otimes h(\tilde{\mathbf{x}}) \tag{4.3}$$

where $t(\tilde{\mathbf{x}})$ is the output of the attention activation function, and $h(\tilde{\tilde{\mathbf{x}}})$ is the hidden layer output. The *softmax* function is defined as:

$$softmax(x_1, x_2, ..., x_n) = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)}$$

Essentially, the attention mechanism serves to apply a weighted arithmetic mean to the hidden layer representation, with the weight representing the relevance based on the provided context.

The internal latent representation of the input with the applied context is reconstructed back to the original form using a decoding function $d(\cdot)$.

$$\hat{\mathbf{x}} = d\big(\mathbf{W}' \cdot a(\tilde{\mathbf{x}}) + \mathbf{b}'\big) \tag{4.4}$$

where the dimension of $\mathbf{W}'$ and $\mathbf{b}'$ is the same as $\mathbf{W}$ and $\mathbf{b}$. The reconstruction of the original input, or the reverse mapping, may be constrained by sharing parameters $\mathbf{W}' = \mathbf{W}^T$. However, this study did not do so, as better results were obtained by having different $\mathbf{W}'$ and $\mathbf{b}'$ parameters at the decoding step.

The *sigmoid* function was selected as the decoding function $d(\cdot)$ as it constraints an input to the $0 - 1$ output range. This gives the probability associated with each item at the output, which is used for ranking in personalized recommendation. However, the *ADCA* model is generic and it can be applied to a rating prediction task by simply selecting any other nonlinear function as the decoding function $d(\cdot)$.

$$\sigma(x) = \frac{1}{1+\exp(-x)}$$

The parameters of the model are trained by minimizing the mean squared error between the original corrupted input vector $\tilde{\mathbf{x}}$ and the reconstructed vector $\hat{\mathbf{x}}$.

$$\min_{\mathbf{W},\mathbf{W}',\mathbf{W}_h,\mathbf{W}_p,\mathbf{W}_q,\mathbf{b},\mathbf{b}'} \frac{1}{m}\sum_{u\in U} \|\tilde{\mathbf{x}}_u - \hat{\mathbf{x}}_u\|^2 \tag{4.5}$$

The parameters are updated using the stochastic gradient descent variant ADAM optimizer [KB15]. *Dropout* [GBC16] was used for regularization to prevent overfitting and improve generalization capacity. The *dropout* rate was set to 0.2, which means that 20% of the hidden units are dropped at random at each training step to prevent co-adaptation.

4.2.2 Top-N Recommendation

The proposed *ACDA* model can be applied to both rating prediction and top-N recommendation by simply changing the decoding function $d(\cdot)$. The decoding function $d(\cdot)$ is set to the *sigmoid* function for top-N recommendation. This study focuses on top-N recommendation, and the generic *ACDA* model is applied to the event recommendation and movie recommendation tasks.

The event recommendation task utilizes the RSVP [2] data from *Meetup*. Users indicate their preference to an event by providing an RSVP, which is used to recommend future events to the user. For the event recommendation task, the user *group* and event *venue* are used as the contextual attributes. Users typically organize themselves into groups in an Event Based Social Network (EBSN) such as *Meetup*, and each event is hosted at a physical venue. The user's preference on existing events in the training set is input into the model as a binary

[2] RSVP is a French expression, which means "please respond"

k-hot encoded vector with a true value for the positive event preferences and false for the negative or unknown event preferences. The input preference is corrupted as discussed earlier in section 4.2.1. In addition to corrupting the input, a fixed number of negative samples are also included by encoding them as positive in the input vector. The negative samples are selected randomly from the training set, and negative sample inclusion is only performed during training, not during evaluation on the cross-validation and test sets. The output of the model is the personalized top-N event recommendation for the user.

For the event recommendation task, the contextual attributes of the model are set as: $\mathbf{c}_p = \mathbf{u}_g$ and $\mathbf{c}_q = \mathbf{i}_v$, where $\mathbf{u}_g \in \mathbb{R}^{|p|}$ denotes the groups that the user belongs to. The parameter $\mathbf{i}_v \in \mathbb{R}^{|q|}$ denotes the venues associated with the events. The parameters $|p|$ and $|q|$ are the number of groups and venues respectively.

The *ACDA* model is also applied to movie recommendation, which is also treated as a top-N recommendation task. The *Movielens* dataset contains the movie ratings on a scale of $1 - 5$, which is converted to a binary scale. The movie binary scale indicates a user's preference on existing movies that is used to recommend other movies to the user. The movie *genre* is selected as a contextual attribute for our model. The *genre* is associated with each movie, and certain movies have multiple genres associated with them. Similar to the event recommendation task, the user's preference is partially corrupted and input into the model as a binary k-hot encoded vector. Negative samples are also used during training. The output of the model is the personalized top-N movie recommendation for the user.

Since there is only one item-related contextual attribute for the movie recommendation task, the model is updated as: $\mathbf{c}_q = \mathbf{i}_r$, where $\mathbf{i}_r \in \mathbb{R}^{|q|}$ denotes the genres associated with the movies preferred by the user. The parameter $|q|$ is the number of genres.

4.3 Experiments

4.3.1 Datasets

The proposed *Attentive Contextual Denoising Autoencoder (ACDA)* model is evaluated on real-world datasets from *Meetup* and *Movielens*. The Meetup dataset is for events from the cities of *New York City*, *San Francisco*, *Washington DC* and *Chicago*. These cities were selected as they are the major metropolitan areas in the United States, and they have a vibrant Meetup community. The event data was collected by using the Meetup API[3] between the time periods January 2016 and May 2016. The proposed model is also analyzed against the publicly available *Movielens (100K)* dataset. The *Movielens* dataset consists of movie ratings provided by the user on the $1-5$ scale. The numeric score is converted to a binary rating for the purpose of top-N recommendation. A score of 5 is converted to a binary rating of 1, and anything less than a 5 is converted to 0. The statistics of the datasets used for the experiments are given in Table 4.1.

Table 4.1: ACDA - Data Statistics

Dataset	Observations	Sparsity	Positive	Negative	Users	Items
Meetup-NYC	73,816	0.9998	70,170	3,646	19,122	36,054
Meetup-SFO	48,972	0.9998	43,637	5,335	18,957	14,445
Meetup-DC	36,451	0.9998	33,541	2,901	10,384	12,359
Meetup-Chicago	22,915	0.9996	20,826	2,089	8,118	9,133
Movielens-100K	100,004	0.9835	15,095	84,909	671	9,066

4.3.2 Experimental Setup

The datasets are split to use 60% as the training set, 20% as the cross-validation set, and 20% as the test set. The evaluation metrics include $P@5$, $P@10$, $R@5$, $R@10$, $NDCG@5$, $NDCG@10$, $MAP@5$ and $MAP@10$ [MRS+08]. These are common metrics for top-N recommendations.

Baselines methods from each of the following categories are considered for comparison against the proposed *ACDA* model with respect to the event and movie recommendation task. The

[3] `http://www.meetup.com/meetup_api/`

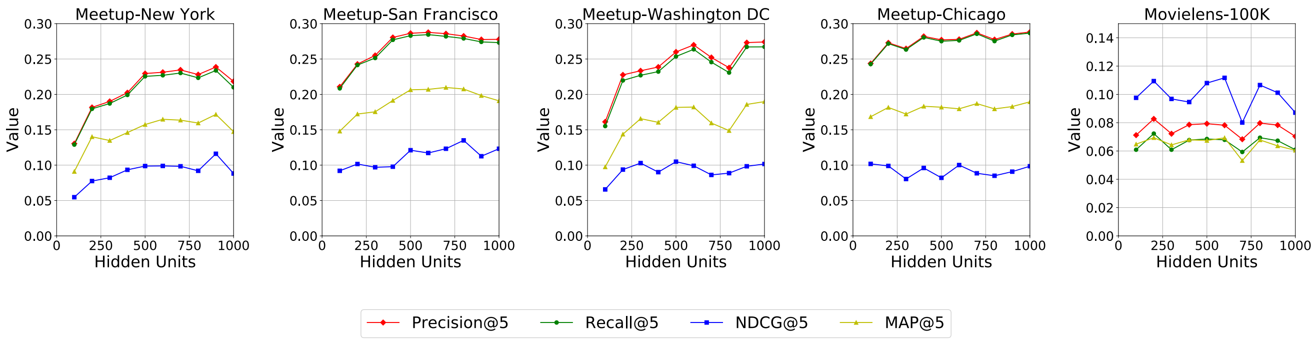

Figure 4.2: Hidden Unit Count Selection

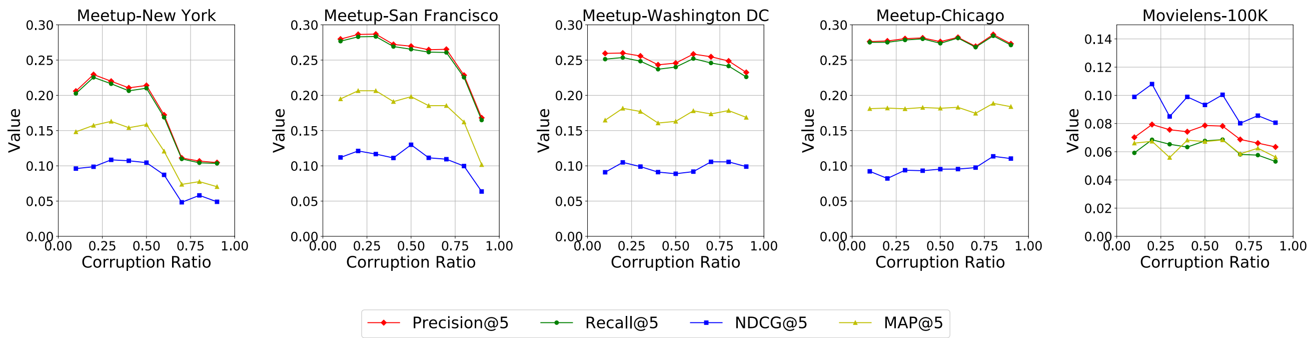

Figure 4.3: Corruption Ratio Selection

results of the proposed models are compared against these baselines methods. The results are presented in this section and the findings are discussed in detail.

- Neighborhood-based Methods (*UserKNN*, *ItemKNN*)
- Model-based Methods (*BiasedMF*, *BPR-MF*, *SVD++*)
- Deep Learning Methods (*CDAE*, *U-AutoRec*)

A popular recommender library, *Librec*[4], is used to obtain results for the neighborhood and model-based methods. A custom implementation is utilized for the deep learning baseline models. The parameter values for the existing methods are similar to the proposed method (to the extent possible).

- *User-KNN*: User k-nearest neighborhood collaborative filtering method that predicts the user preference based on the similarity with the k nearest users. The number of neighbors is set as $k = 10$, as it gave the best results.

[4]`http://www.librec.net`

Table 4.2: ACDA - Experimental Results: New York

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.0069	0.0034	0.0220	0.0237	0.0223	0.0232	0.0186	0.0181
ItemKNN	0.0076	0.0038	0.0241	0.0254	0.0213	0.0222	0.0194	0.0191
Biased-MF	0.0003	0.0002	0.0001	0.0002	0.0002	0.0003	0.0008	0.0007
BPR-MF	0.0501	0.0342	0.1266	0.1524	0.0766	0.0825	0.1061	0.1089
SVD++	0.0005	0.0004	0.0003	0.0006	0.0004	0.0006	0.0001	0.0001
CDAE	0.1035	0.1477	0.1123	0.1657	0.0707	0.0791	0.0788	0.0994
U-AutoRec	0.0527	0.0804	0.0508	0.0790	0.0272	0.0305	0.0334	0.0517
ACDA-V	0.1086	0.1844	0.1066	0.1834	0.0523	0.0541	0.0739	0.1151
ACDA-G	0.1781	0.2320	0.1760	0.2309	0.0860	0.0871	0.1337	0.1738
ACDA-GV	**0.2295**	**0.2905**	**0.2255**	**0.2990**	**0.0987**	**0.0994**	**0.1574**	**0.2116**

Table 4.3: ACDA - Experimental results: San Francisco

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.0166	0.0127	0.0401	0.0615	0.0317	0.0390	0.0232	0.0255
ItemKNN	0.0155	0.0131	0.0394	0.0591	0.0303	0.0381	0.0229	0.0245
Biased-MF	0.0004	0.0004	0.0001	0.0003	0.0005	0.0004	0.0002	0.0001
BPR-MF	0.0552	0.0376	0.1486	0.1809	0.0860	0.0977	0.1217	0.1254
SVD++	0.0014	0.0009	0.0011	0.0017	0.0017	0.0016	0.0009	0.0007
CDAE	0.1109	0.1877	0.1098	0.1909	0.0519	0.0564	0.0804	0.1129
U-AutoRec	0.1045	0.1525	0.1020	0.1513	0.0634	0.0686	0.0730	0.1084
ACDA-V	0.1793	0.2623	0.1773	0.2616	0.0765	0.0789	0.1226	0.1770
ACDA-G	0.1879	3061.0	0.1649	0.3049	0.0602	0.0622	0.1004	0.1825
ACDA-GV	**0.2864**	**0.3708**	**0.2830**	**0.3692**	**0.1211**	**0.1266**	**0.2065**	**0.2743**

- *Item-KNN*: Item k-nearest neighborhood collaborative filtering method that predicts the user preference based on the similarity with the k nearest items. The value is set as $k = 10$ to be consistent with *User-KNN*.

- *BPR-MF*: Bayesian personalized ranking method that utilizes pairwise loss to provide top-N item recommendation using matrix factorization (MF). The latent factor count is set to $l = 50$ as it offered the best performance.

- *Biased-MF*: Basic matrix factorization that includes global mean, user bias and item bias. The latent factor count is set as $l = 50$ to be consistent with the *BPR-MF* method.

- *SVD++*: State-of-the-art matrix factorization method that incorporates implicit feedback from the user into the baseline SVD model for better accuracy. The latent factor count is set as $l = 50$ to be consistent with the *BPR-MF* method.

- *CDAE*: Collaborative filtering technique based on denoising autoencoders that incorpo-

Table 4.4: ACDA - Experimental results: Washington DC

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.0013	0.0006	0.0039	0.0045	0.0023	0.0025	0.0016	0.0017
ItemKNN	0.0011	0.0005	0.0042	0.0042	0.0027	0.0027	0.0018	0.0019
Biased-MF	0.0003	0.0001	0.0003	0.0022	0.0002	0.0010	0.0007	0.0003
BPR-MF	0.0588	0.0466	0.1188	0.1509	0.0688	0.0753	0.1068	0.1098
SVD++	0.0007	0.0007	0.0001	0.0006	0.0012	0.0011	0.0007	0.0004
CDAE	0.0983	0.1860	0.0914	0.1812	0.0459	0.0515	0.0663	0.1089
U-AutoRec	0.0905	0.1136	0.0847	0.1083	0.0560	0.0606	0.0731	0.0885
ACDA-V	0.1737	0.2498	0.1676	0.2455	0.0836	0.0863	0.1177	0.1713
ACDA-G	0.1956	0.2833	0.1886	0.2792	0.0777	0.0794	0.1198	0.1886
ACDA-GV	**0.2600**	**0.3472**	**0.2536**	**0.3430**	**0.1049**	**0.1092**	**0.1816**	**0.2476**

rates the user latent factor as additional input [WDZE16].

- *U-AutoRec*: Collaborative filtering technique based on denoising autoencoders [SMSX15] that has two variants: *I-AutoRec*, which accepts the k-hot encoded item preference vector consisting of users as input, and *U-AutoRec* that accepts the k-hot encoded user preference vector of items. A comparison is made against the *U-AutoRec* variant as it is similar to the proposed *ACDA* model in terms of the user preference on items being provided as input.

The proposed models are evaluated by incorporating the influence of the different contextual attributes for the event and movie recommendation tasks.

- *ACDA-V*: This is the variant of the proposed generic *ACDA* model that incorporates only the event *venue* as a contextual attribute for the event recommendation task.

- *ACDA-G*: A variant of the proposed generic *ACDA* model that just incorporates the user *group* as a contextual attribute for the event recommendation task.

- *ACDA-GV*: This model includes both the user *group* and event *venue* as contextual attributes of the *ACDA* model for the event recommendation task.

- *ACDA-R*: This model includes the movie *genre* as a contextual attribute of the *ACDA* model for the movie recommendation task.

Table 4.5: ACDA - Experimental results: Chicago

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.0065	0.0041	0.0204	0.0213	0.0160	0.0170	0.0122	0.0125
ItemKNN	0.0062	0.0037	0.0193	0.0202	0.0157	0.0166	0.0118	0.0120
Biased-MF	0.0004	0.0003	0.0002	0.0006	0.0002	0.0005	0.0008	0.0001
BPR-MF	0.0498	0.0311	0.1640	0.1925	0.0952	0.1059	0.1277	0.1322
SVD++	0.0020	0.0014	0.0005	0.0012	0.0020	0.0020	0.0010	0.0008
CDAE	0.1271	0.2097	0.1260	0.2095	0.0428	0.0438	0.0724	0.1297
U-AutoRec	0.0879	0.1209	0.0878	0.1209	0.0476	0.0479	0.0621	0.0855
ACDA-V	0.2354	0.3356	0.2339	0.3353	0.0805	0.0796	0.1493	0.2261
ACDA-G	**0.2866**	**0.3994**	**0.2849**	**0.3988**	**0.1375**	**0.1395**	**0.2052**	**0.2835**
ACDA-GV	0.2771	0.3856	0.2752	0.3849	0.0821	0.0847	0.1819	0.2655

Table 4.6: ACDA - Experimental results: Movielens 100K

Method	P@5	P@10	R@5	R@10	NDCG@5	NDCG@10	MAP@5	MAP@10
UserKNN	0.0200	0.0185	0.0052	0.0082	0.0200	0.0204	0.0109	0.0089
ItemKNN	0.0195	0.0164	0.0055	0.0079	0.0205	0.0194	0.0097	0.0081
Biased-MF	0.0008	0.0008	0.0002	0.0004	0.0007	0.0008	0.0002	0.0002
BPR-MF	0.0761	0.0603	**0.1022**	**0.1477**	0.0907	0.0997	0.0611	0.0691
SVD++	0.0017	0.0011	0.0001	0.0001	0.0016	0.0013	0.0010	0.0006
CDAE	0.0595	0.0887	0.0533	0.0869	0.0664	0.0782	0.0428	0.0613
U-AutoRec	0.0691	0.0986	0.0609	0.0962	0.0932	0.1017	0.0631	0.0718
ACDA-R	**0.0827**	**0.1106**	0.0723	0.1070	**0.1094**	**0.1189**	**0.0694**	**0.0835**

The basic *ACDA* model (without the contextual attributes) was not included into the comparison as that is basically the *U-AutoRec* model, which has been considered as a baseline method. The proposed *ACDA* models are trained on training set, and then evaluated on the cross-validation set for selecting the appropriate values for the hyper-parameters. Finally, the model is evaluated on the test set, the results of which are published for comparison with the baselines. The proposed models are developed and trained using Google's tensorflow library[5]. Additional experiments were conducted to determine the optimal value for the hidden unit size and corruption ratio hyper-parameters. The results of the additional experiments are provided in Section 4.3.3.

The $epoch = 200$ was set during training as the model was found to converge at this point. Different learning rates $(0.1, 0.01, 0.05, 0.001, 0.005)$ were experimented with, and the learning rate $\alpha = 0.001$ was found to work best. To prevent the model from just training on positive samples, the positive samples of a user were paired with a configurable number of negative or

[5]http://www.tensorflow.org

unknown samples for the user.

4.3.3 The Effects of Number of Hidden Units and Corruption Ratio

To investigate the effect of the number of hidden units (k) on the performance, the value of k was varied from 100 to 1000 in increments of 100. The results are provided in Figure 4.2. As observed from the plots, the performance of the model plateaus after $k = 500$, with higher values offering no significant gain in performance at a cost of increased training time. While there are certain metrics, such as the *NDCG@5*, that perform slightly better at higher values, $k = 500$ was set as a default choice.

Different values of the corruption ratio were tried, ranging from 0.1 to 0.9 in increments of 0.1. The results, depicted in Figure 4.3, indicate that the performance degrades with higher values of the corruption ratio. The only exception to this is the *Meetup-Chicago* dataset, which does not have a observable degradation in performance at higher values of the corruption ratio. Therefore, the value of the corruption ratio is defaulted to $\rho = 0.2$.

4.3.4 Baseline Comparisons

Tables 4.2, 4.3, 4.4, 4.5, 4.6 contains the results of the different methods, with the best results highlighted in boldface. A general observation is that, other than a few exceptions, the results on the *precision*, *recall*, *NDCG* and *MAP* metrics are consistent across all the datasets. The proposed model performed well on the *Meetup* and *Movielens* datasets, which demonstrates its effectiveness on top-N recommendation tasks.

First, the performance of the baseline methods is discussed. Three different categories of the baseline methods are considered: neighborhood-based, model-based and deep learning based methods. Among these categories, the deep learning based baseline methods are found to perform better than the others. In general, across the baseline methods, the *CDAE* deep learning based method performs better on the precision and recall metrics. The *BPR-MF* is better on the *NDCG* metric. The *CDAE* method is based on the denoising autoencoder, and the results signify its importance to recommender systems. The good performance of the *BPR-MF* method may be attributed to the use of the pairwise loss function. The *BPR-MF* method also performed well against the *Movielens* dataset. However, it is observed that the *U-AutoRec* performs better than *CDAE* against the *Movielens* dataset. This suggests that the user latent factor included in the *CDAE* model does not help to improve the performance against the *Movielens* dataset, but it does so against the *Meetup* dataset. When considering the neighborhood methods, both (*UserKNN* and *ItemKNN*) were found to be similar in performance.

Comparing the baseline methods to the proposed models for the event recommendation task, it is observed that all three variants of the proposed *ACDA* model perform better than the baselines. While a variant of the *ACDA* model with some of the contextual attributes may perform better, in general the model with more contextual attributes offers the better performance. As it is observed for the *Meetup* datasets, the *ACDA-GV* model offers a better performance in three of the four cities. It is also observed that the significance of the contextual attributes is not equal. The influence of the user *group* contextual attribute is higher than the event *venue* attribute, and the model *ACDA-G* performs better than the *ACDA-V* model. This implies that additional contextual parameters may improve the performance further in some cases, however, this may not be always true. With regard to the movie recommendation task, the movie *genre* is utilized as a contextual attribute. The model *ACDA-R* performs better on all metrics except *recall*. The *BPR-MF* method is better on the *recall* metric, perhaps due to the fact that it uses a pairwise loss function. Future work will aim to evaluate the performance of the proposed models using pairwise loss. The results, which are consistent across all datasets, reinforce the assertion that the proposed *ACDA* model performs well on recommendation tasks.

4.4 Summary

This study proposes a deep learning architecture for contextual recommendation based on the denoising autoencoder augmented with a context-driven attention mechanism. The proposed

ACDA architecture is a generic model that can be used for both rating prediction and top-N recommendation. It is demonstrated via comprehensive experiments that the proposed *ACDA* model performs better than the state-of-the-art baselines on the event recommendation and movie recommendation tasks.

Chapter 5

Conclusion & Future Work

5.1 Thesis Summary

This study proposes two novel methods for personalized recommendation. The first model, which is called the *Group-Aware Latent Factor Model (GLFM)*, is a context-aware latent factor model realized using matrix factorization. The *GLFM* model applies contextual attributes from a dual-perspective of both user and item, and the model is extensible to allow for additional contextual attributes. The proposed *GLFM* model is utilized for the task of event recommendation, and evaluated against multiple real-world datasets from *Meetup*, a popular *Event-Based Social Network (EBSN)*. Experimental results demonstrate the superior performance of the *GLFM* model against the state-of-the-art baseline methods.

Additionally, this study also proposes a second method for personalized recommendation, which is based on the denoising autoencoder neural network architecture. The proposed model is referred to as the *Attentive Contextual Denoising Autoencoder (ACDA)*, and it is built on the denoising autoencoder with a context-driven attention mechanism. The *ACDA* model is applied to the task of event recommendation and movie recommendation. For the event recommendation task, the *ACDA* model utilizes the user *group* and event *venue* as contextual attributes. Whereas for the movie recommendation task, the movie *genre* is utilized as a contextual attribute. The *ACDA* model is evaluated on multiple real-world datasets from

Meetup and *Movielens*, and it is found to perform better in comparison to the other state-of-the-art recommenders.

5.2 Key Observations

This study proposes different methods for providing context-aware personalized recommendation. The basis of the study are the proposed machine learning models for recommendation. However, the study also attempts to evaluate the role of context for personalized recommendation and other considerations such as pairwise loss, attention mechanism, etc. The key findings of this report are:

- An important takeaway from this study is that context plays a pivotal role in personalized recommendation. This study evaluates variants of the proposed *GLFM* and *ACDA* models without the contextual attributes. It is observed that the model variant without the contextual attribute does not perform as well the one with contextual attributes. The experiments conducted against the variants of the proposed models also suggest that certain contextual attributes are more important than the others. As an example, the variant of both the *GLFM* and *ACDA* models that considers the *group* contextual attribute performs better than the variants that consider other contextual attributes. This signifies the importance of the *group* contextual attribute for event recommendation. It is also observed that additional contextual attributes help to improve the performance of the model, as variants of the *GLFM* and *ACDA* models that consider all the relevant contextual attributes perform the best against almost all the datasets. These findings validate our assertion that context attributes are essential for a meaningful recommendation.
- Pair-wise ranking is an important consideration for top-N recommendation. Implicit feedback datasets lack negative feedback. The user actions captured implicitly provide only the positive feedback. As an example, the event RSVP provided by the user acts as a positive preference of the user for that event; however, the user may ignore certain events or may not be aware of their existence. Pair-wise ranking performs the evaluation

in positive-negative pairs, where negative or unknown preferences are sampled from the dataset and paired with known positive preferences. The *GLFM* model is evaluated using both pair-wise and point-wise loss, and the performance of the model is much better with a pair-wise loss.

- The attention mechanism has been effective at incorporating the contextual attributes to the *ACDA* model. The attention mechanism is applied to the latent data representation in the hidden layer, which is effective due to the low dimensionality of the data representation. Keeping in mind the number of available contextual parameters, it makes sense to apply it to lower dimensional data representation, instead of the high dimensional input data.

5.3 Future Work

This report is a preliminary study with an objective of ascertaining the impact of contextual attributes on personalized recommendation within the scope of two novel machine learning methods. The work presented here may be extended in the following ways:

- Both the *GLFM* and *ACDA* models presented in this study are used for top-N tasks of event and movie recommendation. The proposed models are generic and extensible, and it would be a good enhancement to extend and evaluate them on rating prediction problems.
- It would be beneficial to study the effect of learning to rank, and evaluate the proposed models by considering other loss functions, such as list-wise loss.
- This study may be extended by applying these novel methods to other recommendation tasks that provide contextual data.
- Since the *ACDA* model is based on the neural network architecture, extending it to be a deep model with additional hidden layers may be considered. This may be performed in

conjunction with experimenting with different activation functions to study their impact on the performance of the models.

- Finally, this study may be extended to utilize social ties for providing recommendations to a group of people. For example, recommending a vacation package to a group of friends on a social network.

Bibliography

[AC09] Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 19–28. ACM, 2009.

[AK14] Alan Agresti and Maria Kateri. Some remarks on latent variable models in categorical data analysis. *Communications in Statistics-Theory and Methods*, 43(4):801–814, 2014.

[ASST05] Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23(1):103–145, 2005.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[BHK98] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[BKV07] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proceedings* *of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104. ACM, 2007.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[Bot10] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT*, pages 177–186. Springer, 2010.

[Cao15] Longbing Cao. Coupling learning of complex interactions. *Information Processing & Management*, 51(2):167–186, 2015.

[CCB15] Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11):1875–1886, 2015.

[CCZ+12] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. Collaborative personalized tweet recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 661–670. ACM, 2012.

[CS16] Chien Chin Chen and Yu-Chun Sun. Exploring acquaintances of social network site users for effective social event recommendations. *Information Processing Letters*, 116(3):227–236, 2016.

[CWSB14] Minmin Chen, Kilian Weinberger, Fei Sha, and Yoshua Bengio. Marginalized denoising auto-encoders for nonlinear representations. In *ICML*, pages 1476–1484, 2014.

[CZH+17] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *ACM SIGIR RDIR*, pages 335–344. ACM, 2017.

[CZL+12] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. Svdfeature: a toolkit for feature-based collaborative filtering. *The Journal of Machine Learning Research*, 13(1):3619–3622, 2012.

[CZW+07] Rui Cai, Chao Zhang, Chong Wang, Lei Zhang, and Wei-Ying Ma. Musicsense: contextual music recommendation using emotional allocation modeling. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 553–556. ACM, 2007.

[dMM14] Augusto Q de Macedo and Leandro B Marinho. Event recommendation in event-based social networks. In *Hypertext, Social Personalization Workshop*, 2014.

[DYM+14] Rong Du, Zhiwen Yu, Tao Mei, Zhitao Wang, Zhu Wang, and Bin Guo. Predicting activity attendance in event-based social networks: Content, context and social influence. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 425–434. ACM, 2014.

[FFCC13] Vreixo Formoso, Diego FernaNdez, Fidel Cacheda, and Victor Carneiro. Using profile expansion techniques to alleviate the new user problem. *Information Processing & Management*, 49(3):659–672, 2013.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[GDFST12] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. Personalized ranking for non-uniformly sampled items. In *KDD Cup*, pages 231–247, 2012.

[GZ16] Yuyun Gong and Qi Zhang. Hashtag recommendation using attention-based convolutional neural network. In *IJCAI*, pages 2782–2788, 2016.

[HKG+15] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701, 2015.

[HLZ+17a] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, 2017.

[HLZ+17b] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, pages 173–182. ACM, 2017.

[HSL14] Longke Hu, Aixin Sun, and Yong Liu. Your neighbors affect your ratings: on geographical neighborhood influence to rating prediction. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 345–354. ACM, 2014.

[JL16] Jyun-Yu Jiang and Cheng-Te Li. Analyzing social event participants for a single organizer. In *Tenth International AAAI Conference on Web and Social Media*, 2016.

[KABO10] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86. ACM, 2010.

[KB15] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015.

[KBV09] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[KGDFST12] 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 fifth ACM international conference on Web search and data mining*, pages 173–182. ACM, 2012.

[KKB16] Ondrej Kassak, Michal Kompan, and Maria Bielikova. Personalized hybrid recommendation for group of users: top-n multimedia recommender. *Information Processing & Management*, 52(3):459–477, 2016.

[Kor08] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[Kor10] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[KT13] Houda Khrouf and Raphaël Troncy. Hybrid event recommendation using linked data and user diversity. In *RecSys*, pages 185–192. ACM, 2013.

[LC16] Shu-hsien Liao and Hsiao-ko Chang. A rough set-based association rule approach for a recommendation system for online consumers. *Information Processing & Management*, in press, 2016.

[LCL+15] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 433–442. ACM, 2015.

[LCLS10] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[LS17] Xiaopeng Li and James She. Collaborative variational autoencoder for recommender systems. In *KDD*, pages 305–314. ACM, 2017.

[LVT+16] Di Lu, Clare R Voss, Fangbo Tao, Xiang Ren, Rachel Guan, Rostyslav Korolov, Tongtao Zhang, Dongang Wang, Hongzhi Li, Taylor Cassidy, et al. Cross-media event extraction and recommendation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL-HLT)*, 2016.

[LY08] Nathan N Liu and Qiang Yang. Eigenrank: a ranking-oriented approach to collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 83–90. ACM, 2008.

[LZY09] Nathan N Liu, Min Zhao, and Qiang Yang. Probabilistic latent preference analysis for collaborative filtering. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 759–766. ACM, 2009.

[MBAG16] Eder F Martins, Fabiano M Belém, Jussara M Almeida, and Marcos A Gonçalves. On cold start for associative tag recommendation. *Journal of the Association for Information Science and Technology*, 67(1):83–105, 2016.

[MMS15] Augusto Q Macedo, Leandro B Marinho, and Rodrygo LT Santos. Context-aware event recommendation in event-based social networks. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 123–130. ACM, 2015.

[MN89] Peter McCullagh and John A Nelder. *Generalized linear models*, volume 37. CRC press, 1989.

[MRS+08] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[MS07] Andriy Mnih and Ruslan Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2007.

[MYHL11] Tao Mei, Bo Yang, Xian-Sheng Hua, and Shipeng Li. Contextual video recommendation by multimodal relevance and user feedback. *ACM Transactions on Information Systems (TOIS)*, 29(2):10, 2011.

[NH10] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[PB07] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[PC13] Weike Pan and Li Chen. Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering. In *IJCAI*, volume 13, pages 2691–2697, 2013.

[PLCZ15] Tuan-Anh Nguyen Pham, Xutao Li, Gao Cong, and Zhenjie Zhang. A general graph-based model for recommendation in event-based social networks. In *Proceedings of IEEE International Conference on Data Engineering*, pages 567–578. IEEE, 2015.

[PN13] Maria S Pera and Yiu-Kai Ng. A group recommender for movies based on content similarity and popularity. *IPM*, 49(3):673–687, 2013.

[Ren10] Steffen Rendle. Factorization machines. In *ICDM*. IEEE, 2010.

[RFGST09] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[RST10] Steffen Rendle and Lars Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 81–90. ACM, 2010.

[SG08] Ajit P Singh and Geoffrey J Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658. ACM, 2008.

[SHYL17a] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *RecSys*, 2017.

[SHYL17b] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. Representation learning of users and items for review rating prediction using attention-based convolutional neural network. In *MLRec*, 2017.

[SK] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques.

[SKKR01] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[SLH13] Yue Shi, Martha Larson, and Alan Hanjalic. Mining contextual movie similarity with matrix factorization for context-aware recommendation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(1):16, 2013.

[SMH07] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *ICML*, pages 791–798. ACM, 2007.

[SMSX15] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *WWW*, pages 111–112. ACM, 2015.

[SWCF15] Jianshan Sun, Gang Wang, Xusen Cheng, and Yelin Fu. Mining affective text to improve social media item recommendation. *Information Processing & Management*, 51(4):444–457, 2015.

[SZW+12] Yue Shi, Xiaoxue Zhao, Jun Wang, Martha Larson, and Alan Hanjalic. Adaptive diversification of recommendation results via latent factor portfolio. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 175–184. ACM, 2012.

[VLBM08] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103. ACM, 2008.

[VTBE15] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164. IEEE, 2015.

[WB11] Chong Wang and David M Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456. ACM, 2011.

[WDZE16] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *WSDM*, pages 153–162. ACM, 2016.

[WWY15] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *KDD*, pages 1235–1244. ACM, 2015.

[WYZ+17] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *SIGIR*, 2017.

[XBK+15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICL*, pages 2048–2057, 2015.

[XYH+17] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *IJCAI*, 2017.

[YSQ+15] Lina Yao, Quan Z Sheng, Yongrui Qin, Xianzhi Wang, Ali Shemshadi, and Qi He. Context-aware point-of-interest recommendation using tensor factorization with social regularization. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1007–1010. ACM, 2015.

[ZMK15] Tong Zhao, Julian McAuley, and Irwin King. Improving latent factor models via personalized feature projection for one class recommendation. In *Proceedings* *of the 24th ACM International on Conference on Information and Knowledge Management*, pages 821–830. ACM, 2015.

[ZW15] Wei Zhang and Jianyong Wang. A collective bayesian poisson factorization model for cold-start local event recommendation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1455–1464. ACM, 2015.

[ZWF13] Wei Zhang, Jianyong Wang, and Wei Feng. Combining latent factor model with location features for event-based group recommendation. In *SIGKDD*, pages 910–918. ACM, 2013.

[ZYS17] Shuai Zhang, Lina Yao, and Aixin Sun. Deep learning based recommender system: A survey and new perspectives. *arXiv:1707.07435*, 2017.

[ZYX17] Shuai Zhang, Lina Yao, and Xiwei Xu. Autosvd++: An efficient hybrid collaborative filtering model via contractive auto-encoders. *arXiv preprint arXiv:1704.00551*, 2017.