Cross Domain Recommender Systems Using Matrix and Tensor Factorizations

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Vahid Pourheidari

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Buildin, 110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan S7N 5C9
Canada

OR

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9
Canada

Abstract

Today, the amount and importance of available data on the internet are growing exponentially. These digital data has become a primary source of information and the peoples life bonded to them tightly. The data comes in diverse shapes and from various resources and users utilize them in almost all their personal or social activities. However, selecting a desirable option from the huge list of available options can be really frustrating and time-consuming. Recommender systems aim to ease this process by finding the proper items which are more likely to be interested by users. Undoubtedly, there is not even one social media or online service which can continue its work properly without using recommender systems. On the other hand, almost all available recommendation techniques suffer from some common issues: the data sparsity, the cold-start, and the new-user problems.

This thesis tackles the mentioned problems using different methods. While, most of the recommender methods rely on using single domain information, in this thesis, the main focus is on using multi-domain information to create cross-domain recommender systems. A cross-domain recommender system is not only able to handle the cold-start and new-user situations much better, but it also helps to incorporate different features exposed in diverse domains together and capture a better understanding of the users preferences which means producing more accurate recommendations.

In this thesis, a pre-clustering stage is proposed to reduce the data sparsity as well. Various cross-domain knowledge-based recommender systems are suggested to recommend items in two popular social media, the Twitter and LinkedIn, by using different information available in both domains. The state of art techniques in this field, namely matrix factorization and tensor decomposition, are implemented to develop cross-domain recommender systems. The presented recommender systems based on the coupled nonnegative matrix factorization and PARAFAC-style tensor decomposition are evaluated using real-world datasets and it is shown that they superior to the baseline matrix factorization collaborative filtering. In addition, network analysis is performed on the extracted data from Twitter and LinkedIn.

Acknowledgements

First, I would like to thank my supervisor, Prof. Ralph Deters, for his trust in me and for his great advice and guidance during my M.Sc. program. His support and mentorship have guided me through all my research steps. And Also, special thanks to Prof. Julita Vassileva, Prof. Chanchal Roy, and Prof. Li Chen for their invaluable comments on my research.

I am really thankful to my amazing wife, Sara Rouhani, for both her emotional and academical support as well.

Contents

List of Tables

List of Figures

List of Equations

List of Abbreviations

ALS	Alternating Least Squares
CP	CANDECOMP/PARAFAC
HOSVD	Higher-Order Singular Value Decomposition
KNN	K-Nearest Neighbor
LBSN	Location-Based Social Network
MAE	Mean Absolute Error
MSE	Mean Squared Error
NMF	Nonnegative Matrix Factorization
NN	Neural Network
POI	Point of Interest
RMSE	Root Mean Squared Error
SGD	Stochastic Gradient Descent
STS	Social Tagging System
SVD	Singular Value Decomposition
TD	Tensor Decomposition
VSM	Vector Space Model

1

INTRODUCTION

1.1 Motivation

With the extension of internet access during the last years, the number of online services and social media websites, as well as the number of their active users, have increased exponentially. The available data are not restricted anymore to those provided by broadcast websites and other service providers, but now, users create and share most of the online digital contents. People utilize diverse resources in the World Wide Web for either their personal activities, such as finding a movie to watch, searching for interesting books to read, or music to listen, following news trend, finding a place for dine, and so on, or other activities with the social aspects, like expanding their professional network, being in contact with their friends, even being familiar with new people and finding a partner. More importantly, people introduce themselves and share their opinions about diverse subjects in many different ways; by creating profiles on different social media, sharing an image, using a hashtag, writing articles on their personal blogs, or by leaving reviews for different products or services. In addition, users are not the only parties who use these data. Exploiting the diverse data and capturing the useful insights have become a necessary skill for businesses to keep themselves productive in this modern, fast pace, and competitive marketplace. Almost it is impossible to find a successful company or service provider which have not considered its users' data as a valuable asset.

Although, this huge amount of data can be utilized to satisfy diverse users' taste, however, finding the right product or service which is suitable for a particular user can be an overwhelmingly hard process. It is when the recommender systems come into play. Recommender systems are programs that aim to predict those items which are more likely to be suitable for a unique user. Users use the recommender systems to find their favorable

options within the large list of available options and online service providers utilize them to keep their users engaged. That is why the examples of recommender systems can be found in almost any online service provider website, from search engines to social media.

Recommender systems mainly rely on using the data, exposed by users in the past, to recommend new interesting items. Based on their implementation, they try to capture the similarities between the users, or the items, or both, in order to understand the users' preferences. However, almost all of the available recommendation techniques face into some important obstacles. The first one is the sparse nature of available data on the internet or even a specific domain. A traditional instance is an online retailer website where only a low percentage of the cells in the user-item matrix are filled and others are unknown. Another problem is when the recommender system does not have a sufficient amount of information about a user (or users) to understand the explicit preferences and the latent relations. In this case, the recommender system cannot produce useful and accurate recommendations. It is called the cold-start problem. The new-user problem is a special case of the cold-start problem when the user is totally new to the system or does not have any previous rating history.

Users share different types of information in diverse domains. According to this fact, recently many studies have tried to integrate that information in order to address the mentioned issues. The sparsity of data can be reduced by sharing the knowledge from one domain to another or by merging different features from diverse domains. Another promising point is that the data integration can potentially increase the accuracy of recommender systems in the cold-start situation. If a user is new to a domain, a cross-domain recommender system can use other previously shared information related to this user or subject to create a precise recommendation for that particular user in the new domain. Although, using the cross-domain recommender system can substantially answer many of the traditional techniques' problem, however, this field is relatively new.

This thesis focuses on developing cross-domain recommender systems to tackle the data sparsity, the cold-start problem, the new-user problem, and also to increase the accuracy of the final recommended items. To investigate the effectiveness of this idea, different algorithms are designed and implemented, including knowledge-based collaborative, content-based, hy-

brid mix, and hybrid recommender systems, a coupled nonnegative matrix factorization-based, and a tensor decomposition-based recommender system. In addition, the performances of these cross-domain recommender systems are evaluated using real datasets, extracted from two popular social media.

1.2 Structure of Document

As explained in the previous section, in this thesis we propose a number of cross-domain recommender systems, comprising four different knowledge-based algorithms, a novel coupled matrix factorization model, and a multilinear tensor decomposition model, which result to more accurate recommendations in the target domain. The needed preliminaries to understand these models as well as detailed information about the modeling, implementation, and evaluation processes are organized in the next chapters as follow:

- **Chapter 2: Problem Statement** In this chapter the main goals of the thesis are presented. In addition, the implemented techniques and final findings of this research study are briefly explained.

- **Chapter 3: Background and Related Works** A comprehensive study about the recommender systems, their classification, and their various techniques are conducted in this chapter. Also, the required mathematical preliminaries of the matrix and tensor factorizations are presented here. In addition, it covers many recently published studies about the application of these techniques in the recommender systems field.

- **Chapter 4: Research Methodologies and Implementations** It includes detailed explanations about the proposed recommender systems. It is divided into four sections. In its first section, the characteristics of the extracted real-world dataset and its diverse domains are explained. The second section introduces four different knowledge-based cross-domain recommender systems. The coupled matrix factorization-based model is presented in the third section. And the final section comprises the detail of the suggested tensor decomposition-based recommender system.

- **Chapter 5: Results and Discussion** This chapter starts with the network analysis of the different domains in our dataset. It then covers the results of all the proposed recommendation models and the related discussions.

- **Chapter 6: Conclusion and Future Directions** The main research questions are reviewed in this chapter and all the significant findings of the thesis are explained again. In addition, the possible future directions for further studies are noted at the last part of this chapter.

2

PROBLEM STATEMENT

During the last years, there has been an increasing popularization of social media services. Nowadays, there is at least on specific-application social media for any particular purpose: from sharing images and videos to create a job-related network. All the available social media and all new start-ups in this area try to invent innovative solutions to attract new users and keep the current users engaged. At the core of all these techniques and solutions, there are recommender systems. They try to create a convenient experience for users by helping them to find their desired products, information, and services. These recommender systems usually use the provided data in a unique domain to make recommendations for that specific domain. However, recently researchers have found that using cross-domain recommender systems, which consider available data in multiple domains (Figure 2.1), can increase the accuracy of the recommended items and also reduces some serious problem such as the data sparsity and the cold-start. With this hypothesis in mind, we state the following specific research goals.

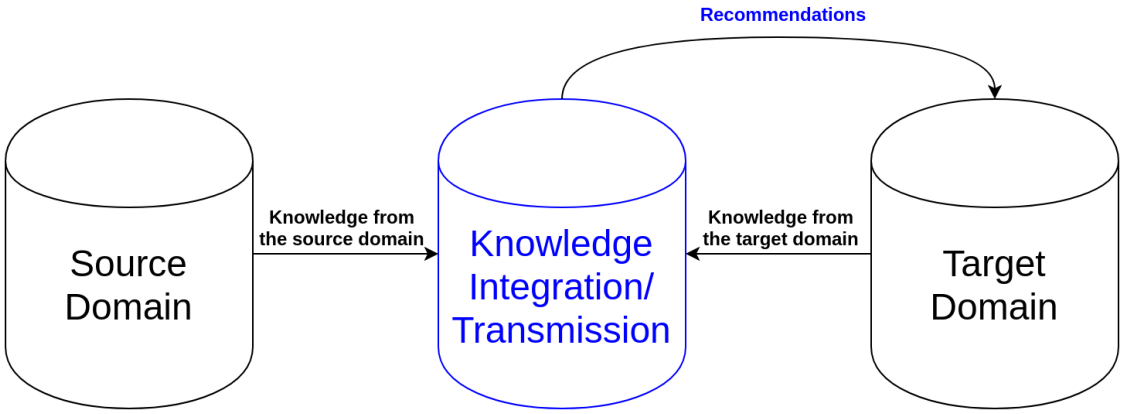

Figure 2.1: Cross-domain recommender system

- **1. Review the importance of recommender systems, their classification, and the reasons users tend to utilize them.** As we are going to produce novel algorithms and propose recommendation models using them, it is necessary to have a solid understanding of the recommender systems' foundations. In order to improve the current state of the art methods on recommender systems, we need to have a good perspective about the available various techniques in this field, how they work, and how they help users.

- **2. Review the cross-domain recommendation concept.** Since using the cross-domain recommender systems is relatively new in comparison to the traditional techniques, there are not still comprehensive definitions of different domains and knowledge-flow between them. We should fill this gap by summarizing the most important definitions of diverse domains and information integration methods, before trying to develop new cross-domain recommender systems.

- **3. Using a pre-clustering stage to reduce the inherent sparse nature of the data in various domains.** One of the common issues of recommender systems is their inability to work well with the sparse data. On the other hand, one of the characteristics of social media data is their sparsity. In order to reduce the negative impact of this conflict, we will propose a pre-clustering step which can properly classify the categorical data in order to reduce the percentage of sparsity in the dataset.

- **4. Develop various knowledge-based cross-domain recommendation models that exploit the users' information in the source domain, or both the source and target domains, in order to suggest items in the target domain.** The conducted studies in the two first questions would help us to understand the type of data we need for this research study. We shall extract this data from diverse social media which represent different features of identical users. Then, we will try to develop some cross-domain recommender systems which rely on the integration of various available data.

- **5. Review the matrix factorization and tensor decomposition techniques**

and their applications on recommender systems. These techniques are literally the state of art on cross-domain recommender system which attracts the researchers' attention in the last few years. We shall start with their required preliminaries and then summarize their recent application in developing recommender systems.

- **6. Develop novel coupled matrix factorization-based and tensor decomposition-based cross-domain recommender systems.** While traditional single-domain recommender systems consider user preferences in on domain, the coupled matrix factorization are able to take different information into account in order to understand the various features of users' preferences. In addition, the tensor decomposition method can be applied to uncover the implicit relationships between users, different items, and there features.

- **7. Analyze the performance of the proposed cross-domain recommender systems.** According to the nature of the recommender systems' job, that is suggesting a list of items which is more likely to be liked by the users, we will evaluate the effectiveness of the proposed recommender systems in term of their precision. The accuracies of these models are compared to each other as well as the baseline collaborative filtering recommender system.

Following the mentioned goals, in this thesis, we utilize different techniques in order to develop and analyze the following cross-domain recommendation methods:

- A knowledge-based collaborative filtering,
- A knowledge-based content-based,
- A knowledge-based hybrid mix,
- A knowledge-based hybrid feature combination,
- A coupled nonnegative matrix factorization-based,
- A tensor decomposition-based.

3

BACKGROUND AND RELATED WORKS

3.1 Recommendation Systems

Due to the exponential growth of available data on the internet and social media as well as the number of active users, recommender systems have become a must-have part for almost all online service providers. Recommender systems provide a convenient experience for the users by suggesting them desired favorable products or services, such as a place to visit or dine, a book or a scientific paper to read, a movie to watch, an education program to attend and so on. Otherwise, without recommender systems, finding a proper option among the huge set of available options is a frustrating and time-consuming process and make users leave that social media. In this section, we will review the fundamental of recommender systems and their various recommending techniques. As the focus of this research study is on developing cross-domain recommender systems, then we will also review the multi-domains recommending approach as well.

3.1.1 Introduction to recommender systems

Recommender systems are programs that aim to provide proper suggestions for products or services that are most likely of interest of particular individuals or businesses [1]. Those products or services are generally called items and the individuals or businesses searching for desirable items are called users [2].

There are some cases that the necessity of recommender systems is more tangible. First and maybe the most important case is when we are dealing with an information overload system [3]. In informal language, the term of information overload simply means having or

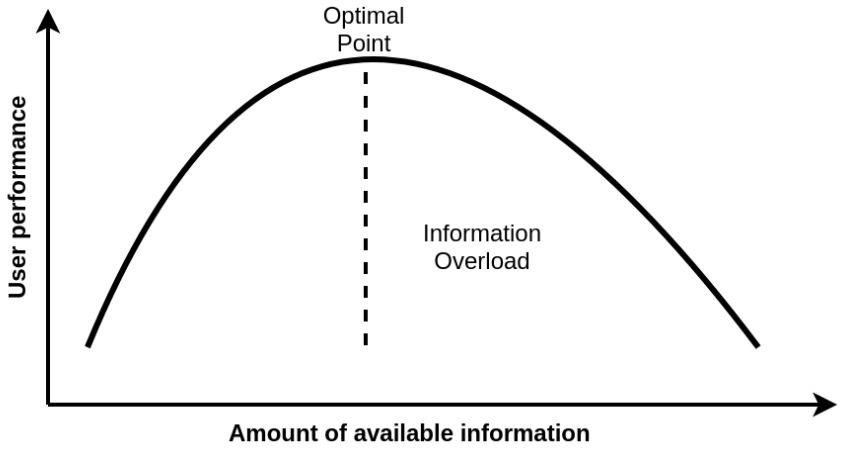

Figure 3.1: Relationship between the amount of available information and the user performance in term of the quality of decisions

receiving lots of information which is hard to understand as a whole. Within the research and scientific community, this term conveys a similar meaning and is applied to various fields such as cognitive overload, communication overload, and knowledge overload systems [4]. Many studies have shown that the performance of users varies with the amount of information they are exposed to. As a general rule, the quality of decisions and choices of users increases with increasing the amount of available information up to a certain point. However, after this threshold, further information cause a sharp drop in the performance of users [5]. Figure 3.1 shows a schematic illustration of this concept.

For example, today, the amount of available information on the internet is growing exponentially. This explosive rise in the accessible data may cause confusion for users and leads to frustrating experience since the overwhelming number of various choices makes finding the items of interest too difficult and even impossible. Recommender systems can effectively tackle this problem by helping users avoid confusion and frustration caused by searching a desirable option in a large list of available options. One common example of an information overload system is social media websites where millions or billions of users, forming online societies, are able to produce and share information and make new connections. As the magnitude of the information of these social media is vast, users usually are not able to find data and social connections they are looking for without the help of recommender systems. For instance, Facebook recommends new connections based on mutual friends, people who

are tagged in the same photo, or people of the group a user has been joined. Without these suggestions, even finding a friend by searching between all Facebook users seems to be tough.

Another case which properly represents the importance of recommender systems is when a user does not have enough experience or qualification to evaluate all items in order to select an item which works best for him or her. The shopping experience on an online retailer website is a good example of this specific case [6].

Recommender systems are used to produce suggestions for different items in various domains. Recommending movies and TV serials [7], music [8], books [9], places [10], services [11] and foods [12] are only some examples of recommendation systems' use cases. Bobadilla et al. [1] enumerate many other instances which recommender systems play a crucial role to create a pleasant experience for the users and improve the performance of different service providers. Also, Lu et al. [13] provide a comprehensive overview of recommender systems' applications in their study. In general, without recommender systems, the users of all social sites suffer from information overload. Therefore, recommender systems are important parts of all highly-rated internet sites and social applications such as Twitter [1], LinkedIn [2], Facebook [3], Netflix [4], Spotify [5], Last.fm [6], YouTube [7], and Amazon [8]. These social media are implementing different techniques to create more effective suggestions to gain benefits by increasing the users' satisfaction, increasing users' loyalty, increasing users' engagement, capturing more human-related data, improving their understanding of what user wants, and finally, increasing the number of sold products or services.

On the other hand, based on the study of Herlocker et al. [14], the usefulness of recommender systems to facilitate many tasks motivates users to utilize these tools. Here, some important usages of recommender systems for users are covered. Users contribute with recommender systems mainly for:

- *Finding good items:* Obviously, users are using recommender systems to find favorable

[1] An online news and social networking service, https://twitter.com/
[2] An online social media with professional networking focus, https://www.linkedin.com/
[3] An online social media and social networking service, https://www.facebook.com
[4] An online streaming of a library of films and television programs, https://www.netflix.com
[5] an online music streaming platform, https://www.spotify.com
[6] An online music service, https://www.last.fm
[7] A video sharing website, https://www.youtube.com
[8] An online e-commerce marketplace, www.amazon.com

items among all alternative items. Especially, when the number of related items is very large or the user does not have enough expertise to distinguish items based on his or her requirements.

- *Expressing themselves:* There are some users who are more interested in contribute with ratings and reviews and express their opinions rather than buying a commodity or service. Recommender systems can raise the level of satisfaction of these users and also increase their fidelity.

- *Helping other users:* Many users believe that their contribution in the recommender systems' information profits all the community. Therefore, this can motivate them to engage with the recommender system. For example, with an online shopping site, a user who has already purchased and used an item is aware that her or his entered review and rating is useful to other users; this contribution does not have any benefit for the user, but, he or she tends to share information in order to improve the recommender system efficiency and help other users.

- *Influencing other users:* There are some users who utilize recommender systems in order to make it bias toward some particular items and influence other users into selecting those products or services.

As mentioned so far, recommender systems are essential parts of nowadays social media and they are implemented to provide a list of suggested items to users. However, since the context where recommender systems can be utilized is highly diverse, for instance, the criteria for recommending a movie and the criteria for recommending a scientific paper are entirely different; therefore, depending on the context, researchers introduced different types of recommender systems. The main difference of recommender systems is characterized by the filtering algorithm. The most common classification divides recommender systems into the content-based, collaborative, demographic, and hybrid algorithms [15]. This classification is shown in Figure 3.2. The mentioned filtering approached are explained in the following section.

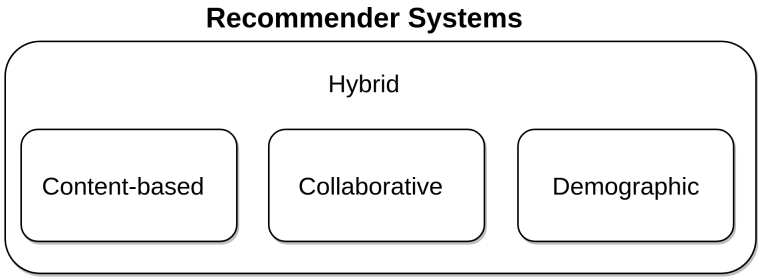

Figure 3.2: Recommender systems classification

3.1.2 Recommender systems techniques

According to the wide usage of recommender systems and the growing number of researches in this area, many diverse recommendation techniques have been suggested. The various recommendation approaches can be categorized based on various criteria, for example, from the algorithmic point of view, different available techniques can be classified into memory-based and model-based methods. Memory-based approaches mainly use some heuristics to compare items and calculate the relevance of the items for a particular user. These methods usually work with the original rating matrix of users-items or use another rating generated before the referral process. Therefore, a memory-based method can be implemented easily because it has a simple logic and follows a particular goal only. On the other hand, it is less flexible in comparison to a model-based approach. In contrast to the memory-based techniques, model-based methods use more complicated algorithms, such as different machine learning tools, to build a prediction model from available data. They use recommender system information to produce a predictor model that generates items suggestion for a particular user. Popular memory-based and model-based methods are mentioned in the Figure 3.3.

There are other possible classifications and as mentioned before the most widely used classification divides all the proposed techniques into the four categories (Figure 3.2):

1. *Content-based recommender systems:* analyze the properties of items and suggest a group of items to a particular user which are most similar to the past selected items by that user.

2. *Collaborative filtering-based recommender systems:* investigate the rating similarities between users and suggest a group of items to a user which are already selected by

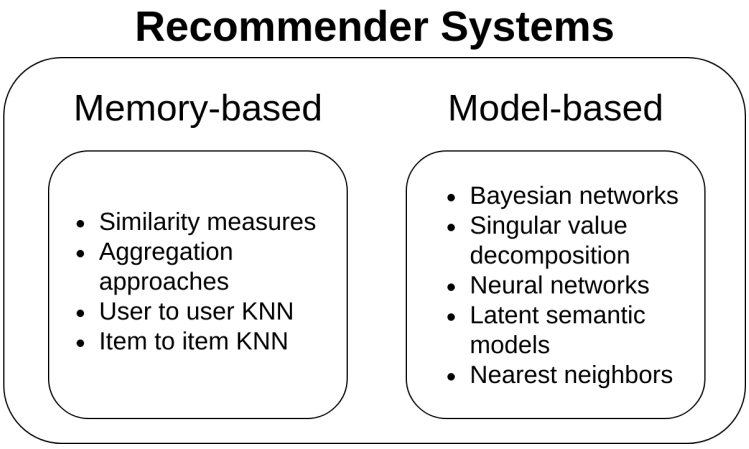

Figure 3.3: Classification of recommender systems from algorithmic point of view

most similar users to that unique user.

3. *Demographic recommender systems:* rely on this hypothesis that users with the same personal characteristics, such as gender, age, nationality, level of education, etc., have the same taste of items.

4. *Hybrid recommender systems:* are combinations of other recommender systems and usually are seen as a combination of collaborative with demographic method [16] or as a combination of collaborative with content-based approach [17, 18].

A content-based recommender system tries to understand the commonalities among the items a user has rated in past, then suggest the items that share the same features [19]. The schematic process of a content-based recommender system is displayed in Figure . However, since these types of recommender systems relay on keywords that describe the content, they are most efficient for text-based content where keywords can be parsed dynamically. Otherwise, it can be highly inefficient to associate those keywords manually.

In order to understand the commonalities among the items, many content-based techniques rely on the popular Vector Space Model (VSM). In VSM we assume each item is a vector of its features $\vec{t_i} = (w_1, w_2, ..., w_n) \in \mathbb{R}^n$, So the i-th item has n various features and the value of w_j, $0 < j < n + 1$, reflects the impact of the j-th feature of the i-th item. Also, another commonly used approach to create vectors of items is the TF-IDF method [20]. When the items vectors are created, the profile of a particular user is defined by aggregating

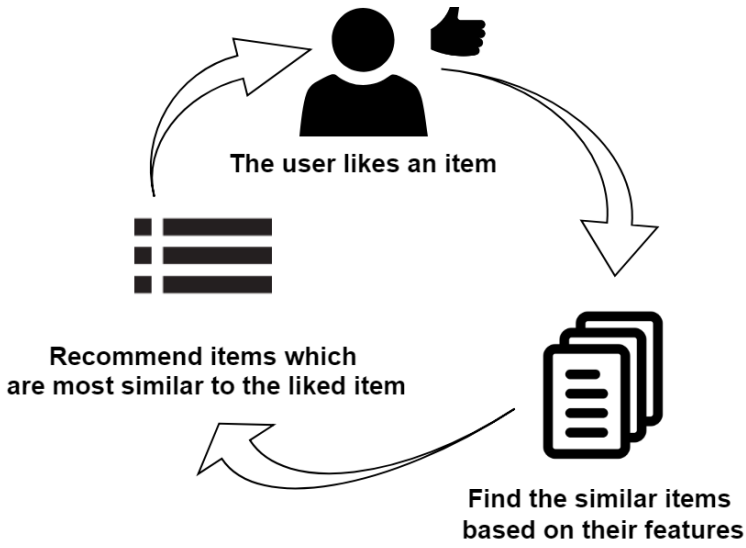

Figure 3.4: A simple representation of content-based recommender system

the models of those items he or she liked before. Finally, the utility of item i for user m can be described by using a similarity measure, such as Cosine or Pearson correlation. The popular Cosine similarity is defined as:

$$cos(\vec{u}_m, \vec{t}_i) = \frac{< \vec{u}_m, \vec{t}_i >}{\|\vec{u}_m\| \cdot \|\vec{t}_i\|} \tag{3.1}$$

where, $\vec{u}_m$ indicates the profile of the user m.

Collaborative filtering-based recommender systems are based on the heuristic that users, who are similar, make choices which are similar (Figure 3.5). Therefore, these methods recommend items for a particular user based on the items previously rated highly by other similar users [21]. The similar users are found based on their histories of rating the same items that the user has seen and rated in the past; by correlating these past rating. One advantage of collaborative filtering is that it just uses rating patterns, so no content information is needed. Different learning methods can be used to create a collaborative filtering-based model, such as k-nearest neighbor (KNN)[22, 23], neural network (NN) [24], singular vector decomposition (SVD) [25], genetic algorithm [26], and latent semantic indexing (LSI) [27] approaches.

Although content-based and collaborative filtering recommender systems have their own advantages and use cases, however, both suffer from a problem known as the cold-start

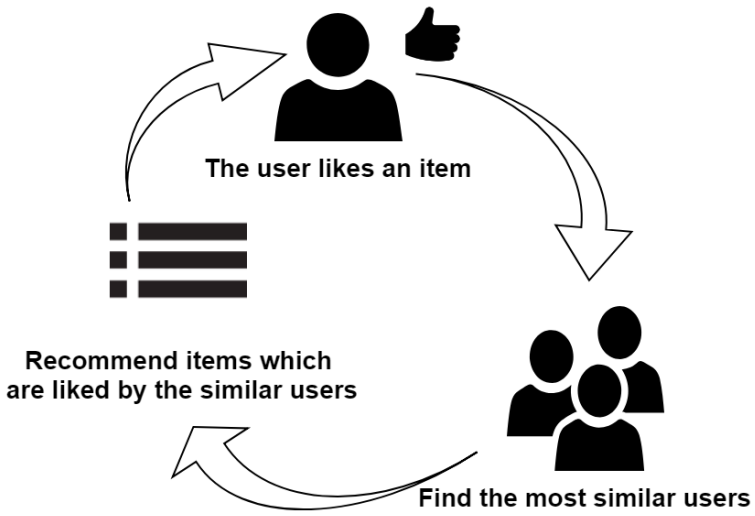

Figure 3.5: A simple representation of collaborative filtering recommender system

problem. Cold-start happens because users have to rate (or buy, view, etc depending on the context) a sufficient number of items before a recommender system can really understand users' preferences and generate reliable recommendations.

Hybrid recommenders seek to tackle the aforementioned issues about content-based and collaborative recommender systems and increase the effectiveness of the predictor model by combining both methods in different ways, such as:

- *Mixed:* produced recommendation from content-based and collaborative recommender systems are combined together and the top frequent items are suggested as the final recommendations.

- *Feature combination:* has some features of the content-based method and some features of the collaborative approach. The output of one of these recommender systems is used as an additional feature for another one.

- *Switching:* based on predefined conditions or context the system chooses among content-based or collaborative methods and applies the selected one.

It should be noted that even though hybrid recommender systems proved to be more accurate in many cases, they do not fully address the cold-start problem. In their study,

Table 3.1: Comparison between different recommender systems' techniques

Recommender systems techniques	**Main feature**	**Main drawback(s)**
Content_based	Suggests new items which are most similar to the past selected items by a particular user	Incapable to find items' quality based on the users' pereference
Collaborative filtering	Recommends new items which are already selected by most similar users to an unique user	Suffers from cold-start problem and possible incorrect users' rating
Demographic	Classifies the users based on their personal characteristics	Suffers from cold-start problem and incorrect personal categorization
Hybrid	combines other recommender systems, especially content-based and collaborative	An unwise combination of recommender systems can reduce the accuracy

Urmela et al. [28] compare various recommender systems techniques and also mention the most import drawback of each approaches. Here, we summarize their findings is Table 3.1.

Many present problems of most of the recommender systems come from the sparsity of data in one domain and also inability or weakness of different algorithms to handle the cold-start or new user situation. One possible solution that in this study is considered is using data integration between two or more various domains in order to develop cross-domain recommender system. It not only can reduce the sensitivity of the model to the cold-start problem, but also, it can increase the accuracy of the designed model by decreasing the sparsity of data. This matter is discussed in the next section.

3.1.3 Cross-domain recommender systems

Recommender systems have been used successfully in almost all social media to suggest desirable items to the users. Each social media website tries its best to develop more efficient algorithms in order to create more accurate recommender systems that accept the data pro-

vided by the users of that particular domain and return a list of suggested items. Likewise, most of the conducted research studies have focused on the single domain recommender systems which use a single domain's information to predict those items that are most likely to be interested of the users of that domain. On the other hand, there is another case, that a recommender system uses information of two or even more different domains to recommend items on one of those. This distinctive approach, which relies on multi-domain data integration to suggest more precise recommendations, is called cross-domain recommender system.

Researchers may consider different definitions for two distinct domains. Generally, we can consider four levels to describe the difference between two separate domains. Based on this classification, two domains may be considered as two distinct domains if they comprise one of the following cases:

1. *Value level: same system, same type, same attributes, different values.* Considering items in a unique system having the same type and same set of attributes, but, various values. For instance, two books in the same system can be considered to be in two distinct domains if they have different genres. Cao et al. [29] suppose various book categories as different domains in their study.

2. *Attribute level: same system, same type, different attributes.* Assuming items in a particular system which have the same type and maybe share some common attributes but they have some different attributes as well. For example, books and magazines have the identical type and many similar attributes, but they still have some specific attributes which cannot find in another one. An actual example is the Winoto and Tang research study [30] which movies and TV shows are considered as separate domains.

3. *Type level: same system, different types.* Happens when the items in a specific system have various types. For instance, Shapira et al. [31] assume movies and music in two distinct domains.

4. *System level: different systems.* Items which belong to two separate systems, such

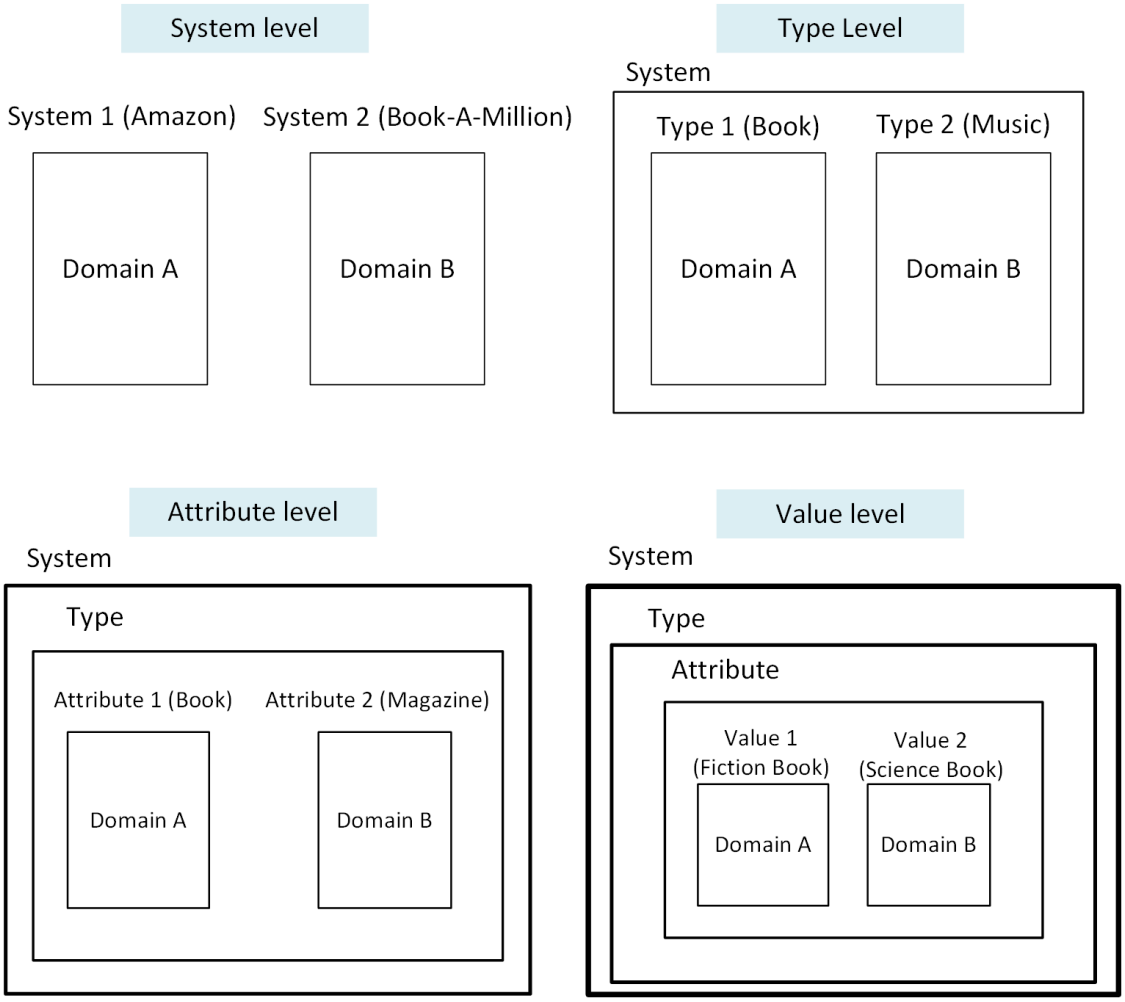

Figure 3.6: Various ways to define distinct domains

as books on Amazon and books on Book-A-Million[9] can be considered as different domains. For example, in the Pan and Yang research [32] movies information extracted from MovieLens[10], MoviePilot[11], and Netflix are considered in different domains based on their source.

Above classification is displayed in the Figure 3.6 as well. Multi-domains approaches utilize various information exposed into separate domains by the same users or by different users to pursue some main goals. The most important one maybe is addressing the cold-start problem. Shapira et al. [31] tackle this problem by using data from two domains which have

[9] A bookstore chain in the United States, www.booksamillion.com
[10] A web-based recommender system for movies, https://movielens.org
[11] An online magazine covering the film industry, https://moviepilot.com

a type-level difference. One special case of the cold-start problem is the new-user problem. It means when a user joins to the community or just starts using the recommender system, this cannot provide accurate personalized suggestions because it does not know anything about the user's preferences. Hu et al [33] extract books, movies, and music domains' information and address the new-user issue in their proposed recommender system. Finally, one common goal of using multi-domain recommender systems is improving the accuracy of the predictor model [29, 34, 35].

In addition, the information flow between two distinct domains can have different shapes. Here we classify them into two major categories and later we will try each of those to create cross-domain recommender systems:

- *Knowledge aggregation* where the information from two domains is aggregated to each other and then is used to suggest items or create a predictor model. One possible manner is merging different features in two domains together.

- *Knowledge transmission* where the source domain is connected to the target domain using a common explicit feature or implicit latent feature, and the transferred knowledge from the source domain is used to produce recommendation is the target one.

3.2 Matrix Factorization

Many recommender systems implement matrix factorization technique to characterize users and items and extract latent factors of the rating matrix. The aim of this section is covering the preliminaries of related algebra and reviewing matrix and coupled matrix factorization techniques and their applications in the recommender systems.

3.2.1 Preliminaries of matrix algebra

As our convention, matrices are shown with uppercase letters and vectors are denoted by lowercase letters. **Matrix** A is a $m \times n$ table with m rows and n columns which contains scalars. In this study, we assume that all cells include only real scalars, so we can say that the set of $m \times n$ real matrices is indicated by $\mathbb{R}^{m \times n}$. The element in the i-th row and j-th

column of matrix A is denoted by A_{ij}. If a matrix have same number of rows and column, it is called a square matrix. A diagonal matrix D is a square matrix which all elements except elements on its main diagonal is equal to zero, so for any element A_{ij} in the matrix D if $i \neq j$ then $A_{ij} = 0$. **Identity matrix** I_n is a $n \times n$ diagonal matrix where its diagonal elements are equal to 1. A column vector is a matrix with only one column. Similarly, a row vector can be defined as a matrix of only one row. the **rank of matrix** $A \in \mathbb{R}^{m \times n}$, denoted by $rank(A)$, is the maximum number of its linearly independent columns (rows) vectors. Matrix A is called **rank-1 matrix** if it is factorizable as an outer product of column vectors u and v, $A = u \otimes v$. In this case, we can say $rank(A) = 1$.

Ho, in his study [36], present a comprehensive overview on matrix and vector linear algebra. Here, we shortly mention some of the main matrix manipulations which are good to know to have a better understanding of the matrix/tensor factorization techniques which are discussed in the following sections.

- **Matrix product -** $A = B.C = BC$, where $A_{ij} = \Sigma_k B_{ik}.C_{kj}$
- **Transpose of matrix -** $[A_{ij}]^T = A_{ji}$
- **Inner product of vectors -** if $u, v \in \mathbb{R}^n$ then:

$$\langle u, v \rangle = \Sigma_i u_i v_i = u^T v \tag{3.2}$$

- **Matrix vectorization -** Lets $A \in \mathbb{R}^{m \times n}$, then:

$$vec(A) = \begin{bmatrix} A_{:1} \\ \vdots \\ A_{:n} \end{bmatrix} \in \mathbb{R}^{mn} \tag{3.3}$$

- **Inner product of matrices -** $\langle A, B \rangle = vec(A)^T vec(B) = \Sigma_{ij} A_{ij} B_{ij}$
- **Inverse of matrix -** B is inverse of A, denoted by $B = A^{-1}$, if $AB = BA = I$

- **Kronecker product -** Assume matrix $A \in \mathbb{R}^{m \times n}$, then Kronecker of A and B is:

$$A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \cdots & A_{1n}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1}B & A_{m2}B & \cdots & A_{mn}B \end{bmatrix} \tag{3.4}$$

- **Outer product of vectors -** Lets $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^m$ be two vectors. Then, their outer product is a matrix W, where $W_{ij} = u_i v_j$ and $W \in \mathbb{R}^{n \times m}$. Outer product is an special case of Kronecker product.

- **Hadamard product -** $A = B \circ C$, where $A_{ij} = B_{ij}C_{ij}$. Hadamard product is an elementwise product and $A, B, C \in \mathbb{R}^{m \times n}$.

- **Eigenvectors -** $\lambda \in \mathbb{R}$ is an eigenvalue and nonzero vector $x \in \mathbb{R}^n$ in an eigenvector of matrix $A \in \mathbb{R}^{n \times n}$ if $Ax = \lambda x$. $\sigma(A)$ is the set of all eigenvalues of matrix A:

$$\sigma(A) = \{x_i | Ax_i = \lambda_i x\} \tag{3.5}$$

  and the spectral radius of matrix A is defined as follow:

$$\rho(A) = max\{|\lambda| \,|\, \lambda \in \sigma(A)\} \tag{3.6}$$

- **Rank-R decomposition of matrix -** the minimal number of rank-1 matrices whose their linear combinations results the matrix A, is called the rabk-R decomposition of the matrix A:

$$A = \sum_{r=1}^{R} \beta_r (u^{(r)} \otimes v^{(r)}) \tag{3.7}$$

  where, β_r is a scalar coefficient and R represents the rank-R decomposition of matrix A.

- **Norms of a vector (or a matrix) -** Norms are used to measure the magnitude of a vector or a matrix. A most common norm is the Frobenius norm:

$$\| x \|_F = \sqrt{\langle x, x \rangle} \tag{3.8}$$

where, x can be either a vector or a matrix.

- **Nonnegative matrix -** matrices which all their elements are equal to or greater than zero are called nonnegative matrices.

3.2.2 Matrix and coupled matrix factorization techniques

Singular Value Decomposition (SVD)

The Singular Value Decomposition (SVD) [37] factorizes the real matrix $A \in \mathbb{R}^{m \times n}$ into three matrices $U \in \mathbb{R}^{m \times m}$, $\Sigma \in \mathbb{R}^{m \times n}$, and $V \in \mathbb{R}^{n \times n}$ (Figure 3.7). The U and V are orthogonal matrices which are called left and right singular matrices and contain the left singular values and right singular values respectively. In fact, U is formed of singular vectors of AA^T and V is formed of singular vectors of A^TA. Matrix Σ is a diagonal matrix which its diagonal elements represent the singular values of A. Therefore, based on the SVD:

$$A = U \Sigma V^T \tag{3.9}$$

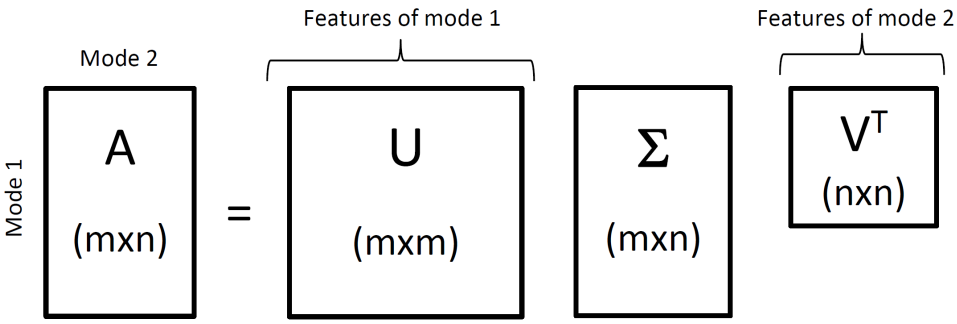

Figure 3.7: Singular Value Decomposition (SVD)

One another important point is that the best low-rank approximation of the matrix A, with respect to the Frobenius norm, is accomplished by truncating its SVD. Eckart and Young [38] proved that the best K-low-rank estimation of the matrix A can be desribed as follow:

$$A = U\,\Sigma_K\,V^T \tag{3.10}$$

where, Σ_K is created by keeping only K largest singular values of matrix Σ and replacing others by zeros.

We can use the U, V, and Σ matrices to reconstruct the original matrix A. The problem here is finding those three matrices such that the difference between A and $\acute{A} = U\,\Sigma\,V^T$ be as smaller as possible. The most used measure is the Frobenius norm:

$$F(A, U\Sigma V^T) = \frac{1}{2} \parallel A - U\Sigma V^T \parallel_F^2 \tag{3.11}$$

Nonnegative Matrix Factorization (NMF)

Although the Nonnegative Matrix Factorization (NMF) first is introduced by Paatero and Tapper [39] , but it seems that the most popular research work in this field is the Lee and Seung study [40] .

Given an nonnegative matrix $A \in \mathbb{R}^{m\times n}$, where $A_{ij} \geq 0$, and a rank r, where $r < min(m, n)$, the NMF factorizes matrix A into two separate nonnegative matrices $U \in \mathbb{R}^{m\times r}$ and $V \in \mathbb{R}^{n\times r}$ (Figure 3.8). Therefore, we have:

$$A = U\,V^T \tag{3.12}$$

Again here we can reconstruct the original matrix A using U and V :

$$\acute{A} = UV^T \tag{3.13}$$

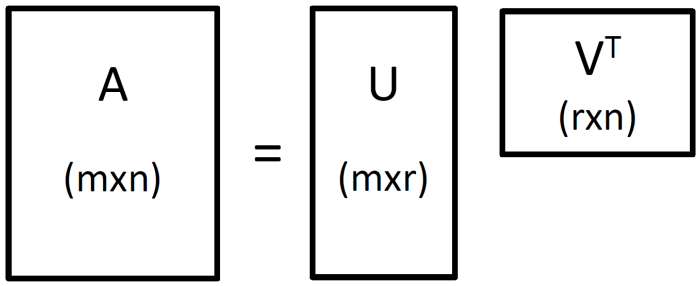

Figure 3.8: Nonnegative Matrix Factorization (NMF)

Using the Frobenius norm to calculate the difference between A and $\acute{A}$ [36], we have:

$$F(A, UV^T) = \frac{1}{2} \parallel A - UV^T \parallel_F^2 = \frac{1}{2} \Sigma_{i=1}^{m} \Sigma_{j=1}^{n} (A_{ij} - [UV^T]_{ij})^2 \tag{3.14}$$

$$\min_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{n \times r}} \frac{1}{2} \parallel A - UV^T \parallel_F^2 \tag{3.15}$$

Coupled matrix factorizations

SVD and NMF are two popular matrix factorization techniques and both have had many successful implementations in the recommender system field [41, 42, 15] and also in other areas. If we want to factorize two matrices which have one mode in common, in order to incorporate different features of these matrices together, we can implement a modified version of SVD or NMF.

The coupled matrix factorization technique, which we are considering in this study, applies same factorization approach on two matrices simultaneously. Assume nonnegative matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{m \times q}$ which have identical features on their first modes, then the SVD of A and B are $U_A \Sigma_A V_A^T$ and $U_B \Sigma_B V_B^T$. However, to factorize them at a same time we consider $U = U_A = U_B$. Then, the coupled SVD corresponding to Equation 3.11 is:

$$F(A, B, U\Sigma_A V_A^T, U\Sigma_B V_B^T) = \frac{1}{2} \left(\parallel A - U\Sigma_A V_A^T \parallel_F^2 \parallel + \parallel B - U\Sigma_B V_B^T \parallel_F^2 \right) \tag{3.16}$$

In addition, the NMF of matrices A and B are $U_A V_A^T$ and $U_B V_B^T$ respectively. Again, lets $U = U_A = U_B$, then, as shown in Figure 3.9 schematically, the coupled NMF corresponding to Equation 3.14 is as follow:

$$F(A, B, UV_A^T, UV_B^T) = \frac{1}{2}\left(\| A - UV_A^T \|_F^2 + \| B - UV_B^T \|_F^2\right) \tag{3.17}$$

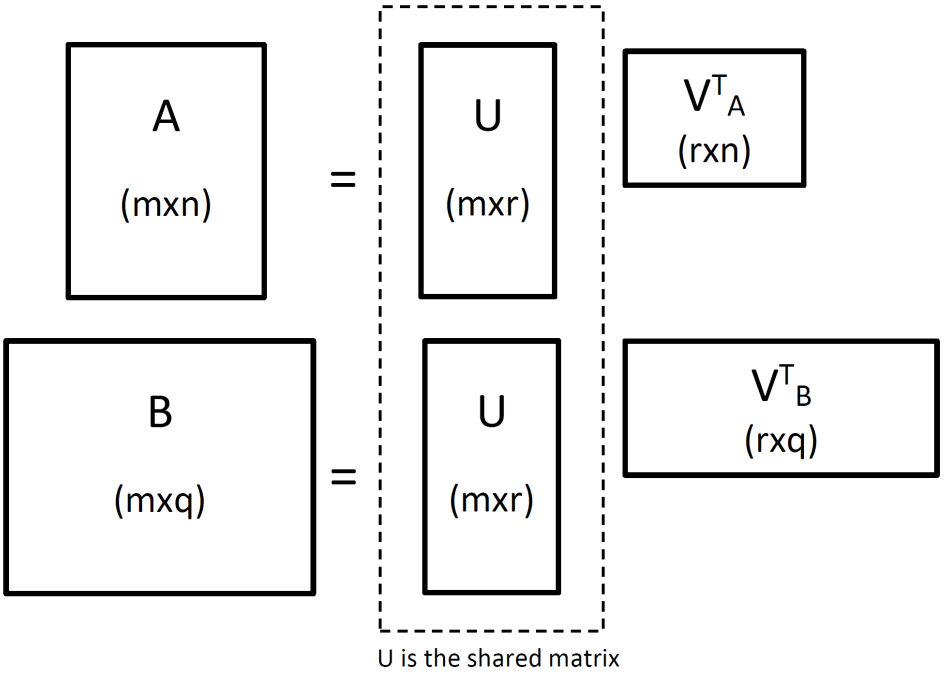

Figure 3.9: Coupled NMF

3.2.3 Matrix and coupled matrix factorization applications in recommender systems

By applying the matrix factorization techniques, such as SVD and NMF, on the user-item rating matrix, these techniques form some low-rank matrices (Figures 3.7 and 3.8) whose their production will approximate the rating matrix. This feature of matrix factorization approaches has a close affinity with this assumption that a small number of latent features affects the rating patterns [41].

Paterek [43] combine different collaborative filtering predicting algorithms, such as regularized SVD, post-processing SVD with KNN and K-means, in order to produce more accu-

rate models. Using the non-negative matrix factorization, Hernando et al. [44] convert the rating (user,item) table to two matrices whose their components lie between 0 and 1. They show that this method convey more understandable probabilistic meaning than classic matrix factorization. In addition, By comparing the accuracy of the predictions and recommendations of the proposed method and classic matrix factorization, the authors indicate that this method produce more accurate results. In another popular research study, Koren et al. [41] develop a matrix factorization-based recommender system and also investigate various learning methods to solving the factorization problem, such as Stochastic Gradient Descent (SGD) and Alternating Least Squares (ALS). They show that matrix factorization models are superior to traditional KNN-based techniques and produce more precise recommendations. Ma et al. [45] take into account both information of users in the social network and rating matrix and implement a probabilistic matrix factorization. Their suggested method shows high accurate recommendation even for sparse data. In order to deal with the high amount of available data, Yu at all. [46] propose a nonparametric SVD matrix factorization which allows the latent factors to be data-driven. Also, they suggest new learning models which allow their nonparametric matrix factorization technique to be highly efficient on large-scale data. More recently, Guo et al [47] show that not only explicit patterns of user-item rating are important but also the implicit influence of rating matrix can play an important role. The authors suggest an SVD-based technique as a collaborative filtering recommender system which takes into account the explicit influence of trust (trust values), the implicit influence of trust (who trust whom), and the user-item rating. The implicit effect of trust is added to the rating matrix as an extra feature. And the explicit impact of trust is used to constrain that user latent factor should be matched with their social trust relationships. In this way, the latent factors of a particular user can be predicted from trust information, even only a few rating are available for that user.

One of the first implementations of coupled matrix factorization is the published study of Long et al [48]. In this study, they propose a model to clustering relational data, which come from different domains, using a coupled matrix factorization. Later, Singh and Gordon [49] conduct a similar study. They evaluate their proposed method by using separate matrices of user-movie, movie-genre, and movie-actor. Based on the highly accurate results, the authors

conclude that combining information from multiple domains and using the coupled matrix factorization leads to better predictions. In another study, Acar et al. [50] consider two different metabolomic matrices of $X \in \mathbb{R}^{I \times J}$ and $Y \in \mathbb{R}^{I \times K}$ which have the first mode in common. They perform a coupled matrix factorization to recommend potential biomarkers for apple intake. Their suggested method is finding the matrices $A \in \mathbb{R}^{I \in R}$, $B \in \mathbb{R}^{J \in R}$, and $C \in \mathbb{R}^{K \in R}$ that minimize the following equation:

$$F(A, B, C) = \| X - AB^T \|_F^2 + \| Y - AC^T \|_F^2 \tag{3.18}$$

More recently, Wang et al. [51] implement a coupled matrix factorization method to predict the response time in logistic services. For the service-response matrix $R \in \mathbb{R}^{N \in M}$ and its corresponding weight matrix W from the first domain, the order feature matrix $X \in \mathbb{R}^{N \times S}$ from the second domain, and finally the driver feature matrix $Y \in \mathbb{R}^{M \times T}$ from the third domain, their method is finding the matrices U, V, G, and H in order to minimize the following equation:

$$F(R, W, X, Y, U, V, G, H) = \frac{1}{2} \| W \circ (R - UV^T) \|_F^2 + \frac{\lambda_1}{2} \| X - UG^T \|_F^2 + \frac{\lambda_2}{2} \| Y - VH^T \|_F^2 \tag{3.19}$$

3.3 Tensor Decomposition

Recently, many researchers have utilized the tensor-based representation of data in order to represent the multi-modal relationships between various features [52] within a single domain or multi-domains. In addition, the tensor decomposition techniques have been used in many studies to capture latent factors of different features and develop predictor and recommender systems. In this section, we will start by a review of multilinear algebra, then we will mention popular tensor decomposition techniques and their applications in the field of recommender systems.

3.3.1 Preliminaries of tensor algebra

A **tensor** is a multidimensional array. Sometimes it is described as a higher-order consideration of vectors and matrices. For example, a 3-way tensor, a cubic of data, can be considered as a list of 2D matrices. More formally, a tensor is a multilinear mapping over a set of vector spaces. for the real tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$, the M is the **order of tensor**. A tensor with the order of M also known as M-way tensor. The **dimension of tensor** $\mathcal{A}$ in its I-th order is the number of distinct features on that order.

The followings are some fundamental definitions which are necessary to understand the math behind the tensor decomposition techniques [53]:

- **Rank-1 tensor -** If $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ is an M-way tensor and there are vectors $u_1, u_2, \cdots, u_M$ such that $\mathcal{A} = u_1 \otimes u_2 \otimes \cdots \otimes u_M$, then tensor $\mathcal{A}$ is rank-1 tensor.

- **mode-m vectors -** the mode-m vectors of the Mth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ is calculated by varying index I_m while keeping the other indices constant.

- **Mode-m unfolding -** Lets $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$, then element $(i_1 \times i_2 \times \cdots \times i_N)$ of tensor $\mathcal{A}$ is mapped to element (i_m, j) of the m-mode unfolding, also known as m-mode matrixizing, of $\mathcal{A}$, which is identified by $A_{[m]} \in \mathbb{R}^{I_1 \cdots I_{m-1} I_{m+1} \cdots I_M}$, with:

$$j \; = \; 1 \; + \sum_{k=1, k \neq m}^{N} \left[(i_k - 1) \prod_{\substack{n=1 \\ n \neq m}}^{k-1} I_n \right] \tag{3.20}$$

- **Mode-m rank -** the mode-m rank of the M-way tensor $A_{[m]} \in \mathbb{R}^{I_1 \cdots I_{m-1} I_{m+1} \cdots I_M}$ is defined as:

$$R_m^{\mathcal{A}} \; = \; rank_m(\mathcal{A}) \; = \; rank(A_{[m]}) \tag{3.21}$$

- **Mode-m product -** consider tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ and matrix $B \in \mathbb{R}^{J_m \times I_m}$, then the mode-$m$ product of them, denoted by $\mathcal{A} \times_m B$, is tensor $\mathcal{C} \in \mathbb{R}^{I_1 \times \cdots \times I_{m-1} \times J_m \times I_{m+1} \times \cdots \times I_M}$

- **Khatri-Rao product -** the Khatri-Rao product of matrices $A \in \mathbb{R}^{I \times J}$ and $B \in \mathbb{R}^{K \times J}$, which is denoted by $A \odot B$, is calculated as:

$$A \odot B = \left[(a^{(1)} \otimes b^{(1)}) \cdots (a^{(l)} \otimes b^{(l)}) \cdots (a^{(L)} \otimes b^{(L)}) \right] \tag{3.22}$$

where, $[A \odot B]_{ik,j} = a_{ij} b_{kj}$ and $A \odot B \in \mathbb{R}^{IK \times J}$ and $a \otimes b$ represents the outer product of a and b.

3.3.2 Tensor decomposition techniques

Allegedly, the idea of representing data through a multi-way model came from Cattell's study [54]. However, the topic became more popular after Tucker's great studies on multi-way factor analysis [55, 56, 57]. More recently, many researches have used tensor decompositions in the wide range of fields such as numerical analysis [58], graph analysis [59], neuroscience [60], image analysis and computer vision [61], social network analysis [62] and many other areas. Here, we review two widely used tensor decomposition techniques which have many successful implementations in the recommend systems as well.

CANDECOMP/PARAFAC (CP) decomposition

The idea of polyadic decomposition of tensors was first proposed by Hitchcock [63]. Further, it rediscovered by other researchers in the form of CANDECOMP (canonical polyadic decomposition) [64] and PARAFAC (parallel factor decomposition) [65]. Later, due to the Kiers study [66], this technique has become popular as the CANDECOMP/PARAFAC (CP) approach.

As in this research study we are using the 3-way tensor as part of our suggested solutions in the next chapter, here we are going to introduce the 3-way CP decomposition. Assume 3-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, then based on CP we can represent the $\mathcal{X}$ as the sum of three-way outer products [67]:

$$\mathcal{X} = \sum_{r=1}^{R} a_r \otimes b_r \otimes c_r \tag{3.23}$$

where, $a_r \in \mathbb{R}^I$, $b_r \in \mathbb{R}^J$, and $c_r \in \mathbb{R}^K$. R is a positive integer which is called the number of components. In addition, The following matrices are identified as the factor matrices which are the combinations of the vectors from the rank-one components:

$$\begin{aligned} A &= [a_1 \ a_2 \ \cdots \ a_R] \\ B &= [b_1 \ b_2 \ \cdots \ b_R] \\ C &= [c_1 \ c_2 \ \cdots \ c_R] \end{aligned} \tag{3.24}$$

In can be shown that [68]

$$\mathcal{X} \simeq AD^{(k)}B^T \tag{3.25}$$

where, $D^{(k)} \equiv diag(c_{k:})$ for $k = 1, 2, \cdots, K$.

The CP decomposition may be regarded as the generalization of the matrix SVD because, as it can be seen in the Equation 3.25, like the SVD which factorizes a matrix to three matrices, the result of the CP decomposition can be represented as the three factor matrices.

In order to solve the CP decomposition, we find the A, B, and C such that they minimize the following equation:

$$F(\mathcal{X}, A, B, C) \ = \ \Big|\Big|\mathcal{X} - \sum_{r=1}^{R} a_r \otimes b_r \otimes c_r\Big|\Big|_F^2 = \Big|\Big|\mathcal{X} - AD^{(k)}B^T\Big|\Big|_F^2 \tag{3.26}$$

One of main the reasons for the popularity of the CP decomposition is that it is ease for interpretation. In fact, each decomposed rank-one component serves as a cluster is data. In addition, one another feature of CP decomposition is that its' result is unique [69].

Tucker decomposition

The Tucker decomposition was first presented by Tucker in 1963 [55] and more comprehensive versions of this idea was introduced later in 1966 [57] by the same author. Although, Tucker offered three different methods in the later study, but the third, which known as Tucker-3, became more prevalent. Therefore, we shortly review this technique here and for the ease refer to Tucker-3 as Tucker decomposition technique.

Assume the 3-way tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$. The Tucker technique decomposes this tensor to three factor matrices $A \in \mathbb{R}^{I \times R_1}$, $B \in \mathbb{R}^{J \times R_2}$, $C \in \mathbb{R}^{K \times R_3}$, and a smaller tensor, also known as core tensor, $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$. Therefore, we have:

$$\mathcal{X} \simeq \mathcal{G} \times_1 A \times_2 B \times_3 C \tag{3.27}$$

$$\mathcal{X}_{i\,j\,k} \simeq \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \mathcal{G}_{r_1\,r_2\,r_3} A_{i\,r_1} B_{j\,r_2} C_{k\,r_3} \tag{3.28}$$

The Tucker problem can be described as finding the $\mathcal{G}$, A, B, and C for selected reduced dimensions R_1, R_2, and R_3 in order to minimize the difference between the original tensor and the reconstructed one:

$$F(\mathcal{X}, \mathcal{G}, A, B, C) = \left\| \mathcal{X} - (\mathcal{G} \times_1 A \times_2 B \times_3 C) \right\|_F^2 \tag{3.29}$$

In contrast with the CP decomposition, the Tucker's result is not unique. However, this decomposition technique provides a good low-rank approximation of tensor, much like the SVD for matrices. Also, it is widely used for dimensional reduction purposes as well. Beside the Tucker-3 [57], another popular algorithm for Tucker decomposition is the Higher-Order Singular Value Decomposition (HOSVD) presented by De Lathauer et al. [70].

Papalexakis et al. [53] summerize popular tensor decomposition techniques and review their applications in many diverse fields, such as social network analysis, recommender systems, computer networks, information retrieval, web mining, healthcare, speach and image processing, and urban computing. Moreover, in this survey paper, the authors cover most important studies that have attempted to design scalable tensor decompositions to handle large datasets and be compatible with the big data. Table 3.2 summerizes their findings.

Table 3.2: Comparison between CP and Tucker decompositions

Tensor decomposition technique	Advantages	Disadvantages	Applications
CP	Easy to interpret, unique under mild conditions	Hard to find the proper rank, hard to determine the global minimum	Explanatory analysis, clustering, capturing latent factors, compressing data which have low-rank multilinear structure
Tucker	Compresses the tensor, extracts non-trilinear variations	Hard to interpret, Nonunique result,	Compressing the data that do not have low tensor rank, analyzing relations between latent components

3.3.3 Applications of tensor decomposition in recommender systems

CP applications

Zheng et al. [71] consider GPS data and develop a collaborative recommendation system to suggest locations and activities based on the users GPS traces. They construct a user-location-activity tensor and apply a nonnegative CP-style decomposition to this tensor along with four supplementary matrices. These matrices, namely user-location, user-user, location-feature, and activity-activity matrices, are integrated with the original tensor to deal with its inherent sparsity. Based on the result, using the additional information represented in the mentioned matrices increases the accuracy of the produced recommendations in comparison to the case that the only input information is the original 3-way tensor.

Kutty et al. [72] propose a tensor-based hybrid recommendation system for the people

network with two different user types, such as male-female, doctor-patient, and employer-employee. They capture the present 2-way relationships in these networks by considering two separate third-order tensors in shape of (sender, receiver, attribute values of the receiver), which sender and receiver are network's users with various types. Based on the interaction between the users, each element of the tensor might be positive or negative. If the message had sent by the sender got a positive response from the receiver the corresponding element is positive. However, if the receiver has negatively responded to the message or ignored it, the element is negative. The authors apply the CP method over two tensors separately. Each reconstructed tensor can be used to create 1-way recommendations, but, to follow the inherent 2-way relationships of the network they only collect those pairs which is recommended in both reconstructed tensors. To Evaluate their people-to-people recommendation system, Kutty et al. use a dataset collected from an online dating site and show that their method can achieve better precision and recall in comparison to former algorithms.

Kao et al. [73] show in their study that how CP decomposition can be applied to reveal topics and relationships in a temporal social network. In their case study, they extract data from Twitter including tweets related to a specific event. Considering various parameters such as user, retweeted user, term, hashtag, and time, they create 3-way and 4-way tensors and decompose them to explore the principal factors, different relationships between terms and users, and topic trend in different time windows.

Yao et al. [74] use the multidimensional information of check-in data from Location Based Social Networks (LBSNs) to develop a context-aware Point of Interest (POI) recommender system. They construct a 3-way user-location-time frame tensor and apply the CP technique to extract the hidden information. They also impose the users' social connections as an additional social regularization to improve the accuracy of the produced predictions. Since in the tensor decomposition technique the natural multidimensional relationship between LBSN's data is considered properly, the suggested method results in more accurate recommendations rather than baseline linear algorithms.

Zheng et al. [75] develop a recommender system using a CP-based probabilistic tensor factorization which takes into account the social relationships, rating, item content, and contextual information by decomposing a 4-way user-item-context-rating tensor. Through

proving the superiority of the proposed approach in comparison with other common context-aware recommender systems, which most consider only two parameters, the authors show that different information types affect the users' behavior simultaneously and considering them together results in more accurate predictions.

In another study, Park et al. [76] develop a tensor decomposition package, running on distributed Hadoop to manipulate large scale tensors. The core ideas in their work are handling the intermediate data explosion problem, reducing the floating point operations, and decreasing the number of redundant MapReduce jobs. The package consists of five different decomposition methods: CP, non-negative CP, Tucker, non-negative Tucker, and coupled matrix-tensor factorization. In order to analyze the obtained factor matrices from huge tensors, they also consider two more points. First, they consider only the top-k highest valued elements in each factor matrix because the number of elements in each matrix is too large to be explored easily. Second, they rank these top-k elements based on the calculated specificity score to avoid biased filtering toward frequent elements in different columns.

Almutairi et al. [77] investigate the application of matrix and tensor factorizations in the learning analytics by developing recommender models to predict students' grades in courses which are new for them. They propose two models based on the coupled matrix factorization method which incorporate a main historical grades matrix with a context matrix. They consider absolute time, measured in semesters, in which the course is taken and student experience, which is equal to the number of semesters the student has been in the current program, as the contextual information. Moreover, they suggest another model based on the tensor decomposition as well. In this approach, they create a tensor by combining those matrices and apply the CP decomposition to predict the students' future grades. The authors show that all these model represent higher accuracy rather than the matrix factorization method because they take extra contextual information into account.

Nakatsuji et al. [78] use 3-way tensors, (user-topic-user), extracted from Twitter to predict links among users and topics discussed among them. Their proposed algorithm consists of an extension of the Variational nonnegative matrix factorization method which is an instance of the CP decomposition technique. In addition, they tackle the biased prediction issue and tensor sparsity problem by connecting topics extracted from tweets to the DBpedia dataset.

They create a single tensor including all the users who tweet or retweet contents about some chosen topics. However, the authors claim that the prediction created using this single tensor tends to be biased toward the popular topics. In addition, they state that this social network tensor is usually sparse because tweets and retweets of each user just include a small set of topics among the wide range of topics that are discussed in the network. Therefore, they suggest the following steps in their method to answer these mentioned problems: 1) Considering crawled tweets about some initial topics and the corresponding DBpedia entities to these topics, first for each chosen topic they find those entities in the DBpedia knowledge base which are in the certain neighborhood radius of that specific entity. If these achieved entities are used in the extracted tweets, they consider them as the related topics to the initial topic. In this way, for each initial topic, they create a domain which is a set of topics. So, they avoid strong biases toward a specific topic or domain by creating a separate tensor for each domain. 2) in order to deal with the sparsity, the authors select the most sparse topics in the obtained tensors and for each of these sparse topics, they find the neighboring entities of the corresponding entity in the DBpedia dataset. Then, they create a newly augmented tensor by adding new (user-topic-user) relationships including the topics of these neighboring entities to the tensor from the previous step. Next, they decompose the tensors obtained from steps 1 and 2 as well as the first large tensor simultaneously and semantically augment tensors in different domains to handle sparsity of the tensors.

Rabanser et al. [79] conduct a case study to show how tensor decomposition can be implemented in the machine learning solutions. In order to estimate the latent variable models, they first apply the Method of Moments over Gaussian Mixture Model and Topic Model to create 3-way moments, then they utilize the Tensor Power Method to uncover the latent structure of these moment tensors.

More recently, Ioannidis et al. [80] develop a new method to decompose a tensor coupled with a graph-style data. They argue that this joint analysis method can demonstrate the latent structure of related heterogeneous data from various information repositories. Their proposed method consists a CP decomposition for the tensor and a nonnegative matrix factorization for the corresponding matrix of the graph. Furthermore, they use an algorithm based on the alternating direction method of multipliers to obtain the latent factor matrices

and predict the missing values of the tensor by considering the graph matrix. They show that this graph-tensor factorization method can also be implemented to detect communities on an incomplete graph by using the recovered factors. To evaluate their method, the authors create an activities recommender system using a real dataset and compare the results with CP, nonnegative tensor decomposition, and matrix-tensor decomposition algorithms. Based on these results, the proposed algorithms achieve more accurate predictions and perform better than those mentioned alternatives.

Tucker applications

Peng et al. [81] suggest a collaborative recommender for social tagging system. They apply Tucker decomposition to extract the lower dimensional representation of users and use them to compute the users' similarity. Instead of using only these similarities to make recommendations, they construct item-tag joints and project them into the item space to make the final recommendation. They evaluate this method using various datasets and show that it gives more precise recommendation than previous user-based methods.

Around the same time, Symeonidis et al. [82] implement the HOSVD method to reduce the dimensionality of a 3-way (user, tag, item) tensor and analyze multiway latent semantic presents in social tagging systems' data. In addition to the tensor decomposition method, they utilize the kernel-SVD smoothing technique to deal with the tensor sparsity. They create recommendation systems using tensor decomposition and kernel-SVD methods to suggest tags, items, and similar users. They show that the tensor decomposition method provides more precise recommendations in comparison with other traditional approaches such as fusion, matrix SVD, item-based, FolkRank, and baseline algorithms. Moreover, they study the influence of the core tensor dimensions on the recommendation accuracy. They demonstrate that by fixing the dimensionality of one modal, the optimal value for other two can be calculated.

Zou et al. [83] exploit the GPUs' flexible programming feature as well as their efficient parallelism to accelerate the tensor decomposition, especially for big data. The suggested GPU-based algorithm partitions a tensor to some smaller blocks and then perform the n-mode production block by block, then calculate the tensor factorization parallelly using HOSVD. It

also comprises an optimization strategy to deal with the intermediate data explosion problem. The authors show that the parallel GPU-based HOSVD can be 10 times faster than the case which this decomposition technique is applied without parallelism using CPU.

Maroulis et al. [84] propose a context-aware POI recommendation system for LBSNs. By considering various contexts such as time, date, and category transition, they construct a user-POI-context tensor and apply the HOSVD. The authors assess the performance of this method by comparing its results to some other techniques which rely on the matrix factorization approach. They show that the tensor-based method provides higher precision and recall rather than matrix-based algorithms.

Symeonidis [85] suggests an approach to deal with some of the drawbacks of the HOSVD technique in social tagging systems. To be more specified, this method reduces the high factorization dimension of HOSVD and addresses the data sparsity in the STSs simultaneously. Technically, it includes a tag clustering step before constructing the tensor. Therefore, HOSVD is applied over the user-tag cluster-item tensor instead of the user-tag-item tensor. This method not only reduces the tensor sparsity and causes more accurate recommendations but it also decreases tag ambiguity and tag redundancy. K-means, spectral, and hierarchical agglomerative methods are compared for the clustering step. Although, using each of them with HOSVD leads better results rather than the HOSVD itself, but higher precision is achieved using the spectral clustering method.

Zheng et al. [86] develop a Tucker-based tensor topic model to capture low-dimensional representations of users, words, and items in textual reviews. To do that, they form a 3-way (user, item, word) tensor and apply the Tucker decomposition to obtain the multi-modal relationships. Afterward, these results are used to construct a probabilistic model for rating prediction.

Ying et al. [87] propose a temporal-aware POI recommendation system consists of two steps. The first step is learning temporal-aware user preferences through a Tucker-based decomposition method; Second is inferring the score of POI using a weighted Hypertext Induced Topic Search (HITS)-based rating approach. Finally, POI recommendations are produced by assembling these two steps, considering user preferences, temporal influences, and social opinions. In order to model the user preferences in the first step, they create a

3-way (user, POI category, time) tensor. Then, the authors suggest a Tucker-based context-aware decomposition approach which can partly handle the sparsity of this tensor and create better estimations, by incorporating the original tensor with three auxiliary user-features, category-time and category-category matrices. The authors evaluate the effectiveness of the proposed approach for user preferences modeling by comparing it with some other options. These options include using the average of all non-zero entries in the related time slot instead of implementing the tensor decomposition method, applying the user-category matrix factorization instead of the tensor decomposition, and the tensor decomposition using only one or two of the three mentioned supplementary matrices. The results show that the suggested context-aware tensor decomposition method outperforms all other options by showing less root mean square error (RMSE) and mean absolute error (MAE).

Another research which investigates the application of tensor decomposition for cross-domain recommendation is the Taneja and Arora study [88]. The authors design a cross-domain recommender system which takes into account the different features of users' interests, expressed within the source and target domains, in order to provide new recommendations. The authors state that using proposed cross-domain multi-dimension tensor factorization method, this recommender system can deal with the sparsity and cold start problems better than traditional single-domain recommenders. In their case study, they consider two 5-way tensors in the source and target domains. To handle the sparsity of tensors, they use the agglomerative hierarchical clustering technique to group data in the source domain based on the available features and then map the common features in the target domain to the corresponding created clusters. In addition, they implement the HOSVD to discover the relations between different modes and create new recommendations.

In addition, recently Zhao et al. [89] present a model for POI recommendation by considering the most important temporal properties, including periodicity, consecutiveness, and non-uniformness. They construct a (user, time label, POI) user-time label-POI tensor and implement a Tucker-based decomposition technique, based on the study of Rendle et al. [90], to capture different temporal latent features. Then, they aggregate these features using a linear convex combination operator to calculate a score function for each given time label, user, and POI.

3.4 Summary

According to the importance of the recommender systems, any online service provider or social media uses some kind of recommendation techniques. These recommendation models may include collaborative filtering to capture the similarities between the users and suggest new items to a particular user based on these similarities, or may include content-based approach to understand the items' contents and recommend new items based on the items' descriptions, or even may be a combination of the collaborative and content-based methods (Figure 3.2). While these recommendation models usually use the information of a specific domain, such as a particular social media, to create a recommendation for that specific domain, they face with some important problems which decrease the accuracy of the recommended items.

Recently, it has found that by integrating different information from separate domains (Figure 3.6) and produce cross-domain recommender system, some of these problems, such as data sparsity and cold-start problem, can be partially addressed. In a cross-domain recommender system, various information exposed in separate domains are incorporated in one of these approaches: transferring knowledge from one domain to another one, or merging knowledge from various domains together.

The coupled matrix factorization technique is an example of knowledge transmission approach. While various matrix factorizations are well-known techniques to create single-domain collaborative filtering recommender systems, factorizing two separate user-item matrices can be used to create cross-domain recommendation models. It actually shares the latent features of these matrices together in order to make more precise recommendations.

On the other hand, tensor decomposition can be considered as an example of knowledge merge approach. By creating the multi-modal data structure using different domains' information, the latent relations between users and various items can be taken into account. Then, various tensor decomposition techniques can be applied to capture the latent features between these modes or to capture a compressed core of data.

4

RESEARCH METHODOLOGIES AND IMPLEMENTATIONS

As it said before, the main purpose of this study is bringing new recommender systems forth, which link various information from different domains together. We are using the state of art techniques, coupled matrix factorization and tensor decomposition, as well as diverse knowledge-based algorithms, to develop those cross-domain recommender systems. In order to implement and evaluate our suggested models, we are using a real dataset extracted from two popular social media websites, namely Twitter and LinkedIn.

In this chapter, we first describe our considered domains and the crawled dataset. The suggested methodologies can be divided into three parts. First part includes four different knowledge-based recommender algorithms which connect the LinkedIn's user-skill dataset as the source domain to the Twitter's user-account dataset as the target domain and vice versa. In the second part, a coupled matrix factorization method is proposed to make item recommendations using these two domains. And finally, the last part comprises a tensor decomposition model which is used to create cross-domain recommendations. The evaluation and results of these methodologies will be discussed in the next chapter.

4.1 Dataset Description

The Twitter and LinkedIn are considered as separate domains and suggested recommender systems connect information from those by either merging users' preferences or sharing latent factors of user-item matrices (Figure 4.1). Based on the Figure 3.6, it can be seen that the difference between the two domains is at the system level.

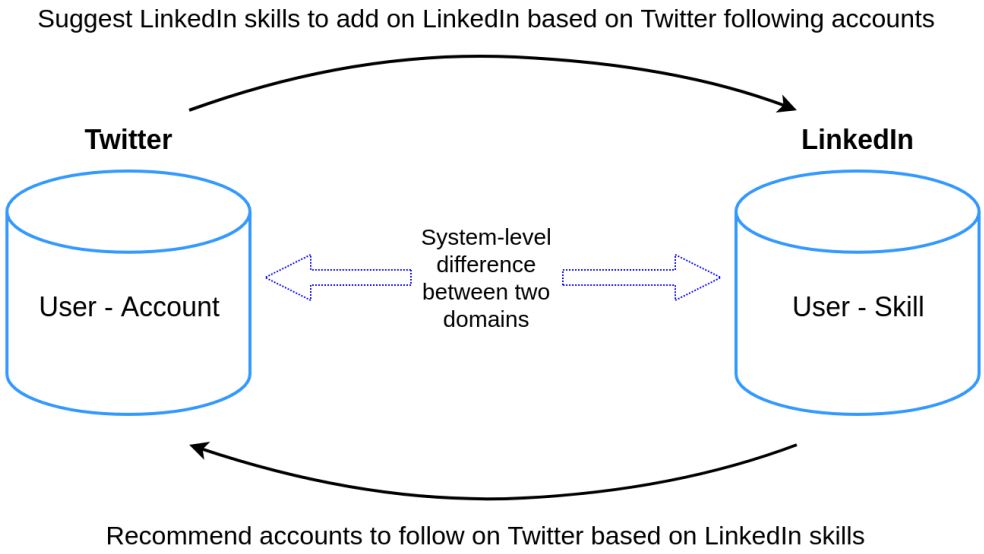

Figure 4.1: Schematic view of separate domains in our study

The first step is extracting desired data from these two domains such that they express various features of identical users. To do this, we chose to extract LinkedIn skills and Twitter following accounts of some computer scientists. The dataset should be about computer scientists who have active accounts on both Twitter and LinkedIn.

The first challenge here is: how to find experts working in the computer science area? Should we search LinkedIn to find who claimed to have computer science-related skills? Undoubtedly, if we just rely on the claimed skills of random people in LinkedIn, the gathered data will not be precise. As a solution, in this study, we used the LinkedIn search engine to find people with computer science-related skills who are working in high-tech companies of the computer industry, like Google, Facebook, Microsoft, IBM, and Amazon, and previously worked in one of these companies as well. Therefore, we assume that these people, who have computer science skills and have been worked in at least two high-tech companies, have enough knowledge and expertise in this field to recognize their own proficiencies with acceptable accuracy.

The second important issue is that finding all corresponding Twitter accounts is almost impossible. Every person who has a LinkedIn account not necessarily has a Twitter account. In addition, comparing LinkedIn and Twitter accounts based on the matching name is not practical too because there might be more than one account with the same real name on

Twitter. Also, finding Twitter accounts using image processing and based on matching pictures of two social medias' profiles might result in high inaccuracy. In this research study, to avoid mentioned errors, we just consider those above computer scientists who declared their Twitter IDs on their LinkedIn profile page. Table 4.1 describes the statistics of the extracted dataset.

Table 4.1: Statistics of the dataset

Description	**Number**
Users	492
All Twitter's following accounts	194053
Unique Twitter's following accounts	143974
Unique Twitter's following accounts which are followed by at least 5 users	3486
All LinkedIn's skills	13986
Unique LinkedIn's skills	2566

The word-cloud of Twitter following accounts and the word-cloud of LinkedIn skills are illustrated in Figure 4.2 and Figure 4.3 respectively. Using these figures we can understand easily which twitter accounts are more interesting for computer scientists and also which skills are more important in professional marketing. Words with larger size are more repeated in our dataset.

According to Table 4.1, more than %74 of Twitter accounts are unique. It means that there are many Twitter accounts followed by only a few users in our dataset. Although these accounts may not be related to computer science field, they actually do not affect the performance of proposed knowledge-based recommender systems, because, for them, we only consider most related cases while recommending Twitter accounts to follow or predicting LinkedIn skills. However, for coupled matrix factorization-based and tensor decomposition-based recommender systems, the unique Twitter's following accounts which are followed by at least five computer scientists are utilized to develop the recommending models.

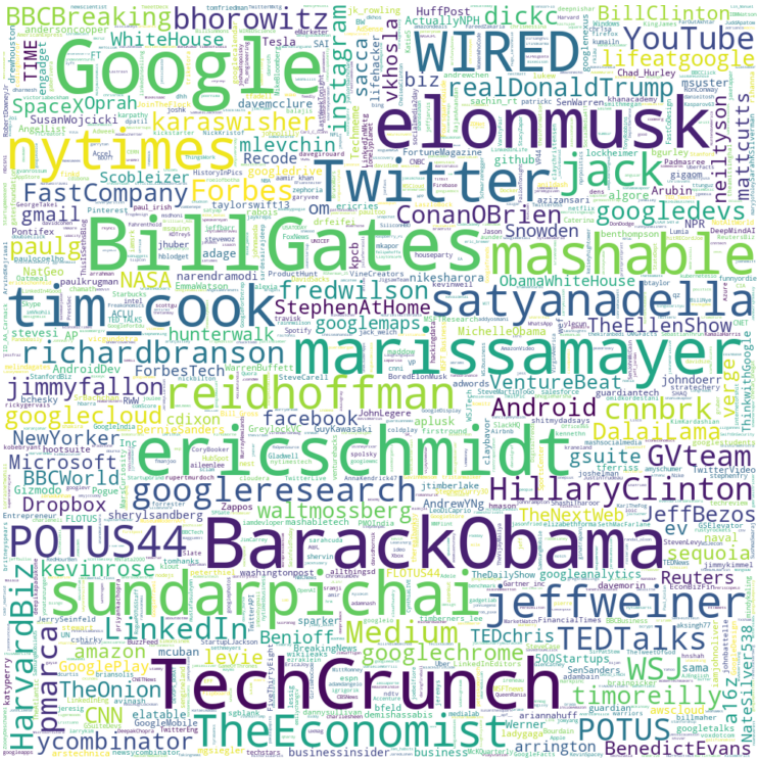

Figure 4.2: The word-clouds of following accounts extracted from Twitter

4.2 Part I: Knowledge-Based Recommender Systems

As we should verify the usefulness of proposed recommender systems in this section, we use train-test-split method and divide our dataset into two separate subsets randomly: a larger set which is used to train our algorithms and a smaller set that is used to evaluate the accuracy of proposed models. The train set includes the information of 90 percent of users in the two distinct domains.

4.2.1 Collaborative filtering recommender system

The first algorithm is collaborative filtering. In this approach, in order to suggest new items in the target domain to a particular user, we use the train set to find the first 10 percent of users who have most similarity with that user in the source domain. Then we suggest 50 Twitter accounts to follow (or 20 LinkedIn skills to add), based on the preferences of these similar users. Assume that we want to recommend some Twitter accounts to one of the users

Figure 4.3: The word-clouds of skills obtained from LinkedIn

of our test set based on his or her skills. Therefore, the input here is a set of LinkedIn skills. The first step is sorting the users of train set based on the matching between their list of LinkedIn skills and the input list. Then we select the top 10 percent computer scientists who have the highest number of common skills with that particular user. In the second step, we accumulate all Twitter following accounts belonging to these top similar users and sort them based on their repetition. In the end, we recommend 50 Twitter accounts which have more frequency in the obtained list of Twitter following accounts. Figure 4.4 shows the flowchart of our collaborative filtering recommender system.

4.2.2 Content-based recommender system

Here as the first step, we create a description for each Twitter account or LinkedIn skill; we want to know that having which skills may cause following a specific Twitter account, and following which Twitter accounts show that the user might have a unique skill. Hence, for each user in the train set, we create all possible skill-account pairs, where the skill belongs to the list of user's LinkedIn skills and the account belongs to the list of user's Twitter accounts.

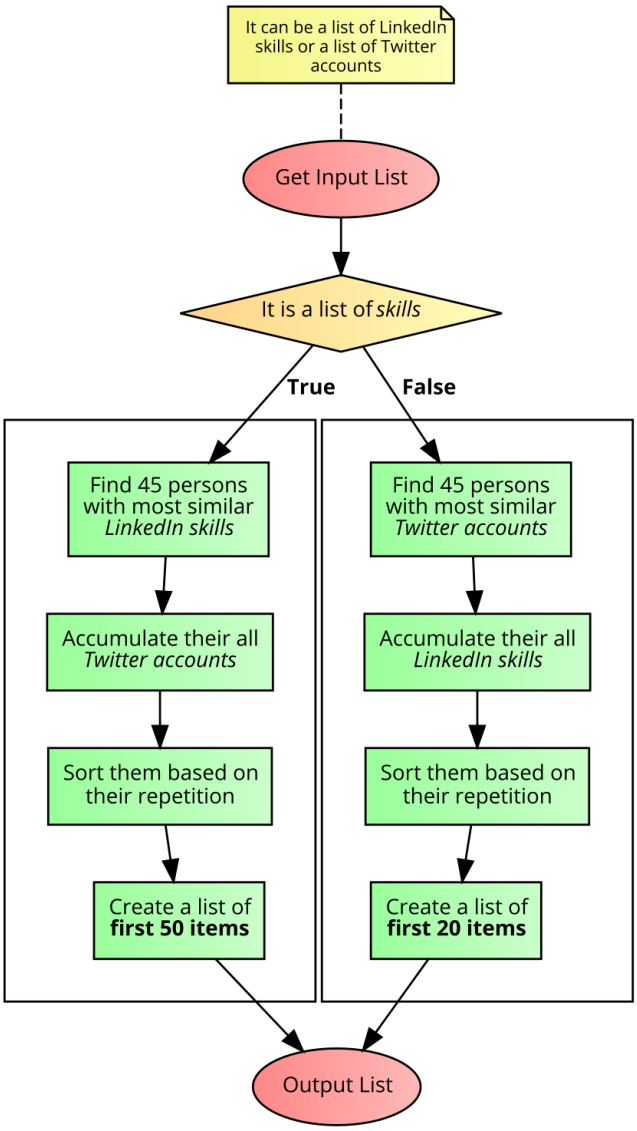

Figure 4.4: Flowchart of the knowledge-based collaborative algorithm

Then we aggregate all pairs related to all users of the train set, count each available pair in the obtained list of pairs, and sort pairs based on their frequency. As a result, by looking to this sorted list we can find which Twitter accounts have a tighter relationship with one specific LinkedIn skill and vice versa. Figure 4.5 shows the flowchart of this step. It should be noted that this step is done only once.

To recommend some Twitter accounts to follow, or predict some LinkedIn skills, we use the acquired weighted list of pairs of the LinkedIn skill and the Twitter following accounts. This list abstracts the features of the items in our train set and explains the relations be-

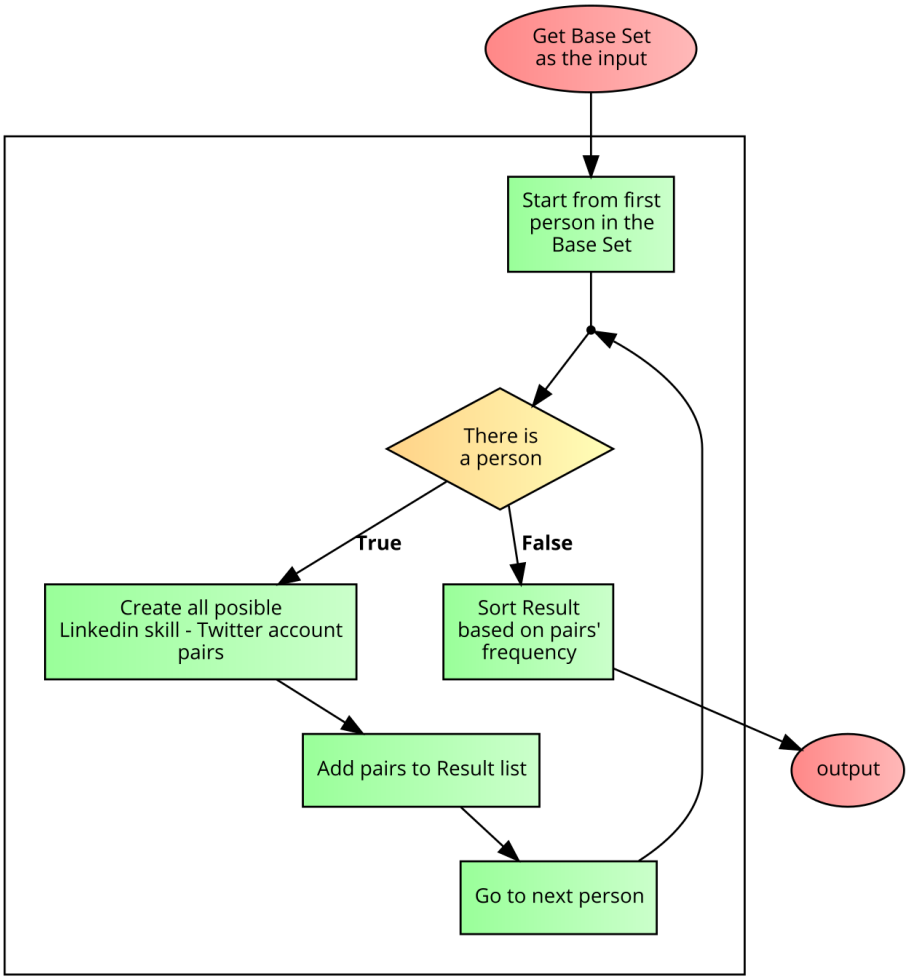

Figure 4.5: Flowchart of creating a description for each item in shape of (LinkedIn skill, Twitter account)

tween LinkedIn skills and Twitter accounts. Therefore, we can consider it to determine the preferences of any particular user.

In the content-based recommender system, we get the input list which can be either a list of LinkedIn skills or a list of Twitter accounts. If it involves some skills, for each skill S in this list we find 50 pairs in the list of weighted pairs, obtained in the first step, where their first component is S. In fact, these 50 pairs represent the 50 Twitter accounts that have the most influence on skill S. Then we accumulate all effective Twitter accounts related to all input skills and sort them based on their frequency. The same process is used when the input list includes some twitter accounts; nut, in this case, we find 20 pairs for each unique Twitter account and recommend 20 LinkedIn skills at the end. The flowchart of the content-based

recommender system is shown in Figure 4.6.

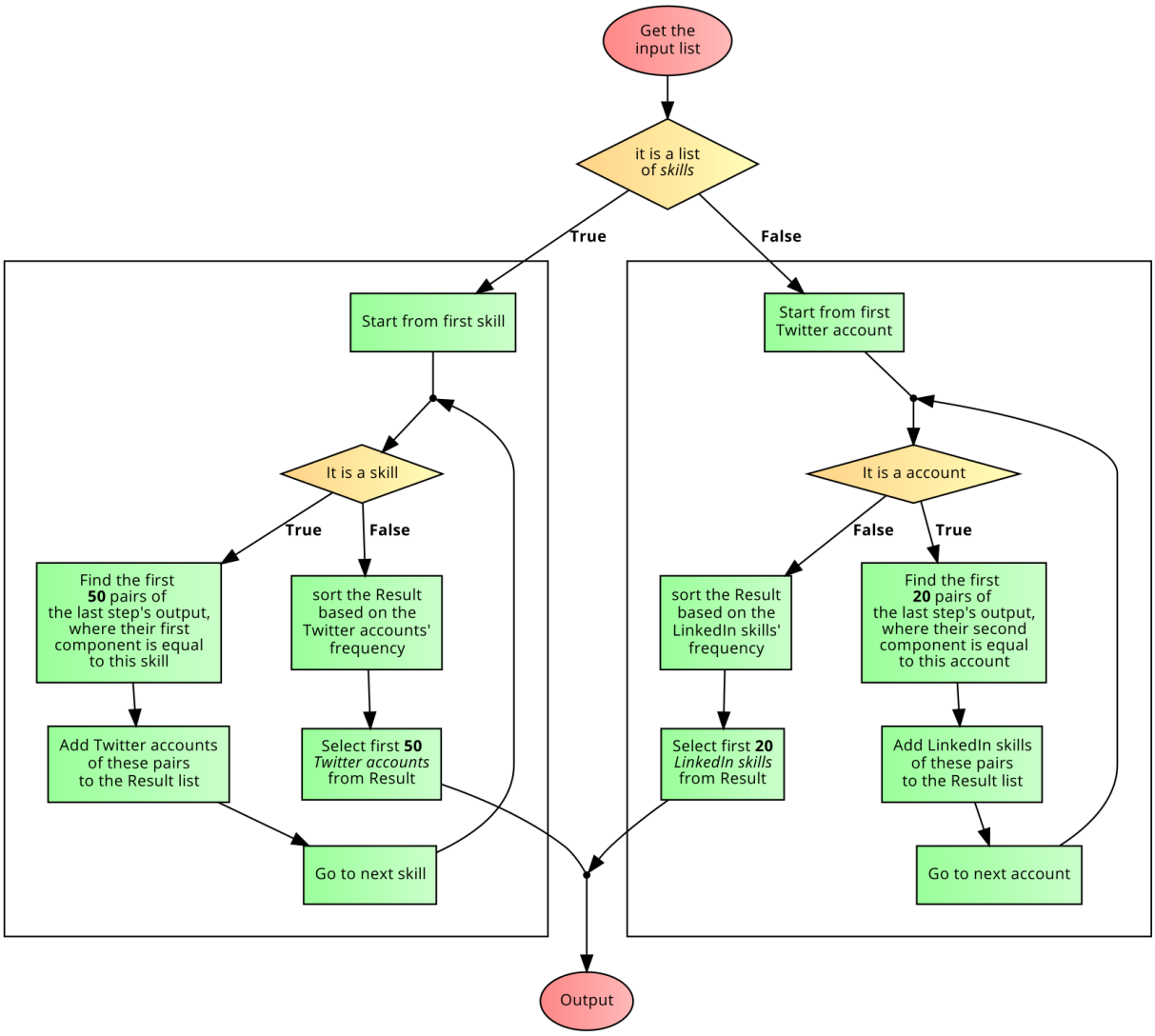

Figure 4.6: Flowchart of the knowledge-based content-based algorithm

4.2.3 Hybrid mix recommender system

The third algorithm, which is a hybrid recommender system, is the union of two previous algorithms. It aggregates the results of collaborative and content-based approaches, sort recommended items again based on their repetition, and suggests 50 Twitter account or 20 LinkedIn skills at the end, depending on the type of input list.

4.2.4 Hybrid feature combination recommender system

The fourth knowledge-based algorithm has some features of the collaborative filtering and some features of the content-based approach method. Like the first one, for any input list, it finds top 10 percent of similar users in the source domain who have more commons with the input list. Afterward, considering only these similar users, it creates a weighted list of possible pairs of LinkedIn skills and Twitter accounts. Then, it follows the content-based algorithm to create the final output. Therefore, this hybrid algorithm utilizes a feature of the collaborative filtering and uses it to modify a feature of the content-based recommender system.

4.3 Part II: Recommender System based on Coupled Matrix Factorization

As discussed in part 3.1.3, one type of information flow in cross-domain recommender systems is the knowledge transmission. A coupled NMF-based recommender system is one example for this type of knowledge flow, where latent features of user-item matrices in different domains are shared together in order to capture the influences of all available features on the users' preferences.

4.3.1 k-modes pre-clustering

One of the main weakness of recommender systems is their inability to work with sparse data. The sparsity may be defined slightly different in various studies. Sometimes it refers to having a few numbers of high-value ratings in the user-item matrix but most of the time it means few available ratings in comparison to all possible rating cases. Today, according to exponential growth in the number of users and also the number of available features of social media, the problem of data sparsity is more tangible. Data sparsity negatively affects the accuracy of the recommender system and increase the cold-start problem. Therefore, many researchers try to address this issue by using diverse methods [91, 92, 93].

Table 4.2 shows the characteristics of sparse information in our separate domains, Twitter domain and LinkedIn domain. It should be noted again, that we consider only those Twitter accounts which are followed by at least five users. As can be seen in this table, the corresponding matrices are large and highly sparse. Developing a cross-domain recommender system based on these user-item matrices, by using the coupled matrix factorization or any other technique, will not produce highly accurate recommendations.

Table 4.2: Sparsity characteristics of the original matrices

Description	**Value**
Number of all elements in the user - twitter account matrix	1715112
Percentage of non-zero elements in the user - twitter account matrix	% 2.53
Number of all elements in the user - skill matrix	1262472
Percentage of non-zero elements in the user-skill matrix	% 1.23

In order to reduce the sparsity problem, we perform clustering on LinkedIn skills as well as on Twitter accounts. Then we apply the coupled matrix factorization (or tensor decomposition) into the reconstructed clustered user-item matrices.

One of the widely used clustering methods is the k-means algorithm [94]. It allows dividing the numerical data in k clusters according to the similarities among them. In order to specify these clusters, the k-means algorithm starts by initializing the centroids of the clusters. Each centroid can be either one of the data points or even an imaginary point within the data range. Then, it computes the differences of each point to those centroids. There are different metrics to measure these differences but the most popular one is the Euclidian distance. Each

data point is assigned to the cluster it is closer to. After that, centroids are redefined to be at the center of the clusters. The distances are calculated again and the data points are re-assigned to the new centroids. This procedure is continued for a specific number of iterations or until the centroids do not move any more ore the sum of centroids displacements is under a certain small threshold. Symeonidis [85] utilizes various clustering methods, such as the k-mode algorithm, to reduce the sparsity of the user-tag matrix before using it to construct a 3-way tensor and create a recommender system by decomposing the tensor. This study is one example of how a wise pre-clustering can increases the accuracy of the recommender system.

As it said, the k-means stands on a mathematical calculation like measuring the Euclidian distance between data points and clusters' centroids or calculating the mean of data points in a cluster to move its' centroid. However, in many cases, like our research study, we are dealing with categorical data. Converting the categorical data into numerical data using label encoding or one-hot encoding and applying the k-means clustering is not appropriate because it could consider close two really different items and results low-quality clusters.

In this study, we implement the k-modes clustering algorithm. This technique which was introduced by Huang [95, 96], is an extension of the k-means. The main differences between these two include distance function and centroids representation [97].

In contrast to the k-means which calculates the Euclidian distances, the k-modes algorithms measures the dissimilarity between the object X and the centroid of a cluster Z described by m categorical attributes, as follow:

$$dissim(X, Z) = \sum_{j=1}^{m} \delta(x_j, z_j) \tag{4.1}$$

where

$$\delta(x_j, z_j) = \begin{cases} 0 & if \quad x_j = z_j \\ 1 & if \quad x_j \neq z_j \end{cases} \tag{4.2}$$

In addition, in k-modes clustering, the centroids are identified by vectors of modes of categorical attributes. the mode vector of a cluster centroids is selected such that it minimizes the sum of the dissimilarities between available objects in that cluster and the centroid.

We apply the k-modes clustering algorithm on LinkedIn skills and Twitter following accounts and divide them into k_l and k_t clusters respectively, which k_l is equal to the one percent of the number of all LinkedIn skills, and similarly k_t is equal to the one percent of the number of Twitter accounts following by at least five users. The clustered user-skill and user-account matrices are created by using these clusters. Table 4.3 shows the sparsity characteristics of these clustered matrices.

Table 4.3: Sparsity characteristics of the clustered matrices

Description	**Value**
Number of all elements in the clustered user - twitter account matrix	17220
Percentage of non-zero elements in the clustered user - twitter account matrix	% 46.58
Number of all elements in the clustered user - skill matrix	12792
Percentage of non-zero elements in the clustered user-skill matrix	% 47.74

By comparing Tables 4.2 and 4.3, it found that the pre-clustering using the k-modes reduces the inherent sparsity of the extracted data from both domains. Next step, is using this clustered matrices to create recommender systems.

4.3.2 Problem formulation

Although using the coupled matrix factorization in recommender systems have become popular in last few years in order to deal with the data sparsity, cold-start, and new user problems as well as increasing the accuracy of recommender systems but it has a long presence in some other fields. Acar et al. [50] use the equation 3.18 to analyze metabolomic information from

two distinct domains. Our suggested method is based on their work.

We have two different domains: Twitter and LinkedIn. In each domain, we have a nonnegative user-item matrix: user-account in the Twitter domain and user-skill in the LinkedIn domain. These two matrices have one mode in common; in other words, they are describing different features of identical users. Assume user-account matrix as $A \in \mathbb{R}^{m \times n}$ and user-skill matrix as $B \in \mathbb{R}^{m \times q}$, we are using a coupled nonnegative matrix factorization technique to calculate lower-rank matrices of $U \in \mathbb{R}^{m \times r}$, $V_A \in \mathbb{R}^{n \times r}$, and $V_B \in \mathbb{R}^{q \times r}$ such that $A \simeq \acute{A} = UV_A^T$ and $B \simeq \acute{B} = UV_B^T$. The $\acute{A}$ and $\acute{B}$ are sharing the matrix U; this matrix represents the latent features of the users. On the other hand matrices, V_A and V_B express the latent features of Twitter following accounts and LinkedIn skills respectively. Our purpose is to calculating the U, V_A, V_B in order to minimize the difference between the original user-item matrices, A and B, and the reconstructed matrices, $\acute{A}$ and $\acute{B}$, ($A - \acute{A}$ and $B - \acute{B}$). Therefore considering the following equation:

$$F(A, B, \acute{A}, \acute{B}) = \frac{1}{2} \parallel A - \acute{A} \parallel_F^2 + \frac{1}{2} \parallel B - \acute{B} \parallel_F^2 + R \tag{4.3}$$

$$F(A, B, U, V_A, V_B) = \frac{1}{2} \left(\parallel A - UV_A^T \parallel_F^2 + \parallel B - UV_B^T \parallel_F^2 \right) + R(U, V_A, V_B) \tag{4.4}$$

we find the answer of this:

$$\min_{U, V_A, V_B} \frac{1}{2} \left(\parallel A - UV_A^T \parallel_F^2 + \parallel B - UV_B^T \parallel_F^2 \right) + R(U, V_A, V_B) \tag{4.5}$$

where, $R(U, V_A, V_B)$ is the Lasso regularization [98] terms:

$$R(U, V_A, V_B) = \lambda \left(\parallel U \parallel_F^2 + \parallel V_A \parallel_F^2 + \parallel V_B \parallel_F^2 \right) \tag{4.6}$$

Here, we use a SGD based [99] algorithm for solving the equation 4.5. First, we calculate

the partial differentiations of the cost function (equation 4.4):

$$\frac{\partial F}{\partial U} = (A - UV_A^T)(-V_A) + (B - UV_B^T)(-V_B) + 2\lambda U$$

$$\frac{\partial F}{\partial V_A} = (A - UV_A^T)^T(-U) + 2\lambda V_A \tag{4.7}$$

$$\frac{\partial F}{\partial V_B} = (B - UV_B^T)^T(-U) + 2\lambda V_B$$

Lets $X = UV_A^T$ and $Y = UV_B^T$, then:

$$\frac{\partial F}{\partial U} = (X - A)U + (Y - B)V_B + 2\lambda U$$

$$\frac{\partial F}{\partial V_A} = (X - A)^T U + 2\lambda V_A \tag{4.8}$$

$$\frac{\partial F}{\partial V_B} = (Y - B)^T U + 2\lambda V_B$$

Then, we iterate over the data and update the following equations using one randomly piece of data in each step, until the algorithm converges:

$$U := U - \alpha \frac{\partial F}{\partial U}$$

$$V_A := V_A - \alpha \frac{\partial F}{\partial V_A} \tag{4.9}$$

$$V_B := V_B - \alpha \frac{\partial F}{\partial V_B}$$

where, α is a scalar, typically between 0.00001 to 1, and it is called the step size or the learning rate.

4.3.3 Handling missing values

One important matter is how to handle the missing values in the LinkedIn's user-skill and Twitter's user-account matrices, First of all, we want to design a general solution which can

be used to create a cross-domain recommender system and real-world rating matrices usually include a number of missing values. The method we choose to replace these missing values is more important if we have a significant number of missing values in our dataset. In addition, to evaluate our proposed models, we want to randomly remove some elements from the rating matrices and see how well those models can predict the missing values.

There are many different ways to deal with the missing values and it really depends on the nature of data, the number of missing values and the knowledge we have about the dataset. One possible method is dropping the data entity or even the feature. Here, we avoid to do this because, as it shown before, the sparsity of the data is reduced using the pre-clustering and the chance to have a lot of missing values in one row or one column of the clustered matrices is very low. Another popular approach is replacing the missing values with zeros. However, a zero element in the user-skill matrix, A_{ij} means that the i-th user does not have the j-th skill; and also, if the element B_{ik} of the user-account matrix is zero, it means that the i-th user does not follow the k-th twitter account. Therefore, since zeros have meaning in our matrices, it seems that replacing missing values with zeros is not a good idea.

So, we need to replace possible missing values with some numbers except zero. We can consider the average of all the present values in the matrix, the average of all available values in the corresponding row, or the average of all the present values in the related column. Our purpose in this section is using the coupled NMF to develop a cross-domain recommender system; therefore, we want to know which case is more appropriate to replace missing values before applying the NMF. To determine which case works best for our data and with the NMF, we randomly remove elements from the clustered user-skill and clustered user-account matrices. Then, the NMF technique is used to predict those missing values of each domain separately. We use the Mean Squared Error (MSE) as the measure to see how close the predicted matrices are to the actual matrices.

$$MSE = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (X_{ij} - \acute{X}_{ij}) \tag{4.10}$$

Tables 4.7 and 4.8 show the obtained results. According to these tables, we will use the average of non-missing values in each column to replace the missing value in that column.

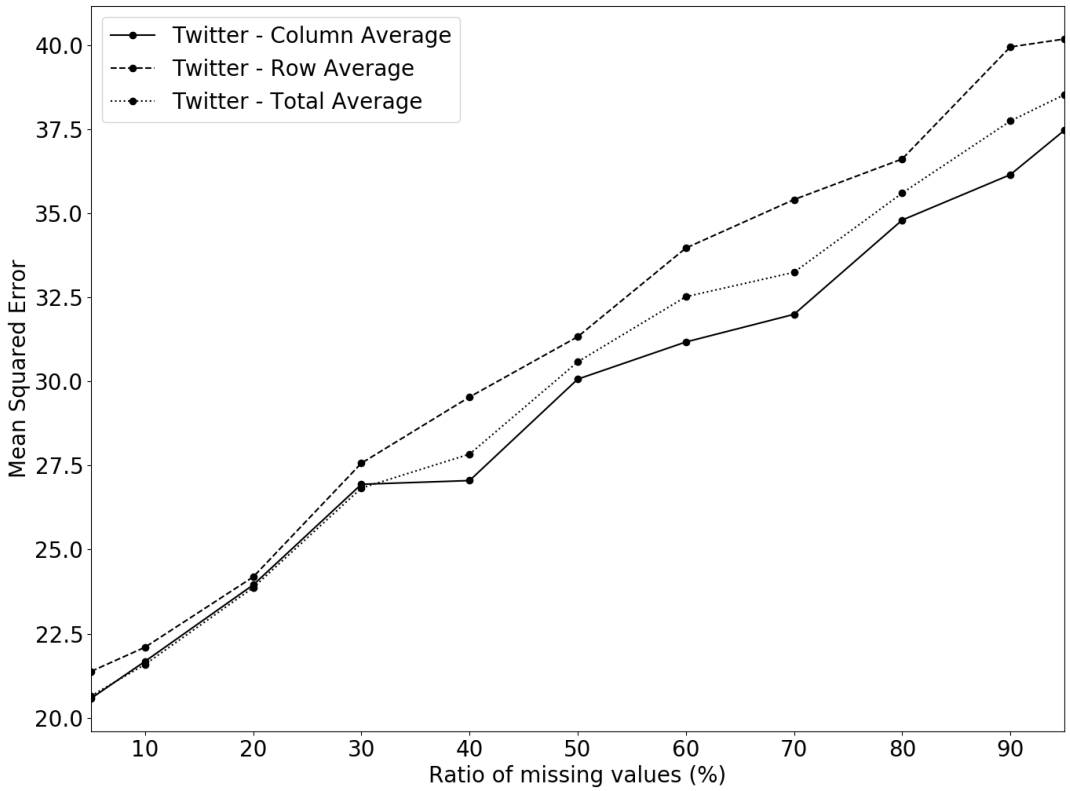

Figure 4.7: Difference between actual user-account matrix and its corresponding predicted matrix using NMF

Each column in the clustered user-skill matrix represents a unique cluster of LinkedIn's skills; likewise, each column of the clustered user-account indicates one specific cluster of Twitter's following account.

4.3.4 Selecting the factorization rank

Assume that k is the factorization rank in the coupled NMF. By other means, if $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{mtimesq}$, then by applying the coupled NMF we have $A \simeq UV_A^T$ and $B \simeq UV_B^T$ where $U \in \mathbb{R}^{m \times k}$, $V_A \in \mathbb{R}^{n \times k}$ and $V_B \in \mathbb{R}^{q \times k}$. The k is a hyperparameter whose its' value needs to be set before the learning process begins. Here, we consider different possible values for k: 25, 50, 100, 200, and 400. For each of these values, we determine the performance of the recommender system in term of precision.

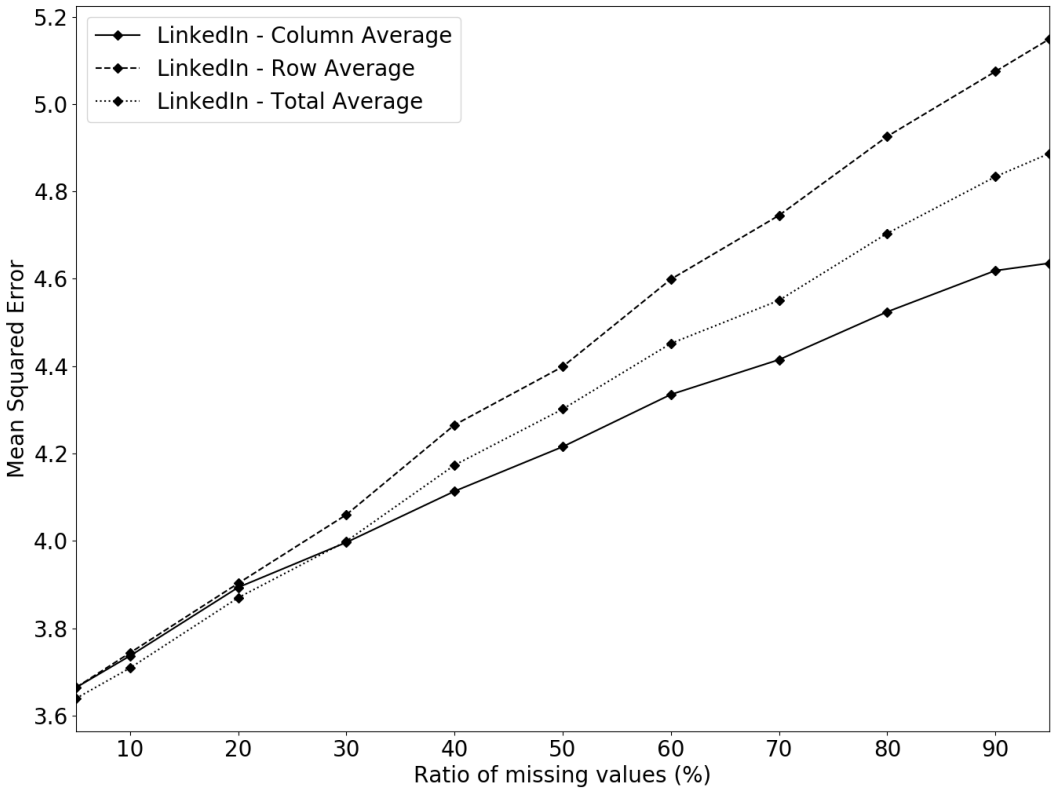

Figure 4.8: Difference between actual user-skill matrix and its corresponding predicted matrix using NMF

Consider one of the domains as the source domain and another as the target domain. We randomly replace some of the elements of the target domain with missing values. The missing values are handled according to the part 4.3.3. Then, coupled NMF-based recommender systems are produced to recommend top 5 items to each user in the target domain. The only difference between these recommender systems is the value of the factorization rank, k. The performances of created recommender systems are compared based on their precision.

Here, the precision of the recommender system, to suggest items to one particular user, is defined as the number of elements in the intersection of recommended items and actual items, divided by the number of recommended items. Hence, the total precision of the recommender

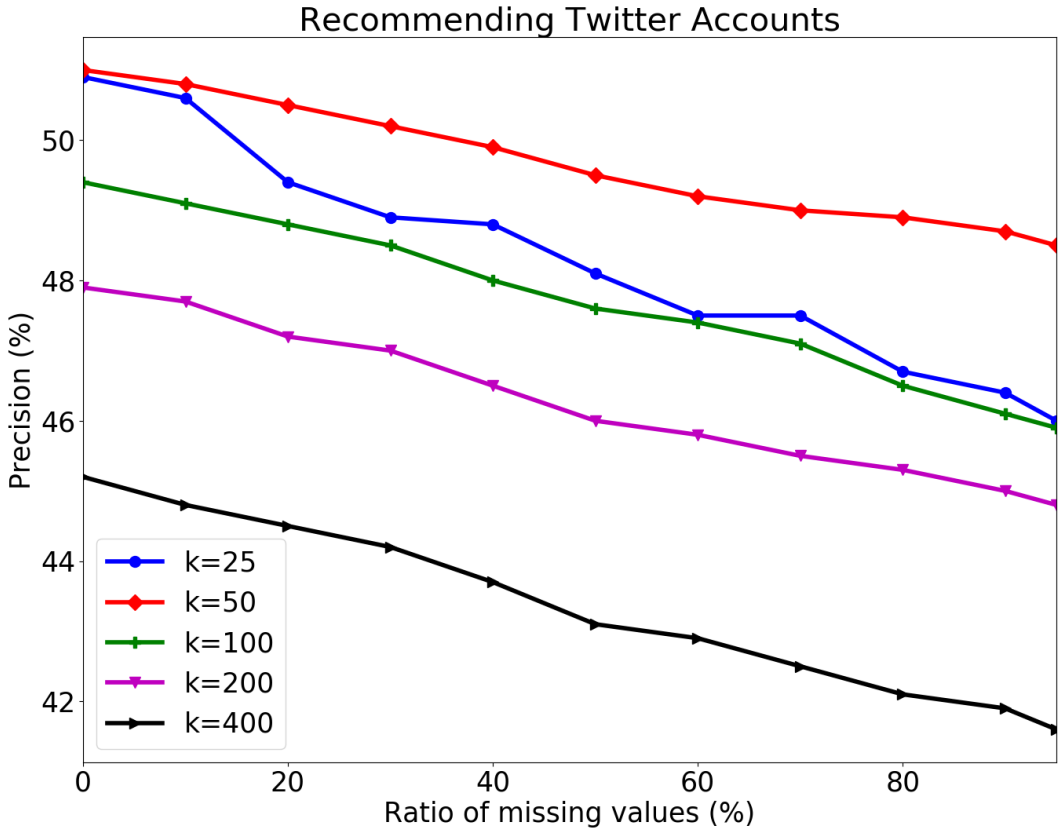

Figure 4.9: Comparison between coupled NMF-based recommender systems with different factorization rank; source domain is the user-skill matrix and target domain is the user-account matrix.

system is calculated as the mean of its precisions for all users.

$$precision = \frac{|recommended \quad items \quad \cap \quad actual \quad items|}{|recommended \quad items|} \tag{4.11}$$

We also consider this constraint that $|recommended\ items| = |selected\ actual\ items|$. In other words, if the number of values greater than zero for the user in the reconstructed matrix using coupled NMF is n and the number of desired top item to recommend is N and $n < N$, then $|recommended\ items| = |selected\ actual\ items| = n$; Otherwise if $n \geq N$, then $|recommended\ items| = |selected\ actual\ items| = N$.

Figure 4.9 illustrates the precision of recommender systems when the LinkedIn dataset is the source domain and Twitter dataset is the target domain. Here, we used recommender

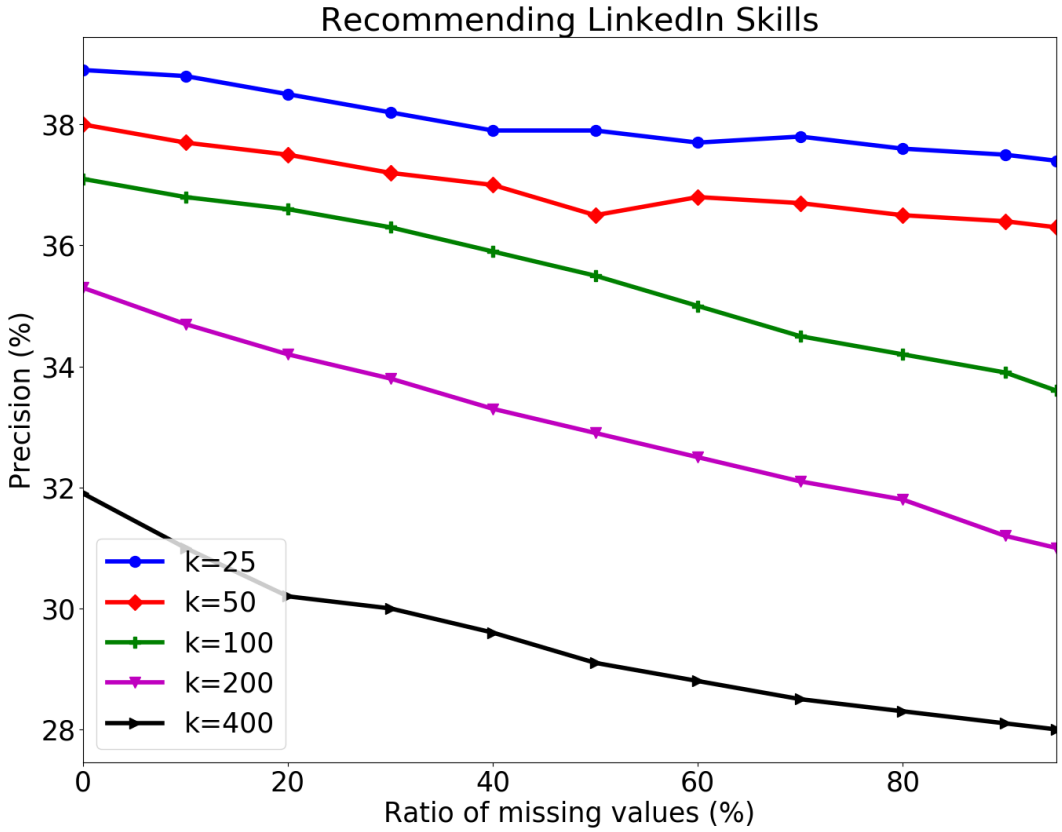

Figure 4.10: Comparison between coupled NMF-based recommender systems with different factorization rank; source domain is the user-account matrix and target domain is the user-skill matrix.

systems based on coupled NMF to recommend top 5 twitter accounts. According to this table, the recommender system with the factorization rank of 50 shows the best performance.

Considering Twitter's user-account matrix as the source domain and LinkedIn's user-skill matrix as the target domain, Figure 4.10 shows the performance of recommender systems to predict the skills of the users. As can be seen in this figure, the recommender system with factorization rank of 25 has the best precision.

4.3.5 Selecting the number of items to recommend

Another parameter which can affect the performance of the coupled NMF-based recommender system is the number of items it recommends to each particular user.

Like the scenario of part 4.3.4, assume one of the domains as the source domain and another one as the target domain. Considering different ratios of missing values in the target domain, we create coupled NMF-based recommender system, with proper factorization rank, to suggest N items in the target domain. As it obtained before, the factorization rank to recommend Twitter's accounts to follow is $k = 50$ and the considered factorization to suggest LinkedIn's skills to add to the profile is $k = 25$. In addition, we deal with missing values based on the result of part 4.3.3.

We compare different numbers of recommended items, N: 3, 5, 10, 15, 20. If the number of values greater than zero for a user, n, is less than N, the n is considered to calculate the precision. According to Figures 4.11 and 4.12 the best numbers of Twitter accounts and LinkedIn skills to recommend are 10 and 15 respectively.

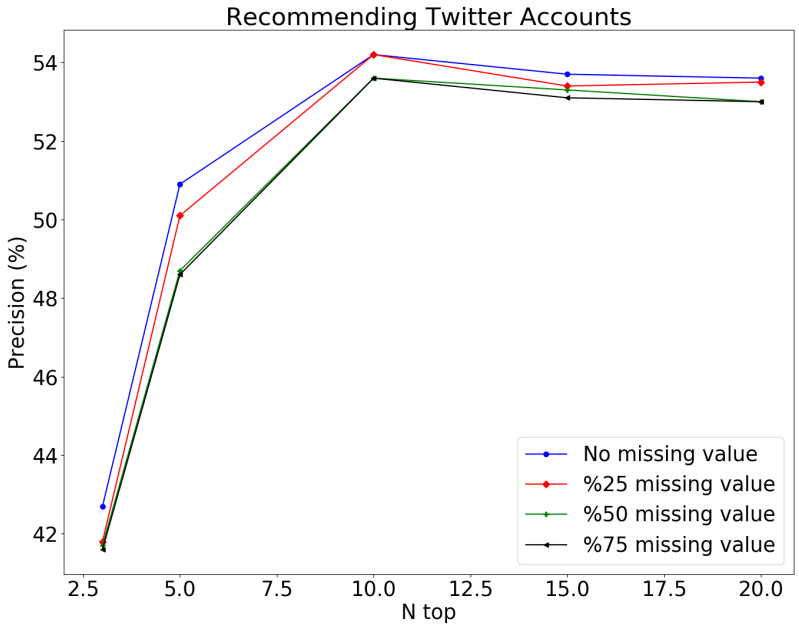

Figure 4.11: Coupled NMF-based recommender systems to recommend N items; source domain is the user-skill matrix and target domain is the user-account matrix.

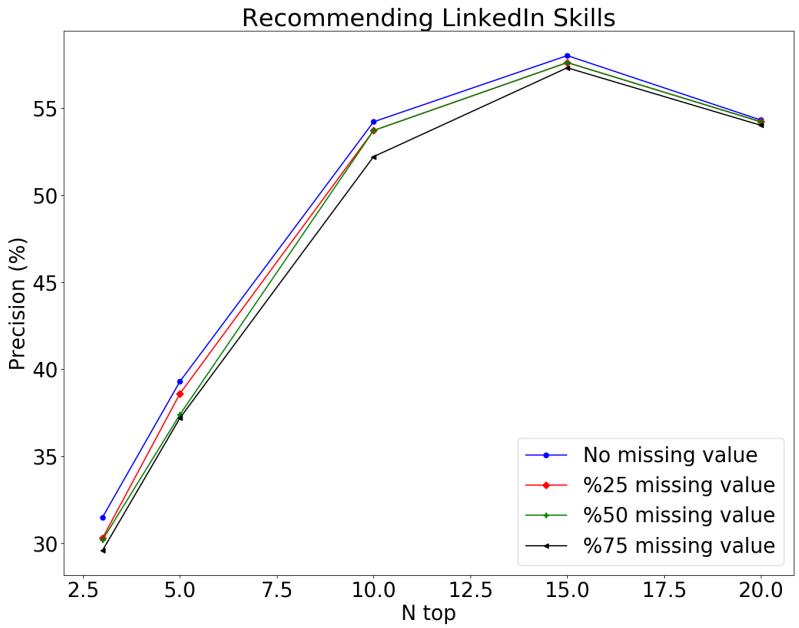

Figure 4.12: Coupled NMF-based recommender systems to recommend N items; source domain is the user-account matrix and target domain is the user-skill matrix.

4.4 Part III: Recommender System based on Tensor Decomposition

One another case of knowledge flow in cross-domain recommender systems is knowledge aggregation (as discussed in the part 3.1.3). The tensor decomposition-based recommender system is an example of knowledge aggregation, where users' preferences in different domains are merged together in order to create more accurate recommendations as well as to deal with data sparsity and cold-start problems.

In this section, we first construct a 3-way tensor of user-account-skill by combining the available information in separate domains. Then, considering one of the domains as the target domain and by using a CP-style decomposition, we design and implement a cross-domain recommender system to suggest items in the target domain.

4.4.1 Problem formulation

k-modes pre-clustering

As discussed before the original user-item matrices in our diverse domains are very large and highly sparse. Creating a 3-way tensor using these matrices results in a huge tensor with even worse sparsity status. Because of the large dimensionality of the original tensor's orders, calculating the decomposition of the tensor and developing the recommender system base on it needs a lot of computational power and time. In addition, the intense sparsity status of this cube of numbers leads to low accurate recommendations. Therefore, similar to part 4.3.1 we apply a k-modes clustering on the original user-skill and user-account matrices; and then, utilize these clustered matrices to construct the clustered tensor. Table 4.4 compares the sparsity properties of the original tensor and clustered tensor together.

Table 4.4: Sparsity properties of original and clustered tensors

Description	**Value**
Number of all elements in the original tensor	4400977392
Percentage of non-zero elements in the original tensor	% 0.03
Number of all elements in the clustered tensor	447720
Percentage of non-zero elements in the clustered tensor	%22.81

Tensor construction

A 3-way tensor $\mathcal{T}$ in created using information in various domains. If $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{m \times q}$ are clustered user-item matrices in two domains, then the constructed tensor is of the shape of $\mathcal{T} \in \mathbb{R}^{m \times n \times q}$. A schematic view of the tensor is showed in Figure 4.13.

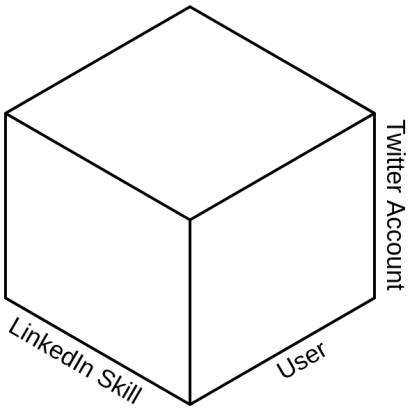

Figure 4.13: Schematic view of the constructed tensor

Handling missing values

We want to implement the CP decomposition technique to factorize the clustered tensor of user-account-skill in order to use the reconstructed tensor to suggest Twitter's following accounts to follow or recommend LinkedIn skills to add.

Assume that the Twitter domain is our target domain; therefore, we can look at the clustered tensor $\mathcal{T}$ as a list of user-account matrices. Each user-account matrix is related to a unique skill. In other words, for each particular skill, the corresponding user-account matrix explains the relationships between users and Twitter accounts for those users who have that particular skill. Similarly, if the LinkedIn domain is the target domain, the tensor $\mathcal{T}$ can be considered as a list of user-skill matrices. the i-th matrix represents that which skills are claimed by users who follow the i-th Twitter account$Account_i$. Figure 4.14 helps to understand these different perspectives. It is important that the two tensors displayed in this figure are mathematically identical; we just looking at a unique tensor from different points of view.

To handle missing values in the clustered tensor, we use our finding in the part 4.3.3. Therefore, if we are going to use the CP decomposition to recommend some Twitter accounts, we replace a missing value with its column's average in the corresponding user-account matrix. For example, if the element $\mathcal{T}_{ijk}$ in the user-account-skill tensor is a missing value, it is replaced by the average of j-th column in the k-th user-account matrix. Similarly, if the LinkedIn domain is the target, a missing value in the j-th matrix is replaced by the average of k-th

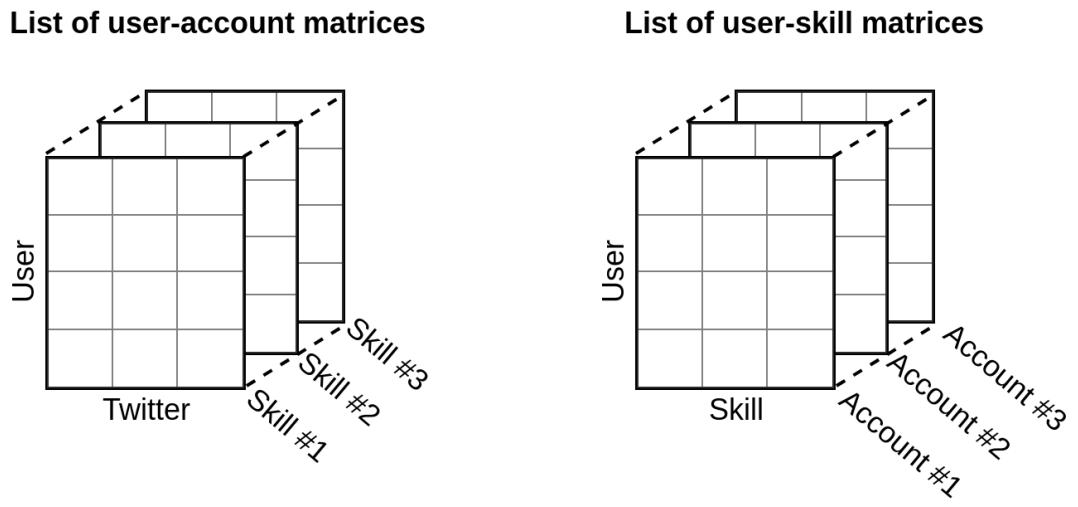

Figure 4.14: The 3-way tensor can be thought as either list of user-account matrices or list of user-skill matrices

column.

Applying CP decomposition

When the tensor is constructed, a nonnegative CP decomposition via alternating least squares is applied on it [100]. Figure 4.15 shows a schematic view of the CP decomposition of a 3-way tensor. The recreated tensor $\acute{\mathcal{T}}$ is used to suggest new recommendations. Assume $\mathcal{T} \in \mathbb{R}^{m \times n \times q}$ as our clustered tensor. According to the CP decomposition, which is discussed in part 3.3.2, we have:

$$\mathcal{T} \simeq \acute{\mathcal{T}} = \sum_{i=1}^{K} a_i \otimes b_i \otimes c_i \tag{4.12}$$

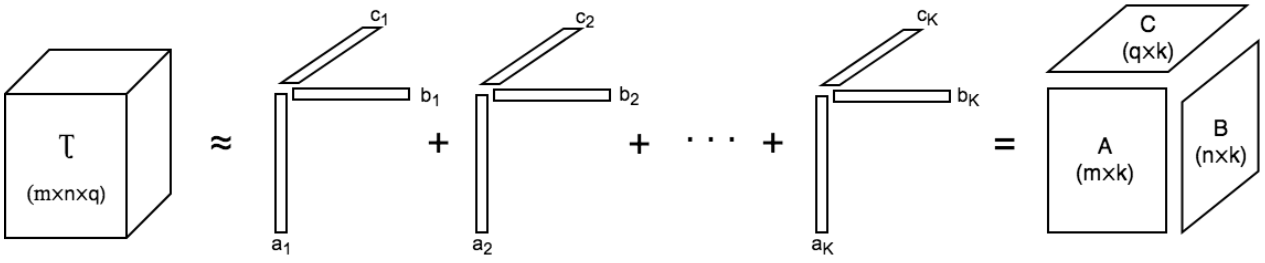

Figure 4.15: CP decomposition of a 3-way tensor

4.4.2 Selecting the number of components

As shown in Equation 4.12 the CP decomposition factorize the 3-way tensor $\mathcal{T}$ into the sum of K component rank-one tensors. The number of components should be selected before applying the CP.

To evaluate the effect of the number of rank-one components on the performance of the final recommender system, we consider various numbers of Ks : 25, 50, 100, 200, 400. In addition, different ratios of missing values are considered in the target domain before constructing the clustered tensor. For each case, we apply the nonnegative CP decomposition-based recommender system to suggest the top 5 items in the target domain. Assume that the Twitter is the target domain and the reconstructed tensor is $\acute{\mathcal{T}} \in \mathbb{R}^{\Updownarrow \times \backslash \times \amalg}$; In order to recommend some twitter accounts to the user i, $i \leq m$, we calculate the following score for each twitter account j, $j \leq n$, in the $\acute{\mathcal{T}}$:

$$score(account_j) \;=\; \frac{1}{q} \sum_{k=1}^{q} \acute{\mathcal{T}}_{ijk} \tag{4.13}$$

Then, the N top score twitter accounts are selected to recommend to the user i. Figure 4.16 displays the content of the Equation 4.13. Likewise, in order to recommend LinkedIn skills to user i, the following score is calculated for each skill k, $k \leq q$:

$$score(skill_k) \;=\; \frac{1}{n} \sum_{j=1}^{n} \acute{\mathcal{T}}_{ijk} \tag{4.14}$$

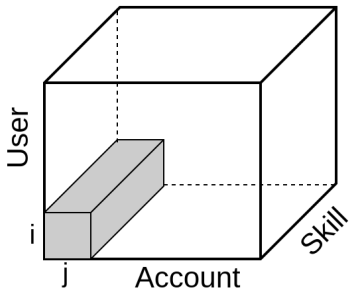

Figure 4.16: Create a score for each item in the target domain using the reconstructed tensor

Obtained results are compared to each other using precision and F-score measures, where:

$$precision = \frac{|\,recommended \quad items \quad \cap \quad actual \quad items\,|}{|\,recommended \quad items\,|} \tag{4.15}$$

$$recall = \frac{|\,recommended \quad items \quad \cap \quad actual \quad items\,|}{|\,actual \quad items\,|} \tag{4.16}$$

$$F\,score = 2\frac{precision.recall}{precision + recall} \tag{4.17}$$

As it can be seen in Figures 4.17 and 4.18 the best K number (number of decomposition components) between the investigated numbers is $K = 400$ when the target domain is Twitter. Likewise, when we want to predict LinkedIn skills, the recommender system with $K = 400$ shows better performance in comparison with other recommender systems which have different number of Ks (Figures 4.19 and 4.20).

4.4.3 Selecting the number of items to recommend

Similar to the last part, consider one of the distinct domains as the target domain. First, we randomly replace some elements of the user-item matrix in the target domain with missing values. Then using this matrix and the rating matrix from the source domain, we construct the 3-way tensor of user-account-skill. By applying the CP decomposition and calculating a score for each item in the target domain, we suggest top N items to each user. The precision of the recommended items for each particular user is determined by Equation 4.15. The precision of the recommender system is the average of these precisions.

The influence of the number of recommended items, N, on the precision of CP-based recommender system is investigated by testing various N values: 3, 5 10, 15, and 20. According to Figures 4.21 and 4.22, Although the precision of the recommender system decreases by increasing the number of recommended items in case of no missing value, however, it seems that the $N = 10$ is the best case when the target domain includes some missing values.

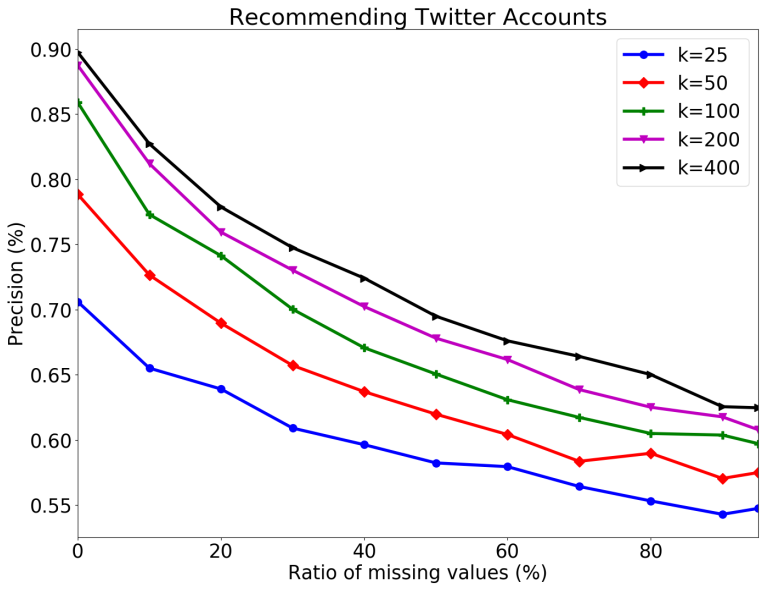

Figure 4.17: Comparison between CP decomposition-based recommender systems with different number of components, based on the precision. Twitter is the target domain.

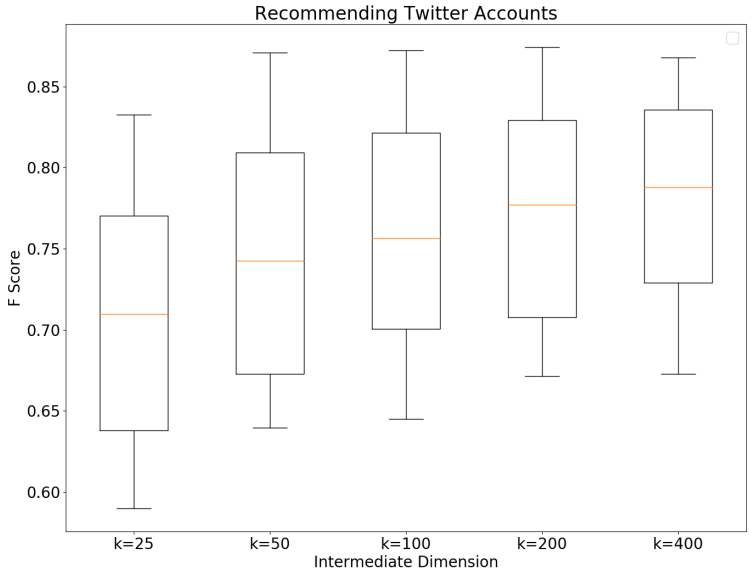

Figure 4.18: Comparison between CP decomposition-based recommender systems with different number of components, based on the F-score. Twitter is the target domain.

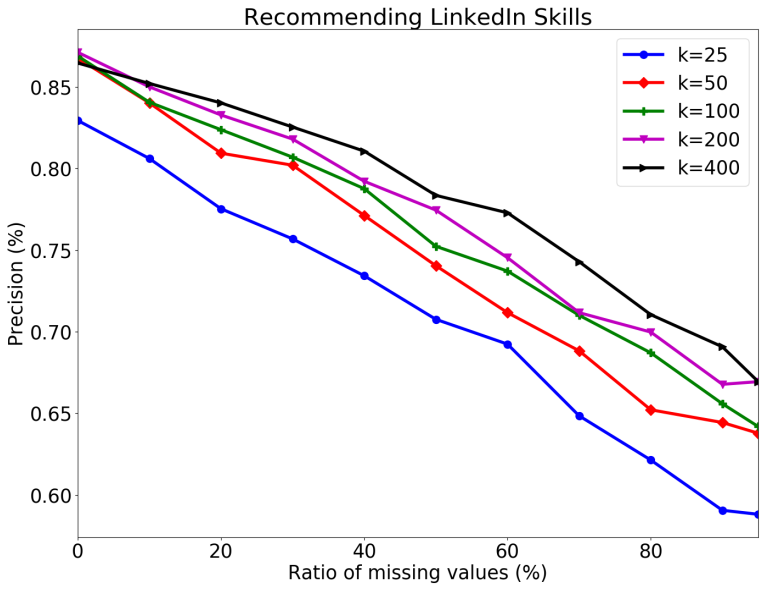

Figure 4.19: Comparison between CP decomposition-based recommender systems with different number of components, based on the precision. LinkedIn is the target domain.

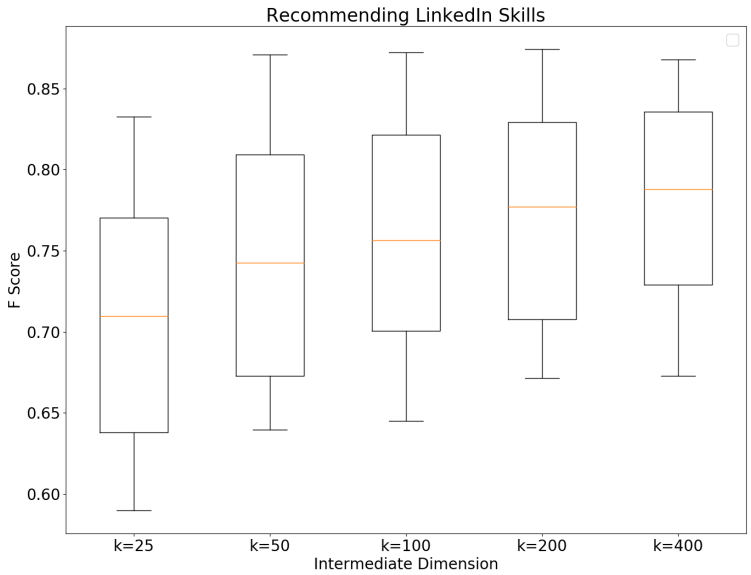

Figure 4.20: Comparison between CP decomposition-based recommender systems with different number of components, based on the F-score. LinkedIn is the target domain.

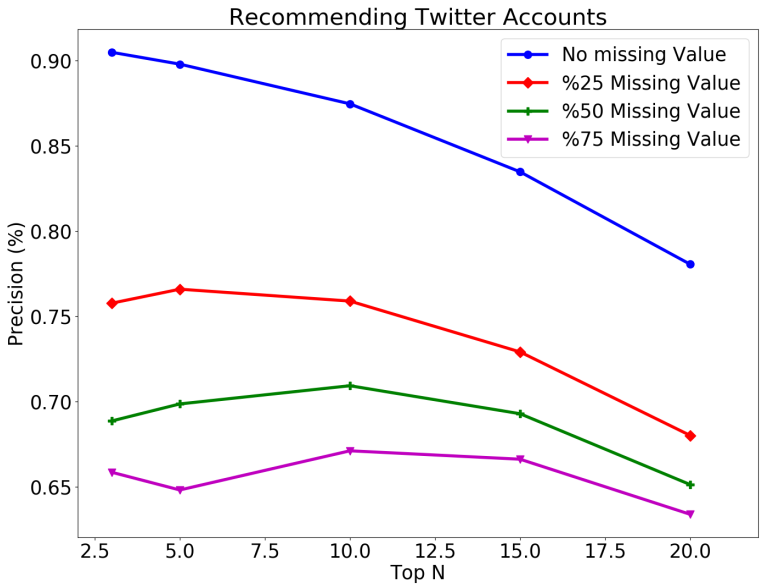

Figure 4.21: CP decomposition-based recommender system to recommend N items; target domain is the Twitter domain.

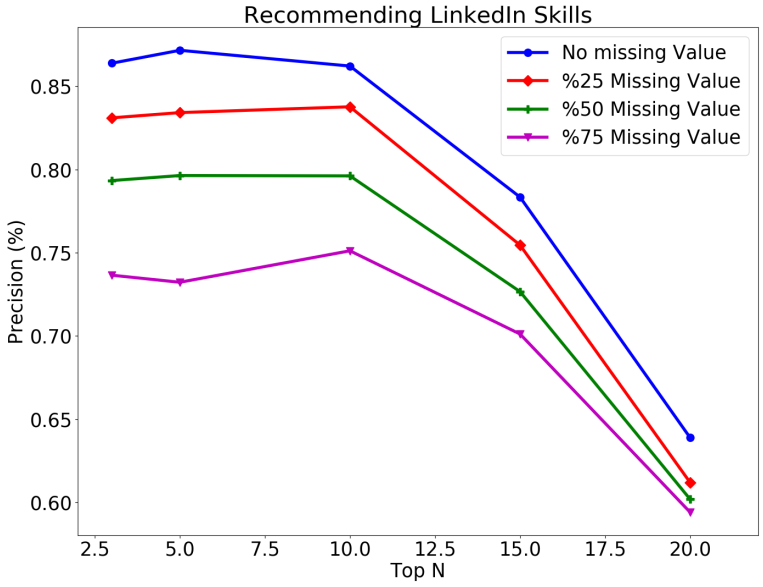

Figure 4.22: CP decomposition-based recommender system to recommend N items; target domain is the LinkedIn domain.

5
RESULTS AND DISCUSSION

In the previous chapter the ideas and the implementation methods of the knowledge-based, coupled NMF-based, and CP-based cross-domain recommender systems were explained. In addition, comprehensive information about two separate domains, the Twitter domain and the LinkedIn domain, were presented.

In the first section of this chapter, we start with the network analysis of different domains. In the second section, results of diverse knowledge-based algorithms, namely collaborative filtering, content-based, hybrid mix, and hybrid feature combination, are discussed and compared to each other. Finally, in the last section, the performance of coupled NMF-based and CP decomposition-based cross-domain recommender systems are compared together.

5.1 Network Analysis

Although the main purpose of this thesis is suggesting the discussed cross-domain recommender systems, but before investigating their results, we have a deeper look at our datasets and their characteristics in this section. In addition, the findings of this section will be used in the last chapter, to answer one of the appeared questions during studying the proposed techniques' results.

The graph of Twitter following accounts and the graph of LinkedIn skills are created using information in their related domains. Bearing in mind two sets of LinkedIn skills and Twitter accounts, the definitions of their corresponding graphs are almost similar, except that for the following accounts graph we just consider those accounts which are followed by at least five users; but, we consider all skills in order to model the LinkedIn skills graph.

In Twitter accounts graph, nodes are Twitter accounts and there is an edge between two

nodes if there is at least one user following both accounts. In addition, the weight of the edge connecting nodes T_A and T_B together is the reciprocal of the number of users who are following both corresponding accounts to these two nodes:

$$W_{AB} = \frac{1}{n_{AB}} \tag{5.1}$$

where, W_{AB} is the weight of edge between nodes T_A and T_B, and n_{AB} is the number of people who follow both related Twitter accounts. Again, both T_A and T_B are Twitter accounts followed by at least five users.

Likewise, in our LinkedIn skills graph, there is an edge between two different skills if someone has both of them (Figure 5.1). Also, the weight of the edge, $\acute{W}_{XY}$, is equal to the multiplicative reverse of the number of users who have both skill L_X and skill L_Y:

$$\acute{W}_{XY} = \frac{1}{\acute{n}_{XY}} \tag{5.2}$$

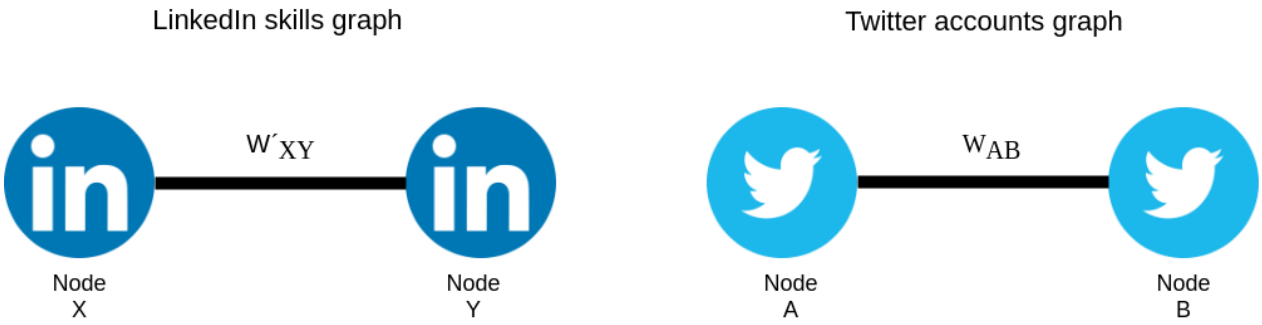

Figure 5.1: Schematic view of an edge in the created graphs

The statistical characteristics of these graphs are presented in Table 5.1. It should be noted that since these two networks are too dense, showing them, showing them here as graphs are not useful.

The distribution of degree centrality of two graphs are illustrated in Figures 5.2 and 5.3. As can be seen in these figures, degree distributions are quite different for these graphs.

As can be seen in Figure 5.2, in the LinkedIn skills graph most of the nodes, which are skills, have a low degree centrality and only a few of those have a centrality degree greater

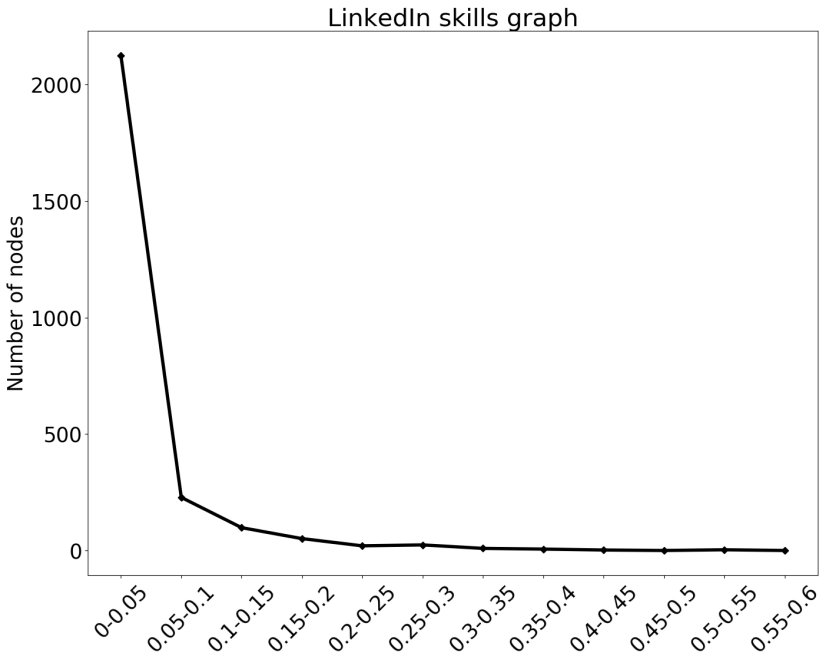

Figure 5.2: Degree centrality versus number of nodes in the LinkedIn skills graph

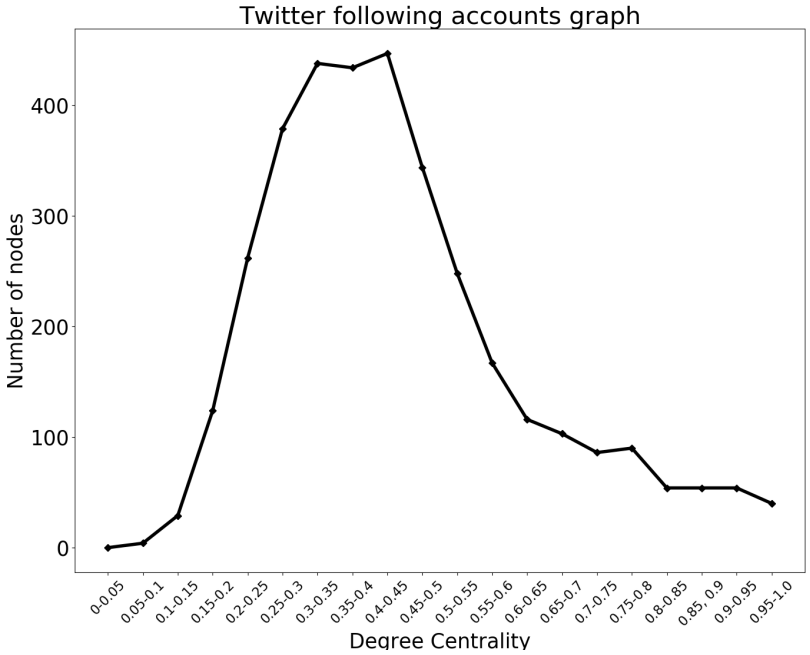

Figure 5.3: Degree centrality versus number of nodes in the Twitter accounts graph

Table 5.1: Number of nodes and edges in following accounts graph and skills graph

Description	Twitter following accounts graph	LinkedIn skills graph
Number of nodes	3486	2566
number of edges	2031172	1222238

than 0.35. It means that there are a few highly linked nodes, while the majority of nodes have a low number of links. In addition, if we consider the distribution of the clustering coefficient for this graph (Figure 5.4), we can see that the clustering coefficient decreases as the node degree centrality rises.

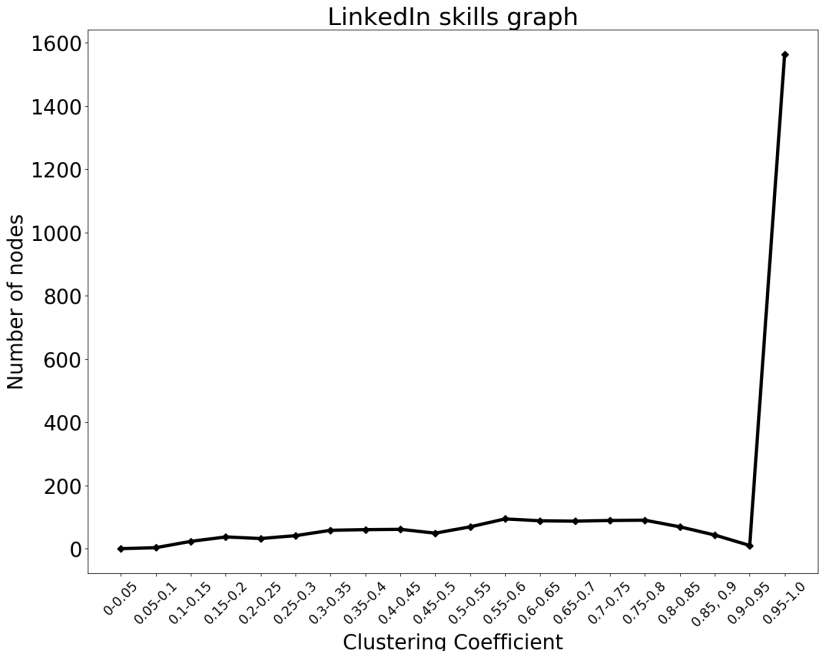

Figure 5.4: Clustering coefficient versus number of nodes in the LinkedIn skills graph

Therefore, this implies that there are some hubs with high degrees in the skills network which connect some sub-graphs to each other where these sub-graphs consist other low-degree nodes. These features belong to a scale-free network. So, we can consider the LinkedIn skills graph as a scale-free network. Table 5.2 shows some nodes in this network that have greater degree centrality, betweenness, or closeness rather than other nodes. Also, Table 5.3 states

the degree centrality and betweenness ranks of some popular programming languages. In contrast with LinkedIn skills, the distribution of degree centrality of the Twitter following account, shown in Figure 5.3, is not like a scale-free network because, in this network, most of the nodes have a medium degree.

Table 5.2: Important nodes in the LinkedIn skills graph

Name of skill	**Degree centrality rank**	**Betweenness rank**	**Closeness rank**
Leadership	1	1	1
Management	2	2	2
Strategy	3	3	3
Project Management	4	5	4
Social Media	5	4	5
Mobile Devices	6	6	6
Business Development	7	7	7
Product Management	8	13	8
Start-ups	9	9	9
Marketing	10	16	10
Cloud-computing	11	8	11
Java	16	10	16

5.2 Knowledge-based Recommender Systems

As said in section 4.2 we use the train-test-split technique to develop and evaluate our knowledge-based recommender systems. The test set includes information about 10 percent of users in both domains. We implement the proposed algorithms to recommend items

Table 5.3: Characteristics of some of the popular languages in the LinkedIn skills graph

Name of skill	Degree centrality rank	Betweenness rank
Java	16	10
JavaScript	21	17
Python	23	15
C++	30	23
SQL	37	25
C	41	27
PHP	45	36
Perl	91	65
C#	99	88
Ruby	175	207
Scala	325	357
Go	632	747
Erlang	962	923

in the target domain for these users; and the suggested items are compared to the actual items in order to determine the performance of the recommender system. The precision of a recommender system can be interpreted as the percentage of recommended options, produced by it, which were already chosen by the user. In addition, we measured the run-time of each knowledge-based recommender system using a computer with Intel Core i7-4765T 2.00GHz×8 processor and memory of 8GiB.

The statistic characteristics of the obtained results are presented in Tables 5.4, 5.5. There are two important points here; first the presented run times in these tables are calculated considering hundred runs' results, and second, the presented precisions show the performance

of these recommender systems under the new-user situation because we do not use any information about the user in the target domain to select recommended items. The obtained results show that the precision and time consumption of each model vary from person to person. Mainly, they depend on the selected target domain and number of available ratings in the source domain.

Table 5.4: Statistic results of the knowledge-based recommender systems for suggesting Twitter accounts

Algorithm	**Precision mean (%)**	**Precision s.d. (%)**	**Time mean (s)**	**Time s.d. (s)**
Collaborative filtering	8.86	6.48	10.6	6.61
Content-based	21.91	19.32	17.5	9.42
Hybrid mix	20.01	17.78	26.84	11.96
Hybrid feature combination	21.33	18.75	8.88	4.37

Table 5.5: Statistic results of knowledge-based recommender systems for suggesting LinkedIn skills

Algorithm	**Precision mean (%)**	**Precision s.d. (%)**	**Time mean (s)**	**Time s.d. (s)**
Collaborative filtering	25.02	15.83	1.37	1.93
Content-based	29.40	19.54	72.44	95.32
Hybrid mix	28.17	18.80	75.35	103.13
Hybrid feature combination	30.60	21.36	28.63	33.86

The precision of the hybrid mix and hybrid feature combination recommender systems are very close, however, using the feature combination algorithm decreases the run time by %67 and %62 on average for recommending Twitter following accounts and LinkedIn skills respectively. Also, it seems that the feature combination system shows the best performance

in comparison to other three recommending systems. Figure 5.5 compares the precision of collaborative filtering, content-based, mix, and feature combination knowledge-based recommender systems together.

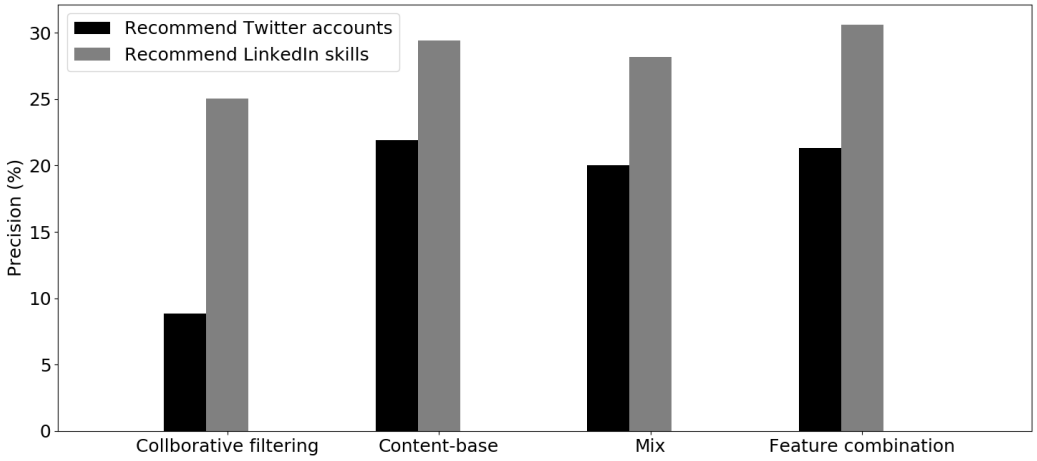

Figure 5.5: Comparison between performance of the proposed knowledge-based recommender systems

5.3 Recommender systems based on matrix and tensor factorizations

The k-fold cross-validation technique is used to evaluate the TD-based and coupled NMF-based cross-domain recommender systems. The data in the target domain are divided into ten folds; considering each fold in the target domain as the test set, we produce recommender systems using the available data in the source domain and other folds in the target domain as well as partial information in the test set. The partial data of the test set is created by replacing different ratios of missing values inside it. For each suggested cluster of items, two more frequent items are recommended. Then, the precision is calculated for each test fold; and finally, the total precision is determined as the average of different folds' precisions. the Figures 5.6 and 5.7 show the result of coupled-NMF based and TD-based recommender systems respectively.

The performances of proposed cross-domain recommender systems are compared to the traditional single-domain collaborative filtering recommender system using matrix factorization. Figures 5.8 and 5.9 display the results of this comparisons.

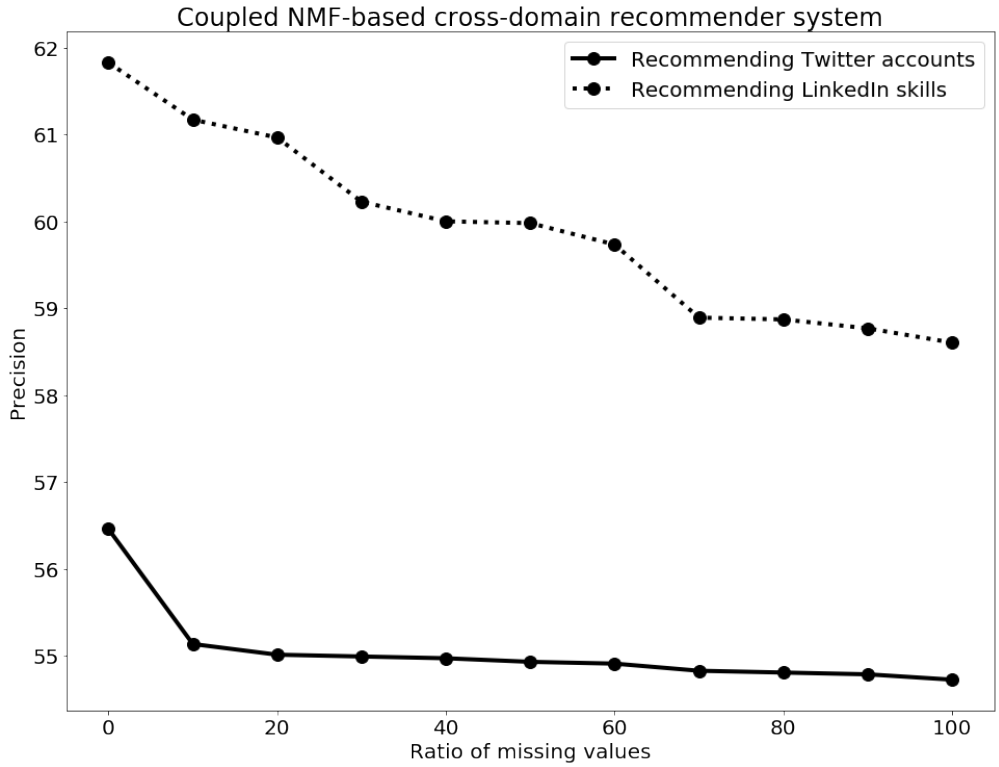

Figure 5.6: Cross-domain recommender systems based on the coupled nonnegative matrix factorization

According to the 5.7 and 5.6 the precision of cross-domain recommender systems decreases with increasing the number of missing values in the target domain. It simply means that recommender systems can create more accurate recommendations when they have more information about the user's preferences. In contrast, If the user has only a few past rating or even no any past rating, the recommended list of items is less precise.

Another interesting point, which can be found from these figures, is that the precision of recommender systems to recommend LinkedIn skills are higher than their precision to suggest Twitter following accounts. We believe that it comes from different natures of data in these two separate domains. Assume that the recommender system wants to produce a list of LinkedIn skills which is likely to be interested by a particular user. As it discussed before, in the section 5.1, the LinkedIn skills form a scale-free network with few highly linked

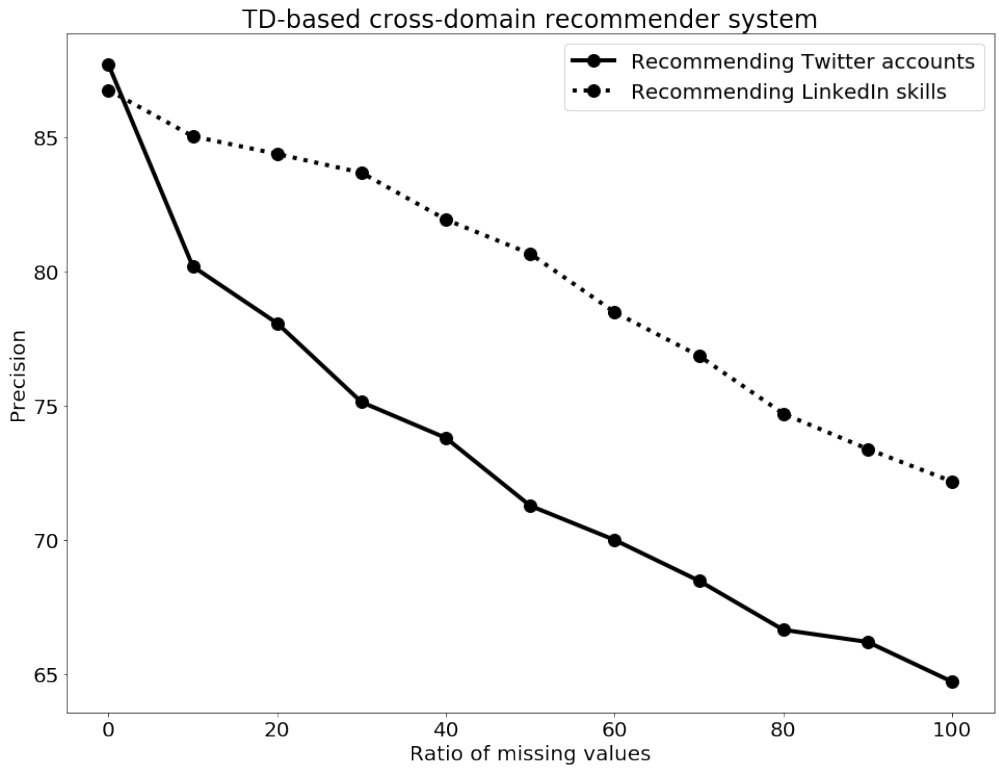

Figure 5.7: Cross-domain recommender systems based on tensor decomposition

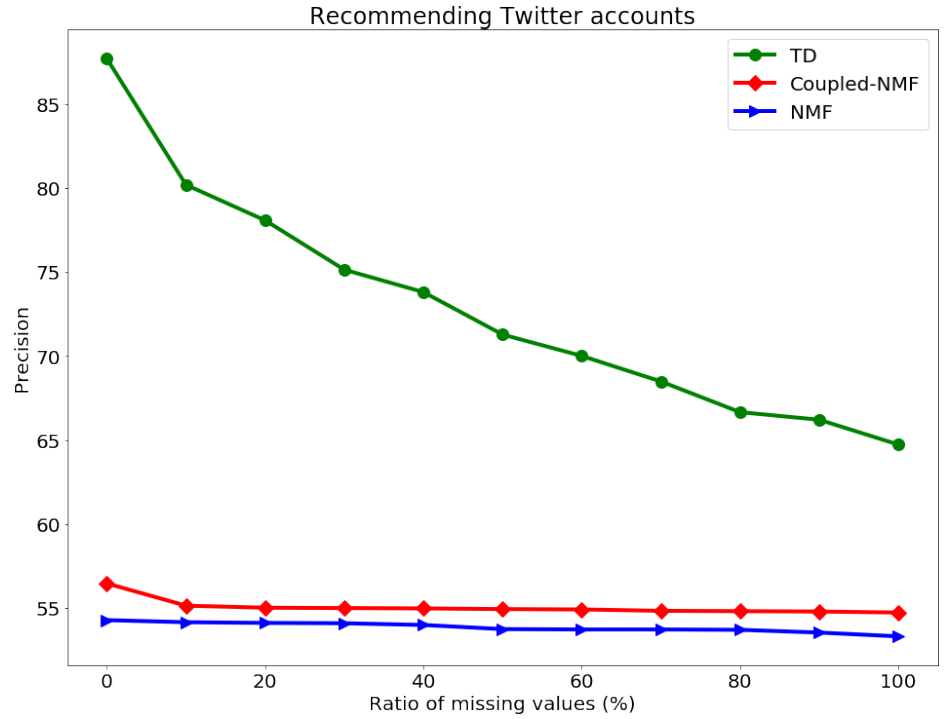

Figure 5.8: Comparison between different recommender systems to suggest Twitter following accounts

nodes. It also can imply that they are some skills in this dataset which are selected by the majority of the users or at least a large portion of them. So, there is a high chance that this unique user wants to claim some of those skills as well. We guess that the proposed coupled NMF-based and TD-based recommender systems can find the most-related popular skills for each user. This guess is also confirmed by our previous finding in the Figure 5.5 where all the suggested knowledge-based recommender systems follow the same trend: they can create more accurate suggestions for LinkedIn rather than Twitter. In contrast, to the LinkedIn skills, Twitter following accounts do not form a scale-free network. By other words, it is hard to find a particular Twitter account which is followed by the majority of the users.

Figures 5.8 and 5.9 shows that the both suggested cross-domain recommendation frameworks, namely coupled NMF-based and TD-based recommender systems, show better performance in comparison with the single-domain NMF-based collaborative filtering technique. The main reason is that they use more information to identify the users' preferences. While the single-domain matrix factorization tries to understand the similarities between the users based on the features in the target domain in order to suggest new items in that specific domain, the cross-domain recommender systems take the benefit of using additional features exposed in the source domain as well. Although, both coupled NMF-based and TD-based recommender systems show higher precision rather than the baseline method, however, the multi-modal TD-based technique superior to the coupled NMF-based algorithm. Like the single-domain NMF, the cross-domain coupled NMF tries to understand the similarities between users but using both rating features in the two domains. Therefore, both of these techniques are the type of collaborative filtering. On the other hand, in the TD-based recommender system, we construct a tensor of data; inside this tensor, all the modes are related to each other. By applying the tensor decomposition on this tensor, the TD-based recommender system tries to find the relationships between all the modes: user-user, user-skill, user-account, and even skill-account and utilize these relationships to suggest a more accurate list of items. Therefore, this techniques has a hybrid type and because it considers multi-modal relationships between the users and different features, it can produce more precise recommendations.

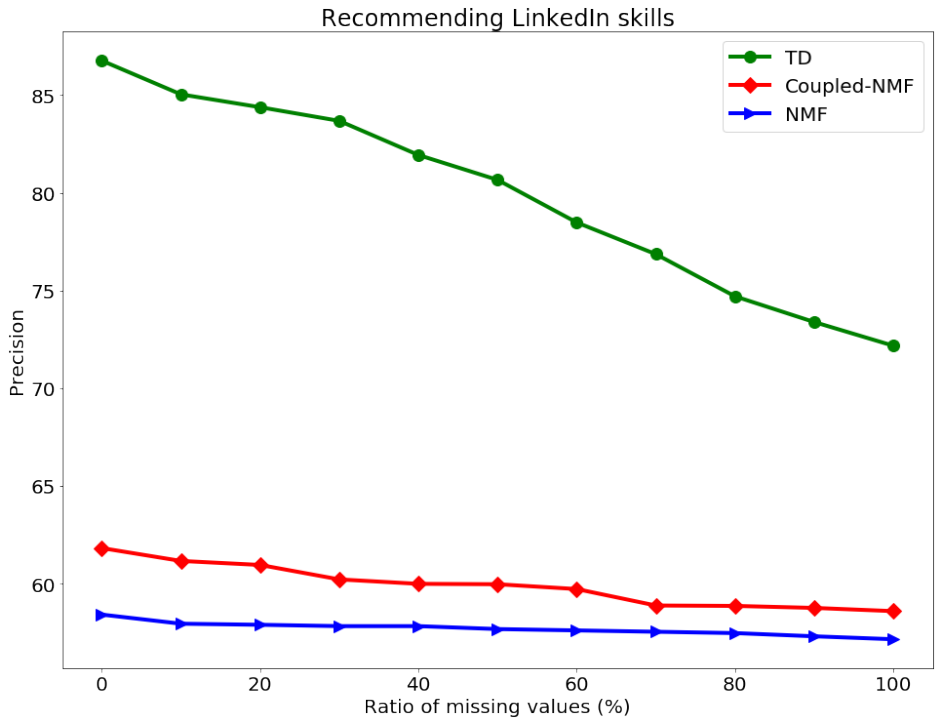

Figure 5.9: Comparison between different recommender systems to suggest LinkedIn skills

6

CONCLUSION AND FUTURE DIRECTIONS

In this thesis, novel cross-domain recommender systems have been proposed which can suggest items to users with high accuracy, even under the cold-start or new-user situations. In addition, it has been shown that by implementing a pre-clustering on available data, the sparsity of the data can be reduced sharply. Here, we present the main conclusions derived from our research study. First, we summarize the findings of the thesis and discuss the obtained contributions; then, we mention potential research directions in this area.

6.1 Summary of Contributions

Considering the research goal number one, stated in chapter 2, we first started our research study by reviewing the recommender systems' concepts, their different classification, and various recommendations techniques. In addition, in chapter 3, we followed the second research goal; and reviewed the idea of cross-domain recommendation. We provided a framework to define the difference between two domains into one of the following levels: system level, type level, attribute level, and value level. In addition, by reviewing previous related studies, we found that the knowledge-flow between different domains is either as a knowledge transmission, where the explicit or implicit features of the source domain are used to make recommendation in the target domain, or as a knowledge aggregation, where different features in distinct domains are combined together in order to create a better understanding of users' preferences. Moreover, chapter 3 covered the fifth research goal as well. Hence, the preliminaries of the matrix and tensor algebra, as well as the most popular matrix factorization and tensor decomposition techniques, are reviewed in this chapter. It showed that these techniques are the state of art methods on the recommender systems and, during last few

years, many studies utilized matrix-based or tensor-based methods to represent multi-modal data and produce recommender or predictor models.

In chapter 4, we first described the extracted data from two different social media, the Twitter and the LinkedIn. These information showed different features of identical computer scientists: their Twitter following accounts and their LinkedIn technical skills. We discussed that the difference between the Twitter and the LinkedIn domains can be considered as a system level dissimilarity. Further, this chapter covered the fourth research goal. Based on the reviewed studies in the previous chapter, four knowledge-based cross-domain recommender systems were proposed to recommend Twitter accounts to follow or LinkedIn skills to add. These recommender systems included a collaborative filtering, a content-based, a hybrid mix, and a hybrid feature combination algorithms. Actually, we have investigate the applicability of the cross-domain recommendation idea by proposing these four knowledge-based algorithms. Although these methods rely on our information about the considered Twitter and LinkedIn domains, however, in the following sections we have proposed the coupled-NMF and TD-based techniques which have the minimum dependency to the domains specific characteristics.

In addition to those knowledge-based recommendation models, chapter 4 included two more important parts: designing and implementing a coupled NMF-based recommender system as well as a TD-based recommender system. In fact, to address the sixth research goal we developed two cross-domain recommendation algorithms: 1) the coupled NMF algorithm which produces recommendation in target domain by sharing the latest features of the source domain, and 2) the CP-style tensor decomposition which suggests item to users in the target domain by merging users' preferences from distinct domains. Their implementation included various steps, such as a pre-clustering stage to reduce the data sparsity, handling the missing values, and tuning different hyperparameters. To the best of our knowledge, it is the first time that the suggested coupled NMF-based formula with its regularization terms is used to develop a cross-domain recommendation system for social media domains. And also, using CP to integrate Twitter and LinkedIn data is a novel idea proposed by this thesis.

To address the last research goal, we analyzed the performance of the proposed cross-domains recommender systems in chapter 6. We compared the precision of the knowledge-

based recommender systems together and showed that, on average, the hybrid feature combination algorithms could suggest more accurate items in the target domain rather than the collaborative filtering, the content-based, and the hybrid mix algorithms. We also analyzed the data in our two domains by constructing the LinkedIn skills graph and the Twitter accounts graph. Based on the obtained results, the LinkedIn skills formed a scale-free network; we recognized the most popular and important skills in this network by looking at their degree centrality as well as their betweenness and closeness degrees.

Furthermore, the performance of the coupled NMF-based and TD-based cross-domain recommender systems were compared together as well as with the single-domain NMF-based traditional recommendation model. We saw that both cross-domain algorithms show better performance in comparison with the single-domain recommender system. The main reason is that these multi-domain models use more features to understand the users' preferences or relationships between users and items. Also, based on the results, The TD-based recommender system was superior to the coupled NMF-based recommender system; because although the coupled NMF-based model use different features to make a more comprehensive description of users' preferences, however, the TD-based model consider the multidimensional relationship between users, Twitter accounts, and LinkedIn skills. In other words, the coupled NMF-based model is a collaborative filtering recommender system and the TD-based model has a hybrid type.

The followings are the thesis's most important findings in nutshell:

- Cross-domain recommender system can partially answer some common issues of current recommendation techniques, such as data sparsity, cold-start problem, and new user-problem.

- Like single-domain recommender systems, the cross-domain models can be implemented as a content-based filtering, collaborative filtering or hybrid recommender systems.

- Using a pre-clustering can reduce the data sparsity significantly. If the data we are working with is a categorical data, or is a combination of categorical and numerical data, the k-modes clustering technique can be a proper choice.

- Both the proposed TD-based and coupled NMF-based recommender systems produce

more precise recommendation rather than collaborative filtering NMF-based model. However, the TD-based recommender systems show far better results.

6.2 Open Research Issues

In this thesis, we have shown that the wise integration of data from diverse domains can increase the accuracy of the recommendations in the target domain. However, selecting the proper auxiliary information from another domain, choosing the type of knowledge-flow, and developing the model itself are not easy and straightforward. There is not such a rubric to determine if the source domain preferences should be considered or not. On the other hand, an inappropriate cross-domain recommender system even may reduce the accuracy of the single-domain model. Therefore, trying to design an intelligent method to ease these steps can be a great achievement.

Assume that we decided to use another domain, as the source, to make a recommendation in the target domain. There might be a variety of different matrix or tensor factorization methods, such as coupled NMF, coupled SVD, CP, Tucker, and etc. One another research direction is to create an unsupervised model to select the appropriate technique.

Finally, the most interesting research direction is developing the frameworks which can properly deal with high-order high-scale data. For example, in this thesis, we proposed a CP tensor decomposition model for the cross-domain recommendation. As discussed before, the considered tensor has three modes: user, Twitter account, LinkedIn skill. Although here this model can create high accurate suggestions, if the number of considered features becomes much higher than three (the tensor has much higher than three orders), then, implementation of any TD-based method needs an unreasonably high storage capacity; therefore, using these models becomes impractical.

References

[1] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013.

[2] Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender systems: introduction and challenges. In *Recommender systems handbook*, pages 1–34. Springer, 2015.

[3] Muhammad Aljukhadar, Sylvain Senecal, and Charles-Etienne Daoust. Information overload and usage of recommendations. In *Proceedings of the ACM RecSys 2010 workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI). CEUR-WS. org*, 2010.

[4] Martin J Eppler and Jeanne Mengis. The concept of information overload: A review of literature from organization science, accounting, marketing, mis, and related disciplines. *The information society*, 20(5):325–344, 2004.

[5] Eugene G Chewning Jr and Adrian M Harrell. The effect of information load on decision makers' cue utilization levels and decision quality in a financial distress decision task. *Accounting, Organizations and Society*, 15(6):527–542, 1990.

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

[7] Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):13, 2016.

[8] Oscar Celma. Music recommendation. In *Music recommendation and discovery*, pages 43–85. Springer, 2010.

[9] Paula Cristina Vaz, David Martins de Matos, Bruno Martins, and Pavel Calado. Improving a hybrid literary book recommendation system through author ranking. In *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*, pages 387–388. ACM, 2012.

[10] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. Time-aware point-of-interest recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 363–372. ACM, 2013.

[11] Sabrina Nusrat and Julita Vassileva. Recommending services in a trust-based decentralized user modeling system. In Liliana Ardissono and Tsvi Kuflik, editors, *Advances in User Modeling*, pages 230–242, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[12] Mouzhi Ge, Francesco Ricci, and David Massimo. Health-aware food recommender system. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 333–334. ACM, 2015.

[13] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: a survey. *Decision Support Systems*, 74:12–32, 2015.

[14] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[15] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[16] Manolis G Vozalis and Konstantinos G Margaritis. Using svd and demographic data for the enhancement of generalized collaborative filtering. *Information Sciences*, 177(15):3017–3037, 2007.

[17] Ana Belén Barragáns-Martínez, Enrique Costa-Montenegro, Juan C Burguillo, Marta Rey-López, Fernando A Mikic-Fonte, and Ana Peleteiro. A hybrid content-based and item-based collaborative filtering approach to recommend tv programs enhanced with singular value decomposition. *Information Sciences*, 180(22):4290–4311, 2010.

[18] Marco Degemmis, Pasquale Lops, and Giovanni Semeraro. A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Modeling and User-Adapted Interaction*, 17(3):217–255, 2007.

[19] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[20] Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1986.

[21] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.

[22] Youngki Park, Sungchan Park, Woosung Jung, and Sang-goo Lee. Reversed cf: A fast collaborative filtering algorithm using a k-nearest neighbor graph. *Expert Systems with Applications*, 42(8):4022–4028, 2015.

[23] Christian Desrosiers and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[24] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1244. ACM, 2015.

[25] Sheng Zhang, Weihong Wang, James Ford, Fillia Makedon, and Justin Pearlman. Using singular value decomposition approximation for collaborative filtering. In *null*, pages 257–264. IEEE, 2005.

[26] Jesus Bobadilla, Fernando Ortega, Antonio Hernando, and Javier Alcalá. Improving collaborative filtering recommender system results and performance using genetic algorithms. *Knowledge-based systems*, 24(8):1310–1316, 2011.

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

[28] S. Urmela, D Joseph, and K. Vaitheki. A survey on web service mining by collaborative filtering and qos. *International Journal of Recent Development in Engineering and Technology*, 3:69–78, 2014.

[29] Bin Cao, Nathan N Liu, and Qiang Yang. Transfer learning for collective link prediction in multiple heterogenous domains. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 159–166. Citeseer, 2010.

[30] Pinata Winoto and Tiffany Tang. If you like the devil wears prada the book, will you also enjoy the devil wears prada the movie? a study of cross-domain recommendations. *New Generation Computing*, 26(3):209–225, 2008.

[31] Bracha Shapira, Lior Rokach, and Shirley Freilikhman. Facebook single and cross domain data for recommendation systems. *User Modeling and User-Adapted Interaction*, 23(2-3):211–247, 2013.

[32] Weike Pan and Qiang Yang. Transfer learning in heterogeneous collaborative filtering domains. *Artificial intelligence*, 197:39–55, 2013.

[33] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Can Zhu. Personalized recommendation via cross-domain triadic factorization. In *Proceedings of the 22nd international conference on World Wide Web*, pages 595–606. ACM, 2013.

[34] Sheng Gao, Hao Luo, Da Chen, Shantao Li, Patrick Gallinari, and Jun Guo. Cross-domain recommendation via cluster-level latent factor model. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 161–176. Springer, 2013.

[35] Amit Tiroshi, Shlomo Berkovsky, Mohamed Ali Kaafar, Terence Chen, and Tsvi Kuflik. Cross social networks interests predictions based ongraph features. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 319–322. ACM, 2013.

[36] Ngoc-Diep Ho. *Nonnegative matrix factorization algorithms and applications.* PhD thesis, PhD thesis, Université catholique de Louvain, 2008.

[37] Horst D Simon and Hongyuan Zha. Low-rank matrix approximation using the lanczos bidiagonalization process with applications. *SIAM Journal on Scientific Computing*, 21(6):2257–2274, 2000.

[38] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[39] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[40] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

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

[42] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[43] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[44] Antonio Hernando, Jesús Bobadilla, and Fernando Ortega. A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model. *Knowledge-Based Systems*, 97:188–202, 2016.

[45] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940. ACM, 2008.

[46] Kai Yu, Shenghuo Zhu, John Lafferty, and Yihong Gong. Fast nonparametric matrix factorization for large-scale collaborative filtering. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 211–218. ACM, 2009.

[47] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. Trustsvd: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In *AAAI*, volume 15, pages 123–125, 2015.

[48] Bo Long, Zhongfei Mark Zhang, Xiaoyun Wu, and Philip S Yu. Spectral clustering for multi-type relational data. In *Proceedings of the 23rd international conference on Machine learning*, pages 585–592. ACM, 2006.

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

[50] Evrim Acar, Gozde Gurdeniz, Morten A Rasmussen, Daniela Rago, Lars O Dragsted, and Rasmus Bro. Coupled matrix factorization with sparse factors to identify potential biomarkers in metabolomics. *International Journal of Knowledge Discovery in Bioinformatics (IJKDB)*, 3(3):22–43, 2012.

[51] Yuqi Wang, Jiannong Cao, Lifang He, Wengen Li, Lichao Sun, and Philip S. Yu. Coupled sparse matrix factorization for response time prediction in logistics services. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 939–947, New York, NY, USA, 2017. ACM.

[52] Qiang Liu, Shu Wu, Liang Wang, et al. Cot: Contextual operating tensor for context-aware recommender systems. In *AAAI*, pages 203–209, 2015.

[53] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):16, 2017.

[54] Raymond B Cattell. parallel proportional profiles and other principles for determining the choice of factors by rotation. *Psychometrika*, 9(4):267–283, 1944.

[55] Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 15:122–137, 1963.

[56] Ledyard R Tucker. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology*, 110119, 1964.

[57] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[58] Boris N Khoromskij. Tensor numerical methods for multidimensional pdes: theoretical analysis and initial applications. *ESAIM: Proceedings and Surveys*, 48:1–28, 2015.

[59] Jimeng Sun, Dacheng Tao, and Christos Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383. ACM, 2006.

[60] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[61] Bo Du, Mengfei Zhang, Lefei Zhang, Ruimin Hu, and Dacheng Tao. Pltd: Patch-based low-rank tensor decomposition for hyperspectral images. *IEEE Transactions on Multimedia*, 19(1):67–79, 2017.

[62] Kun Tu, Bruno Ribeiro, Ananthram Swami, and Don Towsley. Temporal clustering in dynamic networks with tensor decomposition. *arXiv preprint arXiv:1605.08074*, 2016.

[63] Frank L Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics*, 7(1-4):39–79, 1928.

[64] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika*, 35(3):283–319, 1970.

[65] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[66] Henk AL Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 14(3):105–122, 2000.

[67] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[68] Joseph B Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications*, 18(2):95–138, 1977.

[69] Luca Chiantini and Giorgio Ottaviani. On generic identifiability of 3-tensors of small rank. *SIAM Journal on Matrix Analysis and Applications*, 33(3):1018–1037, 2012.

[70] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[71] Vincent Wenchen Zheng, Bin Cao, Yu Zheng, Xing Xie, and Qiang Yang. Collaborative filtering meets mobile recommendation: A user-centered approach. In *AAAI*, volume 10, pages 236–241, 2010.

[72] Sangeetha Kutty, Lin Chen, and Richi Nayak. A people-to-people recommendation system using tensor space models. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 187–192. ACM, 2012.

[73] Anne Kao, William Ferng, Stephen Poteet, Lesley Quach, and Rod Tjoelker. Talison-tensor analysis of social media data. In *Intelligence and Security Informatics (ISI), 2013 IEEE International Conference on*, pages 137–142. IEEE, 2013.

[74] Lina Yao, Quan Z Sheng, Yongrui Qin, Xianzhi Wang, Ali Shemshadi, and Qi He. Context-aware point-of-interest recommendation using tensor factorization with social regularization. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 1007–1010. ACM, 2015.

[75] Cong Zheng, E Haihong, Meina Song, and Junde Song. Cmptf: contextual modeling probabilistic tensor factorization for recommender systems. *Neurocomputing*, 205:141–151, 2016.

[76] Namyong Park, Byungsoo Jeon, Jungwoo Lee, and U Kang. Bigtensor: Mining billion-scale tensor made easy. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 2457–2460. ACM, 2016.

[77] Faisal M Almutairi, Nicholas D Sidiropoulos, and George Karypis. Context-aware recommendation-based learning analytics using tensor and coupled matrix factorization. *IEEE Journal of Selected Topics in Signal Processing*, 11(5):729–741, 2017.

[78] Makoto Nakatsuji, Qingpeng Zhang, Xiaohui Lu, Bassem Makni, and James A Hendler. Semantic social network analysis by cross-domain tensor factorization. *IEEE Transactions on Computational Social Systems*, 4(4):207–217, 2017.

[79] Stephan Rabanser, Oleksandr Shchur, and Stephan Günnemann. Introduction to tensor decompositions and their applications in machine learning. *arXiv preprint arXiv:1711.10781*, 2017.

[80] Vassilis N Ioannidis, Ahmed S Zamzam, Georgios B Giannakis, and Nicholas D Sidiropoulos. Coupled graphs and tensor factorization for recommender systems and community detection. *arXiv preprint arXiv:1809.08353*, 2018.

[81] Jing Peng, Daniel Dajun Zeng, Huimin Zhao, and Fei-yue Wang. Collaborative filtering in social tagging systems based on joint item-tag recommendations. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 809–818. ACM, 2010.

[82] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. *IEEE Transactions on Knowledge and Data Engineering*, 22(2):179–192, 2010.

[83] Benyou Zou, Mengwei Lan, Cuiping Li, Liwen Tan, and Hong Chen. Context-aware recommendation using gpu based parallel tensor decomposition. In Xudong Luo, Jeffrey Xu Yu, and Zhi Li, editors, *Advanced Data Mining and Applications*, pages 213–226, Cham, 2014. Springer International Publishing.

[84] S. Maroulis, I. Boutsis, and V. Kalogeraki. Context-aware point of interest recommendation using tensor factorization. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 963–968, Dec 2016.

[85] P. Symeonidis. Clusthosvd: Item recommendation by combining semantically enhanced tag clustering with tensor hosvd. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(9):1240–1251, Sep. 2016.

[86] Xiaolin Zheng, Weifeng Ding, Zhen Lin, and Chaochao Chen. Topic tensor factorization for recommender system. *Information Sciences*, 372:276 – 293, 2016.

[87] Yuankai Ying, Ling Chen, and Gencai Chen. A temporal-aware poi recommendation system using context-aware tensor decomposition and weighted hits. *Neurocomputing*, 242:195 – 205, 2017.

[88] Anu Taneja and Anuja Arora. Cross domain recommendation using multidimensional tensor factorization. *Expert Systems with Applications*, 92:304 – 316, 2018.

[89] Shenglin Zhao, Irwin King, and Michael R. Lyu. Aggregated temporal tensor factorization model for point-of-interest recommendation. *Neural Processing Letters*, 47(3):975–992, Jun 2018.

[90] Steffen Rendle and Lars Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 81–90, New York, NY, USA, 2010. ACM.

[91] KR Bindu, Rhama Lalgudi Visweswaran, PC Sachin, Kundavai Devi Solai, and Soundarya Gunasekaran. Reducing the cold-user and cold-item problem in recommender system by reducing the sparsity of the sparse matrix and addressing the diversity-accuracy problem. In *Proceedings of International Conference on Communication and Networks*, pages 561–570. Springer, 2017.

[92] Xin Luo, MengChu Zhou, Shuai Li, Zhuhong You, Yunni Xia, and Qingsheng Zhu. A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method. *IEEE transactions on neural networks and learning systems*, 27(3):579–592, 2016.

[93] Giacomo Domeniconi, Gianluca Moro, Andrea Pagliarani, Karin Pasini, and Roberto Pasolini. Job recommendation from semantic similarity of linkedin users' skills. In *ICPRAM*, pages 270–277, 2016.

[94] Charu C Aggarwal and Chandan K Reddy. *Data clustering: algorithms and applications*. CRC press, 2013.

[95] Zhexue Huang. Clustering large data sets with mixed numeric and categorical values. In *Proceedings of the 1st pacific-asia conference on knowledge discovery and data mining,(PAKDD)*, pages 21–34. Singapore, 1997.

[96] Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data mining and knowledge discovery*, 2(3):283–304, 1998.

[97] Joshua Zhexue Huang. Clustering categorical data with k-modes. In *Encyclopedia of Data Warehousing and Mining*, 2009.

[98] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[99] Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 116–, New York, NY, USA, 2004. ACM.

[100] Jean Kossaifi, Yannis Panagakis, and Maja Pantic. Tensorly: Tensor learning in python. *arXiv preprint arXiv:1610.09555*, 2016.