University of Louisville

ThinkIR: The University of Louisville's Institutional Repository

Electronic Theses and Dissertations

8-2017

Accurate and justifiable : new algorithms for explainable recommendations.

Behnoush Abdollahi
University of Louisville

Follow this and additional works at: <https://ir.library.louisville.edu/etd>

Part of the [Artificial Intelligence and Robotics Commons](#), and the [Other Computer Sciences Commons](#)

Recommended Citation

Abdollahi, Behnoush, "Accurate and justifiable : new algorithms for explainable recommendations." (2017). *Electronic Theses and Dissertations*. Paper 2744.
<https://doi.org/10.18297/etd/2744>

This Doctoral Dissertation is brought to you for free and open access by ThinkIR: The University of Louisville's Institutional Repository. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of ThinkIR: The University of Louisville's Institutional Repository. This title appears here courtesy of the author, who has retained all other copyrights. For more information, please contact thinkir@louisville.edu.

**ACCURATE AND JUSTIFIABLE: NEW ALGORITHMS FOR
EXPLAINABLE RECOMMENDATIONS**

By

Behnoush Abdollahi
M.Sc., Computer Engineering and Computer Science,
University of Louisville, Louisville, KY

A Dissertation
Submitted to the Faculty of the
J.B. Speed School of Engineering of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy in Computer Science and Engineering

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

August 2017

Copyright 2017 by Behnoush Abdollahi

All rights reserved

ACCURATE AND JUSTIFIABLE: NEW ALGORITHMS FOR EXPLAINABLE RECOMMENDATIONS

By

Behnoush Abdollahi
M.Sc., Computer Engineering and Computer Science,
University of Louisville, Louisville, KY

A Dissertation Approved On

August 4, 2017

by the following Dissertation Committee:

Dr. Olfa Nasraoui, Dissertation Director

Dr. Nihat Altiparmak

Dr. Adrian Lauf

Dr. Scott Sanders

Dr. Jacek Zurada

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Olfa Nasraoui for her endless support and encouragement. Without her patience, motivation and guidance in all aspects of being a good research scientist, this experience would not have been the same. I would like to extend my gratitude to the rest of my committee members: Dr. Nihat Altiparmak, Dr. Adrian Lauf, Dr. Scott Sanders, and Dr. Jacek Zurada, for their valuable and constructive feedback. In addition, I would like to thank Dr. Ming Ouyang, Dr. Ayman El-baz and Dr. Simone Stumpf who gave me the opportunity to learn from them and provided insight and expertise that greatly assisted my research.

I wish to express my thanks to the Computer Science and Computer Engineering Department and Speed School of Engineering and School of Interdisciplinary and Graduate Studies for providing me the opportunity to pursue my degree and for their unfailing support and assistance throughout my graduate studies. I would also like to thank the Kentucky Science and Engineering Foundation for partially providing the funding for the work.

My sincere thanks goes to my fellow lab-mates, with a special mention to Wen-long Sun and Gopi Chand Nutakki. Our conversations on research and their good-hearted support and friendship will not be forgotten.

Last but by no means least, I am so grateful to my family and friends who have supported me in every possible way unconditionally and selflessly. Many thanks to my sister Behnaz Abdollahi for encouraging me and expressing confidence in my abilities when I could only do the opposite. Moreover, thanks to Milad Nikoukar for motivating me to keep reaching for excellence.

ABSTRACT

ACCURATE AND JUSTIFIABLE: NEW ALGORITHMS FOR EXPLAINABLE RECOMMENDATIONS

Behnoush Abdollahi

August 4, 2017

Websites and online services thrive with large amounts of online information, products, and choices, that are available but exceedingly difficult to find and discover. This has prompted two major paradigms to help sift through information: information retrieval and recommender systems. The broad family of information retrieval techniques has given rise to the modern search engines which return relevant results, following a user’s explicit query. The broad family of recommender systems, on the other hand, works in a more subtle manner, and do not require an explicit query to provide relevant results. Collaborative Filtering (CF) recommender systems are based on algorithms that provide suggestions to users, based on what they like and what other similar users like. Their strength lies in their ability to make serendipitous, social recommendations about what books to read, songs to listen to, movies to watch, courses to take, or generally any type of item to consume. Their strength is also that they can recommend items of any type or content because their focus is on modeling the preferences of the users rather than the content of the recommended items.

Although recommender systems have made great strides over the last two decades, with significant algorithmic advances that have made them increasingly accurate in their predictions, they suffer from a few notorious weaknesses. These include the cold-start

problem when new items or new users enter the system, and lack of interpretability and explainability in the case of powerful black-box predictors, such as the Singular Value Decomposition (SVD) family of recommenders, including, in particular, the popular Matrix Factorization (MF) techniques. Also, the absence of any explanations to justify their predictions can reduce the transparency of recommender systems and thus adversely impact the user’s trust in them. In this work, we propose machine learning approaches for multi-domain Matrix Factorization (MF) recommender systems that can overcome the new user cold-start problem. We also propose new algorithms to generate explainable recommendations, using two state of the art models: Matrix Factorization (MF) and Restricted Boltzmann Machines (RBM). Our experiments, which were based on rigorous cross-validation on the MovieLens benchmark data set and on real user tests, confirmed that our proposed methods succeed in generating explainable recommendations without a major sacrifice in accuracy.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF ALGORITHMS	xiv

CHAPTER

1	INTRODUCTION	1
1.1	Problem Statement	4
1.1.1	Assumptions	5
1.2	Research Contributions	5
2	BACKGROUND AND LITERATURE REVIEW	7
2.1	Recommendation Systems	7
2.1.1	Memory-based Methods	10
2.1.2	Model-based Methods	11
2.1.3	The Sparsity Problem	12
2.1.4	The Cold start problem	13
2.2	Latent Factor Models	14
2.2.1	Principal Component Analysis (PCA)	14
2.2.2	Singular Value Decomposition (SVD)	15
2.2.3	Matrix Factorization (MF)	15
2.2.4	Restricted Boltzmann Machines (RBM)	18
2.3	Comprehensible Classification Models	19
2.4	Explanation Mechanisms in Collaborative Filtering	22

2.4.1	Benefits of Explanation	23
2.4.2	Explanation Approaches and Algorithms	24
2.4.3	Explanation Evaluation	30
2.5	Summary	31
3	PROPOSED WORK	32
3.1	A Generalized Asymmetric MF-based Framework in Collaborative Filtering	33
3.2	A Warm-Up Solution for the Cold-Start Problem	34
3.3	Explanation Generation Using Asymmetric MF	36
3.3.1	NSE-based Explanation Generation in the Latent Space	36
3.3.2	ISE-based Explanation Generation in the Latent Space	37
3.4	Explainability	39
3.4.1	NSE-based Explainability	39
3.4.2	ISE-based Explainability	41
3.4.3	KSE-based Explainability	41
3.5	Explainability Graph	42
3.6	Explainable Matrix Factorization	43
3.6.1	Proof of Convergence of EMF	44
3.6.2	Explainability Effect in the Latent Space	46
3.7	Explainable Restricted Boltzmann Machines	48
3.8	Summary	50
4	EXPERIMENTAL EVALUATION	52
4.1	Cross-modal MF-based CF	52
4.2	Explainable Matrix Factorization (EMF)	53
4.2.1	Recommender System Evaluation	61
4.2.2	Explainability Evaluation	63
4.3	Explainable Restricted Boltzmann Machines (ERBM)	72

4.4	User Study	80
4.4.1	Hypothesis	82
4.4.2	Methods	82
4.4.3	Subject Recruitment	83
4.4.4	Sample Size Estimation	83
4.4.5	Procedures	83
4.4.6	Analysis of Results	86
4.4.7	Hypothesis Testing	92
4.5	Summary	94
5	CONCLUSION	96
	REFERENCES	98
	CURRICULUM VITAE	105

LIST OF TABLES

TABLE		Page
3.1	Number of explainable items in the top-10 recommendation generated using EMF has larger mean and standard deviation than using MF technique. Paired T-test on the difference showed the EMF has improved at 1% level or higher ($p - value = 7.784e - 12$).	49
4.1	Top-5 rated movies by Sample User A, along with the movies' genres. . . .	53
4.2	Top-3 recommended movies to Sample User A, along with the movies' genres.	54
4.3	Top-5 rated movies by Sample User B, along with the movies' genres. . . .	54
4.4	Top-3 recommended movies to Sample User B, along with the movies' genres.	57
4.5	Performance of the EMF when varying θ	71
4.6	Performance of the ERBM when varying θ	74
4.7	Top-5 rated movies by a sample user along with the movies' genres. . . .	80
4.8	Likert scale survey questions.	86
4.9	Demographic questions.	86
4.10	Categorization of the survey questions from Table 4.8 according to the research questions.	93
4.11	Tukey multiple comparisons of means at 95% family-wise confidence interval for transparency.	94
4.12	Tukey multiple comparisons of means at 95% family-wise confidence interval for satisfaction.	94

LIST OF FIGURES

FIGURE		Page
1.1	Four different explanation style examples.	3
2.1	A user-book ratings matrix, representing ratings of books on a scale of 1-5.	9
2.2	Collaborative filtering process.	10
2.3	Decomposed matrices with rank 2.	16
2.4	An RBM network with n hidden and m visible units.	20
3.1	Example of multiple-domain movie data.	32
3.2	Asymmetric multi domain MF.	35
3.3	An NSE-based explanation showing the user's neighbors' ratings.	37
3.4	An ISE-based explanation showing the user's ratings to the recommended item's neighbors.	38
3.5	Four different explanation style formats: (1) NSE,(2) ISE ,(3) NSE,(4) KSE.	40
3.6	An example of explainability graph. The blue and the purple nodes are the users and all the other nodes are the items. For the sample user (in color blue), the explainable items are shown in color green.	43
3.7	Red points = explainable items, green points = relevant items, Cyan points = remaining items, black point = sample user. For two sample users (top and bottom rows), the items are represented in the latent space ($f = 2$, for visualization purpose). In each case, EMF has resulted in explainable items to be located closer to the user and in the green area (relevant items for recommendations).	48
3.8	Conditional RBM for explainable recommendations.	50
4.1	Genre frequency of the rated movies by Active User A.	54

4.2	Corresponding Top-3 NSE-based explanations generated for the recommendations shown in Table 4.2. Recommendations/explanations are generated using asymmetric MF for Sample User A.	55
4.3	Corresponding Top-3 ISE-based explanations generated for the recommendations shown in Table 4.2. Recommendations/explanations are generated using asymmetric MF for Sample User A.	56
4.4	Genre frequency of the rated movies by Active User B.	57
4.5	Corresponding Top-3 NSE-based explanations generated for the recommendations shown in Table 4.4. Recommendations/explanations are generated using asymmetric MF for Sample User B.	58
4.6	Corresponding Top-3 ISE-based explanations generated for the recommendations shown in Table 4.4. Recommendations/explanations are generated using asymmetric MF for Sample User B.	59
4.7	RMSE calculated over the testing data with each iteration decreases. The setting for this experiment is $\alpha = 0.001$ and $\beta = 0.01$, when varying f	60
4.8	Comparison of accuracy when varying the percentage of data used in the test. . . .	61
4.9	Objective function, J , calculated over the training data in each iteration decreases until convergence. The setting for this experiment is $f = 5$, $ N = 50$, and $\theta = 0$ for all three explanation styles.	62
4.10	Comparison of accuracy when varying f	63
4.11	Comparison of accuracy when varying $ N $	64
4.12	Comparison of user based xP , xR , and xF_score when varying f . For EMF techniques we set $ N = 50$	65
4.13	Comparison of item based xP , xR , and xF_score when varying f . For EMF techniques we set $ N = 50$	66
4.14	Comparison of feature based xP , xR , and xF_score when varying f . For EMF techniques we set $ N = 50$	67
4.15	Comparison of user based xP , xR , and xF_score when varying $ N $	68

4.16	Comparison of item based xP , xR , and xF_score when varying $ N $	69
4.17	Comparison of accuracy when varying f . For EMF techniques we set $ N(u) = 50$	73
4.18	Comparison of accuracy when varying $ N $	74
4.19	Comparison of user based xP , xR , and xF_score when varying f . For $ERBM$ techniques we set $ N = 50$	75
4.20	Comparison of item based xP , xR , and xF_score when varying f . For $ERBM$ techniques we set $ N = 50$	76
4.21	Comparison of feature based xP , xR , and xF_score when varying f . For $ERBM$ techniques we set $ N = 50$	77
4.22	Comparison of user based xP , xR , and xF_score when varying $ N $	78
4.23	Comparison of item based xP , xR , and xF_score when varying $ N $	79
4.24	Example of an automated explanation generated for the recommended movie: “Lost Highway (1997)” using the EMF_{NSE} technique.	80
4.25	Example of the automated explanation generated for the recommended movie: “McHale’s Navy (1997)” using the EMF_{ISE} technique.	81
4.26	Example of the automated explanation generated for the recommended movie: “That Darn Cat! (1965)” using the EMF_{KSE} technique.	81
4.27	Comparison between two explanations for a test user where the explainability of the item i_1 on the left is higher than item i_2 one on the right.	82
4.28	Example of movies shown to the user to rate.	84
4.29	Example of a recommendation with explanation presented to the user.	85
4.30	Horizontal bar graph of the answers to the questions in Table 4.8.	87
4.31	Horizontal bar graph of the answers of people in the group “low” to the questions in Table 4.8.	88
4.32	Horizontal bar graph of the answers of people in the group “medium” to the questions in Table 4.8.	88

4.33	Horizontal bar graph of the answers of people in the group "high" to the questions in Table 4.8.	88
4.34	Heat-map plot of the answers to the questions in Table 4.8.	89
4.35	Heat-map plot of the answers of people in the group "low" to the questions in Table 4.8.	89
4.36	Heat-map plot of the answers of people in the group "medium" to the questions in Table 4.8.	90
4.37	Heat-map plot of the answers of people in the group "high" to the questions in Table 4.8.	90
4.38	Distribution of the participants' gender.	91
4.39	Distribution of the participants' age.	91
4.40	Distribution of the participants' weekly hours watching movies.	91
4.41	Distribution of the participants' familiarity with recommender systems. . .	92
4.42	Distribution of the participants' favorite genres.	92
4.43	Visualization of group pairs' differences for "transparency".	95
4.44	Visualization of group pairs' differences for "satisfaction".	95

LIST OF ALGORITHMS

ALGORITHM	Page
2.1 Basic Alternating Least Square (ALS) Algorithm	17
2.2 Basic SGD Algorithm for MF	18
3.1 NSE-Based Explanation Generation Algorithm in the Latent Space	37
3.2 ISE-Based Explanation Generation Algorithm in the Latent Space	38
3.3 Explainable Matrix Factorization (EMF)	47

CHAPTER 1

INTRODUCTION

Machine learning (ML) models are being used increasingly in many sectors, ranging from health and education to e-commerce and criminal investigation. Hence, these Artificial Intelligence (AI) systems are starting to affect the lives of more and more human beings. Examples include risk modeling and decision making in insurance, education (admission and success prediction), credit scoring, medical, criminal investigation and predicting recidivism, etc. Without the intelligent systems' ability to *explain* their decisions and actions to the human users, the effectiveness of these systems can be limited. Users may require *understanding* and *trusting* predictions made by these systems before making decisions with an inherent risk, as a result of these predictions [1].

Furthermore, AI models are susceptible to bias that stems from the data itself or from systemic social biases that generated the data (e.g. recidivism, arrests). As such, models that are learned from real world data can become unethical if their outputs discriminate, albeit unintentionally, against a certain group of people. While building ethical and fair models seems like the ultimate and ideal goal, the minimum and urgent criterion that ML models should satisfy is *transparency*, and this could be the first step in the direction toward fair and ethical models. Therefore, designing *explainable intelligent systems* that facilitate conveying the reasoning behind the results is of great importance.

Recommender systems are a special kind of ML AI systems that suggest interesting items to users in various decision making situations, with suggested items ranging from songs and books to courses and news. Recommender systems represent a valuable means of helping online users with information overload and they have recently become powerful pillars in e-commerce and information retrieval (IR). Collaborative filtering (CF) recommendation

systems try to suggest items of interest (e.g., movies, songs, books, applications, websites and travel destinations) to a user based on their user profile which can be explicit (e.g. user ratings) or implicit (e.g. their browsing or purchase history) [2,3]. CF recommender systems provide recommendations to users based on the similarity between users or between items, giving rise to neighborhood-based CF approaches, which can be user-based or item-based. Other sources of information such as users' demographic data and items' features may be used to provide a user with a list of items that maximize the user's utility or satisfaction. These recommender systems exploit rating or purchase history as their main source of input information. When there is not enough history available for new users or new items in the system, a CF system cannot provide good recommendations. This is known as the cold-start problem, and can be considered as an extreme case of sparse input data.

It is important for a recommender system to provide explanations for recommendations. In fact, most commercial recommender systems provide simple explanations to the users, and this can enhance the user's trust and acceptance. Even long before automated recommender systems, a medical expert, system's explanation of the reasoning behind a suggestion, has been found to be critical to the users' acceptance of the system's suggestion [4]. Amazon's recommender system shows similar items that the user (or other similar users) have bought or viewed, when recommending a new item. The Netflix recommender system justifies its movie suggestions by listing similar movies obtained from the user's social network. When an interpretable model is used in a recommender system, it has been shown that explanations can help users make more accurate decisions; hence, improving user satisfaction and acceptance of recommendations [5–7].

In Figure 1.1, four different explanation examples are shown, where (1) and (3), are based on the ratings of the active user's most similar users (neighbors) that is called Neighbor Style Explanation (NSE); (3) is a feature-based explanation that is based on the content and keyword features, and is called Keyword Style Explanation (KSE); while (4) is based on the active users ratings of items that are similar to the recommended item, and is called Item Style Explanation (ISE) [6].

Your neighbors' ratings for this movie

Rating	Number of Neighbors
★	0
★★	0
★★★	3
★★★★	4
★★★★★	2

(1)

Our recommendation is "Pulp Fiction", because you rated similar movies:

Movie	Your Rating
From Dusk Till Dawn (1996)	2
Seven (Se7en) (1995)	4
Usual Suspects The (1995)	4

(2)

Ratings of the People Who Share Your Interests and Have Watched This Movie

(3)

Our Justified Recommendations:

Movie	The reason is	Because you rated
Scream 2	horror	25 movies with this feature
Kiss the Girls	Freeman, Morgan	19 movies with this feature
The Peacemaker	Clooney, George	10 movies with this feature

(4)

Figure 1.1: Four different explanation style examples.

Machine Learning techniques used for recommender systems can be categorized in two families: White-Box methods (WB), such as Decision Trees [8,9], and production rule learners [10,11], which are interpretable and essentially come with explanations. On the other hand, Black-Box (BB) methods such as Artificial Neural Networks [12], Support Vector Machines [13], Ensemble Methods [14], and Matrix Factorization (MF) [15,16] produce opaque models which are not easily interpretable by humans. Most accurate recommender systems, available nowadays, developed mainly in research labs and in some cases for commercial use, are BB models. The most popular BB models, for recommender systems, nowadays are based on Matrix Factorization models, with deep learning networks, also recently becoming another popular BB model. MF methods are accurate CF approaches that map users and items to low-dimensional feature vectors [17]. MF methods fit a model to the known ratings and use the model for further predictions. In most MF-based techniques, predictions are not interpretable and cannot be justified to the user as easily as in neighborhood-based CF methods. Although a few of these provide some form of explanation, it is not clear why the system is recommending a specific item, which may result in users not trusting the suggestions of the recommender system. One way to communicate

explanations would be based on identifying similar users and/or items in the latent space and presenting the most similar users and/or items as the explanation. The drawback of this method is that the way the explanation is generated does not necessarily comply with the learned ML model. This is because the solution to the MF optimization problem does not guarantee that the most similar users to an active user, who have liked the system’s suggestion, are necessarily the active user’s neighbors in the latent space. Thus, the only way to assess the quality of the recommendation for the user is to try the item. This, however, is contrary to one of the goals of a recommendation system, which is reducing the time that users spend on exploring items.

It would be very desirable and beneficial to design recommender systems that can give accurate suggestions, which, at the same time, facilitate conveying the reasoning behind the recommendations to the user. However, a main challenge in designing a recommender system is whether to choose an explainable technique with moderate prediction accuracy or a more accurate technique (such as MF) which does not give explainable recommendations.

1.1 Problem Statement

Due to the trade-off between a machine learning model’s prediction accuracy and its interpretability [18], it is generally very challenging to achieve both accuracy and explainability. Therefore, it is common, in many applications, to opt for accuracy at the expense of interpretability.

Our research question is: can we design a model-based recommender engine that suggests items that are *explainable*, while recommendations remain accurate? In order to answer this question, we define explainability and hypothesize that recommendations that have higher explainability value, help users make better decisions, therefore they are more effective.

1.1.1 Assumptions

We assume that the BB recommender system is mainly rooted in a latent factor-based or deep learning model using ratings as the primary input and possibly augmented with item or user attributes into a hybrid CF system. We also assume that the input to the recommendations and the input to the explanations are not constrained to be exactly the same. This is because an input can contribute to empowering a BB model’s training, while not necessarily being a good source of human explanation. Most studies in the literature study the effect of explanations as a separate module from the recommender module [5,6,19]. The focus of their studies are mostly on the types and formats or visualizations of the explanations regardless of the recommender module [19,20].

Our current scope is limited to CF recommendations where explanations for recommended items are in one of the three main explanation styles: NSE, ISE or KSE, as shown in Figure 1.1. We encode the user-item explainability relationship in a graph. While other methods in the literature have used graph structures to find a better representation of data points in lower spaces, we further incorporate the explainability graph in the design of the MF model to be able to generate *explainable* recommendations.

1.2 Research Contributions

1. We present a cross-modal recommender engine that leverages multiple domains of data to retrieve similar items and recommend the most relevant items to the user. We show how this approach can automatically generate explanations for the recommendations and also has the potential to alleviate the cold-start problem, one of the most notorious limitations of Collaborative Filtering (CF) techniques.
2. We propose a probabilistic formulation for measuring the explainability of Neighbor-Style Explanation (NSE), Influence Style Explanation (ISE), and feature Style Explanation (KSE) for recommendations.
3. We propose an Explainable-Matrix Factorization (EMF) model for providing explain-

able recommendations that can leverage the accurate predictions of MF and the transparency of neighborhood-based CF algorithms. In our method, explainability can be directly formulated based on the rating distribution within the user’s or item’s neighborhood. If many neighbors have rated the recommended item, or the user have rated many similar items to the recommended item, then this provides a basis upon which to explain the recommendations.

4. We propose an explainability metric based on Keyword Style Explanation that uses item content features to generate explanations and recommend explainable items.
5. We encode the user-item explainability relationship in a graph. While other methods in the literature have used graph structures to find a better representation of data points in lower spaces, we further incorporate the explainability graph in the design of the MF model to be able to generate explainable recommendations.
6. We propose an explanation-aware neural network using constrained Restricted Boltzmann Machines (RBM) for CF to recommend items that are explainable.
7. We propose offline metrics to evaluate the explainability of recommender systems.
8. We present user study experiments for online evaluation of our proposed explainable recommender system.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

“We are leaving the Information Age and entering the Recommendation Age [21].” In the past, gathering information to make effective and efficient decisions was difficult, and recommendations from others and their experience were a guide to us through this lack of knowledge. Today, we are overloaded with information which means we have more choices and therefore making choices is becoming even harder. Recommender systems can act as shortcuts through the information to help us narrow our choices, and obtain relevant information.

Recommender systems are used in many websites to personalize the experience of a user based on their past history, item content, or user demographic information. Among recommender systems, collaborative filtering (CF) is an effective recommendation approach in which the preference of a user can be predicted based on information from other users that share similar interests, without requiring any content information about the recommended items. This chapter presents a thorough overview of the methods in the literature. In Section 2.1, we review recommender system approaches, and their challenges such as the cold-start problem and the importance of providing explanations. Section 2.2 is devoted to latent factor models including Matrix Factorization and Restricted Boltzmann Machines. Section 2.4 presents the review of recommendation explanation mechanisms. Finally, in Section 2.5, we summarize the chapter.

2.1 Recommendation Systems

The recommender problem may be defined as approximating a utility function that predicts how a user will like an item automatically. Utility is usually represented by user

ratings, however it could be any function. The recommendation process is based on the past behavior of the users, the relations between the users or the items, item similarity based on the content, and finally context [22].

Formal definition of a recommender system: let U be the set of users, I be the set of possible recommendable items, and f be the utility function measuring the usefulness of item i to user u . For each user u belonging to U , we want to choose items belonging to I , that maximize f , i.e.,

$$\forall u \in U, \quad i^* = \operatorname{argmax}(f(u, i)) \quad \text{s.t. } i \in I \quad (2.1)$$

Recommender systems can be divided based on which data and which mechanism they use to make recommendations. The main categories of recommender systems are *content-based filtering* (CBF), *collaborative filtering* (CF), and *hybrid* methods. Content-based filtering makes recommendations based on the content of the items (e.g. text or images) and the content of the user profiles. The similarity between an object of interest and the items that the user has bought, viewed, or ranked before, is calculated based on a user or item profile, and is used as the basis for the recommendation. Content-based approaches require ratings made by only the user herself in contrast to collaborative filtering models that also use the ratings of other users who are similar to the target user in order to derive the recommendations. Collaborative filtering algorithms exploit the historical ratings and preferences of the active user's preference and the like-minded users to suggest items or to predict the utility of an item for a particular user. Content-based characteristics can be combined with collaborative filtering models to improve the results of recommendation. This type of recommendation is known as a hybrid recommendation [23].

In a typical CF framework, there are m users: $U = \{u_1, u_2, \dots, u_m\}$ and n items: $I = \{i_1, i_2, \dots, i_n\}$. The set of items that the user u_i has expressed her opinions about is I_{u_i} . Note that it is possible for I_{u_i} to be a null set. The opinions can be explicitly provided by the user as a rating score, or can be implicitly derived from the purchase logs or web history of the user [24, 25]. In 2.1, an example of a utility matrix is shown that represents the users'

	★		★★			
				★★	★★★	?
		★★★		?	★	
		★★★		★	★★	
					?	

Figure 2.1: A user-book ratings matrix, representing ratings of books on a scale of 1-5.

ratings of books on a scale of 1-5, with 5 being the highest rating. Users are shown in the rows and books are shown in the columns. The number of stars in each cell represents the rating of the user in the corresponding row on the specific book in the corresponding column, while blank cells show the situation where no rating is available. CF methods estimate the appropriate value for some of the blank cells (cells filled in with a question mark).

For the active user u_a , the result of the CF algorithm can take two forms (that do not necessarily have direct relationship [26]):

1. A numerical value, which is a predicted value for the likeness of item $i \notin I_{u_a}$ for the active user u_a .
2. A list of top-N recommended items for the active user u_a . The set of top-N recommended items and the items that the user has already purchased or rated must not have any overlap.¹

2.2 shows the general framework for the collaborative filtering process on the rating data that is shown in 2.1.

CF techniques can be classified into two groups: *memory-based* methods and *model-based* methods [27].

¹The exception rule is ignored in some cases, such as certain eBay and e-learning transactions, as well as some news and multi-media retrieval and recommendations.

Figure 2.2: Collaborative filtering process.

2.1.1 Memory-based Methods

Memory-based methods provide recommendations based on the similar-user or similar-item neighborhood around the target user or the target item. Depending on whether the obtained neighbors are similar users or items, memory based methods are classified as user-based algorithms or item-based algorithms. Similar to the idea of nearest neighbor classification, after computing the similarities, the neighbors are aggregated to get the top-N most similar users or items [28–30]. The similarity w between users a and u can be computed using *Pearson's correlation coefficient* [31]:

$$w(a, u) = \frac{\sum_{j=1}^m (r_{a,j} - \bar{r}_a)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{j=1}^m (r_{a,j} - \bar{r}_a)^2 \sum_{j=1}^m (r_{u,j} - \bar{r}_u)^2}} \quad (2.2)$$

where m is the total number of items that both a and u have rated. $r_{a,j}$ is the rating that user a gave for item j and $r_{u,j}$ is the rating of user u on item j . $\bar{r}_a$ and $\bar{r}_u$ are the averages of user a 's ratings and user u 's ratings for all items, respectively. Similarly, the similarity between items k and i can be computed using the following formula:

$$w(k, i) = \frac{\sum_{j=1}^m (r_{j,k} - \bar{r}_k)(r_{j,i} - \bar{r}_i)}{\sqrt{\sum_{j=1}^m (r_{j,k} - \bar{r}_k)^2 \sum_{j=1}^m (r_{j,i} - \bar{r}_i)^2}} \quad (2.3)$$

where m is the total number of users that rated both items k and i . $r_{j,k}$ is the rating that user j gave to item k and $r_{j,i}$ is the rating of user j to item i . $\bar{r}_k$ and $\bar{r}_i$ are the average ratings of items k and i , respectively.

The *Cosine similarity* is another similarity measure that can be used to find the

similarity between two users or two items. The similarity between two users a and u is defined as:

$$Cos(a, u) = \sum_{j=1}^m \frac{r_{a,j}}{\sqrt{\sum_{j=1}^m r_{a,j}^2}} \frac{r_{u,j}}{\sqrt{\sum_{j=1}^m r_{u,j}^2}} \quad (2.4)$$

where m is the total number of items that both a and u have rated. $r_{a,j}$ is the rating that user a gave for item j and $r_{u,j}$ is the rating of user u on item j . Similarly, the similarity between items k and i , can be computed using the following formula:

$$Cos(k, i) = \sum_{j=1}^m \frac{r_{j,k}}{\sqrt{\sum_{j=1}^m r_{j,k}^2}} \frac{r_{j,i}}{\sqrt{\sum_{j=1}^m r_{j,i}^2}} \quad (2.5)$$

where m is the total number of users that rated both items k and i . $r_{j,k}$ is the rating that user j gave to item k and $r_{j,i}$ is the rating of user j on item i .

Recommender Systems traditionally use the Pearson's correlation or cosine similarity, however, many other similarity measures can be used. Ellen Spertus et al. [32] evaluated six different similarity measures in the context of the Orkut social network. The best results were for recommendations generated using the cosine similarity. On the other hand, Lathia et al. [33] studied several similarity measures in the context of recommender systems and concluded that the prediction accuracy was not affected by the choice of the similarity measure.

One drawback of memory-based methods is that computing accurate similarities. Another drawback is that the search for neighbors among a large user population of potential neighbors forms a bottleneck in the user-based neighborhood CF methods [28]. Item-based techniques may be able to avoid this bottleneck because item neighborhoods are relatively static compared to user neighborhoods.

2.1.2 Model-based Methods

Model-based CF methods use the rating data to train a system to make predictions based on a learned model. The model can be built using different machine learning al-

gorithms such as Bayesian networks [34, 35], clustering [36], matrix factorization [15], and rule-based approaches [37].

Some representative methods include [3, 38–40]. Among the model-based CF methods, matrix factorization based methods [27] have been the most popular in recent years, and they have proven to address collaborative filtering challenges such as scalability and sparsity effectively [16, 41, 42]. Memory-based and model-based CF approaches can be combined to form hybrid CF approaches to improve the recommendations [43, 44]. More recent work uses textual reviews and opinions to improve the performance of rating prediction [45–47].

Even though CF tasks have achieved some success in providing recommendations to the users, there are additional challenges that they still need to address to produce high quality predictions, such as data sparsity and the cold-start problem (new items or new users) [48]. Usually, in recommender systems, there is a large set of items to evaluate. This results in an extremely sparse user-item matrix for collaborative filtering [27]. The cold-start problem happens when a new user or a new item enters the system. The system cannot recommend items to the new user until some of their rating information becomes available; similarly, new items cannot be recommended until some users have rated them [43, 48].

2.1.3 The Sparsity Problem

Recommender systems generally have to deal with data in a very high dimensional space. This space is usually very sparse and only a limited number of features are available for each object. For example, in Amazon.com, only a few ratings out of billions of items are typically available from each user. In a high dimensional space, all objects tend to be dissimilar and finding objects that form groups with similar properties becomes critical. This is because there is no intuitive notion of density or distance between points as in low dimensional spaces. This is known as the *Curse of Dimensionality* [22].

To overcome this problem, dimensionality reduction techniques are typically used to convert the high dimensional space to a new lower-dimensional space. The most relevant methods for dimensionality reduction in recommender systems are Matrix Factorization

(MF) methods. These techniques will be presented in section 2.2.

2.1.4 The Cold start problem

The cold-start problem is a notorious problem for CF systems. It happens when recommendations have to be made for new users in the system, or when new items that no one has rated yet need to be recommended. Providing an accurate and efficient recommendation result in the cold start case is a key challenge in CF [49]. Cold start problems can occur in three cases: (a) recommendation on existing items for new users, (b) recommendation for existing users on new items, and (c) recommendations on new items for new users.

When new users enter the system, they are asked to provide their ratings for different items to acquire information for CF. Usually in CBF, the system asks new users a series of questions to generate her initial profile with her explicitly stated preferences. As a user likes or consumes more items, the system can update her profile and give more weights to the content features of the items that she consumed. In these approaches, the recommended items are similar to the items that were previously consumed by the user. This can result in lack of diversity and therefore low satisfaction with recommendation results.

One possible remedy for the cold-start problem is to exploit user and item attributes because they can help create a bridge between existing users or items and new users or items. Rashid et al. [50] propose several strategies that can be incorporated in CF algorithms to learn about the new users in the system. These techniques, which present items to new users, range from random item selection to methods that exploit database queries such as choosing popular items. Park and Chu [51] proposed a predictive regression model that incorporates demographic information as well as content features to tackle the cold-start problem. Shaw et al. [52] used association rule mining to expand user profiles and provide more accurate recommendations for new users. However, extracting rules from large datasets with a very large number of multi-valued attributes is often computationally prohibitive. Golbandi et al. [53] presented a model for profiling new users in the system by eliciting the opinion of users about items. The core of their method is an efficient decision-tree-based recommender

algorithm, suitable for an adaptive bootstrap process. In [54], a functional matrix factorization (fMF) was proposed. An initial interview is performed to acquire information from the new user. To construct the adaptive interview, a decision tree is learned with each node being an interview question. Zhang and Li [55] proposed a solution for the cold start that makes use of social tags. Their recommendation algorithm considers social tags as a bridge that connects users and items. Shein et al. [56] proposed a probabilistic model for combining collaborative filtering and content information to recommend new items to the users. They used the EM algorithm to fit the model to the data.

2.2 Latent Factor Models

As described in Section 2.1.3, one drawback of memory-based CF methods is that it is solely based on the co-rated items and does not consider the hidden interests and topics that similar users and items share. Latent factor models, that are also known as Matrix Factorization (MF) models, are model-based techniques that leverage the idea that ratings are influenced by a set of lower rank *factors*, such as movie genres, characters, etc. In these type of models, both users and items are projected into a new latent feature space using an objective function. Similar to MF techniques, Restricted Boltzmann Machines (RBM) also perform latent factor discovery and can be categorized into latent factor models. These latent factors are solved using optimization techniques. In this section, an overview of some of the most common latent factor models in the literature is given.

2.2.1 Principal Component Analysis (PCA)

PCA is a statistical method that converts a set of data points observed from possibly correlated variables into a set of points with linearly uncorrelated variables that are called principal components. The amount of variance captured by the first component is larger than the second component and so on. PCA is applicable when the data set is mostly Gaussian.

2.2.2 Singular Value Decomposition (SVD)

SVD is another powerful method for dimensionality reduction in Recommender Systems, which allows for extracting concepts from the high dimensional data in a new space. Given the matrix $X(n \times m)$, X can be decomposed into $X = U\Sigma W^T$, where Σ is a k -by- k rectangular diagonal matrix of positive numbers, called the singular values of X ; U is an n -by- k matrix, the columns of which are orthogonal unit vectors of length n called the left singular vectors of X ; and W is a m -by- k matrix whose columns are orthogonal unit vectors of length m and called the right singular vectors of X . The k Eigenvalues in Σ are ordered in decreasing magnitude. To approximate X , the Eigenvalues matrix Σ can be truncated up to row i , resulting in X_i : $X_i = U_i \Sigma_i W_i^T$ which is the closest rank- i matrix to X . In terms of factorization computation, W is equivalent to the eigenvectors of $X^T X$, since $X^T X$ can be written as: $X^T X = W \Sigma U^T U \Sigma W^T = W \Sigma^2 W^T$. Similarly, U is equivalent to the eigenvectors of XX^T .

Although Matrix Factorization methods such as PCA and SVD can be used in preprocessing to lower the dimensionality, in recent years they have been employed as independent approaches to Recommender Systems.

2.2.3 Matrix Factorization (MF)

MF is a family of latent factor algorithms where a data matrix, X , is factorized into two lower rank approximated matrices P and Q as follows:

$$\mathbf{X}_{n \times m} \simeq \mathbf{P}_{n \times f} \mathbf{Q}_{m \times f}^T \quad (2.6)$$

f is the rank of matrices P and Q , and it is selected such that $f \ll \min(m, n)$, so that the number of elements in the decomposition matrices is far less than the number of elements of the original matrix: $nf + fm \ll nm$.

One of the common applications of matrix factorization is in collaborative filtering recommender systems [15]. The Netflix prize competition has contributed a lot to the popularity of matrix factorization in recommender systems [15]. In this context, MF takes

Figure 2.3: Decomposed matrices with rank 2.

as input the user-item rating data and decomposes users and movies into a set of latent factors that define a new latent space, which can be thought of as concepts or categories. 2.3 shows an example of user and item decomposition matrices with $f = 2$, where MF projected onto a lower dimensional latent space extracted using matrix factorization. An effectively intuitive interpretability of the resulting factors, as shown in 2.3 in this fictitious example, can make MF especially interesting for many applications.

MF algorithms learn the factors $p_u \in \mathbb{R}^f$ and $q_i \in \mathbb{R}^f$, which are the lower-rank representations of user u and item i in a latent space of reduced dimensionality f . In order to estimate p_u and q_i , the following objective function is minimized [17]:

$$J_{MF} = \sum_{(u,i) \in R} (r_{u,i} - p_u q_i^T)^2 + \frac{\beta}{2} (\|p_u\|^2 + \|q_i\|^2) \quad (2.7)$$

where R is the set of available ratings for the (u, i) pairs. The regularization term $(\|p_u\|^2 + \|q_i\|^2)$ with the regularization coefficient, β , control the smoothness and sparseness of the resulting factors. This optimization problem is not convex; however, it is convex with respect to matrix P only or matrix Q only. It can therefore be solved by using arithmetic or multiplicative updates for P and Q . In Sections 2.2.3.1 and 2.2.3.2, two of the most common algorithms in the literature, for solving MF, are reviewed.

2.2.3.1 Alternating Least Squares

Alternating Least Square (ALS) is an iterative approach, which fixes P or Q in each iteration. Although 2.7 is not convex, it is convex in either P or Q . However, it does not guarantee that the matrices found are the global solutions and provides only a local minimum to the MF problem. The ALS method is outlined in Algorithm 2.1 [16, 57].

Algorithm 2.1 Basic Alternating Least Square (ALS) Algorithm

Input: data matrix $\mathbf{R}$, number of factors f

Output: optimal matrices $\mathbf{P}$ and $\mathbf{Q}$

1. Initialize matrix $\mathbf{P}$ (for example randomly)
 2. **Repeat**
 3. Solve for $\mathbf{Q}$ using: $\mathbf{P}^T \mathbf{P} \mathbf{Q} = \mathbf{P}^T \mathbf{R}$
 4. Solve for $\mathbf{P}$ using: $\mathbf{Q} \mathbf{Q}^T \mathbf{P} = \mathbf{Q} \mathbf{R}^T$
 5. **Until** the cost function converges
-

2.2.3.2 Stochastic Gradient Descent

In the basic Stochastic Gradient Descent (SGD) approach, p_u and q_i are modified in each iteration by an amount proportional to the value of the gradient [58]. The derivative of J_{MF} with respect to p_u and q_i is:

$$\frac{\partial J_{MF}}{\partial p_u} = -2(r_{u,i} - p_u q_i^T) q_i + \beta p_u \quad (2.8)$$

$$\frac{\partial J_{MF}}{\partial q_i} = -2(r_{u,i} - p_u q_i^T) p_i + \beta q_i \quad (2.9)$$

Given Formula 2.8 and 2.9 the SGD update rules are as follows:

$$p_{u+1} \leftarrow p_u + \alpha(2(r_{u,i} - p_u q_i^T) q_i - \beta p_u) \quad (2.10)$$

$$q_{i+1} \leftarrow q_i + \alpha(2(r_{u,i} - p_u q_i^T)p_u - \beta q_i) \quad (2.11)$$

where α and β are the learning rates. Algorithm 2.2 shows the application of SGD update rules iteratively until convergence.

Algorithm 2.2 Basic SGD Algorithm for MF

Input: data matrix $\mathbf{R}$, number of factors f , learning rates α and β

Output: optimal matrices $\mathbf{P}$ and $\mathbf{Q}$

1. Initialize matrix $\mathbf{P}$ and $\mathbf{Q}$ (for example randomly)
 2. **Repeat**
 3. **For every** (u,i) pair
 4. update $p_u : p_u^{new} \leftarrow p_u + \alpha(2(r_{u,i} - p_u q_i^T)q_i - \beta p_u)$
 5. update $q_i : q_i^{new} \leftarrow q_i + \alpha(2(r_{u,i} - p_u q_i^T)p_u - \beta q_i)$
 6. **End for**
 7. **Until** the cost function converges
-

Choosing the right values for α and β is very important in the gradient decent algorithms. If these values are too small, the updates will converge too soon, and if large values are selected, it will not converge.

2.2.4 Restricted Boltzmann Machines (RBM)

Restricted Boltzmann Machines (RBM) is another kind of latent factor model that extracts a smaller set of hidden variables from the input data, that can be used as data representation. RBM is a two layer stochastic neural network consisting of visible and hidden units that are connected bidirectionally. Each visible unit is connected to all the hidden units in an undirected form. No visible/hidden unit is connected to any other visible/hidden unit. The stochastic, binary visible units encode user preferences on the items from the training data, therefore the state of every visible unit is known. Hidden units are also stochastic, binary variables that capture the latent features. A probability

$p(v, h)$ is assigned to each combination hidden unit, h , and visible unit, v :

$$p(v, h) = \frac{e^{-E(v, h)}}{Z} \quad (2.12)$$

where E is the energy of the system and Z is a normalizing factor [59]. E can be defined as follows:

$$E(v, h) = - \sum_{i=1}^n \sum_{j=1}^m w_{ij} h_i v_j - \sum_{j=1}^m b_j v_j - \sum_{i=1}^n c_i h_i \quad (2.13)$$

where m and n are the numbers of visible units and hidden units, respectively, w_{ij} is a real valued weight associated with the edge between the units of visible units and hidden units, and b_j and c_i are real valued bias terms associated with the visible units and hidden units respectively. The network graph representation of an RBM is shown in Figure 2.4. The conditional probabilities $p(h_j = 1|v)$ and $p(v_i = 1|h)$ are defined as:

$$p(h_i = 1|v) = \sigma\left(c_i + \sum_{j=1}^m v_j w_{ij}\right) \quad (2.14)$$

$$p(v_j = 1|h) = \sigma\left(b_j + \sum_{i=1}^n h_i w_{ij}\right) \quad (2.15)$$

where $\sigma(x)$ is the logistic function $\frac{1}{(1+e^{-x})}$. To train for the weights, a Contrastive Divergence method was proposed by Hinton [59].

Salakhutdinov et al. [60], proposed an RBM framework for CF. Their model assumes one RBM for each user and takes only rated items into consideration when learning the weights. They presented the results of their approach on the Netflix data and showed that their technique was more accurate than Netflix's own system. The focus of this RBM approach was on evaluating the performance of the proposed system in terms of rating prediction.

2.3 Comprehensible Classification Models

In this section, we review interpretable classification models because explanations are related to interpretability of predictive models.

Figure 2.4: An RBM network with n hidden and m visible units.

In the context of machine learning, interpretability means “explaining or presenting in understandable terms” [61]. In addition, interpretability and explanations can help to determine if qualities such as fairness, privacy, causality, usability and trust are met [62].

Doshi-Velez and Kim [62] presented a taxonomy of approaches for the evaluation of interpretability in ML models in general: application-grounded, human-grounded, and functionality-grounded. Application-grounded and human grounded evaluation approaches are both user-based, while the functionality-grounded approach does not require human evaluation and uses some definition of the interpretability for the evaluation. Experiments can be designed based on different factors, such as global vs local, which considers the general patterns existing in the model as global, while considering local reasoning behind the specific prediction of the model as local [62]. The former is usually helpful for the designer and developer of the model when understanding or detecting bias or causality in the model. The latter can be aimed at the end user of the systems to understand the justifications of the system decisions.

Ribeiro et al. [63] proposed an explanation technique that explains the prediction of the classifiers locally, using a learned model. Their proposed explanation conveys the relationship between the features (such as words in texts or parts in images) and the predictions; and helps in feature engineering to improve the generalization of the classifier. This can help in evaluating the model to be trusted in real world situations, in addition to using the offline accuracy evaluation metrics.

Freitas [64] reviewed comprehensibility or interpretability of five classification mod-

els (decision trees, decision tables, classification rules, nearest neighbors, and Bayesian network classifiers). It is important to distinguish understanding or interpreting an entire model (which the paper does) from explaining a single prediction (which is the focus of this dissertation). In addition, we note that Freitas overviews the problem from several perspectives and discusses the motivations for comprehensible classifier models, which are:

1. Trusting the model: Regardless of accuracy, users are more prone to trusting a model if they can comprehend why it made the predictions that it did.
2. Legal requirements, in some cases like risk modeling, require a justification in case of denying credit to an applicant.
3. In certain scientific domains such as bioinformatics, new insights can be gained from understanding the model, and can lead to new hypothesis formation and discoveries.
4. In some cases, a better understanding can help detect learned patterns in the classification model that are not really stable and inherent in the domain, but rather result from overfitting to the training data, thus they help detect the data shift problem: i.e., when the new instances deviate in their distribution from past training data; we note that concept drift (i.e. when a previously learned and accurate model no longer fits the new data because of changes in patterns of the data) can be considered as a special case of the data shift.

Understanding the logic behind the model and predictions (in other words, comprehension) can reveal to the user the fact that the (new) data has outpaced the model. The user can then realize that the model has gotten old and needs to be updated with a new round of learning on new data. Various interpretation methods exist depending on the family of classifier models: decision trees, rule sets, decision tables, nearest neighbors. Different studies have shown that the interpretability of entire classifier models depends on the application domain and the data, with findings that sometimes contradict each other. Regardless of all the findings in interpreting models, we note that the task of interpreting an “entire classifier

model” (e.g. a complete decision tree or a set of 500 rules) is different from that of one user trying to understand the rationale behind a “single prediction/recommendation” instance.

That said, we find Freitas’ review to be very important for this work: first he argues that model size alone is not sufficient to measure model interpretability, as some models’ complexity is beyond mere size and small models can actually hurt the user’s trust in the system (a notorious example is decision stump models (1-level trees) in the medical or legal domain which do not offer enough evidence to be able to judge a prediction model and would actually lead to the same explanation for each new test instance. Also, extremely small models would likely suffer in accuracy. Second, the work on interpreting rule-based models and nearest neighbor models can be useful to us because it is closest to the CF mechanisms we study. For nearest neighbor models, Freitas [64] mentions that attribute values of nearest neighbors can help provide explanations for predictions, and that showing these values in decreasing order of relevance (based on an attribute weighting mechanism) is a sensible strategy. Another strategy is to show the nearest prototypes of training instances, for example after clustering the training instances. However, in both of these strategies, Freitas [64] was motivating types of entire models rather than individual prediction explanations in the context of recommending thousands of items.

2.4 Explanation Mechanisms in Collaborative Filtering

Most recommendations are the result of black box systems which do not provide the reasoning behind their suggestions to the user. While the user might trust a recommendation to listen to a song, s/he is less willing to act based on recommendations in higher risk content domains such as renting a holiday resort. Herlocker et al. presented evidence that showed explanations improves the acceptance of recommender systems by the user [65]. The explanation *may*, or *may not* reflect the underlying algorithm used by the system. Explanations can be given using words related to item features or user demographic data, but these cannot be done easily in CF approaches. They vary between simple explanation formats such as: “people also viewed” in e-commerce websites [66] to the more recent so-

cial relationships and social tag based explanations [67, 68]. Bilgic & Mooney [69] showed how explaining recommendations can improve the user’s estimation of the item’s quality and help users make more accurate decisions (i.e. user satisfaction). In black box models, the technology behind generating recommendations may not be conveyable to the user. However, one way to provide explanations to justify any recommender system’s output is by providing the system’s past performance. Explanations help provide transparency into these type of systems.

2.4.1 Benefits of Explanation

Explanations can help the user detect errors in recommender systems. The output of recommender systems is prone to error which can be due to data errors or the model not matching the user’s requirements. Data errors are intrinsic characteristics of collaborative filtering systems which can happen because of missing and sparse data, poor or bad data, and high variance data.

In addition to trust and error detection benefits, explanations can add justification and acceptance. Justification gives the user an understanding of the reasoning in the recommendation process which helps the user decide how much confidence to put in the output. Acceptance of a recommender system can be improved after an explanation since the system’s limits and strengths become more visible to the users.

Tintarev and Masthoff [70] studied additional aims of explanations such as trust, effectiveness, and satisfaction. A good explanation would increase the perceived quality of the recommendation and adoption of the recommender system by the user.

We can summarize the benefits of explanation in the following list:

- *Perceived quality and satisfaction:* According to Pu et al., the key factor in a successful recommendation system is its quality, which will affect the acceptance of the system by the user and interaction with the recommender system [71]. Explanation is another factor in increasing the perceived quality. Descriptions and explanations have been found to be correlated with the perceived usefulness of the recommender system,

thereby increasing the overall satisfaction of the user [72].

- *Trust*: Studies by [65, 70, 73] have shown that a good explanation interface could increase users' trust and satisfaction by providing information to justify the recommendations. This in turn increases user involvement and educates users on the internal logic of the system. Trust in the recommender system also depends on the accuracy of the recommendation. Good explanations may not compensate for poor recommendations, however if a user understands *why* a bad recommendation has been made, they may be more confident in the system.
- *Transparency and scrutability*: Explanations can provide the user with the logic behind the recommendation to clarify how a recommendation was chosen. The importance of transparency also known as “visibility of system status” has been confirmed by user studies of recommender systems [72]. Following transparency, recommender systems should allow the user to correct the reasoning and assumptions made where needed.
- *Persuasiveness*: The recommender system could persuade the user into buying or trying a recommended item, whether the prediction is accurate or not.
- *Effectiveness*: Effectiveness is highly dependent on the accuracy of the recommendation. An effective explanation can assist the user to make better decisions according to their preference, by giving the user a broad range of options, including a new domain, or new range of products.

2.4.2 Explanation Approaches and Algorithms

Most recommender systems in the literature are black boxes. Although, a few of them provide some form of explanation, it is still not clear to the user why a specific item is being suggested. This results in users not trusting the suggestions of the recommender system. However, the way that white box models perform are close to the conceptual model of the human recommendation process. Usually white box collaborative filtering systems are

neighborhood based, in which neighboring users are selected based on a similarity measure. The prediction is performed as the process of aggregation of the ratings of the neighbor. This process could end up giving weak recommendations which could be discovered with good explanations.

Based on [69], three different approaches to explanations can be delineated:

1. **Neighbor Style Explanation (NSE):** compile a chart in CF that shows the active user’s nearest CF neighbors’ ratings on the recommended item: these are grouped into 3 categories: bad (ratings of 1-2), neutral (rating 3), and good (4-5). Then showing a histogram of these categories among the nearest 30 neighbors.
2. **Influence style Explanation (ISE):** present a table of those items, that had the most impact on computing the current recommendation. They can be used in both CBF and CF.
3. **Keyword style Explanation (KSE):** analyze the content of recommended items and the user’s profile (interests) to find matching words in CBF.

NSE approaches: Generally, neighbor style explanations can be categorized into 3 approaches: item based, user based, and feature based. *Item based* explanations are presented based on the relationship between the user and the set of related items, usually the items that the user has rated. *User based* explanations are presented based on the relationship between the user and similar users. *Feature based* approaches use characteristics of the recommended item.

An in-depth analysis on how to provide explanations of a user-similarity based CF was performed for the MovieLens project [65]. Several ways of presenting auxiliary information about the ratings were scrutinized, such as expert critics, assessment of accuracy or links to ratings made by the correlated users. It was demonstrated that providing good explanations raised a user’s trust; however, some explanations may be actually more harmful than the lack of explanation. A well known style of explanations in collaborative filtering is used by Amazon: “*Customers who bought this item also bought ..*”. This model implies

that the system retrieved and recommended items from similar users in the neighborhood (i. e. who also bought a common item). Figure 3.5 (3), shows an example of a neighbor style explanation (NSE) for a recommended movie based on the user’s neighbors. This user-based example presents the ratings distribution of the user’s neighbors on three rating levels.

Giving the user information about what type of data is used in the system encourages the user to provide more helpful data of that kind, such as preference ratings. Information about the neighbors selected as the predictors could give the user a chance to examine their ratings and to disregard the recommendations if the right neighborhood is not selected. A good explanation could also help discover weak predictions. Distribution of the ratings of the neighbors on a target item is helpful in identifying whether the prediction is based on enough data or not.

Herlocker et al. [65] compared 20 other explanation systems and found histograms to perform best based on promotion only.

KSE and ISE approaches: Bilgic & Mooney [69] proposed a book recommendation system (LIBRA). They argued that the quality of explanation can be measured using two different approaches: the promotion approach or the satisfaction approach. The *promotion approach* favors the explanation that would convince the user to adopt an item, while the *satisfaction approach* favors an explanation that would allow the user to assess the quality of (or how much they like) an item best.

Bilgic & Mooney’s experiments contradicted Herlocker et al.’s findings. One reason is that Herlocker’s experiments measured only “promotion” while Bilgic & Mooney studied both promotion and satisfaction. They proposed KSE and ISE evaluation approaches in addition to NSE. for the ISE approach. They compute 2 influence scores: content and collaborative influence scores, and then scale each one to a common $[-100,100]$ range and average the two. The Content influence of an item = difference between the recommendation scores computed using a Bayesian classifier trained with and without that item. The Collaborative Influence of an item = difference between the CF scores computed with

and without using that item in computing the Pearson’s correlations that are combined to obtain the CF score.

The conclusion from Bilgic & Mooney is that while the NSE style explanations were top performers in Herlocker et al. from the point of view of “promotion”, Bilgic & Mooney found KSE and next ISE explanations to be the top performers from a “satisfaction” perspective.

Other than [69], Vig et al. [68] proposed a KSE explanation based on two key components: 1) tag relevance: the degree indicating how the tag describes an item, and 2) tag preference: the user’s sentiment toward a tag. They introduce *tagsplanaton*, which is generating explanations based on community tags. In their method, they consider a form of content-based explanation. The average of a given user’s ratings of the movies with a specific tag defines how relevant a tag is for that user.

Another KSE approach is presented by McCarthy [74]. Their explanation is knowledge and utility based; that is, based on the users’ needs and interests. The explanation is presented by describing the matched item for the specified requirements from the user. Zhang et al. [75] proposed an Explicit Factor Model (LFM) to generate explainable recommendations. They extracted explicit product features and user opinions using sentiment analysis. [76] studied the impact of personalizing feature-based explanations on effectiveness and satisfaction. Their results showed that personalization was detrimental to effectiveness, though it may improve user satisfaction.

Ardissonoa et al. [77] built a recommendation system that suggests places to visit based on the travelers’ type (e.g. children, impaired). In this case, the explanation comes in the form of the presentation of the recommendation to the user. The demographic information of the user is utilized to group users, and the explanation is focused on the most meaningful types of information for each group.

Billsus and Pazzani [78] presented a keyword style and influence style explanation approach for their news recommendation system which synthesizes speech to read stories to the users. The system generates explanations and adapts its recommendation to the user’s

interests based on the user’s preferences and interests. They ask for a feedback from the user on how interesting the story had been to the user or if the user needs more information. The explanation is then constructed from the retrieved headlines that are closest to the user’s interests. An example of their explanation is: *“This story received a [high — low] relevance score, because you told me earlier that your were [not] interested in [closest headline].”*

Symeonidis et al. [79] proposed an objective metric, computed as the coverage ratio for a user U for whom a recommendation is being made. This metric is the ratio of the sum of the relevant features in the justification list $J(U)$ to the total sum of relevant features that exist in the user’s feature profile. Starting with a top- n list L of recommended items $L(U) = I_1, \dots, I_n$, the justification list $J(U)$ is the list of ordered pairs $J(U) = (f_1, c_1), \dots, (f_m, c_m)$, used as justification for the recommendations in L . Each ordered pair contains an item content feature f , matching the user profile features, with its frequency c inside recommended list L .

Their explanations have the form: *“Item x is recommended, because it contains features $a, b, \dots$, which are included in items $z, w, \dots$, that you have already rated.”* Thus their justification style combines the keyword-based style explanations KSE (because of listing features) and the influence style explanations ISE (because of listing items), that were each defined previously in [69]. Their recommendation system was the feature-weighted nearest bicluster (FWNB) algorithm, and they measured the accuracy of the recommendation using precision and recall. Their recommendation is based on finding biclusters containing item content features that have strong partial similarity with the test user. Similarity is a convex combination of two similarities $S = (1 - a)S_1 + (a)S_2$. Similarity S_1 is based on agreement between the user and bicluster’s ratings. Similarity S_2 , referred to as justifiable, is based on agreement between user’s item content features (from the user profile) and a bicluster’s content features. The convex combination can lean toward ratings or justifications in weighing the two similarities by tuning parameter a (this was done empirically to maximize recommendation precision and coverage). The item content features can later be

used for justifying the recommendations.

They extracted the dominant features that influence recommendations by first constructing matrix F_b with element $F_b(i; f)$ denoting the influence of feature f of item i in the biclusters' neighborhood of a user u , while B_u is the set of nearest biclusters of user u . The dominant features are those with highest total influence. Then to find the total influence of an item in the user's neighborhood, they add $F_b(i; f)$ elements of matrix F_b for each individual item i , thus, revealing the items that contain the most dominant features. Then they keep the j ($j > N$) items with the highest aggregated values creating the B_b set of items and furthermore exclude those items from B_b that have already been rated by the test user. The justifiable list consists of the items from the recommend list L that are the most influential in the feature profile of the test user.

They used the 100K movielens benchmark data set and extracted the movie content features from the Internet movie database (imdb). The join process lead to 23 different genres, 9847 keywords, 1050 directors and 2640 different actors and actresses.

Their survey-based user study measured the user satisfaction against KSE, ISE and their own style, called KISE. They designed a user study with 42 pre- and post-graduate students of Aristotle University, who filled out an online survey. Each target user was asked to provide ratings for at least five movies that exist in the Movielens data set. Then, they recommended to each target user a movie, justifying their recommendation by using the three justification styles (a different style each time). Finally, target users were asked to rate (in 1-5 rating scale) each explanation style separately to explicitly express their actual preference among the three styles. Subsequent analysis of the mean and standard deviation of the users' ratings for each explanation style, found KISE to outperform all other styles. Paired t-tests also concluded that the difference between KISE from KSE and ISE was statistically significant at $p = 0.01$ level.

Although the findings in [79] did not compare with NSE, their study and experiments were similar to those of Bilgic&Mooney [6] who previously found KSE to be the top performer, followed closely by ISE (then by a margin, NSE). However it is worth mentioning

that the data sets in the two studies were different: movielens for [79] vs. books for [69]. Thus, their item content features are different (genre, keywords, directors, actors collected from imdb for movies vs. keywords in the author, title, description, subject, related authors, related titles, that are crawled from Amazon for books). It is easy to see that the content features for the books in LIBRA draw significantly more on Human Expert knowledge (Subject, Related authors and book titles) compared to the imdb-sourced content features of movies in Symeonidis (no related movie titles or related producers).

2.4.3 Explanation Evaluation

Evaluation of explanations in recommender systems require user-based metrics to evaluate the perceived quality of the explanation and the efficiency of the justification of the recommendation provided to the user by the explanation. Pu et al. [71] proposed a method that consists of 60 questions to assess the perceived quality of the recommendations such as usefulness, users’ satisfaction, influence on the users’ intention to purchase the recommended product, and so on. However, this questionnaire was designed for user-based evaluation of the *recommender system* and not the *explanation*. Herlocker et al. [65] provided some initial explorations into measuring how explanations can improve the filtering performance of users, but their study was more focused on different aspects of the explanation generation than their evaluation.

The user-based experiments in the two studies are different in two perspectives: Symeonidis et al. [79] used both (i) a quantitative (objective) metric for justification (coverage ratio) which is based on the amount of influence from content features in the justified recommendation list, and (ii) direct user’s 1-5 scale ratings about “how satisfied they are” with each explanation style (KSE, ISE or KISE), while Bilgic & Mooney [69] collected the user’s satisfaction via analysis on their ratings of the explanations before and after examining the recommended item in question.

Furthermore [69] collected the user’s satisfaction without showing them which explanation method was used and most importantly, they collect the user’s satisfaction without

even showing them the item identity before examining the item but only an explanation of why it was recommended) thus allowing to measure the user’s satisfaction with the explanation itself and not merely the recommendation. Bilgic & Mooney’s measure of the quality of an explanation is based on how similar the user’s ratings of the recommendation are before and after examining the recommended item, thus measuring the power of the explanation to convey the true nature of the recommended item, even in cases where the recommended item was rated low by the user, and not merely a promotion-based explanation (which accounts only for highly rated recommended items!).

Despite the apparent limitation of Symeonidis study [79], it remains easier to implement because it does not require that the user examines the item being recommended, and (ii) because it also computes an objective quantitative measure (based on total contribution of the influence of recommended items’ dominant content features relative to the dominant user profile features) and these can be computed directly from the ratings data, recommended lists, and explanations, all of which do not even require actual user-based tests.

2.5 Summary

In this chapter, we reviewed the main collaborative filtering and latent factor techniques in the literature. Also, we reviewed some major explanation techniques used in collaborative filtering. We find that most collaborative filtering approaches use a special type of data, namely ratings. Also, unlike neighborhood-based approaches, model-based collaborative filtering approaches lack a unified way of providing explanations for the recommendations because their underlying means of suggesting an item is mainly based on the user-item ratings and not the actual variables or features describing users and items. In the next chapter, we propose a unified multi-domain MF-based framework to generate explanations while also overcoming the cold-start problem. Also, we present novel explanation-aware latent factor recommender systems based on MF and RBM.

CHAPTER 3

PROPOSED WORK

The different variations of Collaborative Filtering (CF) recommender systems, described in the previous chapter, are generally applied to ratings data. However, in many cases, we have multiple domains or sources of information such as item attributes and user demographic data that we can leverage to improve the performance of the recommender system. Figure 3.1 shows two domains for movie data. Section 3.1 describes our multi-domain recommender system using MF to generate a common latent space based on multiple domains of data, where both recommendations and explanations can be generated. In Section 3.2, we explain how this technique solves the cold-start problem. Most recommender systems generate explanations independent of the recommender module and do not incorporate explainability of items when generating recommendations. In Section 3.4, we propose a novel formulation for explanations in recommender systems. Using explainability as a basis for building recommender systems, we present two explainable latent factor recommender system approaches: Explainable Matrix Factorization (EMF) and Explainable Restricted Boltzmann Machines (ERBM) as presented in Sections 3.6 and 3.7. Finally, in Section 3.8 we conclude this chapter.

Figure 3.1: Example of multiple-domain movie data.

3.1 A Generalized Asymmetric MF-based Framework in Collaborative Filtering

In many cases, we could have multiple domains of data that can be combined to build a more accurate prediction model. In CF, the rating matrix is usually very sparse; however, the item attributes are a great source of information that can be integrated to build a multi-domain model. Koren et al. [15] used movie attributes and demographic data to build a matrix factorization based collaborative filtering model that considers all data domains simultaneously. We propose a multi-domain Asymmetric MF-based approach to exploit multi-modal interactions between user ratings and other domains such as movie genre (in the case of movie recommendation). We use this model to solve the new item cold start problem.

In Asymmetric MF, the latent semantic space is first derived using one domain, then follows an adaptation phase in which the second domain is utilized to fit the former latent space. This results in a common space where the two domains co-exist.

In the case of movie recommendations, the movie attributes, such as movie genres, are almost always available. On the other hand, for some users and movies, no rating data is available and therefore, the rating matrix is very sparse. The ratings can be used as the second domain to adapt to the previously obtained latent space computed from the movie content data. More specifically, the two domains are: Genre-Movie matrix ($\mathbf{R}_{t_1}(g \times m)$) and Movie-Ratings matrix ($\mathbf{R}_{t_2}(n \times m)$), where g is the number of genres, m is the number of movies, and n is the number of users. The two main steps of the algorithm are as follows:

1. *Build a latent space model based on the item attributes (genre):* This step is the application of the MF to the Movie Genre data ($R_{t_1}(g \times m)$). The decomposed matrix in the latent space is obtained as follows:

$$\mathbf{R}_{t_1} \simeq \mathbf{P}_{t_1} \mathbf{Q}_{t_1}^T \quad (3.1)$$

In this formula, $P_{t_1}(g \times f)$ is the basis matrix of transforming the genre data to the latent space, while $Q_{t_1}(m \times f)$ is the representation of movie items in latent space.

2. *Adapt the basis data in the latent space:* In this step $\mathbf{Q}_{t_1}$ is transferred as fixed latent factor coefficients computed from the first step, while estimating only the basis $P_{t_2}(n \times f)$ for the second domain, consisting of the user-item ratings. This constructs a new latent space based on the two domains, Movie Ratings as well as the Movie Genre. Consequently, $\mathbf{Q}_{t_1}$ is found to span the semantic space for both ratings and item attributes.

$$\mathbf{R}_{t_2} \simeq \mathbf{P}_{t_2} \mathbf{Q}_{t_1}^T \quad (3.2)$$

The objective function to optimize for step one is the standard MF formulation as presented in Eq. 2.7. To solve for this objective function Algorithm 2.2 for using gradient descent is used.

Step one can be solved using gradient descent. With a proper choice of step size, gradient descent converges to a local minimum. In step 2, we note that matrices $\mathbf{R}_{t_2}$ and $\mathbf{Q}_{t_1}^T$ are already known, thus the problem of solving for $\mathbf{P}_{t_2}$ is convex, and can be solved using gradient descent while fixing $\mathbf{Q}_{t_1}$. Similar to step one, the update rules to solve for $\mathbf{P}_{t_2}$ would be as follows:

$$\mathbf{P}_{t_2}^{(t+1)} \leftarrow \mathbf{P}_{t_2}^{(t)} + \alpha(2(\mathbf{R}_{t_2} \mathbf{Q}_{t_1}^T) \mathbf{Q}_{t_1}) - \beta \mathbf{P}_{t_2}^{(t)} \quad (3.3)$$

where α is the learning rate and β is the regularization coefficient. $\mathbf{Q}_{t_1}$ has the role of transferring knowledge from the first domain to the second domain.

The contribution of asymmetric MF to recommender systems is transferring the data from one domain in the latent space to another domain to use the knowledge from both domains and build a richer latent space. Figure 3.2 shows the asymmetric MF framework in collaborative filtering.

3.2 A Warm-Up Solution for the Cold-Start Problem

The rating matrix is usually very sparse. When new users enter the system there is not enough information regarding their interest. Also, new items in the system start with

Figure 3.2: Asymmetric multi domain MF.

no ratings. This issue is called the cold-start problem which is the most prevalent problem of CF recommender systems. Using multiple domains of data, as is done in some hybrid systems, can help overcome this problem. In this section, we explain how our multi-domain asymmetric MF-based recommender system overcomes the new item cold-start problem [80].

In the case of movie recommendations, the Movie-Genre domain is selected as the first domain to build the genre-movie latent space. Having the genre data available for all movies, regardless of the availability of ratings, this latent space can incorporate new movies as well. The Movie-Ratings is selected as the second domain, which adapts the user-movie basis into the previously built latent space. The learning phases of the algorithm would be:

Training: The two domains used in this framework are the two modalities of the data that share the same data points. Thus, both domains should be split such that equal points in both domains are selected in the training set. To test the cold start, some percentage (e.g. 30%) of the movies are selected as the testing movies, such that no ratings for them are used in the training and only genre features of them are available. The genre features are used as the first domain. For the second domain, the rating data of the same movies from the first domain is used, noting that only movies with at least one ratings are used.

Testing: The ratings for the testing movies (e.g. 30% of the movies) are calculated in this phase. To estimate for the ratings, simply the corresponding row and column of

the target movie in the two decomposed factors are multiplied together. Having the actual ratings of the cold start movies as the ground truth is used in computing and reporting the error.

3.3 Explanation Generation Using Asymmetric MF

In this work, we propose to use the Asymmetric MF (technically, a BB model) to generate explanations in addition to recommendations. This approach makes it possible to use different sources of data for generating explanations and for computing recommendations. We consider our data sources/domains to be the ratings matrix and the item content attributes. Once a user is provided with a recommendation, a chart that shows how the active user’s neighbors have rated the recommended item is presented to the user as an explanation. This method was first proposed and tested in [65]. However, this method gives meaningful results only when the recommendation system is a neighborhood-based CF technique (user-based or item-based), and cannot be used to explain purely content-based approaches. Using our multi-domain MF-based model, it is possible to project multiple domains of data, such as item content attributes (e.g. movie genre) and user ratings onto a shared latent space that provides a common space where data from multiple domains can be compared. Using this feature, we present two neighborhood-based explanation methods: One is user based (known as NSE), and works with the active user’s neighbors in the latent space. The other is item based, and uses the recommended item’s neighbors in the latent space (known as ISE).

3.3.1 NSE-based Explanation Generation in the Latent Space

MF takes the input matrix and decomposes it into two matrices ($\mathbf{R} \simeq \mathbf{PQ}^T$). These matrices, $\mathbf{P}$ and $\mathbf{Q}$, encode the new coordinates in a new latent space for the rows and columns (users and items, respectively) of the rating matrix $\mathbf{R}$. Since $\mathbf{P}$ encodes the users in the new space with k features, the similarity $S_{i,j}$ between users i and j can be computed as follows (but we can use any other similarity measure):

$$S_{i,j} = P_{i,:} \cdot P_{j,:} = \sum_{k=1}^f P_{i,k} P_{j,k} = P_{i,1} P_{j,1} + P_{i,2} P_{j,2} + \dots + P_{i,f} P_{j,f} \quad (3.4)$$

This can be used to find the top l similar users to the active user, who rated the recommended item. Algorithm 3.1 summarizes the steps of generating the NSE-based explanation in the latent space. An example of the histogram-based presentation of the NSE, proposed by Herlocker et al. [5] is shown in Figure 3.3.

Figure 3.3: An NSE-based explanation showing the user’s neighbors’ ratings.

Algorithm 3.1 NSE-Based Explanation Generation Algorithm in the Latent Space

Input: active user (u), recommended item (i), $P_{n \times f}$, number of neighbors for the active user (l)

Output: NSE-based histogram explanation

1. Calculate the similarity matrix S : $S_{n \times n} = P_{n \times f} \times P_{f \times n}^T$
 2. Sort $S_{u,:}$ in a descending order
 3. $N_u \leftarrow$ Take the first l elements in $S_{u,:}$ that have rated item i
 4. Generate the ratings histogram from the ratings of N_u on the item i
-

3.3.2 ISE-based Explanation Generation in the Latent Space

An ISE-based explanation, like the NSE-based histogram explanation, requires computing the neighboring items of the recommended item, and therefore computing the similarity between the items. To compute the similarity $T_{u,v}$ between items u and v , we use

their encoding coefficients Q_u and Q_v in the latent space computed by MF:

$$T_{u,v} = Q_{u,:}Q_{v,:} = \sum_{k=1}^f Q_{u,k}Q_{v,k} = Q_{u,1}Q_{v,1} + Q_{u,2}Q_{v,2} + \dots + Q_{u,f}Q_{v,f} \quad (3.5)$$

Once the top l similar items have been located from the set of items that the active user has rated before, we generate the item histogram-based explanation (see Algorithm 3.2). Figure 3.4 shows an example of the ISE-based histogram explanation.

Figure 3.4: An ISE-based explanation showing the user's ratings to the recommended item's neighbors.

Algorithm 3.2 ISE-Based Explanation Generation Algorithm in the Latent Space

Input: active user (u), recommended item (i), $Q_{f \times m}$, number of neighbors (l)

Output: ISE-based histogram explanation

1. Calculate the similarity matrix T : $T_{m \times m} = Q_{m \times f}^T Q_{f \times m}$
 2. Sort $T_{i,:}$ in a descending order
 3. $N_i \leftarrow$ Take the first l elements in $T_{i,:}$ that active user a have rated
 4. Generate the ratings histogram from the rating of user u on the set N_i
-

3.4 Explainability

As described in Section 2.4.2, the three most common explanation styles are Neighbor Style Explanation (NSE), Influence Style Explanation (ISE), and Keyword Style Explanation (KSE) approaches [6], that are based on similar users, similar items, and feature attributes, respectively. NSE and ISE-based formats are usually utilized when the underlying recommender module is user-user or item-item Collaborative Filtering (CF).

An example of NSE is: “Users with similar interest to you have liked this item” which can be represented to the user in the form of a table or histogram (Figure 3.5.1 and 3.5.3).

An example of ISE is: “You will like this item because you rated/liked X,Y,Z,..” as can be presented to the user using the table shown in Figure 3.5.2. NSE is based on similar users and ISE is based on similar items. Note that there is a similar explanation type which is used by some recommender systems, and of the form: “Customers who liked item X also liked item Y,” is not a personalized explanation and it is based on the most commonly bought/clicked items together *regardless of the user*. Therefore we do not consider this explanation in this study. In Figure 3.5.4, an example of the Keyword Style Explanation (KSE) format is shown [81]. This style is mainly used for content-based or hybrid recommender systems, where the model is built from the features of the movies and user profiles.

Given the definition of NSE, ISE, and KSE, we propose a novel explainability metric for measurement of the explainability of these styles.

3.4.1 NSE-based Explainability

In the NSE type, the explanation is created from the histogram of ratings of similar users’ ratings on the recommended item, as shown in Figure 3.5 (3). This is the general form of the NSE format.

If we divide the histogram by the total counts, we will obtain the empirical density distributions of the similar users’ ratings on the recommended item i . Equivalently, this is

Your neighbors' ratings for this movie

Rating	Number of Neighbors
★	0
★★	0
★★★	3
★★★★	4
★★★★★	2

(1)

Our recommendation is "Pulp Fiction", because you rated similar movies:

Movie	Your Rating
From Dusk Till Dawn (1996)	2
Seven (Se7en) (1995)	4
Usual Suspects The (1995)	4

(2)

Ratings of the People Who Share Your Interests and Have Watched This Movie

(3)

Our Justified Recommendations:

Movie	The reason is	Because you rated
Scream 2	horror	25 movies with this feature
Kiss the Girls	Freeman, Morgan	19 movies with this feature
The Peacemaker	Clooney, George	10 movies with this feature

(4)

Figure 3.5: Four different explanation style formats: (1) NSE,(2) ISE ,(3) NSE,(4) KSE.

the empirical conditional probability of ratings of item i given the set of similar users for user u , denoted as N_u . For each rating value k in the set of ratings, κ , we can write this probability as:

$$\mathbf{Pr}(r_{u,i} = k | N_u) = \frac{|N_u \cap U_{i,k}|}{|N_u|} \quad (3.6)$$

where $U_{i,k}$ is the set of users who have given rating k to item i . $|N_u \cap U_{i,k}|$ is the number of users who have given rating k to item i , and are similar to user u .

Given Eq. 3.6, for each NSE we can calculate the expected value of the ratings given by the similar users to the recommended item i as follows:

$$\mathbf{E}(r_{u,i} | N_u) = \sum_{k \in \kappa} k \times \mathbf{Pr}(r_{u,i} = k | N_u) \quad (3.7)$$

The expected rating of similar users gives a reasonable and intuitive measure of goodness or strength of an histogram-based NSE explanation. In order to keep the expected values between zero and one, ratings are normalized. We can incorporate this value in our recommendation algorithm to recommend items that have higher value for this expected value. Therefore, *explainability* of item i for user u can be defined as:

$$Expl_{NSE}(u, i) = \mathbf{E}(r_{v,i}|N_u) \quad (3.8)$$

3.4.2 ISE-based Explainability

For ISE, the set of items similar to the recommended item, that user u has rated are presented as the explanation. Therefore, if the user has rated more similar items with higher ratings, the ISE would be more explainable or effective. Therefore for ISE, similar to NSE, the histogram of ratings of user u on similar items can be used to obtain the empirical conditional probability of user u 's ratings on item i . Given the set of similar items to item i , denoted as N_i , we can write this probability as:

$$p(r_{u,i} = k|N_i) = \frac{|N_i \cap I_{u,k}|}{|N_i|} \quad (3.9)$$

where $I_{u,k}$ is the set of items that were given rating k by user u and $|N_i \cap I_{u,k}|$ is the number of items that were given rating k , given by user u , and are also similar to item i .

Given Eq. 3.9, the expected value for ISE can be calculated as:

$$\mathbf{E}(r_{u,i}|N_i) = \sum_{k \in \kappa} k \times \mathbf{P}(r_{u,i} = k|N_i)$$

When ratings are normalized $\mathbf{E}(r|N_i)$ will be between zero and one. Similar to NSE, the expected value of the ratings of similar items, gives an intuitive metric for measuring goodness or strength of an ISE explanation. We will incorporate this value in our recommendation algorithm as an explainability score and strive to recommend items with higher values for this explainability score.

Therefore, *explainability* of item i for user u can be defined as:

$$Expl_{ISE}(u, i) = E(r|N_i) \quad (3.10)$$

3.4.3 KSE-based Explainability

Keyword style explainability is generally used when a content data is also available in addition to the ratings. For example, the genres of the movies can be used as features

to generate recommendations along with explanations. Similar to NSE and ISE, in KSE, we also create user-item pairs that are created using the features' domain and the values measuring the score of explainability of an item for a specific user. For this purpose, user profiles are created using a bag of words technique from the features. A user profile is a row of a users by features matrix and the value of each feature is the count of that feature in the items that the user has rated. Therefore, the explainability score of item i for user u , using the KSE technique, can be easily calculated by computing the dot product between the user-feature vector of user u and the item-feature vector of item i :

$$Expl_{KSE}(u, i) = User_Feature_u \cdot Item_Feature_i \quad (3.11)$$

3.5 Explainability Graph

Given a set of users U , a set of items I , and a set of ratings r_{ui} given by user u to item i , we capture the explainability of an item relative to a user in a bipartite graph $G = (V, E)$, with the set of vertices $V = U \cup I$, and the set of edges E from the user nodes $u \in U$ to the item nodes $i \in I$, $E = \{e_{ui} | u \in U, i \in I\}$. The edge weights in the explainability graph are stored in matrix W which represents the explainability of the items to the users. Ideally, the edge weights should be higher for items that can be easily explained and low in the opposite case. We will try to capture this mutual explainability between an item and a user in an explainability score, $Expl_{u,i}$, that will depend on the particular rationale that is chosen of the explanations. This can be based on any explanation style: $Expl_{NSE}$, $Expl_{ISE}$ or $Expl_{KSE}$. Based on the $Expl_{u,i}$, we define the edge weights in the explainability graph, and further prune any edges with low explainability. We thus define the *Explainability Matrix*, W , between user-item pairs in the Explainability graph, as follows:

$$W_{u,i} = \begin{cases} Expl_{u,i} & \text{if } Expl_{u,i} \geq \theta \\ 0 & \text{otherwise} \end{cases} \quad (3.12)$$

where $Expl_{u,i}$ is NSE, ISE or KSE and θ denotes a threshold above which we accept item i to be explainable for user u . $W_{u,i}$ thus measures the explainability of item i

Figure 3.6: An example of explainability graph. The blue and the purple nodes are the users and all the other nodes are the items. For the sample user (in color blue), the explainable items are shown in color green.

for user u . In Figure 3.6, an example of the explainability graph is shown.

3.6 Explainable Matrix Factorization

MF is a family of latent factor models that have been used with success in CF recommender system [17]. Using MF, a data matrix, R , is factored into two lower-rank approximated matrices P and Q , in a joint latent space of dimensionality, f , that is much lower than the typically large number of users or items:

$$R_{n \times m} \simeq P_{n \times f} Q_{m \times f}^T \quad (3.13)$$

MF algorithms learns the factors $p_u \in \mathbb{R}^f$ and $q_i \in \mathbb{R}^f$, which are the lower-rank representations of user u and item i in dimensionality f . To solve for p_u and q_i , different approaches, such as stochastic gradient descent (which is particularly attractive for big data), can be used to minimize the error between the approximation $p_u q_i^T$ and the input rating $r_{u,i}$, that was given by user u on item i [17].

Given the MF definition and the explainability matrix, the objective function for explainable MF to be minimized over the set of known ratings, can be formulated as:

$$J = \sum_{u,i \in R} (r_{u,i} - p_u q_i^T)^2 + \frac{\beta}{2} (\|p_u\|^2 + \|q_i\|^2) + \frac{\lambda}{2} \|p_u - q_i\|^2 W_{u,i} \quad (3.14)$$

where R is the set of user-item pairs for which the ratings are available, $\frac{1}{2}(\|p_u\|^2 + \|q_i\|^2)$ is an

$L2$ regularization term weighted by the coefficient β , and λ is an explainability regularization coefficient that controls the smoothness of the new representation and trade-off between explainability and accuracy [82, 83]. The idea here is that if item i is explainable for user u , meaning $W_{u,i} > \theta$, then their representations in the latent domain (q_i and p_u) should be close to each other, or $p_u - q_i$ is close to zero, in order for the objective function to be minimized.

To minimize the objective function, we use stochastic gradient descent, which has been used successfully to solve MF for CF with big data sets [84–86]. For a given training r_{ij} , the updates for $p_u^{(t+1)}$ and $q_i^{(t+1)}$ can be obtained as follows:

$$\begin{aligned} p_u^{(t+1)} &\leftarrow p_u^{(t)} + \alpha(2(r_{u,i} - p_u q_i^T)q_i - \beta p_u - \lambda(p_u - q_i)W_{u,i}) \\ q_i^{(t+1)} &\leftarrow q_i^{(t)} + \alpha(2(r_{u,i} - p_u q_i^T)p_u - \beta q_i + \lambda(p_u - q_i)W_{u,i}) \end{aligned} \quad (3.15)$$

where α is the step size. With a proper choice of step size, gradient descent converges to a local minimum. Proof of convergence of Eq. 3.14 is provided in Section 3.6.1. The overall algorithm for Explainable-MF is presented in Algorithm 3.3.

3.6.1 Proof of Convergence of EMF

To prove the convergence of EMF using stochastic gradient descent, we can write the objective function J (Eq. 3.14) in matrix format as follows:

$$\begin{aligned} J = Tr(RR^T) - 2Tr(RQP^T) + Tr(PQ^TQP^T) + \frac{\beta}{2}Tr(PP^T) + \frac{\beta}{2}Tr(QQ^T) \\ + \frac{\lambda}{2}(Tr(P^T D^p P) + Tr(Q^T D^q Q) - 2Tr(P^T W Q))_x \end{aligned} \quad (3.16)$$

where $D_{u,u}^p = \sum_l W_{u,l}$ and $D_{i,i}^q = \sum_l W_{l,i}$.

Theorem: The objective function J is nonincreasing under the updating rules in Eq. 3.15.

We need to prove that J is non-increasing under the updating rules. To prove this, we use an *auxiliary function*. $G(p, p')$ is an *auxiliary function* for $F(p)$ if the following conditions are satisfied:

$$G(p, p') \geq F(p), \quad G(p, p) = F(p). \quad (3.17)$$

Lemma: If G is an auxiliary function of F , then F is non-increasing under the update:

$$p^{(t+1)} = \underset{p}{\operatorname{argmin}} G(p, p^{(t)}). \quad (3.18)$$

Proof: $F(p^{(t+1)}) \leq G(p^{(t+1)}, p^{(t)}) \leq G(p^{(t)}, p^{(t)}) = F(p^{(t)})$.

Since J is not a convex function, minimization can be performed iteratively with respect to P and Q separately in each iteration. To prove the convergence of J , we show that each update rule converges separately.

Proof with respect to P :

Considering any element $p_{u,a}$ in P , we use $F_{u,a}$ to denote the part of J that is only relevant to $p_{u,a}$. Since the updates are element-wise, it is sufficient to show that $F_{u,a}$ converges under the update rule (Eq. 3.15). Therefore, the derivative of J with respect to $p_{u,a}$ can be obtained as:

$$F'_{u,a} = \left(\frac{\partial J}{\partial P} \right)_{u,a} = (-2RQ + 2PQQ^T + \beta P + \lambda D^p P - \lambda WQ)_{u,a} \quad (3.19)$$

$$F''_{u,a} = \left(\frac{\partial^2 J}{\partial P^2} \right)_{u,a} = (2QQ^T)_{a,a} + \beta + (\lambda D^p)_{u,u} \quad (3.20)$$

Lemma: Function

$$\begin{aligned} G(p, p_{u,a}^{(t)}) &= F_{u,a}(p_{u,a}^{(t)}) + F'_{u,a}(p_{u,a}^{(t)})(p - p_{u,a}^{(t)}) \\ &\quad + \frac{(2PQQ^T + \beta P + \lambda D^p P)_{u,a}}{p_{u,a}^{(t)}}(p - p_{u,a}^{(t)})^2 \end{aligned} \quad (3.21)$$

is an auxiliary function for $F_{u,a}$.

Proof: It is obvious that $G(p, p) = F_{u,a}(p)$. Therefore, we need to only show that $G(p, p_{u,a}^{(t)}) \geq F_{u,a}(p)$. Using the Taylor series expansion of $F_{u,a}$, we have

$$F_{u,a}(p) = F_{u,a}(p_{u,a}^{(t)}) + F'_{u,a}(p_{u,a}^{(t)})(p - p_{u,a}^{(t)}) + (2QQ^T + \beta + \lambda D^p)_{u,a}(p - p_{u,a}^{(t)})^2 \quad (3.22)$$

therefore we need to only show that

$$\frac{(2PQQ^T + \beta P + \lambda D^p P)_{u,a}}{p_{u,a}^{(t)}} \geq (2QQ^T + \beta + \lambda D^p)_{u,a} \quad (3.23)$$

because

$$2(PQQ^T)_{u,a} = 2 \sum_{l=1}^f p_{u,l}^{(t)} (QQ^T) \geq 2p_{u,a}^{(t)} (QQ^T) \quad (3.24)$$

and

$$\lambda(D^p P)_{u,a} = \lambda \sum_{l=1}^n D_{u,l}^p p_{l,a}^{(t)} \geq \lambda D_{u,u}^p p_{u,a}^{(t)} \quad (3.25)$$

Thus, Eq 3.22 holds. To prove the theorem, we replace $G(p, p^{(t)})$ in eq. 3.17 by eq. 3.20, and keep only the terms that depend on p :

$$\begin{aligned} \underset{p}{\operatorname{argmin}} G(p, p^{(t)}) = & \underset{p}{\operatorname{argmin}} \{ F'_{u,a}(p_{u,a}^{(t)})p + [\frac{(2PQQ^T + \beta P + \lambda D^p P)_{u,a}}{p_{u,a}^{(t)}}]p^2 \\ & - 2[\frac{(2PQQ^T + \beta P + \lambda D^p P)_{u,a}}{p_{u,a}^{(t)}}]pp_{u,a}^{(t)} \} \end{aligned} \quad (3.26)$$

$$= \underset{p}{\operatorname{argmin}} \{ [\frac{p_{u,a}^{(t)}}{(2PQQ^T + \beta P + \lambda D^p P)_{u,a}}] F'_{u,a}(p_{u,a}^{(t)})p \quad (3.27)$$

$$+ p^2 - 2pp_{u,a}^{(t)} \} \quad (3.28)$$

$$= \underset{p}{\operatorname{argmin}} \{ [\frac{p_{u,a}^{(t)}}{(2PQQ^T + \beta P + \lambda D^p P)_{u,a}}] F'_{u,i}(p_{u,a}^{(t)})p \quad (3.29)$$

$$+ p^2 - 2pp_{u,a}^{(t)} \} \\ = \underset{p}{\operatorname{argmin}} \{ [p - (p_{u,a}^{(t)} - \eta F'_{u,a}(p_{u,a}^{(t)}))]^2 \}$$

where

$$\eta = \frac{p_{u,a}^{(t)}}{2(2PQQ^T + \beta P + \lambda D^p P)_{u,a}} \quad (3.30)$$

It is easy to see that this function is minimized at the equation for the value of $p_{u,a}^{(t+1)}$.

Proof with respect to Q :

Similar to P , considering any element $q_{a,i}$ in Q , we use $F_{a,i}$ to denote the part of J that is only relevant to $q_{a,i}$.

$$F'_{a,i} = \left(\frac{\partial J}{\partial Q} \right)_{a,i} = (-2R^T P + 2QPP^T + \beta Q + \lambda D^q Q - \lambda PW)_{a,i} \quad (3.31)$$

$$F''_{a,i} = \left(\frac{\partial^2 J}{\partial Q^2} \right)_{a,i} = (2PP^T)_{a,a} + \beta + (\lambda D^q)_{i,i} \quad (3.32)$$

The rest of the proof is straightforward and similar to the case of P .

3.6.2 Explainability Effect in the Latent Space

The explainability term used in the objective function J , results in items with higher explainability for the user to be located close to that user in the latent space, while keeping

Algorithm 3.3 Explainable Matrix Factorization (EMF)

Input: data matrix R , number of factors f , number of neighbors k , α , β , and λ .

Output: P and Q

1. **for** each user u :
 - (a) calculate $N_k(u)$ using Cosine similarity
 2. **end for**
 3. **for** each user-item pair (u, i) :
 - (a) calculate $W_{u,i}$ using eq. 3.12
 4. **end for**
 5. initialize matrices P and Q
 6. **for** each $r_{u,i}$ from the training:
 - (a) solve for p_u^{new} and q_i^{new} using the update rule in eq. 3.15
 7. **end for**
-

the prediction error of the ratings small. This can be shown with an example. For $f = 2$ latent factors, we can plot the items and users in the latent space in a two-dimensional plot. In Figure 3.7, we plot the items and two sample users, A and B, in the latent space obtained using EMF and standard MF which excludes the explainability term. A sample user node is shown in black. The nodes in green color, are similar items in the latent space such that their cosine similarity with the user is more than 0.7. Items with normalized explainability value larger than 0.7, calculated using EMF_{NSE} , when $N(u) = 20$, are shown in red. All the other item nodes are shown in cyan color. Both EMF and MF techniques have predicted a sufficient set of relevant items close to the test users that can be suggested. However, the main difference is in the placement of explainable items in red. Using EMF, for both test users, explainable items, shown as red points, are factorized and located close to the user, however using MF which excludes the explainability term when estimating the factors f_1 and f_2 , red points are spread throughout the whole space. In other words, EMF captures explainability in the latent factors and factorizes the users and items such that explainable items are located among the relevant items for the recommendation, without decreasing the

Figure 3.7: Red points = explainable items, green points = relevant items, Cyan points = remaining items, black point = sample user. For two sample users (top and bottom rows), the items are represented in the latent space ($f = 2$, for visualization purpose). In each case, EMF has resulted in explainable items to be located closer to the user and in the green area (relevant items for recommendations).

accuracy in recommending relevant items with large enough cosine similarity.

Table 3.1, we list the mean and the standard deviation of the number of explainable items in the top- n recommendation for all the test users, calculated using EMF and MF. It can be observed that EMF has significantly increased the number of explainable items selected in the recommendation lists compared to the standard latent factor model (MF).

3.7 Explainable Restricted Boltzmann Machines

Unlike the traditional RBM, which is agnostic to explanations, we propose a constrained RBM model that takes explainability into account with an additional visible layer, v' , with m nodes, where m is the number of items. Each node, v' has a value between

TABLE 3.1

Number of explainable items in the top-10 recommendation generated using EMF has larger mean and standard deviation than using MF technique. Paired T-test on the difference showed the EMF has improved at 1% level or higher ($p\text{-value} = 7.784e - 12$).

t-test	MF	EMF	$diff(EMF, MF)$
mean	9.058	9.169	0.111
std.	0.305	0.394	0.4931

0 and 1, indicating the explainability score of the relative item to the current user in the iteration, calculated as explained in Section 3.5. The idea is to define a joint distribution over hidden nodes and visible nodes, conditional on the explainability scores. The energy function of the system, E , can be defined as follows:

$$E(v, h) = - \sum_{i=1}^n \sum_{j=1}^m w_{ij} h_i v_j - \sum_{j=1}^m b_j v_j - \sum_{i=1}^n c_i h_i - \sum_{i=1}^n \sum_{j=1}^m w'_{ij} h_i v'_j - \sum_{j=1}^m b'_j v'_j \quad (3.33)$$

where m and n are the numbers of visible units and hidden units respectively, w_{ij} is a real valued weight associated with the edge between the units of visible units and hidden units, w'_{ij} is a real valued weight associated with the edge between the units of explainability units and hidden units; and b_j , b'_j and c_i are real valued bias terms associated with the visible, explainability and hidden units, respectively.

Figure 3.8 presents the conditional RBM model with explainability. Similar to [59], the $p(h_j = 1|v, v')$, $p(v_i = 1|h)$, and $p(v'_i = 1|h)$ are defined as:

$$p(h_i = 1|v, v') = \sigma(c_i + \sum_{j=1}^m v_j w_{ij} + \sum_{j=1}^m v'_j w'_{ij}) \quad (3.34)$$

$$p(v_j = 1|h) = \sigma(b_j + \sum_{i=1}^f h_i w_{ij}) \quad (3.35)$$

$$p(v'_j = 1|h) = \sigma(a_j + \sum_{i=1}^f h_i w'_{ij}) \quad (3.36)$$

where a , b , and c are biases of the nodes, m and f are the numbers of visible and hidden units, respectively, and w and w' are the weights of the edges in the network. $\sigma(x)$ is the logistic function $\frac{1}{(1+e^{-x})}$ [87].

Figure 3.8: Conditional RBM for explainable recommendations.

To learn the values in w and w' , we follow an approximation to the gradient of a different objective function called Contrastive Divergence (CD) [88]:

$$\Delta w_{ij} = \epsilon_w (\langle v_j h_i \rangle_{data} - \langle v_j h_i \rangle_{recom}) \quad (3.37)$$

where w_{ij} is an element of a learned matrix that models the effect of ratings on h . Learning w' , which is the effect of explainability on h , using CD, is similar and takes the form:

$$\Delta w'_{ij} = \epsilon_{w'} (\langle v'_j h_i \rangle_{data} - \langle v'_j h_i \rangle_{recom}) \quad (3.38)$$

3.8 Summary

In this chapter, we proposed new explainable latent factor-based recommendation algorithms based on Matrix Factorization and on Restricted Boltzmann Machines. We also presented a MF-based method for building multiple-domain recommendation systems. In addition to providing explanations based on multiple domains, this approach also addresses the most common problems of CF recommender systems which is the cold-start problem. We explained how our approach solves the new item cold-start problem even if no rating data is available at all for a specific item. In addition to the ability to make predictions for new users or new items, any recommender system should have the ability to effectively explain its recommendations to users. Without explanation, even if the results are very

accurate, there is a lower chance for the system to be adopted by the users. One goal of any recommender system is to help users make more accurate decisions when choosing an item and this is more achievable when an explanation is provided. We showed how our proposed recommender system integrates *explainability* with other input data such as ratings or genre data to suggest explainable recommendations. In the next chapter, we will present experiments that confirm the superiority of our proposed techniques compared to existing methods in the literature.

CHAPTER 4

EXPERIMENTAL EVALUATION

In this chapter, we present the results for the methods explained in Section 3. First, we present the proposed multi-domain MF-based recommender system results its capability to handle the cold-start problem; then, the results for EMF and ERBM techniques are presented. Finally, we present the user study experiment design and results. The chapter ends with the summary section.

4.1 Cross-modal MF-based CF

We tested our cross-modal asymmetric MF technique for CF on the Movielens [89] dataset which consists of 100,000 ratings, on a scale of 1 to 5, for 1700 movies and 1000 users.

The data is first split into training and test sets such that 10% of the latest ratings from each user are selected for the test set and the remaining 90% of the ratings are used in the training set.

We vary f and present the results in terms of Root Mean Square Error (RMSE). RMSE is given by:

$$RMSE = \sqrt{\frac{\sum_{r_{i,j} \in R^{test}} (r_{ij} - p_i q_j^T)^2}{|R^{test}|}} \quad (4.1)$$

Figure 4.7 shows that by increasing f , the RMSE on test data has increased. For each f , the RMSE has decreased monotonically in each iteration, until convergence. In the second experiment, we investigate the performance of our approach on the item-based cold start problem. The test set is selected by choosing a different percentage of all 1700 movies (from 5% to 50%) randomly. All the ratings of the selected movies are changed to unrated to test

TABLE 4.1

Top-5 rated movies by Sample User A, along with the movies' genres.

Top-5 rated movies	Genres
Fargo (1996)	Comedy, Romance
The English Patient (1996)	Crime, Drama, Thriller
Postino II (1994)	Drama, Romance, War
Leaving Las Vegas (1995)	Drama, Romance
L.A. Confidential (1997)	Drama, Romance

the impact of cold-start. We compared our algorithm with the following approaches:

- Standard latent factor model as Matrix Factorization
- Content Boosted Collaborative Filtering (CBCF) [90]

CBCF is the state of the art hybrid technique using both ratings and content data. For MF and Asymmetric MF, f is set to 5. All the experiments were repeated 10 times and the average metrics are reported in Figure 4.8.

For a sample user, A, the top-5 rated items and their genres are shown in Table 4.1. The frequency of the genres of the movies rated by this user is shown in Figure 4.1. The three most rated genres of user A are: drama, comedy and romance. The recommendations along with their explanations are obtained using the proposed asymmetric MF, when $f = 5$ and the number of neighbors around the users or the items for generating NSE or ISE is set to 50. Top-3 recommendations along with their genres are presented in Table 4.2. Corresponding NSE and ISE-based explanations generated in the latent space for the top-3 recommendations are shown in Figures 4.2 and 4.3.

For another sample user, B, results similar to user A are obtained and presented in Tables 4.3 and 4.4, and Figures 4.4, 4.5, and 4.6.

4.2 Explainable Matrix Factorization (EMF)

We tested our approach on the benchmark MovieLens [89] ratings data which consists of 100,000 ratings, on a scale of 1 to 5, for 1700 movies and 1000 users. The ratings data is

TABLE 4.2

Top-3 recommended movies to Sample User A, along with the movies' genres.

Top-3 recommendations	Genres
In Love and War (1996)	Romance, War
Friday (1995)	Comedy
Little City (1998)	Comedy, Romance

Figure 4.1: Genre frequency of the rated movies by Active User A.

TABLE 4.3

Top-5 rated movies by Sample User B, along with the movies' genres.

Top-5 rated movies	Genres
Hate (Haine La) (1995)	Drama
Bottle Rocket (1996)	Comedy
Trainspotting (1996)	Drama
The Last Supper (1995)	Drama, Thriller
The Godfather (1972)	Action, Crime, Drama

Figure 4.2: Corresponding Top-3 NSE-based explanations generated for the recommendations shown in Table 4.2. Recommendations/explanations are generated using asymmetric MF for Sample User A.

Figure 4.3: Corresponding Top-3 ISE-based explanations generated for the recommendations shown in Table 4.2. Recommendations/explanations are generated using asymmetric MF for Sample User A.

TABLE 4.4

Top-3 recommended movies to Sample User B, along with the movies' genres.

Top-3 recommendations	Genres
Pete's Dragon (1977)	Adventure, Animation, Musical
The Cook the Thief His Wife and Her Lover (1989)	Drama
Boomerang (1992)	Comedy, Romance

Figure 4.4: Genre frequency of the rated movies by Active User B.

first split into training and test sets such that 10% of the ratings from each user are selected for the test set and the remaining 90% of the ratings are used in the training set. For the content data in feature-based techniques, the benchmark movie-genre dataset from the same Movielens data is used. The content consists of 19 genres for the movies. Each movie is represented with one or more genres. The list of genres available are as follows: action, adventure, animation, children's, comedy, crime, documentary, drama, fantasy, film-noir, horror, musical, mystery, romance, sci-fi, thriller, war, western, and other.

We compare our results with several nearest neighbor based or MF-based collaborative filtering and hybrid baseline methods that are most related to our approach:

- A standard latent factor model, namely Matrix Factorization (MF) [17]

Figure 4.5: Corresponding Top-3 NSE-based explanations generated for the recommendations shown in Table 4.4. Recommendations/explanations are generated using asymmetric MF for Sample User B.

Figure 4.6: Corresponding Top-3 ISE-based explanations generated for the recommendations shown in Table 4.4. Recommendations/explanations are generated using asymmetric MF for Sample User B.

Figure 4.7: RMSE calculated over the testing data with each iteration decreases. The setting for this experiment is $\alpha = 0.001$ and $\beta = 0.01$, when varying f .

- Probabilistic Matrix Factorization (PMF) [91]
- User-Based (UB) top- n CF [92]
- Item-Based (IB) top- n CF [93]
- Content Boosted Collaborative Filtering (CBCF) [90]

We compare our approach to the baselines in terms of their top- n recommendation list. Therefore, for MF-based techniques, the top- n recommendation list for each user is generated by selecting the top- n items with highest dot product score between the user and the items' representations in the latent space. Standard user-based and item-based nearest-neighbor techniques are the state of the art top- n CF techniques that do not require content data. We use the cosine similarity to find the similar users/items when generating the explainability matrix. The parameters are tuned using cross-validation. Figure 4.9, shows the objective function decreasing in each iteration and eventually converging. In this experiment, for all three EMF techniques, we set $f = 5$ and set the neighborhood size around users or items, collectively denoted as $|N|$ to 50. The explainability threshold θ is set to zero.

Figure 4.8: Comparison of accuracy when varying the percentage of data used in the test.

4.2.1 Recommender System Evaluation

To evaluate the top- n recommendation results, we use two of the most common top- n metrics: Mean Average Precision (MAP) at cutoff 50 and Area Under Curve (AUC), calculated by varying the size of the top- n recommendation list. Table 4.10 shows the MAP and AUC results when varying the number of factors, f . For EMF techniques, the neighborhood size around users or items, collectively denoted as $|N|$, is set to 50. Other

Figure 4.9: Objective function, J , calculated over the training data in each iteration decreases until convergence. The setting for this experiment is $f = 5$, $|N| = 50$, and $\theta = 0$ for all three explanation styles.

parameters are tuned and the best are found to be as follows: $\alpha = 0.001$, $\beta = 0.01$, $\lambda = 0.005$. For $f = 5$, PMF has higher accuracy than EMF_{NSE} , but by increasing f , the MAP and AUC of PMF exceed those of EMF_{NSE} . For $f = 5$, MF outperforms both EMF methods. This can be attributed to the impact of the explainability constraint on the learned hidden factors that result in a decrease in the accuracy for small f . Except for $f = 5$, EMF_{ISE} has the lowest accuracy comparing to EMF_{NSE} and EMF_{KSE} . This is because people usually tend to rate a small set of items. Therefore ISE-based explainability values, which are based on the similar set of items, are usually lower than those of NSE and KSE. Table 4.11 presents the MAP and AUC results when the neighborhood size around the users or items, denoted as $|N|$ is varied. Both CF techniques, user-based (UB) and item-based (IB) nearest neighbor techniques have lower MAP and AUC values compared to other techniques, when varying $|N|$. This is because memory based approaches generally have lower accuracy due to considering only local samples for generating recommendation lists. Since EMF_{KSE} is not neighborhood-based, its MAP and AUC do not change when varying $|N|$. For $|N| = 5$, EMF_{KSE} outperforms other techniques. By increasing $|N|$, EMF_{ISE} outperforms other techniques. However, increasing $|N|$ can affect the NSE based explainability values more than ISE and therefore decrease the accuracy.

Figure 4.10: Comparison of accuracy when varying f .

Using asymmetric MF technique both NSE and ISE can be generated automatically for the recommended item, as opposed to neighbor-based CF or the item-item CF technique where it is only NSE-based or ISE-based.

4.2.2 Explainability Evaluation

To further assess the quality of the proposed approach, it is important to compare the results with other approaches in terms of their explainability. Note that in this work,

Figure 4.11: Comparison of accuracy when varying $|N|$.

we are not proposing a new explanation format that requires user evaluation. However, we can evaluate the top- n recommendations in terms of explainability of the suggestion list. We measure explainability using the mean explainability precision and mean explainability recall metrics [82]. At top- n recommendation, *Explainability Precision*, denoted as xP , is defined as the proportional of explainable items in the top- n recommendation list for each user, u :

Figure 4.12: Comparison of user based xP , xR , and xF_score when varying f . For EMF techniques we set $|N| = 50$.

Figure 4.13: Comparison of item based xP , xR , and xF_score when varying f . For EMF techniques we set $|N| = 50$.

Figure 4.14: Comparison of feature based xP , xR , and xF_score when varying f . For EMF techniques we set $|N| = 50$.

Figure 4.15: Comparison of user based xP , xR , and xF_score when varying $|N|$.

Figure 4.16: Comparison of item based xP , xR , and xF_score when varying $|N_i|$.

$$xP = \frac{|\{i : i \in \text{top} - n, \text{Expl}_{u,i} > \theta\}|}{|\text{top} - n|} \quad (4.2)$$

Similar to the recall metric, we define *Explainability Recall*, denoted as xR , as the proportion of explainable items in the top- n recommendation list for each user:

$$xR = \frac{|\{i : i \in \text{top} - n, \text{Expl}_{u,i} > \theta\}|}{|\text{Expl}_{u,i} > \theta|} \quad (4.3)$$

The mean xP and Mean xR , which are the average values of xP and xR over all users, are calculated with the three different explainability formulations: UB, IB and KSE.

To combine both xP and xR metrics, we define the Explainability F-score, noted as xF_score which is the harmonic mean of xP and xR , as follows:

$$xF_score = 2\left(\frac{xP \cdot xR}{xP + xR}\right) \quad (4.4)$$

Based on the explainability type used in calculating $\text{Expl}_{u,i}$, the explainability metrics xP , xR and xF_score can be either user-based, item-based or feature-based. Figures 4.12, 4.13, and 4.14 present the xP , xR and xF_score results, when varying f for each type of explainability, NSE, ISE, and KSE, respectively. For EMF techniques, $|N|$ is set to 50. Figure 4.12 presents the comparison results of explainability metrics between the EMF_{NSE} technique and PMF and MF methods, when varying f . To compute xP , xR and xF_score for PMF and MF, $\text{Expl}_{u,i}$ between the user-item pairs is calculated using Eq. 3.8. EMF_{NSE} has higher xP and xR values comparing to other techniques. As a result, the xF_score of EMF_{NSE} is higher. Figure 4.13 presents the comparison results of explainability metrics between the EMF_{ISE} technique and PMF and MF methods, when varying f . To compute xP , xR and xF_score for PMF and MF, $\text{Expl}_{u,i}$ between the user-item pairs is calculated using Eq. 3.10. When $f < 50$, EMF_{ISE} has higher xP and xR and therefore xF_score values than other techniques. For $f = 50$, MF has higher values for all explainability metrics. This can be due to the lower effect of the explainability constraint when a larger number of factors is used in the optimization of the error between the actual ratings and estimations. Figure 4.14 presents the comparison results of explainability

TABLE 4.5: Performance of the EMF when varying θ .

EMF_{UB}					EMF_{IB}				EMF_{KSE}			
θ	MAP	AUC	xP	xR	MAP	AUC	xP	xR	MAP	AUC	xP	xR
0.2	0.0113	0.5976	0.3287	0.2448	0.0119	0.6027	0.3074	0.0941	0.0141	0.6695	0.9782	0.0601
0.4	0.0118	0.6156	0.1714	0.2018	0.0111	0.6257	0.1437	0.082	0.0141	0.6845	0.9782	0.0601
0.6	0.0115	0.6128	0.0713	0.2001	0.0097	0.5574	0.0433	0.0569	0.0138	0.6645	0.8313	0.0508
1	0.0117	0.6373	0	0	0.0104	0.5831	0	0	0.0148	0.5432	0.3398	0.0205
Avg.	0.0115	0.6158	0.1429	0.1614	0.0107	0.5922	0.1236	0.0582	0.0141	0.6404	0.7821	0.0478

metrics between the EMF_{KSE} technique and PMF and MF methods, when varying f . To compute xP , xR and xF_score for PMF and MF, $Expl_{u,i}$ between the user-item pairs (u,i) is calculated using Eq. 3.11 for the MovieLens movie-genre dataset. Using feature-based or keyword style explainability for calculating xP , xR and xF_score metrics, MF has higher xP , but lower xR comparing to EMF_{KSE} . However, in terms of xF_score , which combines xP and xR , EMF_{KSE} outperforms other techniques.

Similar to f , $|N|$ is varied and the explainability results are shown in Figures 4.15 and 4.16 when explainability is calculated using NSE and ISE, respectively. Note that KSE-based explainability is independent of the variable $|N|$. For EMF techniques, f is set to 10. Figure 4.15 shows that EMF_{NSE} outperforms other techniques in terms of explainability, when varying the neighborhood size around the users, $|N_u|$. Figure 4.16 also shows that EMF_{ISE} outperforms other techniques in terms of explainability, when varying the neighborhood size around the items, $|N_i|$.

To study the effect of the threshold in Eq. 3.12, θ , on the explainability and accuracy of the recommendations, we varied θ , while fixing all the other parameters (Table 4.5). For all three EMF techniques, xP and xR decrease by increasing the threshold, θ . EMF_{KSE} has a higher xP comparing to EMF_{NSE} and EMF_{ISE} , while EMF_{NSE} has higher xR comparing to the other two EMF techniques when $\theta < 1$. When $\theta = 1$, almost no user-based or item-bases neighbor had a value of one. This is because users usually tend to rate a small set of items.

4.3 Explainable Restricted Boltzmann Machines (ERBM)

Similar to EMF, we tested our approach on the MovieLens ratings data which consists of 100,000 ratings on a scale of 1 to 5, for 1700 movies and 1000 users [89]. For the content data in feature-based techniques, the benchmark movie-genre dataset from the same MovieLens data is used. The dataset consists of 19 genres for the movies. Each movie is represented with one or more genres. The list of genres available are as follows: action, adventure, animation, children's, comedy, crime, documentary, drama, fantasy, film-noir, horror, musical, mystery, romance, sci-fi, thriller, war, western, and other.

The data is split into training and test sets such that 10% of the latest ratings from each user are selected for the test set and the remaining are used in the training set. Ratings are normalized between 0 and 1, to be used as RBM input. Each Experiments is run 10 times and the average results are reported. Figure 4.17 shows the MAP and AUC results of the three ERBM techniques comparing to the standard RBM.

Figures 4.17 and 4.18 present the accuracy of the ERBM techniques in terms of MAP and AUC techniques when varying f or $|N|$. ERBM techniques have higher MAP but lower AUC compared to the RBM model.

Figure 4.19 presents the comparison of $ERBM_{UB}$ with RBM in terms of xP , xR , and xF_score when the explainability is calculated using the $Expl_{NSE}$ formulation (Eq. 3.8) and f is varied. In the cases where RBM outperforms $ERBM_{UB}$ in terms of xP , RBM has lower xR results. As a result, $ERBM_{UB}$ has a higher $xFscore$ comparing to RBM.

Figure 4.20 shows the results of comparing $ERBM_{IB}$ with RBM when the explainability metrics are calculated using the ISE formulation (Eq. 3.10) and when f is varied. $ERBM_{IB}$ outperforms RBM in terms of xP , xR , and $xFscore$.

Figure 4.21 presents the comparison of $ERBM_{KSE}$ with RBM using the keyword style explainability formulation (Eq. 3.11), when varying f . RBM outperforms $ERBM_{KSE}$ in terms of xP , but has lower xR values, when varying f . In terms of $xFscore$, which combines both xP and xR , $ERBM_{KSE}$ outperforms RBM.

In Figures 4.22 and 4.23, the neighborhood size around the user or the neighbor,

Figure 4.17: Comparison of accuracy when varying f . For EMF techniques we set $|N(u)| = 50$.

$|N_u|$, is varied and the explainability metrics calculated using NSE and ISE formulations are reported. For NSE, $ERBM_{NSE}$ outperforms RBM when varying $|N_u|$.

Similar to EMF, to study the effect of the threshold θ in Eq. 3.12, on the explainability and accuracy of the recommendations, we varied θ , while fixing all the other parameters (Table 4.6). $ERBM_{NSE}$ has larger xP and xR values than $ERBM_{ISE}$ and $ERBM_{KSE}$, on average. When $\theta = 1$, almost no explainability metric had a value of one. This is because

Figure 4.18: Comparison of accuracy when varying $|N|$.

TABLE 4.6: Performance of the ERBM when varying θ .

θ	$ERBM_{NSE}$				$ERBM_{ISE}$				$ERBM_{KSE}$			
	MAP	AUC	xP	xR	MAP	AUC	xP	xR	MAP	AUC	xP	xR
0.2	0.019	0.7118	0.4050	0.1970	0.023	0.7383	0.0726	0.0608	0.0288	0.7482	0.2521	0.0184
0.4	0.0188	0.6926	0.2114	0.2679	0.0231	0.7321	0.0647	0.1089	0.0314	0.775	0.2605	0.0304
0.6	0.0186	0.7094	0.0831	0.3507	0.0283	0.79	0.0629	0.1070	0.025	0.7302	0.1313	0.0406
1	0.0177	0.6787	0	0	0.0297	0.7896	0	0	0.0294	0.7889	0	0
Avg.	0.0185	0.6981	0.1749	0.2039	0.026	0.7625	0.05	0.0692	0.0286	0.7605	0.1609	0.0223

Figure 4.19: Comparison of user based xP , xR , and xF_score when varying f . For $ERBM$ techniques we set $|N| = 50$.

Figure 4.20: Comparison of item based xP , xR , and xF_score when varying f . For $ERBM$ techniques we set $|N| = 50$.

Figure 4.21: Comparison of feature based xP , xR , and xF_score when varying f . For $ERBM$ techniques we set $|N| = 50$.

Figure 4.22: Comparison of user based xP , xR , and xF_score when varying $|N|$.

Figure 4.23: Comparison of item based xP , xR , and xF_score when varying $|N_i|$.

TABLE 4.7

Top-5 rated movies by a sample user along with the movies' genres.

Top-5 rated movies	Genres
My Best Friend's Wedding (1997)	Comedy, Romance
Men in Black (1997)	Action, Adventure, Comedy, Sci-Fi
Jerry Maguire (1996)	Drama, Romance
Ransom (1996)	Drama, Thriller
White Squall (1996)	Adventure, Drama

Figure 4.24: Example of an automated explanation generated for the recommended movie: “Lost Highway (1997)” using the EMF_{NSE} technique.

users usually tend to rate a small set of items.

For a test user, the top-5 rated movies and their genres, are presented in Table 4.7. Explanations can be presented to the users using the format shown in Figure 3.5. Figure 4.24 shows the recommendation and the NSE explanation generated for this user using the EMF_{NSE} technique. Similarly, Figures 4.25 and 4.26 present the recommendations and explanations generated using the ISE and KSE styles, respectively.

4.4 User Study

In this section, the validity of the explainability metric, as defined in Section 3.4 is studied by means of a user experiment. Given the definition of the explainability metric in

Movie	Your Rating
Quest The (1996)	2
Down Periscope (1996)	3
Eraser (1996)	3
Rock The (1996)	4
Multiplicity (1996)	2
Scream (1996)	4

Figure 4.25: Example of the automated explanation generated for the recommended movie: “McHale’s Navy (1997)” using the EMF_{ISE} technique.

The reason is	Because you rated
Children's	6
Comedy	25
Mystery	1

Figure 4.26: Example of the automated explanation generated for the recommended movie: “That Darn Cat! (1965)” using the EMF_{KSE} technique.

Eq 3.7, the research questions are as follows:

- Does an explanation with higher explainability increase perceived transparency? Transparency means providing information so that the user can comprehend how the system works and the justification behind the recommendation [94].
- Does the explainability value of the explanation, whether it is low or high, impact the user satisfaction?

Figure 4.27: Comparison between two explanations for a test user where the explainability of the item i_1 on the left is higher than item i_2 one on the right.

4.4.1 Hypothesis

Suppose we have two items i_1 and i_2 along with their respective explanations as the justification of why they were recommended by an automated intelligent system. Given the definition of the explainability metric in Eq 3.7, if i_1 has higher computed explainability value than i_2 (Figure 4.27), does recommending i_1 result in a better satisfaction as judged by the user? Our hypothesis can be summarized as follows: Recommending an item with higher computed explainability metric (defined in Section 3.4) will lead to higher user satisfaction.

4.4.2 Methods

This user study is performed through a web platform similar to commercialized recommender engines (e.g. Netflix for recommending movies). The application focuses primarily on recommending movies selected from the MovieLens dataset.

Based on the computed explainability metric values, the explanations are divided into three groups of low, medium, and high explainability as follows:

- low: explainability value < 2 .
- medium: $2 \leq$ explainability value < 4 .
- high: explainability value ≥ 4 .

4.4.3 Subject Recruitment

Volunteers were recruited through fliers or email across the University of Louisville to join in this experiment. The participants were provided an online link to the web application to participate in the experiment. The incentives for participation was a drawing for three \$50 gift cards from the University of Louisville.

4.4.4 Sample Size Estimation

To estimate the sample size, a statistical power analysis was performed. The effect size in this study was considered to be large using Cohen's [95] criteria. When α is set to 0.05, and power is set to 0.8, the sample size needed is approximately 10.

31 people participated in this experiment where they were assigned randomly to a group. The number of people in each group are as follows:

- Low = 10
- Medium = 11
- High = 10

4.4.5 Procedures

The flow of the experiment is as follows:

1. The user is asked to rate (ratings from 1 to 5) at least 10 movies they have watched previously, from a selection of movies. This data is collected only to allow the recommendation system to suggest films that the user may enjoy and will not be used to answer the study's research question.
2. Based on the group the user was assigned to, a recommendation along with an explanation are presented in a format similar to Figure 4.27. This recommendation along with the explanation will be selected from a pool of recommendations that are calculated using the method proposed in Section 3.6, such that the computed explainability

Figure 4.28: Example of movies shown to the user to rate.

metric value falls in the range of the experimental group that the user was assigned to (low, medium, or high).

3. The user is asked to fill out a Likert Scale questionnaire. Table 4.8 presents the questions used in this questionnaire
4. Demographic information is collected from the user. This data is: age, gender, major of study, weekly hours watching movies, and favorite movie genres. This information is requested to study potential confounding factors on the user's satisfaction with the explanations. Table 4.9 shows the demographic questions used in this experiment.
5. After submission, the user is redirected to a Google Form and asked to enter their contact information for the drawing.

The duration of the experiment will be less than 30 minutes. Snippets of the application are shown in Figures 4.28 and 4.29.

Our recommendation is:

Taxi Driver, 1976

Our explanation for this recommendation is:

Figure 4.29: Example of a recommendation with explanation presented to the user.

TABLE 4.8

Likert scale survey questions.

question 1	“Based on the ratings of people with similar interest to mine, this is a good recommendation.”
question 2	“This explanation helps me understand why this movie was recommended.”
question 3	“Based on the ratings of people with similar interests to mine, I will watch this movie.”
question 4	“Based on the ratings of people with similar interests to mine, I can determine how well I will like this movie.”
question 5	“This explanation helps me understand how the recommender system works.”

TABLE 4.9

Demographic questions.

gender	“What is your gender?”
age	“What is your age?”
weekly hours	“How many hours per week do you watch movies on average?”
favorite genres	“What are your favorite genres?”
familiarity	“How familiar are you with automated recommender systems?”

4.4.6 Analysis of Results

Users answered 5 different statement questions, as shown in Table 4.8, using the five-level Likert scale. Figure 4.30 shows the distribution of the answers to each question using a horizontal bar graph. “somewhat agree” has the highest frequency for all the questions. For questions 4 and 5, no “strongly disagree” answers were recorded. Answers are also plotted for each group and displayed in Figures 4.31, 4.32, and 4.33. In the group “low”, most people’s answers to questions 1 and 3, were “strongly disagree” and no answer was “strongly agree”. Only the answers to questions 4 and 5 were towards the more positive scale. As a result, people in the group “low” were not very satisfied with the recommendation based on the explanation. However, based on questions 4 and 5, they still believed that the

Figure 4.30: Horizontal bar graph of the answers to the questions in Table 4.8.

explanations were useful for determining how well they will like the movie and for providing a justification of how the system works.

In the group “medium”, most replies were “neither agree nor disagree” or “agree” to all the questions. Only a few answers to questions 2 and 5 were “strongly agree”. Also, one reply to question 4 was “somewhat disagree”. At the conclusion, people in group “medium” seemed to be generally satisfied with their recommendations along with the explanations.

In group “high”, most answers were “somewhat agree” or “strongly agree” to all the questions. Only one reply to question 4 and one reply to question 5 were recorded to be “neither agree nor disagree”. Similar to the group “medium”, people in the group “high” were also satisfied with their recommendations and explanations, however their answers were more positive using the Likert scale.

Figure 4.34 shows the heat-map of the answers for the questions. “Somewhat agree” has the highest frequency for all the questions. In questions 4 and 5, the number of “strongly disagree” answers was zero. A heat-map graph for each group is displayed in Figures 4.35, 4.36, and 4.37.

In the group “low”, question 1 has the most number of “strongly disagree” answers. For questions 1, 2, and 3, people have answered towards the disagree range, as opposed to questions 4 and 5, where most answers were towards the “agree” range. In the group “medium”, “somewhat agree” has the highest frequency for all the questions. In the group

Figure 4.31: Horizontal bar graph of the answers of people in the group "low" to the questions in Table 4.8.

Figure 4.32: Horizontal bar graph of the answers of people in the group "medium" to the questions in Table 4.8.

Figure 4.33: Horizontal bar graph of the answers of people in the group "high" to the questions in Table 4.8.

Figure 4.34: Heat-map plot of the answers to the questions in Table 4.8.

Figure 4.35: Heat-map plot of the answers of people in the group "low" to the questions in Table 4.8.

"high", "somewhat agree" and "strongly agree" had the highest frequencies and only two people have answered "neither agree nor disagree". The heat-map in this group is strongly positive towards the agree range.

Figures 4.38, 4.39, 4.40, 4.41, and 4.42 present the distribution of the participants' answers to the demographic questions. Note that replying to these questions (Table 4.9) was optional and a choice of "other" was available for the gender type question.

Figure 4.36: Heat-map plot of the answers of people in the group "medium" to the questions in Table 4.8.

Figure 4.37: Heat-map plot of the answers of people in the group "high" to the questions in Table 4.8.

Figure 4.38: Distribution of the participants' gender.

Figure 4.39: Distribution of the participants' age.

Figure 4.40: Distribution of the participants' weekly hours watching movies.

Figure 4.41: Distribution of the participants' familiarity with recommender systems.

Figure 4.42: Distribution of the participants' favorite genres.

4.4.7 Hypothesis Testing

As the plots in Section 4.4.6 show, the participants' answers varied based on the explainability group they were assigned to randomly. Before evaluating the significance of the difference between the answers of the groups, it is important to assess the reliability of the Likert scale questionnaire which is closely associated with the validity of the experiment [96]. For this purpose, the Cronbach's alpha test is performed. The standardized Cronbach's alpha based upon the correlations for the survey questionnaire with 5 questions and 31 samples was 0.77, which is in the acceptable range (> 0.7). When question 4 was removed

TABLE 4.10

Categorization of the survey questions from Table 4.8 according to the research questions.

Explanation aspect	Question
Transparency	Q2 and Q5
Satisfaction	Q1 and Q3

Cronbach’s alpha increased to 0.8. To address the three research questions on the impact of explainability on “transparency”, “satisfaction”, and “effectiveness”, as perceived by the users, survey questions (Table 4.8) were grouped accordingly. Table 4.10 presents this categorization.

To study the effect of the explainability variable on each of the three explanation aspects, Analysis of Variance (ANOVA) was conducted for each aspect separately. The null hypothesis for this test is whether the means of the three groups, “low”, “medium”, and “high”, are equal.

- *Transparency:* Using Questions 2 and 5 in the ANOVA test, with degrees of freedom equal to 2, the F-value is 7.881 and the p-value is 0.00193 which is less than 0.05. Hence we can conclude that for our confidence interval, there is a significant relationship between the explainability and transparency. The eta-squared measure of effect size for the ANOVA is 0.3601. To determine which groups are different from the others, Tukey’s HSD post-hoc test is conducted. At 95% family-wise significance interval the differences and adjusted p-values are reported in Table 4.11. There is a significant difference in transparency between the group “low” and the group “high”. However, there is no significant difference between the “medium” and “high” groups, nor between the “medium” and “low” groups (p-value > 0.05). Figure 4.43 provides a visualization of the differences in the group pairs.
- *Satisfaction:* Questions 1, 3 and 4 are used in the ANOVA test for evaluating user satisfaction. The degrees of freedom, F-value, and the obtained p-value are 2, 47.07, and $1.11e - 09$, respectively. We thus clearly reject the null hypothesis of equal

TABLE 4.11

Tukey multiple comparisons of means at 95% family-wise confidence interval for transparency.

Group pairs	difference	adjusted p-value
Low-High	-1.2000	0.0013
Med-High	-0.5454	0.1738
Med-Low	0.6545	0.0863

TABLE 4.12

Tukey multiple comparisons of means at 95% family-wise confidence interval for satisfaction.

Group pairs	difference	adjusted p-value
Low-High	-1.9000	0.0000
Med-High	-0.7545	0.0014
Med-Low	1.1454	0.0000

means for all three explainability groups. The eta-squared measure of effect size for the ANOVA is 0.7707. Similar to the transparency, Tukey’s HSD post-hoc test is conducted to determine which groups are significant. Table 4.11 as well as the difference visualization in Figure 4.44, show that there is a significant difference in the satisfaction within all three paired groups.

4.5 Summary

In this chapter, we presented the experimental results validating the methods and algorithms proposed in Chapter 3. We showed that different explainability styles can be incorporated in generating recommendations. In addition to offline evaluation, our user experiments evaluated the explainability metric proposed in Chapter 3.

Figure 4.43: Visualization of group pairs' differences for "transparency".

Figure 4.44: Visualization of group pairs' differences for "satisfaction".

CHAPTER 5

CONCLUSION

We presented a cross-modal recommender engine that leverages multiple domains of data to retrieve similar items and recommend the best items to the user. We showed how our approach can automatically generate multimodal explanations for the recommendations and also has the potential to alleviate the cold-start problem, one of the most notorious limitations of Collaborative Filtering (CF) techniques.

Furthermore, we proposed a probabilistic formulation for capturing the explainability of Neighbor-Style Explanations (NSE), Influence Style Explanations (ISE), and feature Style Explanations (KSE) for recommendations. This user-item explainability relationship is encoded in a bipartite graph structure. While other methods in the literature have used graph structures to find a better representation of data points in lower spaces, we further incorporated the explainability graph in the design of the MF model to be able to generate explainable recommendations.

We then proposed an Explainable-Matrix Factorization (EMF) model for providing explainable recommendations that can leverage the accurate predictions of MF and the transparency of neighborhood-based CF algorithms. In our method, explainability can be directly formulated based on the rating distribution within the user’s or item’s neighborhood. Our rationale is that if many neighbors have rated the recommended item, or if the user has rated many items that are similar to the recommended item, then this provides a basis upon which to explain the recommendations. In addition to neighbor-style explanations, we proposed an explainability metric based on Keyword Style Explanation that uses content and features to generate explanations and recommend explainable items. The rationale in the KSE-based explainability is that if a user is similar to an item in a specific

content feature space, such as genres in the movie recommendation context, then those features can be used to provide explanations. We focused our research on CF recommender systems which have been shown to perform better than Content Based (CB) filtering methods [97]. Our EMF results showed that using explainability has increased the number of explainable items that got recommended, in addition to improving accuracy in some cases. Adding the explainability constraint created a balance between accuracy and explainability and by tuning the appropriate parameters in each context, the optimized solution can be obtained.

Similar to EMF, we proposed an explanation-aware neural network using constrained Restricted Boltzmann Machines (RBM) for CF that recommends items that are explainable. Our results showed the trade-off between the accuracy and explainability when varying several parameters. Similar to EMF, it is important to choose the best set of parameters for the appropriate situation when priority is given to either the accuracy or explainability.

We proposed offline metrics to evaluate the explainability of recommender systems that can be easily used for any explainable recommender system.

We then designed a user study experiment for online evaluation of the explainability metric and explainable recommendations. When the explainability score was low, participants perceived less transparency and satisfaction from the explanation. The results showed that people who received recommendations with higher explainability perceived more transparency and showed more satisfaction compared to people whose explanations had lower explainability; therefore, the explanation by itself does not guarantee user satisfaction.

In the future, we plan to extend our technique with other explainability formulations. In addition, content data and user reviews can be considered as a prominent domain of data to generate explanation-aware recommender systems.

REFERENCES

- [1] David Gunning, “Explainable artificial intelligence (xai),” *Defense Advanced Research Projects Agency (DARPA)*, nd Web, 2017.
- [2] Xiaoyuan Su and Taghi M. Khoshgoftaar, “A survey of collaborative filtering techniques,” *Adv. in Artif. Intell.*, vol. 2009, pp. 4:2–4:2, Jan. 2009.
- [3] Daniel Billsus and Michael J. Pazzani, “Learning collaborative information filters,” in *ICML ’98: Proceedings of the Fifteenth International Conference on Machine Learning*, San Francisco, CA, USA, 1998, pp. 46–54, Morgan Kaufmann Publishers Inc.
- [4] W. Swartout, C. Paris, and J. Moore, “Explanations in knowledge systems: Design for explainable expert systems,” *IEEE Expert: Intelligent Systems and Their Applications*, vol. 6, no. 3, pp. 58–64, 1991.
- [5] Jonathan L Herlocker, Joseph A Konstan, and John Riedl, “Explaining collaborative filtering recommendations,” in *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. ACM, 2000, pp. 241–250.
- [6] Mustafa Bilgic and Raymond J Mooney, “Explaining recommendations: Satisfaction vs. promotion,” in *Beyond Personalization Workshop, IUI*, 2005, vol. 5.
- [7] Nava Tintarev and Judith Masthoff, “Designing and evaluating explanations for recommender systems,” in *Recommender Systems Handbook*, pp. 479–510. Springer, 2011.
- [8] J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
- [9] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, Belmont, CA, 1984.
- [10] Peter Clark and Tim Niblett, “The cn2 induction algorithm,” *Mach. Learn.*, vol. 3, no. 4, pp. 261–283, Mar. 1989.
- [11] William W. Cohen, “Fast effective rule induction,” in *In Proceedings of the Twelfth International Conference on Machine Learning*. 1995, pp. 115–123, Morgan Kaufmann.
- [12] Jacek M Zurada, *Introduction to artificial neural systems*, West St. Paul, 1992.
- [13] Corinna Cortes and Vladimir Vapnik, “Support-vector networks,” *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sept. 1995.
- [14] Leo Breiman, “Bagging predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [15] Y. Koren, R. Bell, and Ch. Volinsky, “Matrix factorization techniques for recommender systems,” *IEEE Computer* 42, 8, pp. 30–37, 2009.
- [16] Gábor Takács, István Pilászy, Botyán Németh, and Domonkos Tikk, “Investigation of various matrix factorization methods for large recommender systems,” in *Proceedings of the 2Nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, New York, NY, USA, 2008, NETFLIX ’08, pp. 6:1–6:8, ACM.

- [17] Yehuda Koren, Robert Bell, and Chris Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, , no. 8, pp. 30–37, 2009.
- [18] Max Kuhn and Kjell Johnson, *Applied predictive modeling*, vol. 810, Springer, 2013.
- [19] Fatih Gedikli, Dietmar Jannach, and Mouzhi Ge, “How should i explain? a comparison of different explanation types for recommender systems,” *International Journal of Human-Computer Studies*, vol. 72, no. 4, pp. 367–382, 2014.
- [20] Alexis Papadimitriou, Panagiotis Symeonidis, and Yannis Manolopoulos, “A generalized taxonomy of explanations styles for traditional and social recommender systems,” *Data Mining and Knowledge Discovery*, vol. 24, no. 3, pp. 555–583, 2012.
- [21] George Lorenzo, Diana Oblinger Oblinger, and Charles Dziuban, “How choice, co-creation, and culture are changing what it means to be net savvy,” in *EDUCAUSE Learning Initiative 4*, 2006.
- [22] Xavier Amatriain, Alejandro Jaimes, Nuria Oliver, and Josep M. Pujol, “Data mining methods for recommender systems,” in *Recommender Systems Handbook*, pp. 39–71. 2011.
- [23] Robin Burke, “Hybrid recommender systems: Survey and experiments,” *User Modeling and User-Adapted Interaction*, vol. 12, no. 4, pp. 331–370, Nov. 2002.
- [24] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, John Riedl, and High Volume, “GroupLens: Applying collaborative filtering to usenet news,” *Communications of the ACM*, vol. 40, pp. 77–87, 1997.
- [25] Loren Terveen, Will Hill, Brian Amento, David McDonald, and Josh Creter, “Phoaks: a system for sharing recommendations,” *Communications of the ACM*, vol. 40, pp. 59–62, 1997.
- [26] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin, “Performance of recommender algorithms on top-n recommendation tasks,” in *Proceedings of the Fourth ACM Conference on Recommender Systems*, New York, NY, USA, 2010, RecSys ’10, pp. 39–46, ACM.
- [27] Xiaoyuan Su and Taghi M. Khoshgoftaar, “A survey of collaborative filtering techniques,” *Adv. in Artif. Intell.*, vol. 2009, pp. 4:2–4:2, 2009.
- [28] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Reidl, “An algorithmic framework for performing collaborative filtering,” *SIGIR ’99 Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 230–237, 1999.
- [29] Matthew R. McLaughlin and Jonathan L. Herlocker, “A collaborative filtering algorithm and evaluation metric that accurately model the user experience,” in *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2004, SIGIR ’04, pp. 329–336, ACM.
- [30] 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*, New York, NY, USA, 2001, WWW ’01, pp. 285–295, ACM.

- [31] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, New York, NY, USA, 1994, CSCW '94, pp. 175–186, ACM.
- [32] Ellen Spertus, Mehran Sahami, and Orkut Buyukkokten, "Evaluating similarity measures: a large-scale study in the orkut social network," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*, 2005, pp. 678–684.
- [33] Neal Lathia, Stephen Hailes, and Licia Capra, "The effect of correlation coefficients on communities of recommenders," in *Proceedings of the 2008 ACM Symposium on Applied Computing*, New York, NY, USA, 2008, SAC '08, pp. 2000–2005, ACM.
- [34] C.C. Aggarwal, J.L. Wolf, K.L. Wu, and P.S. Yu, "Hotting hatches an egg: A new graph-theoretic approach to collaborative filtering," *Proc. of the Fifth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 99)*, pp. 201–212, 1999.
- [35] Zan Huang, Wingyan Chung, and Hsinchun Chen, "A graph model for e-commerce recommender systems," *Journal of the American Society for Information Science and Technology*, vol. 55, pp. 259–274, 2004.
- [36] Manh Cuong Pham, Yiwei Cao, Ralf Klamma, and Matthias Jarke, "A clustering approach for collaborative filtering recommendation using social network analysis," *J. UCS*, vol. 17, no. 4, pp. 583–604, 2011.
- [37] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Riedl, "Analysis of recommendation algorithms for e-commerce," in *EC*, 2000, pp. 158–167.
- [38] M. Pazzani and D. Billsus, "Learning and revising user profiles: The identification of interesting web sites," *Machine Learning*, vol. 5, pp. 313–331, 1997.
- [39] John S. Breese, David Heckerman, and Carl Kadie, "Empirical analysis of predictive algorithm for collaborative filtering," in *Proceedings of the 14 th Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 43–52.
- [40] Raymond J. Mooney, Paul N. Bennett, and Loriene Roy, "Book recommending using text categorization with extracted information," in *Proceedings, AAAI-98 Workshop on Recommender Systems*, Madison, WI, July 1998, AAAI.
- [41] Nathan Srebro, Jason D. M. Rennie, and Tommi S. Jaakola, "Maximum-margin matrix factorization," in *Advances in Neural Information Processing Systems 17*. 2005, pp. 1329–1336, MIT Press.
- [42] Jason D. M. Rennie and Nathan Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proceedings of the 22Nd International Conference on Machine Learning*, New York, NY, USA, 2005, ICML '05, pp. 713–719, ACM.
- [43] Kai Yu, Anton Schwaighofer, Volker Tresp, Xiaowei Xu, and Hans-Peter Kriegel, "Probabilistic memory-based collaborative filtering," *IEEE Trans. on Knowl. and Data Eng.*, vol. 16, no. 1, pp. 56–69, Jan. 2004.
- [44] David Pennock, Eric Horvitz, Steve Lawrence, and C Lee Giles, "Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach," in *In Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*. 2000, pp. 473–480, Morgan Kaufmann.

- [45] Nikolaos Pappas and Andrei Popescu-Belis, “Sentiment analysis of user comments for one-class collaborative filtering over ted talks,” in *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2013, SIGIR ’13, pp. 773–776, ACM.
- [46] Claudiu Cristian Musat, Yizhong Liang, and Boi Faltings, “Recommendation using textual opinions,” in *IJCAI*, Francesca Rossi, Ed. 2013, IJCAI/AAAI.
- [47] Julian McAuley and Jure Leskovec, “Hidden factors and hidden topics: Understanding rating dimensions with review text,” in *Proceedings of the 7th ACM Conference on Recommender Systems*, New York, NY, USA, 2013, RecSys ’13, pp. 165–172, ACM.
- [48] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *IEEE Trans. on Knowl. and Data Eng.*, vol. 17, no. 6, pp. 734–749, June 2005.
- [49] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades, “Facing the cold start problem in recommender systems,” *Expert Syst. Appl.*, vol. 41, no. 4, pp. 2065–2073, Mar. 2014.
- [50] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl, “Getting to know you: Learning new user preferences in recommender systems,” in *Proceedings of the 7th International Conference on Intelligent User Interfaces*, New York, NY, USA, 2002, IUI ’02, pp. 127–134, ACM.
- [51] Seung-Taek Park and Wei Chu, “Pairwise preference regression for cold-start recommendation,” in *Proceedings of the Third ACM Conference on Recommender Systems*, New York, NY, USA, 2009, RecSys ’09, pp. 21–28, ACM.
- [52] Gavin Shaw, Yue Xu, and Shlomo Geva, “Using association rules to solve the cold-start problem in recommender systems,” in *Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part I*, Berlin, Heidelberg, 2010, PAKDD’10, pp. 340–347, Springer-Verlag.
- [53] Nadav Golbandi, Yehuda Koren, and Ronny Lempel, “Adaptive bootstrapping of recommender systems using decision trees,” in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, New York, NY, USA, 2011, WSDM ’11, pp. 595–604, ACM.
- [54] Ke Zhou, Shuang-Hong Yang, and Hongyuan Zha, “Functional matrix factorizations for cold-start recommendation,” in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2011, SIGIR ’11, pp. 315–324, ACM.
- [55] K. Zhang and H. Li, “Fusion-based recommender system,” in *Proceedings of the 13th conference on information fusion (FUSION)*, 2010.
- [56] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock, “Methods and metrics for cold-start recommendations,” in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2002, SIGIR ’02, pp. 253–260, ACM.
- [57] Yifan Hu, Yehuda Koren, and Chris Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on*. Ieee, 2008, pp. 263–272.
- [58] Simon Funk, “Netflix update: Try this at home,” 2006.

- [59] Geoffrey Hinton, “A practical guide to training restricted boltzmann machines,” *Momentum*, vol. 9, no. 1, pp. 926, 2010.
- [60] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton, “Restricted boltzmann machines for collaborative filtering,” in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 791–798.
- [61] Pedro Antunes, Valeria Herskovic, Sergio F Ochoa, and Jose A Pino, “Structuring dimensions for collaborative systems evaluation,” *ACM Computing Surveys (CSUR)*, vol. 44, no. 2, pp. 8, 2012.
- [62] Finale Doshi-Velez and Been Kim, “Towards a rigorous science of interpretable machine learning,” 2017.
- [63] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, “Why should i trust you?: Explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1135–1144.
- [64] Alex A. Freitas, “Comprehensible classification models: A position paper,” *SIGKDD Explor. Newsl.*, vol. 15, no. 1, pp. 1–10, Mar. 2014.
- [65] J. L. Herlocker, J. A. Konstan, and J. Reidl, “Explaining collaborative filtering recommendations,” *CSCW ’00 Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pp. 241–250, 2000.
- [66] Nava Tintarev and Judith Masthoff, “Designing and evaluating explanations for recommender systems,” in *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, Eds., pp. 479–510. Springer, 2011.
- [67] Amit Sharma and Dan Cosley, “Do social explanations work?: Studying and modeling the effects of social explanations in recommender systems,” in *Proceedings of the 22Nd International Conference on World Wide Web*, Republic and Canton of Geneva, Switzerland, 2013, WWW ’13, pp. 1133–1144, International World Wide Web Conferences Steering Committee.
- [68] J. Vig, Sh. Sen, and J. Riedl, “Tagsplanations: Explaining recommendations using tags,” *Proceedings of the 14th International Conference on Intelligent User Interfaces*, pp. 47–56, 2009.
- [69] M. Bilgic, “Explaining recommendations: Satisfaction vs. promotion,” in *In Proceedings of Beyond Personalization 2005, the Workshop on the Next Stage of Recommender Systems Research(IUI2005)*, 2005, pp. 13–18.
- [70] Nava Tintarev and Judith Masthoff, “A survey of explanations in recommender systems,” in *Proceedings of the 23rd International Conference on Data Engineering Workshops, ICDE 2007, 15-20 April 2007, Istanbul, Turkey*, 2007, pp. 801–810.
- [71] Pearl Pu, Li Chen, and Rong Hu, “A user-centric evaluation framework for recommender systems,” in *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, 2011, pp. 157–164.
- [72] Rashmi Sinha and Kirsten Swearingen, “The role of transparency in recommender systems,” in *CHI ’02 Extended Abstracts on Human Factors in Computing Systems*, New York, NY, USA, 2002, CHI EA ’02, pp. 830–831, ACM.

- [73] Kirsten Swearing and Rashmi Sinha, “Interaction design for recommender systems,” in *Designing Interactive Systems*, 2002.
- [74] K. McCarthy, J. Reilly, L. McGinty, and B. Smyth, “Thinking positively - explanatory feedback for conversational recommender systems,” Tech. Rep., In Proceedings of the ECCBR 2004 Workshops, 2004.
- [75] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma, “Explicit factor models for explainable recommendation based on phrase-level sentiment analysis,” in *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, New York, NY, USA, 2014, SIGIR ’14, pp. 83–92, ACM.
- [76] Nava Tintarev and Judith Masthoff, “Evaluating the effectiveness of explanations for recommender systems,” *User Modeling and User-Adapted Interaction*, vol. 22, no. 4-5, pp. 399–439, Oct. 2012.
- [77] L. Ardissonoa, A. Goya, G. Petronea, M. Segnana, and P. Torassoa l, “Intrigue: Personalized recommendation of tourist attractions for desktop and hand held devices,” *Applied Artificial Intelligence*, pp. 687–714, 2003.
- [78] Daniel Billsus and Michael J. Pazzani, “A personal news agent that talks, learns and explains,” in *Proceedings of the Third Annual Conference on Autonomous Agents*, New York, NY, USA, 1999, AGENTS ’99, pp. 268–275, ACM.
- [79] P. Symeonidis, R. Nanopoulos, and Y. Manolopoulos, “Justified recommendations based on content and rating data,” *WebKDD Workshop on Web Mining and Web Usage Analysis*, 2008.
- [80] Behnoush Abdollahi and Olfa Nasraoui, “A cross-modal warm-up solution for the cold-start problem in collaborative filtering recommender systems,” in *Proceedings of the 2014 ACM conference on Web science*. ACM, 2014, pp. 257–258.
- [81] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos, “Movixplain: a recommender system with explanations,” in *Proceedings of the third ACM conference on Recommender systems*. ACM, 2009, pp. 317–320.
- [82] Behnoush Abdollahi and Olfa Nasraoui, “Explainable matrix factorization for collaborative filtering,” in *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 5–6.
- [83] Behnoush Abdollahi and Olfa Nasraoui, “Using explainability for constrained matrix factorization,” in *Proceedings of the 2017 ACM Conference on Recommender Systems*. 2017, RecSys ’17, ACM.
- [84] 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*. ACM, 2008, pp. 426–434.
- [85] Arkadiusz Paterek, “Improving regularized singular value decomposition for collaborative filtering,” in *Proceedings of KDD cup and workshop*, 2007, vol. 2007, pp. 5–8.
- [86] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk, “Major components of the gravity recommendation system,” *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 80–83, 2007.

- [87] Behnoush Abdollahi and Olfa Nasraoui, “Explainable restricted boltzmann machines for collaborative filtering,” *arXiv preprint arXiv:1606.07129*, 2016.
- [88] Geoffrey E Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.
- [89] F. Maxwell Harper and Joseph A. Konstan, “The movielens datasets: History and context,” *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 19:1–19:19, Dec. 2015.
- [90] Prem Melville, Raymond J Mooney, and Ramadass Nagarajan, “Content-boosted collaborative filtering for improved recommendations,” in *Aaai/iaai*, 2002, pp. 187–192.
- [91] Ruslan Salakhutdinov and Andriy Mnih, “Probabilistic matrix factorization,” Citeseer, 2011.
- [92] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl, “An algorithmic framework for performing collaborative filtering,” in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 230–237.
- [93] 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*. ACM, 2001, pp. 285–295.
- [94] Nava Tintarev and Judith Masthoff, “A survey of explanations in recommender systems,” in *Data Engineering Workshop, 2007 IEEE 23rd International Conference on*. IEEE, 2007, pp. 801–810.
- [95] Jacob Cohen, “Statistical power analysis for the behavioral sciences . hilsdale,” *NJ: Lawrence Earlbaum Associates*, vol. 2, 1988.
- [96] Mohsen Tavakol and Reg Dennick, “Making sense of cronbach’s alpha,” *International journal of medical education*, vol. 2, pp. 53, 2011.
- [97] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl, “Evaluating collaborative filtering recommender systems,” *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 5–53, 2004.

CURRICULUM VITAE

NAME: Behnoush Abdollahi

ADDRESS: Computer Engineering & Computer Science Department
Speed School of Engineering
University of Louisville
Louisville, KY 40292

EDUCATION:

Ph.D., Computer Science & Engineering, August 2017

University of Louisville, *Louisville, Kentucky*

M.Sc., Computer Science & Engineering, August 2011

University of Louisville, *Louisville, Kentucky*

B.Sc., Computer Engineering, February 2009

University of Tehran, *Tehran, Iran*

WORK EXPERIENCE:

Research Assistant, University of Louisville

Teaching Assistant, University of Louisville

Machine Learning Engineer, Gen Nine Inc.

PUBLICATIONS:

1. **B. Abdollahi** and O. Nasraoui, “*Transparency and Explainability in Latent Factor Models*”, 2017 (under submission).

2. **B. Abdollahi**, O. Nasraoui, and S. Sanders, “*Study of Explainability on User Satisfaction in Collaborative Filtering*”, 2017 (under submission).
3. **B. Abdollahi** and O. Nasraoui, “*Using Explainability for Constraint Matrix Factorization*”, ACM Conference on Recommender Systems (ACM RecSys), 2017, (acceptance rate 16%).
4. **B. Abdollahi** and O. Nasraoui, “*Explainable Restricted Boltzmann Machines for Collaborative Filtering*”, ICML Workshop on Human Interpretability, 2016.
5. **B. Abdollahi** and O. Nasraoui, “*Explainable Matrix Factorization for Collaborative Filtering*”, World Wide Web (WWW), 2016, (acceptance rate 16%).
6. **B. Abdollahi** and O. Nasraoui, “*A cross-modal warm-up solution for the cold-start problem in collaborative filtering recommender systems*”, Proceedings of the 2014 ACM conference on Web science, ACM SIGWEB WebSci, 2014, (acceptance rate 20%).
7. **B. Abdollahi**, M. Badami, G. C. Nutakki, W. Sun, O. Nasraoui, “*A Two Step Ranking Solution for Twitter User Engagement*”, Proceedings of the 2014 Recommender Systems Challenge, ACM RecSys, 2014.
8. G. C. Nutakki, O. Nasraoui, **B. Abdollahi**, M. Badami, W. Sun, “*Distributed LDA based Topic Modeling and Topic Agglomeration in a Latent Space*”, 23rd International World Wide Web Conference, Social News on the Web, (SNOW-DC @ WWW), 2014.
9. **B. Abdollahi**, A Civelek, XF Li, J Suri, A El-Baz, “*PET/CT nodule segmentation and diagnosis: A survey*”, Multi Detector CT Imaging, 2014.
10. A. El-Baz, G. M. Beache, G. Gimel’farb, K. Suzuki, K. Okada, A. Elnakib, A. Soliman, **B. Abdollahi**, “*Computer-aided diagnosis systems for lung cancer: challenges and methodologies*”, International journal of biomedical imaging, 2013.
11. **B. Abdollahi**, A Soliman, AC Civelek, XF Li, G Gimel’Farb, A El-Baz, “*A Novel 3D Scale Space-Based Joint MGRF Framework for Precise Lung Segmentation*”, 19th

IEEE International Conference on Image Processing (IEEE-ICIP), 2012.

12. **B. Abdollahi**, A Soliman, AC Civelek, XF Li, G Gimel'Farb, A El-Baz, "*A novel 3D joint MGRF framework for precise lung segmentation*", International Workshop on Machine Learning in Medical Imaging, 2012.
13. **B. Abdollahi**, "*Stereoscopic vision in vehicle navigation*", 2011.
14. **B. Abdollahi**, "*Stereoscopic vision in augmented reality and vehicle driving*", Kentucky Academy of Science, 2010.

HONORS AND AWARDS:

1. CSE Doctoral Award, UofL, 2017
2. Dissertation Completion Award, UofL, 2017
3. Outstanding Graduate Student Award, UofL, 2015
4. ACM Student Travel Grant, ACM Web Science, 2014
5. Broadening Participation in Data Mining (BPDm) Scholarship, KDD 2013
6. Grace Hopper Celebration Scholarship, 2011-2013
7. Grosscurth Doctoral Fellowship, UofL, 2011