Similarity Metrics in Neighborhood-Based Methods for the Completion of Partially Observed Data Matrices

ORTAL VITE Patricia Isabel

Acknowledgments

This dissertation would not have been possible without the support of my mentors, friends, and family.

I would like to express my sincere gratitude to my academic advisor Edahiro Masato, who motivated me to continue moving forward and, no matter the time of the day or the day of the week, was available to guide and support me.

Furthermore, I wish to extend my gratitude to Tohru Ishihara and Naoki Nishida for the time they devoted to reading and evaluating this dissertation.

I would also like to thank my dearest friend Jorge Trevino for always been there to support me. I am extremely grateful for all these years of advice, encouragement and challenges that have been invaluable to my growth.

Finally I want to thank my family who patiently supported me during this very busy period.

Abstract

Matrix completion is a widespread task in the machine learning field because of the often occurrence of missing data in real-world scenarios. It involves the estimation of unobserved values of partially observed data represented in tabular form. As part of the collaborative filtering techniques, neighborhood-based methods are widely used to address the data matrix completion problem. While neighborhood-based methods heavily depend on similarity metrics to compute accurate estimations, few studies have focused on the design of similarity measures and instead rely on conventional metrics regardless the problem to solve. This dissertation presents two neighborhood-based methods to solve two different practical data matrix completion problems respectively. The proposed approaches consist on leveraging the known information available from different sources, computing clusters that exhibit close patterns or trends, and then estimating the unknown values based on these groups and similarities. Furthermore, the metrics to establish the degree of similarity between agents are designed to be domain-specific, and free from the limitations of conventional similarity measures. In particular, this work focuses on evaluating the effects that custom measures have in terms of prediction performance. The first proposed method encodes products as a sequence of attributes, each of which represents a different dimension of the consumer perception. The second method is a switching hybrid recommender that

estimates item ratings addressing the sparsity problem that affects the performance of collaborative filtering techniques. The results of experiments conducted using real-world and synthetic data indicate that the proposed methods have superior performance compared to conventional approaches in terms of mean absolute error (MAE) and root mean square error (RMSE).

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The matrix completion problem involves the reconstruction of a low-ranked matrix based on the entries that are observed. Image completion and restoration, localization in IoT networks, and signal phase retrieval are some of the applications that require completing missing information of partially observed matrices [1]. Data matrices that have empty values in one or more variables for one or more records are said to be partially observed. Missing data can occur due to several reasons such as errors when gathering or storing information, deliberate non-exhaustive data collection processes, or lack of available information. The estimation of missing values in data matrices became a prevalent task in the machine learning field, because often the data gathered in real-world problems is represented in tabular form.

One of the most representative approaches to addressing a data matrix completion problem is given by collaborative filtering (CF). CF is a group of techniques mostly known for their application in the field of recommender systems (RS), where they are used to predict products or services a person might like. CF however, is not applied in RS, in a more general sense, CF algorithms aggregate and filter information from different data sources in order to generate estimations of unseen (partially observed)

data.

CF techniques are generally classified into two categories: model-based, and neighborhood-based techniques. In the model-based technique, the training and prediction phases are distinct. First, a model is created based on observed matrix data. There are various algorithms to build CF models such as: latent semantic models, singular value decomposition, Markov decision processes or neural networks, among others. Once the model is created, it is used to generate predictions. Neighborhood-based methods, on the other hand, generate predictions directly using the data represented as a matrix of interactions. First, calculating the similarity between rows (or columns), and then, using the computed similarity to cluster similar row vectors together. These are later aggregated in order to estimate unknown elements from the original matrix. While some model-based methods are more space-efficient and tend to overfit less than neighborhood-based methods, they are more complex to implement and maintain. More importantly, they lose the explainability power that neighborhood models have, acting as a black boxes. Also, neighborhood-based models are more intuitive and stable which make them more mainstream in real-world applications. Because the similarity computation is a critical component of the neighborhood-based approach, numerous measures have been proposed as a result of research efforts, and applied to different domains with various degrees of performance.

The primary purpose of this dissertation is to evaluate the effects that different neighborhood-based methods have in terms of prediction accuracy when using custom similarity measures. In other words, whether custom similarity measures have a significant impact on the performance of neighborhood-based approach regardless of the task. This study proposes to employ contextual information in order to design

custom similarity measures depending on the type of matrix data that is observed. Because the choice of similarity greatly influences the performance of predictions, it is expected that a custom similarity helps the algorithm to provide better predictions than conventional similarity measures. Furthermore, this dissertation discusses two different practical matrix completion tasks, and presents two different methods, one per problem, to estimate the elements of each task's interaction matrix. The proposed methods apply the CF key principles while relying on different custom similarity measures based on the semantic meaning of the interaction data to estimate unobserved values.

Similarity metrics, as a measure of the strength of relationship between two objects, are widely used in the computer science field, especially data mining, where object classification and data clustering are two of their most common tasks. Moreover, neighborhood-based CF techniques are often used to gain insights regarding the relationships between the rows and columns of interaction matrices. For instance, when rows and columns represent users and items, it is possible to find users of similar tastes or similar items that one user might like.

Chapter 3 describes a method to estimate the elements of a vector representing the attributes of products, based on user purchase history. In the proposed method, first, a base product set with known attribute values is built based on survey data. Then, new product attribute vectors are estimated using product similarity. The proposed new similarity measure addresses the limitations of conventional similarity measures. Compared to other approaches, the method outlined in Chapter 3 is not model based, and it does not require thousands of observations to show good performance.

Chapter 4 presents a novel hybrid recommender system, that predicts the elements

of a matrix of movie ratings based on implicit and explicit feedback from users. This chapter outlines how the proposed recommender addresses the common sparsity problem prevalent in CF, and offers an explanation on the proposed similarity metric based on the semantic interpretation of the rating values. Different from other approaches, the proposal uses an architecture consisting on two ratings. One calculated using global statistics and the other one calculated using the neighborhood approach using a custom similarity measure that avoids the drawbacks of conventional similarity measures. The selection of the final rating depends on a sparsity criterion that mitigates the adverse effects of highly sparse data.

Finally, Chapter 5 concludes this dissertation with a summary and discusses the future work of this research.

Chapter 2

Technical Background

2.1 Collaborative Filtering

Collaborative filtering (CF) is a technique often used in recommender systems (RS) to predict what kind of products or services a user might like. RS became popular in the recent years because the digital era created a channel to access vast amounts of information, including countless different choices. These choices range from what book to buy, what news to read, and which movie to watch, to the next vacation destination, the company to work for, or even a potential new friend.

CF makes recommendations from explicit feedback in the form of ratings and reviews, or implicit preferences from historical behavior; such as past purchases or browsing history. The key idea is that the rating of a target user for an item is likely to be similar to that of another user, if both users have rated other items in a similar manner [2]. Thus, CF has been regarded as one of the electronic surrogates of the traditional word-of-mouth [3], where like-minded users influence each other on the consumption of related products or services.

One of the earliest applications of collaborative filters emerged owing to the need

to aid overwhelmed email users in sorting through their huge stream of incoming content. The Tapestry system [4] allowed users to collaboratively annotate documents with tags that are later used to build filters. The filters ensured that users received only potentially interesting information, allowing them to manage emails more efficiently. Another early example of a platform that implemented CF techniques is the GroupLens system [5], which was designed to recommend news articles to readers based on the ratings of other users. CF is a method that matches people with similar interests and then finds what people like based on this preference clusters.

CF techniques are generally classified into two categories: neighborhood-based and model-based techniques. Neighborhood-based methods (also referred to as memory-based methods) compute recommendations for a target user from the known scores of other users with similar rating patterns. These methods are further classified as 'user-based' when they employ users' ratings to cluster like-minded users together and 'item-based' if they attempt to find the items that are most similar to a target one. Model-based approaches, conversely, use the ratings to build predictive models. Some examples include Bayesian clustering, Boltzmann machines [6], and latent factor models such as singular value decomposition [7] or matrix factorization [8], [9]. Matrix factorization (MF) algorithms aim to represent the user-item matrix as a product of matrices, with user vectors reduced to a lower-dimensional latent space. Once the user-item matrix has been factorized in this manner, the lower-dimensional matrices can be used to directly estimate the ratings. This approach was first proposed by Funk [8] and is usually referred to as FunkMF. Other methods building on these ideas, such as SVD++ [9], have also exhibited good recommendation performance. Generally, MF methods tend to be more accurate than neighborhood-based methods;

however, they lack the explainability and versatility that make neighborhood-based methods prevalent in commercial applications [2], [10]. Explainable methods are important because, unlike black-box models, through them we can understand how predictions are made, the relationships between variables and how they influence predictions. For example, e-commerce websites can recommend items based on "the products that you bought before", justifying the choice in recommendation. Furthermore, each new observation, such as the addition of a new user, affects the choice of latent space and requires the user-item matrix to be factorized again; this is a costly operation, and thus, predictions cannot be updated in real-time as new ratings are added to the dataset.

The approaches discussed in this dissertation are based on the underlying assumption of collaborative filters, however, their objectives are different. The objective of the method presented in Chapter 3 is to estimate a full vector of attributes based on similar objects, while the method described in Chapter 4 employs the CF neighborhood-based approach to estimate unknown values on an incomplete vector.

2.2 Neighborhood-based rating prediction

The rating prediction for item i by user u is derived directly from an $n \times m$ sparse user-item rating matrix R, where n denotes the number of users and m the number of items. The elements of R correspond to the ratings if available, or zero if not, and it has the following structure:

$$
\begin{array}{c} \\ u_1 \\ u_2 \\ \vdots \\ u_n \end{array}
\begin{array}{c}
\begin{array}{cccc} i_1 & i_2 & ... & i_m \end{array} \\
\left[\begin{array}{cccc}
0 & 0 & \dots & 3 \\
0 & 0 & \dots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 1 & \dots & 0
\end{array}\right]
\end{array}
$$

It is common to use the weighted average of the ratings of a set of neighboring users, as follows:

$$\hat{r}_{u,i} = \frac{\sum\limits_{v \in N(u,i)} \text{sim}(u,v) \cdot r_{v,i}}{\sum\limits_{v \in N(u,i)} \text{sim}(u,v)}. \tag{2.1}$$

Here, $\text{sim}(u,v)$ represents the similarity between users u and v, and $N(u,i)$ denotes the set of k-nearest neighbors (k-NN) for user u (i.e., users with the highest similarity with respect to u) who have rated item i. Different users have different tendencies when rating items. Some may tend to give higher ratings than others, which would negatively affect the performance of Eq. 2.1. To account for these differences, a mean-centered version of this approach is adopted, which is defined as follows:

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum\limits_{v \in N(u,i)} \text{sim}(u,v) \cdot (r_{v,i} - \bar{r}_v)}{\sum\limits_{v \in N(u,i)} \text{sim}(u,v)}, \tag{2.2}$$

where $\bar{r}_u$ and $\bar{r}_v$ denote the average ratings of users u and v, respectively.

2.3 Similarity Measures

Similarity measures play a critical role in various disciplines; from biology and sociology to data analysis and machine learning. The choice of an effective similarity measure depends on the nature of the data [11]. For this reason, numerous measures have been proposed as a result of research efforts in several domains.

In general, similarity measures quantify the degree of similarity between two objects. They are particularly important in the context of CF for recommendation systems, because they serve as a set of criteria to select groups of similar users, whose preferences are aggregated and exploited as a basis to infer other users' tastes. As seen in Eq. 2.1, the similarity score is used to weight the contribution of other user ratings and to determine which users will influence the predicted result. Consequently, the similarity computation has a direct and significant influence on the performance of CF methods [12].

The similarity index is often calculated between vectors of values. For instance, in natural language processing tasks, similarity measures are calculated between vectors that represent words in a dictionary. Whereas in the neighborhood-based approach, the similarity is calculated between user vectors, where the elements of the vectors are the ratings given by the user to different items.

Numerous similarity measures have been proposed to better capture the relationship patterns between users and items. The Pearson correlation coefficient (PCC), cosine similarity (COS), and the Jaccard index (JAC) are the most widely adopted similarity measures in the literature [13],[10],[14],[15],[16].

Given two user vectors u and v, $\mathrm{PCC}(u,v)$ is defined as follows:

$$\mathrm{PCC}(u,v) = \frac{\sum\limits_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum\limits_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum\limits_{i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}, \tag{2.3}$$

where I_u and I_v denote the sets of items rated by users u and v, respectively. PCC measures the strength of the linear relationship between two vectors, regardless of the magnitude of their elements, implying that even when users have different average

ratings, PCC will consider them to be similar as long as they have similar trends. A value of $+1$ indicates a strong positive correlation, whereas a value of -1 indicates a strong negative correlation. However, users with negative correlations are sometimes filtered out as a heuristic enhancement [10].

COS calculates the similarity of two vectors by measuring the cosine of the angle between them. Given two vectors u and v, the cosine similarity is calculated as follows:

$$\mathrm{COS}(u,v) = \frac{\sum\limits_{i \in I_u \cap I_v} r_{u,i} r_{v,i}}{\sqrt{\sum\limits_{i \in I_u \cap I_v} r_{u,i}^2} \sqrt{\sum\limits_{i \in I_u \cap I_v} r_{v,i}^2}}, \tag{2.4}$$

A COS value of 0 implies that the vectors are orthogonal and thus completely dissimilar, whereas a value of 1 corresponds to vectors pointing in the same direction and therefore that are maximally similar. Unlike PCC, COS does not explicitly contain the means and variances for each user's ratings in its definition.

JAC measures the similarities between two vectors calculating its overlap. Given two vectors u and v, it is defined as follows:

$$\mathrm{JAC}(u,v) = \frac{|I_u \cap I_v|}{|I_u| + |I_v| - |I_u \cap I_v|}, \tag{2.5}$$

where I_u and I_v are the set of items rated by user u and v respectively. JAC is naturally bound between 0 and 1. It assigns a value of 0 when there are no items rated simultaneously by both users, and a value of 1 when both users have rated the same items. The major weakness of JAC is that it does not consider the actual values of the ratings; it only considers the items that were actually rated. Even when two users have diametrically opposed preferences, JAC will assign a score of 1 if they rated the same items.

There has been extensive research comparing the advantages and disadvantages of these measures, such as in [17] and [18]. The study of [15] summarized the drawbacks of these similarities as follows:

- Flat-value problem: If all the ratings in a vector have the same value, PCC cannot be computed because the denominator in Eq. 2.3 becomes 0. Similarly, if both vectors have constant ratings, COS is always 1 even if the constant value differs between users.

- Opposite-value problem: When two user vectors have completely opposite values, PCC will always be -1, even if the user preferences are not extremely opposite in terms of rating semantics.

- Single-value problem: If two users rated only one item in common, PCC cannot be calculated, whereas COS and JAC always yield a value of 1 irrespective of the actual rating values.

- Cross-value problem: If two users rated only two items in common, PCC will either yield a value of -1 if the values cross each other or a value of 1 otherwise.

This dissertation describes the proposal of two similarity measures that overcome the drawbacks of these conventional metrics. The proposed similarity metrics are applied to two different use cases. These practical applications are discussed in detail in Chapter 3 and Chapter 4. In both cases, the objective is to estimate unknown values based on the categorization of objects that have similar vector representations.

Chapter 3

Similarity Measure for Product Attribute Estimation

This chapter presents a method to encode products as a sequence of attributes based on survey data. In the proposed method, first, a base product set with known attribute values is built based on consumers' perceptions. Then, new product attribute vectors are estimated using product similarity. The proposed method also incorporates a new similarity measure that is based on purchase behavior which is suitable for estimating product attribute vector distances. Because it takes into account the magnitude of the individual components of the vectors under comparison, the proposed method is free from the limitations of conventional similarity measures. The results of experiments conducted using real-world data indicate that the proposed method has superior performance compared to conventional approaches in terms of mean absolute error (MAE) and root mean squared error (RMSE).

3.1 Introduction

Comprehensive market analysis and deep understanding of end users is essential to create valuable market positions and stay ahead of competitors. Identifying the consumer subjective perceptions, motivations, and preferences helps companies improve existing products so they can be customized accordingly. In-depth knowledge of consumer sentiment could also be exploited to uncover unmet product needs and generate new product development opportunities.

Representing products in terms of attributes that denote consumer perceptions is also crucial for improving the quality of e-commerce site search. Attributes are the building blocks of the product catalogues that allow e-commerce retailers to identify, organize, standardize, and display information to users [19]. They are implemented to build more efficient product recommendation systems [20], [21], filter search results more effectively [22], and ultimately improve product discoverability, thereby positively influencing sales [23]. Therefore, it is of great interest to devise methodologies that could accurately estimate product attributes.

An example of a field that addresses questions of these kinds is text mining. Text mining is an active area of research that focuses on the development of statistical techniques and machine learning algorithms that extract meaningful information from unstructured or semi-structured text. Attribute extraction involves generating attribute-value pairs from text available in product descriptions or reviews. However, these methods rely heavily on large quantities of text data [24], [25], [26], and are inadequate for inferring the attributes of products that are newly introduced to the market.

In conjunction with text mining, surveys constitute a common source of information on the perceptions and attitudes that a population might have towards certain products. For decades, companies that carry out qualitative research have been relying on surveys to understand the factors that influence buying behavior. However, conducting surveys becomes more expensive and time-consuming as the number of products and attributes increases. For online retail companies, which often have a large inventory of products, it becomes unfeasible to build an attribute product catalogue based on surveys exclusively. For this reason, it is of great interest to propose methodologies that can accurately estimate product attributes without the need for large sources of user generated text or annotated data.

The main contribution of the study in this chapter is a method for encoding products as a sequence of attributes, each of which represents a different dimension of the consumer perception. In the proposed method, user perceptions on a predefined product base are first obtained (via surveys) and analyzed. Then, the attribute vectors of different products from the same group are estimated using product similarity. In particular, one of the main objectives of this research was to propose a new similarity measure that is suitable for estimating product attribute vector distances based on purchase behavior.

The practical value of the proposed method is best appreciated in marketing and e-commerce applications where resources are limited and acquiring additional data can be time-consuming or expensive. Other approaches like the Tag Genome discussed in Section 3.2 use a regression model, or other machine learning algorithms like AdaBoost or rule mining. However, these approaches need thousands of labeled

examples usually extracted from crowd-sourced data to achieve good prediction performance. Rather than increasing the survey size, the proposal generalizes from already available data as explained in Section 3.4. In this manner, marketers can use the results of their existing product sentiment surveys to estimate the attributes of similar products that were not originally surveyed. By building a more exhaustive product attribute catalogue in this manner, marketers may, for example, obtain advanced insights through analyses of sales trends by specific product attributes. The proposed approach can also be used to complete missing attribute information in e-commerce product catalogues. Cleaner and more comprehensive catalogues can increase customer satisfaction through a better shopping experience with superior search results and more accurate product descriptions.

3.2 Related work

Categorization allows us to grasp the maximum amount of useful information with the least cognitive effort [27]. Representing objects in terms of their attributes helps us to simplify our perception of the world in an efficient manner. We can filter out useless information and quickly identify objects with respect to their differences and similarities.

Tagging is a form of attribute assignment, since tags are used to describe particular characteristics of objects. Whereas annotating digital content has been possible for years [28], it only became popular when the weblogging community needed new ways to organize their information for easier recall and discovery. This form of tagging, where users explicitly add keywords in the form of metadata to shared content, is commonly known as collaborative tagging [29]. Some of the limitations of this

classification system include the use of ambiguous terms, imprecision, and a lack of consistency, given the absence of a reference hierarchical structure [30], [31], [32].

Product attribute vectors can also be built automatically. This area of research falls within the domain of information extraction, which focuses on developing different approaches to automatically extract information from unstructured or semi-structured text. Some of the work in this field include different techniques to find values of predefined target attributes that describe products' intrinsic properties, such as brand name, color, or size [33], [19], [22], [21], [34]. The sources of these attributes are commonly product profiles, titles, descriptions, or reviews. An advantage of these techniques is that product attributes do not need to be defined in advance, hence, new concepts can be discovered or the most popular characteristics can be identified. However, these characteristics rarely contain user opinions [35].

Sentiment analysis, on the other hand, focuses on analyzing text to identify the affective state of a user towards specific products or services. Aspect extraction is a subtask aimed at the identification of aspects (attributes) and their associated sentiments. State-of-the-art research in this field investigates ways to efficiently extract aspects, identify the associated opinions, determine the polarity of the opinions, and do so in multiple granularities. However, owing to the high complexity of this procedure, there is little work combining all these tasks at once [36]. Moreover, existing algorithms are not adequately accurate; they are usually limited to specific segments, such as electronic products or hotels reviews [37].

In the context of consumer behavior analysis, both collaborative and automatic tagging offer a powerful way to understand the general perception and sentiment of end users towards particular products. However, these approaches are not sufficient

when the objective of the research is to investigate specific product attributes, because the attributes might never appear as tags or as parts of reviews. Moreover, representing product attributes as a collection of tuples of attribute-value pairs, is not informative of the strength (intensity) with which each attribute describes each element, but only whether the property exists or not. Finally, because these methods rely on tags or text data available online, if there are no reviews available for a product, no attributes can be extracted.

The Tag Genome in [38] introduced the concept of "item genome" to encode movies' most relevant attributes. Similar to how organisms are described by a sequence of genes, the item genome encodes items in an information space based on its relation to a common set of attributes. The Tag Genome G is defined as a collection of relevance values for all tag-item pairs in $T \times I$, represented as a tag-item matrix, where T is a set of tags and I is a set of items. The relevance is calculated using a regression model on predefined features, and has been shown to have high prediction performance. However, this approach relies on training the model on thousands of labeled examples extracted from expert-maintained data or collaborative tags and text sources such as reviews and blogs.

Similar to the Tag Genome, there are a few other works in the literature that approach the prediction of tag data using different machine learning techniques. Examples of these include the prediction of tags in music using AdaBoost [39], the use of nearest neighbor classifiers to predict video features [40], and the tagging of web pages through association rule mining [41]. However, all of them rely on considerable amounts of crowd-sourced data.

The present research seeks to encode product attribute values using a matrix representation similar to the one in [38]. However, a different approach was undertaken for calculating such values. The data came exclusively from a survey, and the attributes were fixed by domain experts. Because surveys constitute a costly data-collection method, the data source was very limited. To train machine learning models with acceptable performance, a considerable number of samples were required, hence, this approach was discarded. Instead, inspired by the assumptions in the collaborative filtering technique, the similarity between products was first calculated, and subsequently the new product attribute vector was computed as a weighted average of the base products, with the weights corresponding to the similarity between products.

3.3 Problem statement

A method is proposed to infer the attribute values of a list of target products from a set of base products with known attribute values obtained through a survey. The attributes were selected by marketing experts with the intent to explain the most representative aspects that characterize the consumer sentiment towards a product. These attributes have an inherently low overlap, since the experts were tasked with choosing different key features related to consumer perception. For this reason, and to simplify the analysis, these were assumed to be independent.

The survey results were normalized and aggregated into a matrix with a column for each attribute $a \in A$ and a row for each product $p \in P$. The elements of the matrix are numbers between 0 and 1, indicating how relevant each attribute is to each product. A value of 0 implies no relevance, and a value of 1 indicates the highest relevance. An example is shown in Table 3.1. The second row shows the attribute vector for product

Table 3.1: $N \times M$ product-attributes matrix. The elements of the matrix, numbers between 0 and 1, represent how relevant to each product each attribute is.

$$\begin{array}{c} \\ p_1 \\ p_2 \\ \vdots \\ p_N \end{array} \begin{array}{c} \begin{array}{cccc} a_1 & a_2 & \dots & a_M \end{array} \\ \begin{bmatrix} 0.3 & 0.4 & \dots & 0.1 \\ 0.4 & 0.8 & \dots & 0.2 \\ \vdots & \vdots & \vdots & \vdots \\ 0.7 & 0.2 & \dots & 0.4 \end{bmatrix} \end{array}$$

p_2 ($[0.4, 0.8, \dots, 0.2]$), which is hence characterized by attribute 1 with 0.4 relevance, attribute 2 with 0.8 relevance, etc.

The task is to estimate the attribute vectors of products that have not been surveyed but belong to the categories for which data are available. The proposal focuses on the users who purchased the products, and applies the basic idea of collaborative filters [10]. In other words, users who chose what to buy based on their set preferences. In such a case, a product purchased by a given user is more likely to have attributes in common with other products purchased by the same user. This allows us to compute similarity scores between products, and infer unobserved values from the available ones.

In summary, the underlying assumptions of the proposal are as follows:

- Consumers buy products with attributes that are consistent with their tastes.
- Similar products have similar attributes.

The frequency of purchase was considered an indicator for preference towards a product. Therefore, the total number of times each item was bought by each user within a fixed period of time was examined. These are the data used to calculate

product similarities.

The conventional similarity measures applied in collaborative filtering have inherent drawbacks that could lead to misleading computed similarities. In particular, they consider only the overall tendencies, thereby failing to consider the contributions of each vector component. This is exemplified in Fig. 3.1 by the quantities in which two products, p_A and p_B, were purchased by two users. PCC cannot be calculated if a product is purchased in the same quantities by all users. Furthermore, it can take on negative values, which is inconsistent with the assumption of positive relevance scores for independent attributes. For the purpose of this study, all negative values were considered to be zero. COS overestimates similarity by ignoring the total number of purchases. For example, cases (a) and (b) have the same COS similarity of 0.7; however, p_A and p_B were purchased by the same user only once in the former case, and 10 times in the latter case. Intuitively, we would expect the higher purchase frequency to indicate a higher similarity. Likewise, JAC considers only the number of users who purchased both products, irrespective of purchase quantities. For example, case (d) has a maximum value of 1 for the JAC measure, although both products were purchased in substantially different quantities.

A similarity measure is defined to avoid the drawbacks of conventional similarities. The proposal is bound between 0 and 1, and assigns an increasing similarity as the total number of purchases increase. In particular, the focus was more on products that had been purchased with the same frequency as that of the target product, and less on those for which the number of purchases disagree. This similarity is referred to as matched preference ratio (MPR).

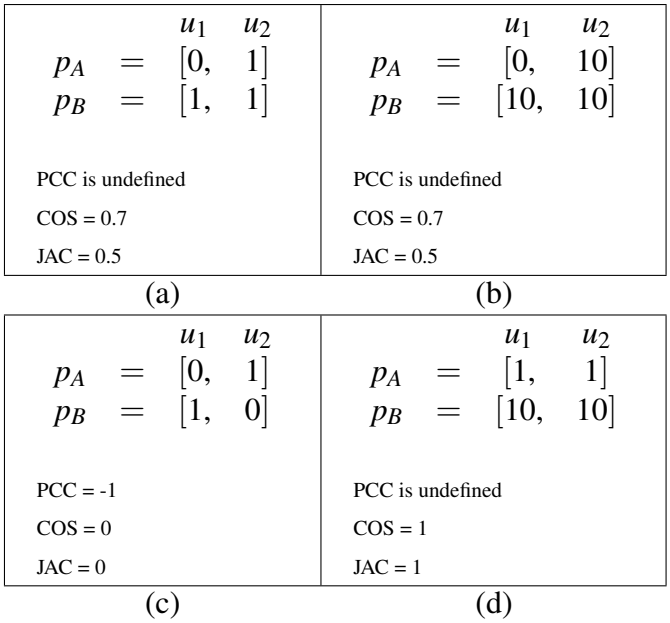

Figure 3.1: Similarity scores between two product vectors, p_A and p_B. Each element of the vector corresponds to purchase quantities by two users, u_1 and u_2. PCC cannot be calculated in cases (a), (b), and (d), and takes on a negative value in case (c). COS overestimates similarities in cases (a), (b), and (d). JAC assigns the same similarity value in cases (a) and (b), and the maximum similarity value in case (d), because it considers exclusively whether a user bought both products or not.

3.4 Proposed method

Considering a set of products $P = \{p_1, p_2, \cdots, p_N\}$ with known M-dimensional attribute vectors $\mathbf{a}(p_n)$ (for each $n \in [1, N]$). The unknown attribute vector of a product q with known similarities to the products in P can be estimated as follows:

$$\mathbf{a}(q) = \frac{\sum_{i=1}^{N} w_i(q)\mathbf{a}(p_i)}{\sum_{i=1}^{N} w_i(q)}, \tag{3.1}$$

where w_i is the similarity of products p_i and q. In particular, this was calculated as follows:

$$\begin{aligned} w_i(q) &= \mathrm{MPR}(p_i, q) \\ &= \frac{\sum_{u \in U} I_a(p_i, q) c_{u,p_i} c_{u,q}}{\sum_{u \in U} I_a(p_i, q) c_{u,p_i} c_{u,q} + \sum_{u \in U} |c_{u,p_i} - c_{u,q}|}. \end{aligned} \tag{3.2}$$

Here, c_{u,p_i} represents the purchase count of product p_i by a user u in the set of all users U. $I_a(p_i, q)$ is an indicator function of agreement, defined as follows:

$$I_a(p_i, q) = \begin{cases} 1 & \text{if } c_{u,p_i} = c_{u,q} \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

The numerator in Eq. 3.2 includes a nonzero term for every user who purchased product p_i in equal quantity as q. Therefore, the similarity score has a first-order relationship with the number of users who purchased both products in equal quantities. On the other hand, each term is the product of purchase amounts for p_i and q; this means that the relationship between the purchase quantity and the similarity score is quadratic. Concerning the denominator, the first sum ensures that the measure is normalized to values between 0 and 1. Meanwhile, the second sum penalizes the score in proportion to the number of users who purchased p_i and q in different quantities. The result is that two products are assigned a higher similarity score when more users purchase them in equal amounts and these quantities are large. In contrast, they are considered less similar if a large number of users purchased them in different quantities; particularly if the difference in purchase quantities is relatively large. These properties mean that the proposed similarity measure depends not only on the direction of the item vector, but also on the magnitude of its components (i.e., the purchase counts). This ensures that the proposal is free from the drawbacks of conventional

similarity measures, outlined in Section 3.3.

If products were classified into categories whose members can be assumed to share similar attributes, the model could be further improved by adding a term to Eq. 3.1; that takes into account the average of all products within the target product category. That is,

$$\mathbf{a}(q) = \frac{\alpha \sum_{i=1}^{N} w_i(q)\mathbf{a}(p_i)}{\sum_{i=1}^{N} w_i(q)} + \frac{(1-\alpha)}{|\mathrm{Cat}(q) \cap P|} \sum_{\rho \in \mathrm{Cat}(q) \cap P} \mathbf{a}(\rho), \tag{3.4}$$

where α is a parameter estimated empirically, and acts as an adjusting factor that controls the relative contribution of each term; $\mathrm{Cat}(q)$ represents the set of products in the same category as q.

3.5 Experiments

3.5.1 Procedure

The data used in this chapter were provided by Rakuten Group's marketing research department, who conducted a survey among a sample of 18,000 Rakuten Ichiba users aged between 20 and 69 years old. The respondents were asked to evaluate 67 different brands of Japanese food products from different categories, such as chocolates, beers, and yogurts. Participation in the survey was voluntary and all respondents agreed to sign a consent form. Respondents were given a list of predefined attributes and asked to select those that characterized each product in the survey. If the respondent did not know the product or did not have any particular opinion about a product, they were encouraged to skip it. The attributes were carefully designed by marketing experts with the intent to describe the most representative features that reflect, at a fundamental level, the consumer sentiment towards a product. Marketing specialists

Table 3.2: Example of the survey format with products and attributes

How do you feel about the following products?
(Check all that apply)

	Tastes sweet	Feels healthy	Is refreshing	...
Product A	☐	☐	☐	☐
Product B	☐	☐	☐	☐
Product C	☐	☐	☐	☐
...	☐	☐	☐	☐

narrowed down the attributes to a total of 39, selecting those that were considered to be the most insightful for the product research process. Examples of the attributes used are "It tastes sweet," "It feels healthy," and "It is refreshing." All products shared the same attributes. Table 3.2 gives an example of the survey format.

The survey results were processed in two steps. In the first step, the replies were aggregated by product to obtain the total counts per attribute. The counts will be referred to as votes, as each count represents a respondent stating that an attribute is relevant to a product. Therefore, a count of zero implies that no respondent associated a particular product with the corresponding attribute. Because not every respondent would review every product, the total votes per product also varied depending on the popularity of the item being evaluated.

The second step involved the normalization of the votes by the total number of respondents per product to account for the effect of the popularity. This was necessary in order to be able to compare attributes between products in terms of the proportion of replies, instead of total number of votes. Without the normalization, the similarity score between products would be affected by the dominant influence of popular ones over those that are less known. That is, the normalization allows for a fair comparison

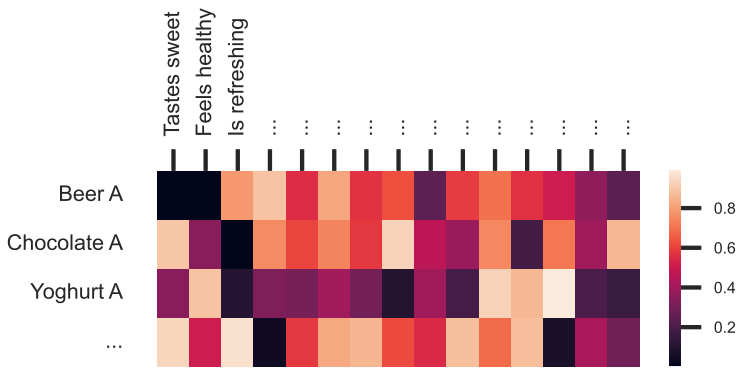

Figure 3.2: Visual representation of products and their attributes. Lighter colors indicate stronger relevance. According to the figure, the attribute "Is refreshing" is considered to be very relevant to "Beer A," whereas attributes "Is sweet" and "Is tasty" are not very relevant. Similarly, attributes "Is healthy" and "Is sweet" are considered to be relevant to "Yogurt A".

of the attributes of different products, regardless of their degree of popularity.

This process allowed us to build a collection of product encodings, in which each product is associated with a vector of attributes. These vectors capture the strength with which each attribute was related to each product on a continuous scale from 0 to 1. For example, the "Is refreshing" attribute for a beer product with a value close to 1 means that most of the respondents found the attribute "Is refreshing" very relevant, whereas the same attribute for a chocolate product could have a value close to 0 (not very relevant), which means most of the respondents did not find the chocolate refreshing. Fig. 3.2 shows a visual representation of each product and its attributes.

To build the product similarity matrix that would be used in later stages to estimate unknown attributes, an item-user matrix similar to that in Table 3.3 was created. The purchase information provided by Rakuten Ichiba in the transaction history logs was employed as an implicit measure of preference; a higher number of items bought indicates stronger preference. Therefore, each element in the item-user matrix is the

Table 3.3: $n \times m$ item-user matrix, where the elements of the matrix represent the total number of items bought per user in a fixed period of time.

$$
\begin{array}{c} \\ i_1 \\ i_2 \\ \vdots \\ i_n \end{array}
\begin{array}{c}
\begin{array}{cccc} u_1 & u_2 & ... & u_m \end{array} \\
\left[\begin{array}{cccc} 0 & 2 & \dots & 1 \\ 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 8 & \dots & 0 \end{array}\right]
\end{array}
$$

total number of items bought per user in a fixed period of time. Subsequently, the similarity between products could be calculated with pairs of product vectors from the item-user matrix and the equations presented in Section 2.3 and 3.4.

The performance of different product attribute estimation models was evaluated. Each of the models calculated attributes based on different similarity measures. The experiments were performed several times using the leave-one-out cross-validation (LOOCV) technique [42]. The reported results are the average of all trials. The result of the proposed similarity measure was compared against three baselines: PCC, COS, and JAC.

Each model was built using a set of "base products," which served as a seed to estimate new attribute vectors. Therefore, in every trial, the set of survey products was split in two disjoint sets. The first set contained all products minus one; this was the "base products" set. The second set contained one single item, the "target product." This product was used to evaluate the accuracy of the estimations.

As it was intended to measure how much the estimations deviated from the real values, the mean absolute error (MAE) and root mean square error (RMSE) were selected as the evaluation metrics. In both cases, a lower value indicates a

better performance; however, RMSE is more sensitive to deviations in individual components.

The procedure to calculate the evaluation metrics using LOOCV comprises the following steps:

1. Considering the set of products $P = \{p_1, p_2, \cdots, p_N\}$ defined in Section 3.4, then for each target product p_t where $t \in [1, N]$ in the survey:

   1.1. Identify the set of all other products $P_{\text{base}} = P - \{p_t\}$ which represent the known products.

   1.2. Compute an estimate for the unknown attribute vector of the target product $\hat{\mathbf{a}}(p_t)$ using Eq. 3.1. Here, p_t and P_{base} represent instances of q and P respectively.

   1.3. Determine the error vector $\boldsymbol{\varepsilon}_t = |\hat{\mathbf{a}}(p_t) - \mathbf{a}(p_t)|$ as the vector of distances between the estimated and the actual attribute values.

   1.4. Compute the MAE and RMSE per target product as follows:

$$\text{MAE}_t = \frac{\|\boldsymbol{\varepsilon}_t\|_1}{M} \tag{3.5}$$

   and

$$\text{RMSE}_t = \frac{\|\boldsymbol{\varepsilon}_t\|_2}{\sqrt{M}} \tag{3.6}$$

   where $\|\boldsymbol{\varepsilon}_t\|_1$ and $\|\boldsymbol{\varepsilon}_t\|_2$ are the ℓ_1 and ℓ_2 norm of $\boldsymbol{\varepsilon}_t$ respectively.

   1.5. Repeat steps 1-4 for all products in P.

2. Average the MAE and RMSE values obtained for all target items as follows:

$$\text{MAE} = \frac{1}{N} \sum_{t=1}^{N} \text{MAE}_t \tag{3.7}$$

and

$$\text{RMSE} = \frac{1}{N}\sum_{t=1}^{N}\text{RMSE}_t \tag{3.8}$$

The experiments considered the coefficient of determination r^2 as the square of the Pearson correlation coefficient. For two series x and y, this is given as:

$$r^2 = \left(\frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum(x_i-\bar{x})^2}\sqrt{\sum(y_i-\bar{y})^2}}\right)^2, \tag{3.9}$$

with x_i and y_i denoting the values in the series, while $\bar{x}$ and $\bar{y}$ stand for their respective average values. The sums run over all elements in the series.

3.5.2 Results

Table 3.4 shows MAE and RMSE values for each model using MPR and the three baseline measures. In this experiment, the purchase history over a period of 50 days was considered. This is also represented in Fig. 3.3 using bar plots with confidence intervals of 99%. The results show that MPR outperformed all other models with the lowest MAE and RMSE values. The plots in Fig. 3.4 show the performance metrics with 99% confidence intervals of the models built with purchase history periods of different lengths. The shortest period was one week, whereas the longest period was 90 days. It is possible to observe that the model built with the proposed similarity measure outperformed all baselines consistently. The low performance exhibited for shorter periods can be attributed to the lack of data to make reliable estimations. When the observations are limited to only a few days, not enough users make their purchases of the relevant products so as to reveal purchasing patterns. For the particular dataset used in the evaluation, the user diversity increased more than 10-fold by day 50 compared to the first 7 days for 11% of all the products. By

Table 3.4: Performance metrics for various models using different similarity measures considering a purchase period of 50 days. The MPR model outperforms all other models, with the lowest MAE and RMSE.

Similarity	MAE	RMSE
MPR	0.040254	0.067911
JAC	0.044553	0.074688
COS	0.044946	0.076635
PCC	0.059692	0.100329

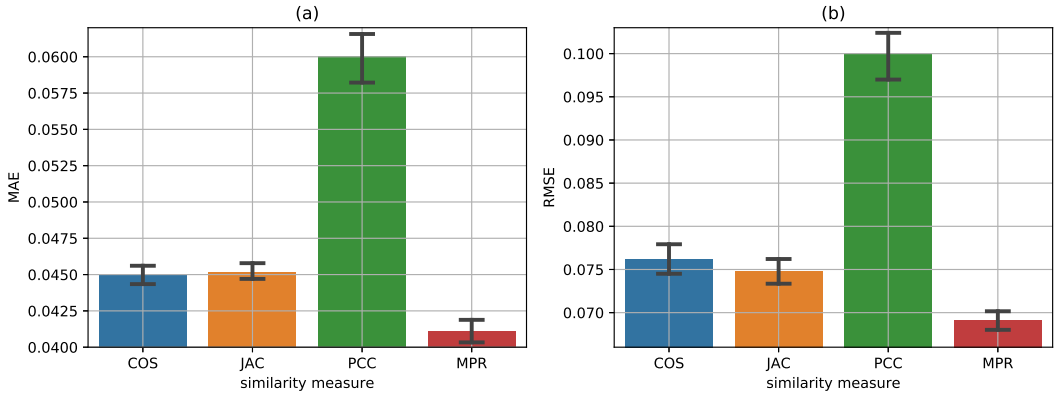

Figure 3.3: (a) MAE and (b) RMSE for various models using different similarity measures considering a purchase period of 50 days.

day 50, the number of items bought is 6.49 times larger than the total items bought during the first week. For all models, the performance increased for larger periods, up to approximately 40 or 50 days; as the confidence intervals did not overlap, the difference was statistically significant. At that point, the item-user matrix provided a reliable estimate of similarity.

Fig. 3.5 shows the performance of various models using different purchase periods. Each bar represents the proportion of times each model achieved the lowest MAE per trial. It is possible to see that, for all periods, the model relying on MPR similarity achieved the lowest error with the highest frequency.

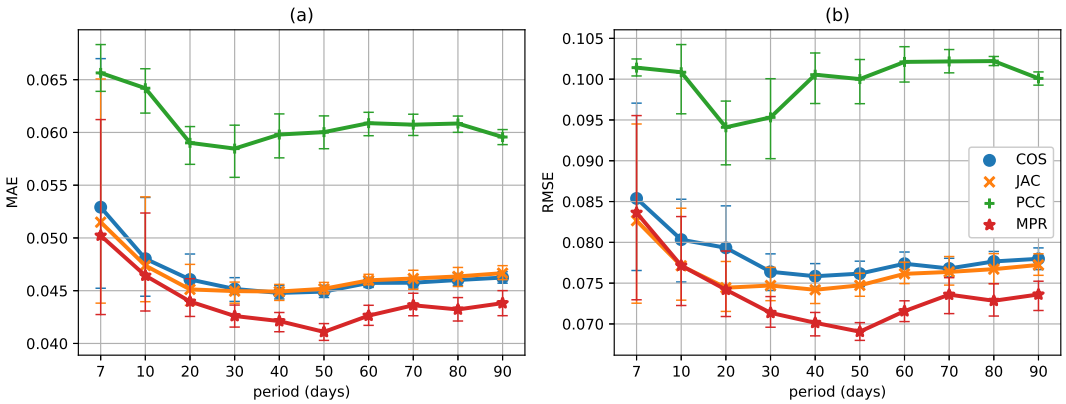

Figure 3.4: Performance comparison in terms of the (a) MAE and (b) RMSE of the baseline and proposed models using different purchase periods.

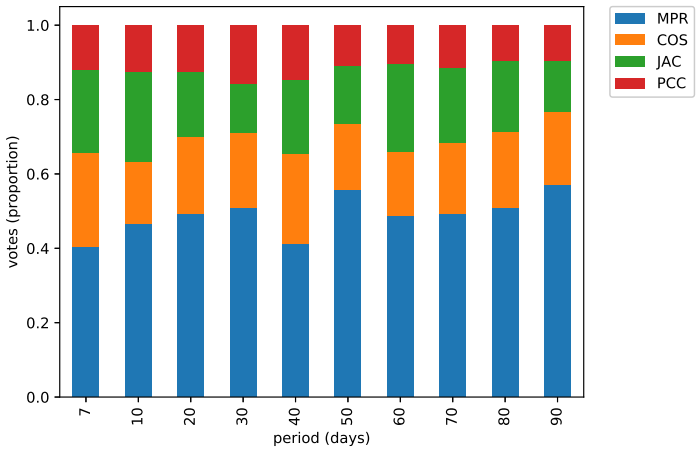

Figure 3.5: Performance of different models regarding the proportion of times each model achieved the lowest MAE per trial.

The experiments presented so far use the attribute information of similar products to compute estimations, regardless of whether they belong to the same category or not. Products from different categories can contribute to the predicted attribute values as long as they are considered similar. To complement this analysis, the impact of using the statistics of the target product's category on the performance of the estimation was investigated. The products were manually classified into categories

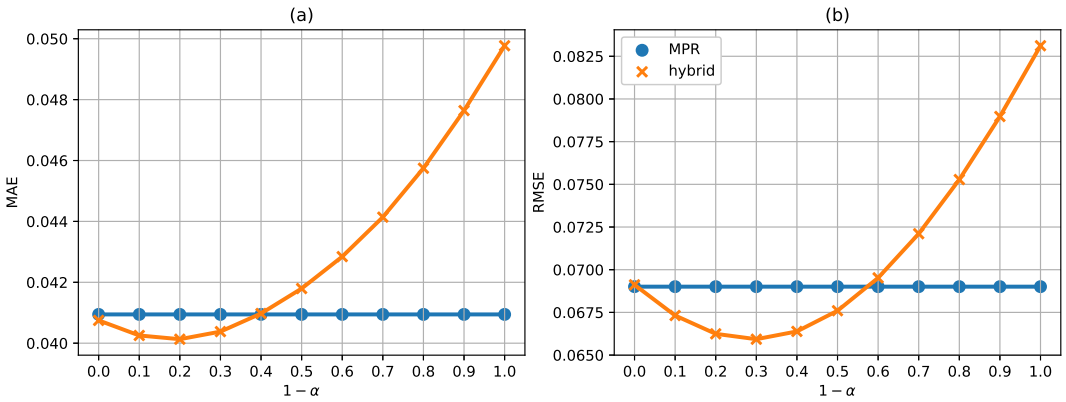

Figure 3.6: Effect of category information on the performance of the estimation. The performance of the hybrid model (Eq. 3.4) improves and worsens as a function of α. When category information is available, some improvement in performance can be achieved by combining both similarity and category information.

by product marketing specialists. However, this is typically costly and unfeasible for large catalogues, for which classification errors usually can expected. To simulate this, a random shuffling of two products per category was performed. Fig. 3.6 shows the MAE (a) and RMSE (b) of two models, with a one-month purchase period, averaged over 10 trials. The model that uses MPR similarity is plotted as a reference and it remains constant because it does not depend on α. In contrast, the performance of the hybrid model (described earlier in Eq. 3.4) improves up to some point as $1-\alpha$ increases, and thereafter deteriorates and becomes worse than the reference performance. This indicates that when category information is available, some improvement in performance can be achieved by combining both similarity and category information. However, the category information alone is not sufficient to produce accurate estimations.

Fig. 3.7 demonstrates the estimated attribute values of two products using the MPR model versus the true attribute values. Product A and product B are examples of

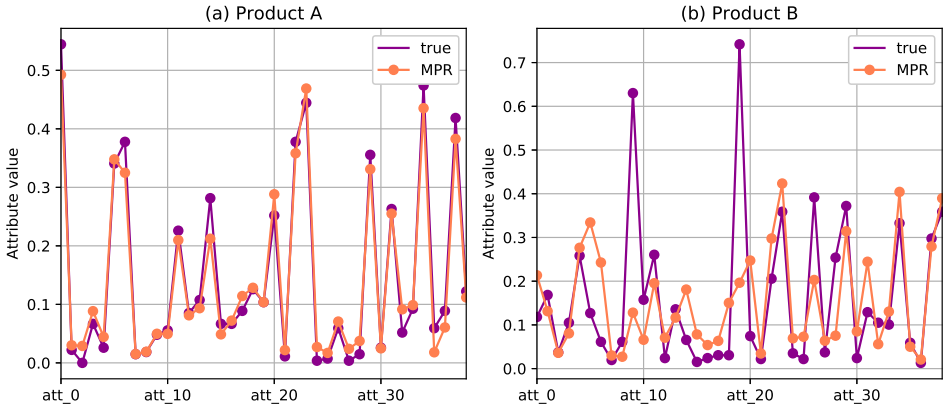

Figure 3.7: Estimated attribute values of two products using the MPR model versus the true attribute values for two products. (a) Example of an estimation with low error. (b) Example of an estimation with high error.

estimations with low and high error, respectively. It is apparent that the estimations of product A follow the true values very closely. However, this is not the case for Product B. Product B is an example of a product with unique attributes, even for its own category. The proposed model failed to give close estimations for such cases because the MPR model computed the unknown attributes as a linear combination of those of similar items. When there were no examples of similar items, the estimations were unsatisfactory.

Fig. 3.8 shows the correlations between the proposed similarity measure and the baseline similarity measures, with the coefficient of determination r^2 calculated according to Eq. 3.9. It is evident that JAC is correlated; however, it tends to assign lower similarity values than MPR. The correlation between JAC and MPR arises from the fact that most items were purchased in smaller quantities. One of the main features of MPR is the weight it gives to instances where items were purchased with higher frequency, or when the difference between purchase amounts was larger. This gives MPR an edge over JAC. COS exhibits a similar trend, but there is a subset of

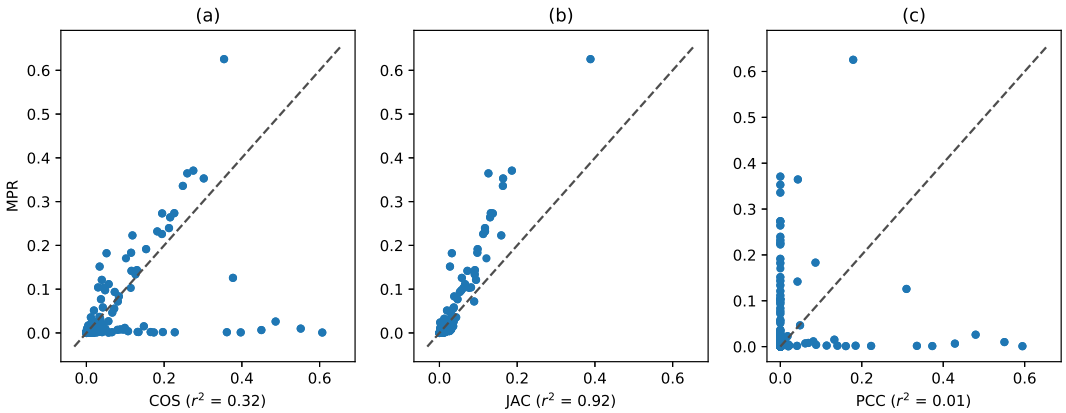

Figure 3.8: Correlation between MPR similarity measure and the baseline similarity measures. (a) MPR assigns low similarity in a subset of products, whereas COS varies widely. (b) MPR and JAC are considerably correlated, however, most points lie above the diagonal in the plot. This means that JAC tends to assign lower similarity values when compared to MPR (i.e. it underestimates similarity values compared to MPR). (c) MPR and PCC are uncorrelated and appear to be looking at different aspects of the data.

products for which MPR gives a low similarity, whereas the COS measure varies widely. This corresponds to products that were purchased in smaller quantities; therefore, MPR assigns a lower weight to them, whereas COS considers only the angle between row-vectors of the item-user matrix, which can take any value, irrespective of purchase numbers. Finally, PCC is uncorrelated; it appears that both measures focus on independent features of the data.

Fig. 3.9 shows a scatter plot of the predicted values against the actual values for each attribute and every product. The estimated values of a perfect model would all fall on the diagonal. The coefficient of determination r^2 is calculated as shown in Eq. 3.9. The MPR model achieves the highest r^2 among all models, meaning that the error in its estimations is lower than that of the other models.

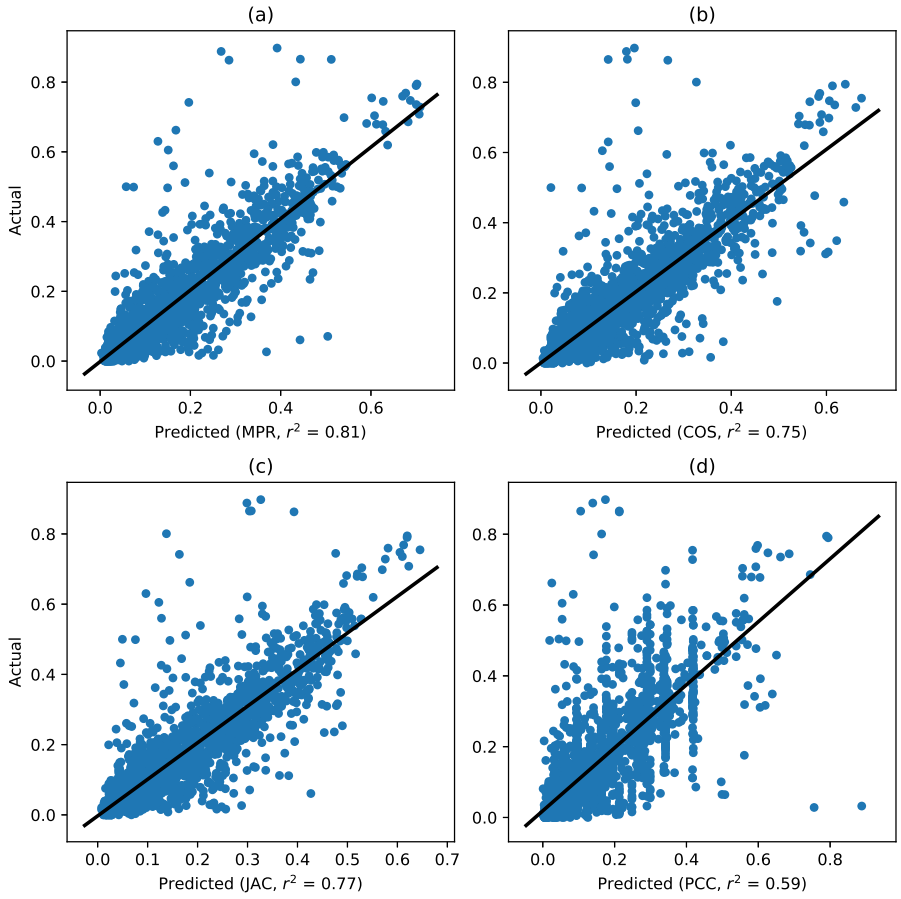

Figure 3.9: Predicted versus actual scatter plots for each similarity measure. Each point is the value of an attribute for every product. MPR achieves the highest r^2 among all models.

3.6 Conclusions

In this chapter, a method for estimating product attribute vectors based on survey data and purchase history was proposed. This is particularly useful for practical applications in marketing or e-commerce where resources are limited and acquiring additional data to train conventional machine learning models is time-consuming and expensive. The proposed method involves estimating unknown attribute vectors as the weighted average of those of known products. The values of those weights depend on

the degree of similarity between the target product and the base products, i.e., products with known attribute values. Furthermore, a new similarity measure that is designed based on domain-specific assumptions was proposed. The described approach relies on the total number of items bought as an implicit measure of preference. The proposed method assigns a larger similarity score to products that were bought in equal amounts, and a smaller one as the difference in purchase amounts increases. This feature allows us to overcome the weaknesses of the conventional similarity measures. The proposed method was applied to data surveyed by Rakuten Group's marketing research department and the purchase history from Rakuten Ichiba. The results obtained indicate that the proposed method outperforms the conventional approaches in terms of MAE and RMSE.

Chapter 4

Switching Hybrid Method Based on User Similarity and Global Statistics for Collaborative Filtering

This chapter presents a method to improve the performance of Collaborative Filtering (CF) techniques. In particular, by proposing a custom similarity measure that considers the semantic nuances of the ratings. This is achieved by weighting the contributions of ratings in proportion to the users' degree of indifference towards the items. Additionally, to address the pervasive sparsity problem in CF, this chapter introduces a switching hybrid method that predicts user ratings based on either the custom similarity measure or through user and item biases. The proposed method was evaluated on six different datasets and compared it with other CF methods using in terms MAE and RMSE. The results show that the proposed recommender consistently outperforms those using conventional similarity measures when the sparsity of the dataset is high.

4.1 Introduction

Decision making involves the evaluation and selection of an option between alternatives based on specific criteria. Classical decision theory is associated with the identification of optimal decisions, considering an ideal decision maker who is internally consistent and completely rational. However, such a logical system fails to explain real-world scenarios [43]. Under conditions where available information is incomplete or overly complex, decision makers rely on various simplifying heuristics or efficient rules of thumb rather than extensive analytical processing [44]. However, in highly difficult situations, they may be unable to make a decision; this is known as analysis paralysis. One of the main causes of analysis paralysis is choice overload. Recent research argues that situations in which several options are presented can negatively affect consumer expectations. This leads to choice deferral and lower satisfaction with the selected option [45].

E-commerce websites rapidly grew with the digital revolution. Online stores now provide users with a more comprehensive search than their brick-and-mortar counterparts, delivering information for an immense number of products and services instantaneously. However, this high number of choices has become one of the biggest problems that e-commerce users face [46]. Rather than being a benefit, having several options frequently overwhelms users, leading them to make poor decisions [2]. This choice paradox is an important catalyst for the development of recommender systems (RS) technology.

RS are personalized information agents that provide suggestions for products or services likely to interest a particular user [2], [47]. RS assists users in filtering the available alternatives and narrowing down their choices, leading to a reduction in

Table 4.1: Classes of recommendation techniques and their knowledge sources

Technique	**Knowledge source**
Collaborative-based	Rating information of different users
Content-based	Item features with descriptions similar to others that the target user liked in the past
Knowledge-based	Inferences regarding users' needs and preferences based on the target user's explicit feedback
Community-based	Social relationships between users and preferences of the target user's friends
Demographic	Demographic profile of the target user
Hybrid	Combination of two or more sources

information overload and preventing analysis paralysis. The essence of RS is the assumption that customer interests can be inferred from different sources of data and that significant dependencies exist between user and item interactions. Because user attitudes toward products have been demonstrated to exhibit a certain degree of consistency [48], they are useful indicators of future choices. The purpose of RS is not only to provide users with more relevant choices but also to help the user discover new, particularly interesting, unexpected, and diverse content. Moreover, this enhances the overall shopping experience and contributes significantly to customer satisfaction [49] and loyalty [50].

A conventional method for classifying recommendation techniques [2], [47] is shown in Table 4.1. CF methods are of interest because they are considered to be one of the most popular and widely implemented recommendation techniques, often exhibiting high predictive accuracy [51], [52], [13], [17].

In particular CF neighborhood-based approaches, described in Section 2.1, are often preferred in practice because they are intuitive, relatively simple to implement, stable, and highly explainable [53], [54], [10]. However, the performance of these methods is known to be highly sensitive to the number of observed ratings. When the

rating sparsity is high, it becomes difficult to identify suitable matching neighbors, leading to less accurate recommendations. This is closely related to the cold-start problem, where it is not possible to provide reliable recommendations to new users or about new items because of the initial lack of ratings [55], [56].

As explained in Section 2.3, another critical aspect that has a significant impact on both the accuracy and performance of neighborhood-based recommenders is the choice of the similarity metric. The similarity metric identifies reliable neighbors and weights their relevance in the prediction. Its definition must capture the aspects of the relationships between users and items that are most representative of user preferences.

The present dissertation proposes a novel CF approach for predicting user rating values. It is designed to avoid common problems of conventional similarity-based methods and mitigate the adverse effects of highly sparse data. In particular, the contributions of the study in this chapter are as follows:

- A new custom similarity measure that considers the semantic meaning of the ratings, with a score bounded between 0 and 1 for higher interpretability.
- The use of the global average, item average, and user average to compute user and item biases that in turn are used to compute ratings.
- The proposal of a switching hybrid method that chooses between two predicted ratings based on a sparsity criterion.

The main difference between the proposed method and existing methods lies in the novel similarity measure introduced in Section 4.3.1. The novelty aspect consist primarily in that the similarity was designed to consider the user's sentiment toward the items, interpreted in terms of how extreme their ratings are. Moreover, it avoids the

drawbacks of conventional similarity measures. In addition, an RS architecture based on this similarity is proposed. This is different from other approaches in the sense that the recommender, described in Section 4.3.3, uses a switching hybrid method to handle sparsity and complements its predictions with those of a recommender using Jaccard similarity. This architecture is shown to outperform conventional ones, as described in Section 4.4.3.

Section 4.2 offers an overview of the main problems and recent research associated with conventional similarity measures, as well as the difficulty in making meaningful predictions for highly sparse datasets. Section 4.3 proposes the Switching Hybrid with Biases (SHB) recommender comprising a CF method with a custom similarity measure and a ratings prediction approach based on global statistics. This is used in combination with a CF method using Jaccard similarity. The custom similarity measure of the SHB recommender was designed to consider the semantic information in the ratings to generate more meaningful similarity scores, leading to better performance even if the dataset sparsity is high. Section 4.4 outlines the experimental settings used in the evaluation of the proposal, including the details of the six datasets used. Section 4.4.3 analyzes the performance of the proposed recommender in terms of root mean square and mean absolute errors and compares it to those of conventional similarity-based CF methods. Section 4.5 summarizes the findings and conclusions.

4.2 Related work

This section describes in more detail the technical context of the work discussed in this chapter. Furthermore it includes related methods that address problems in the area of similarity measures applicable to RS technology, and different methodologies to

tackle the common sparsity problem. In particular, influential work is examined and the limitations are reviewed.

4.2.1 Similarity measures

As introduced in Section 2.3, while PCC, COS, JAC are some of the most broadly used similarity measures, they are known to have inherent drawbacks that might hurt the performance of RS.

To overcome these deficiencies, researchers have proposed different approaches. Candillier et al. [13] demonstrated that weighting similarity measures such as COS and PCC with JAC significantly improves the methods' performance. In this context, JAC ensures that the rating pairs share sufficient attributes for the similarity to be reliable. Ahn [57] developed a heuristic similarity measure based on three aspects –proximity, impact, and popularity (PIP)– representing domain-specific interpretations of user ratings. PIP weights the semantic agreements and disagreements between users and exhibits a superior performance in cold-start conditions. However, its formula is expensive to compute, and because its score is not normalized, it can assume values greater than 1 making it less intuitive and difficult to interpret.

Liu et al. [58] proposed another similarity metric termed as NHSM to address the shortcomings of PIP. This proposal uses the definitions of proximity, significance, and singularity, combined with JAC, along with the mean and variance for the ratings, to achieve a remarkable improvement over PIP. Said et al. [59] proposed and investigated the effects of two weighting schemes that consider the degree of popularity of items. The results indicate that the weighting approaches have a negligible effect on COS but have a significant influence on PCC when the users have more than a few ratings

in common.

Bobadilla et al. [56] presented a similarity measure that combines conventional similarity metrics; however, it optimizes the weights of each similarity through neural networks. The results show an improvement in the prediction quality in cold-start situations. This metric is superior to PIP in terms of performance and computation time. Laveti et al. [60] proposed a weighted hybrid ensemble similarity metric combining two or more conventional approaches that demonstrates an improvement in recommendation accuracy; however, it relies on a high number of rating samples and neighbors to achieve high performance. Guo et al. [15] defined a Bayesian similarity measure based on the Dirichlet distribution that considers the direction and length of the rating vectors while also considering the rating semantics of all rating pairs. Although the approach can perform well and can attain good generality, it includes a number of hyperparameters that require tuning to achieve these results.

4.2.2 Sparsity in neighborhood-based models

In practice, users tend to rate a small subset of the item catalogue, consequently making user-item matrices sparse. Sparsity is a major problem in RS because when most ratings are unspecified, finding reliable neighbors is difficult, and the number of co-rated items between users is small. In scenarios with high levels of sparsity, recommendations are often biased and inaccurate. One of the earliest approaches for tackling sparsity in rating matrices is "default voting," proposed by Breese et al. [61]. Default voting involves using predetermined values to replace missing data and increase the number of mutually rated items between users. Default votes are only applied in cases in which two users are compared and at least one of the users rated

the item. However, replacing missing data usually introduces a significant amount of bias that affects the performance of the estimations.

Wang et al. [62] developed an algorithm that combines ratings from both similar users and items to reduce the dependency on missing data. The experiments demonstrate robustness against data sparsity and improved prediction accuracy when compared with pure user-based or item-based approaches. Zhang et al. [63] proposed a recursive algorithm that can make coarse predictions for the missing rating values of neighboring users, thus alleviating data sparseness. The proposed approach shows promising results, achieving higher prediction accuracy than the conventional approach using PCC. However, the performance of this algorithm also depends on several parameters tuned to every specific dataset, and it has a high computational cost.

Luo et al. [54] addressed the data sparsity problem by describing relationships between users through local and global user similarities. Local similarities are represented as edges of a user graph and are determined based on surprisal-based vector similarity. Global similarities are calculated as the maximin distance of any two nodes in the graph. The results show that, under sparse dataset conditions, the global user similarity can improve the performance of algorithms that use only local similarities.

Another approach for solving the data sparsity problem was proposed by Bessa et al. [64], who advanced a method that considers user communities with similar tastes and predicts new relations within these communities. The results reveal that although this method did not increase the global coverage, it improved the predictions of already covered items, alleviating some of the drawbacks of the sparsity problem. Hawashin et al. [65] proposed a hybrid similarity measure based on explicit user

interests; the proposed method achieved good performance even when no co-rated items existed between two users. However, it did not consider the semantic meanings of the ratings, and it depended on the existence and quality of explicit user interests. Moreover, the process of calculating user interests was computationally expensive.

4.3 Proposed method

In this section, the proposed CF approach for predicting user rating values is detailed. The method was inspired by the strengths of previous approaches while also considering their weaknesses. However, the proposed method retains the characteristics that make neighborhood models widely used; that is, their relative simplicity, ease of maintenance, and high explainability. In real-world use scenarios, neighborhood approaches should remain easy to tune and should not depend on complex optimization procedures that require long computation times, especially if they must be updated every time a new user or item is added.

The proposed recommender, SHB, is an ensemble of a switching hybrid and a basic collaborative filter with JAC. The switching hybrid consists of a CF method that uses the proposed similarity measure, and a CF technique that computes the prediction based on global statistics. To determine the final prediction, first, the target user is given similarity scores with respect to the other users. These serve as the weights in Eq. 2.2 for predicting the desired rating. The proposed similarity measure used in this step considers the distance between user ratings as well as the deviation between these ratings and the median of the rating scale. If the user-item matrix is highly sparse, the target user might not have sufficient neighbors to make a meaningful prediction; that is, the recommender will rely on users with low similarity with the target, which

may often lead to inaccurate predictions. In this situation, the switching hybrid relies instead on the biases for the target user and item as well as on the global mean. As a rule of thumb, the switching criterion considers a minimum of 10 neighbors as the threshold. Finally, an alternative prediction is derived from a neighborhood-based recommender using JAC. This is combined with the prediction of the switching hybrid to yield the final rating.

4.3.1 Proposed similarity measure

As explained in Section 4.2.1, several studies have exhibited promising results by incorporating the semantic meaning of the ratings when defining new similarity measures. Studying the semantic nuances of user ratings is essential for developing a measure of similarity that can realistically quantify the resemblance among user tastes. The purpose of a rating scale is to allow respondents to express both the direction and strength of their opinions regarding a topic [66]. Ratings, unlike other implicit measures of interest, such as total clicks or number of items purchased, are ordinal categorizations that can be assumed to reflect the user's degree of indifference toward an item. It is possible to observe that to compare two users' ratings, it is not only useful to consider the difference between the ratings but also how far the ratings are from the rating scale's median. The intuition behind this is that users who give ratings closer to the median tend to feel more indifferent toward the item, whereas users with more extreme ratings reflect stronger preferences. Furthermore, the fact that stronger preferences provide more reliable information for determining whether or not two users have similar tastes is considered.

A new similarity measure that considers two main aspects: the difference between

user ratings and how extreme these are, is proposed. For users u and v who have rated item i with ratings $r_{u,i}$ and $r_{v,i}$, respectively, the distance $\text{sqd}(r_{u,i}, r_{v,i})$ is determined as follows:

$$\text{sqd}(r_{u,i}, r_{v,i}) = (r_{u,i} - r_{v,i})^2. \tag{4.1}$$

The squared difference is used to ensure that this is a smooth and symmetric measure of the distance between ratings. Without loss of generality, let $r_s = \{1, 2, ..., \text{R}_s\}$ represent an integer rating scale with a median given as follows:

$$\tilde{r}_s = \frac{R_s + 1}{2}. \tag{4.2}$$

The rating strength r_p for a pair of user ratings is defined as follows:

$$r_p(r_{u,i}, r_{v,i}) = \frac{\max(|r_{u,i} - \tilde{r}_s|, |r_{v,i} - \tilde{r}_s|)}{\tilde{r}_s - 1}. \tag{4.3}$$

This satisfies $0 \leq r_p \leq 1$ and reflects how extreme the ratings are. Higher r_p values indicate more extreme ratings. A value of 1 indicates that at least one of $r_{u,i}$ or $r_{v,i}$ is equal to either the lowest or highest possible rating. Meanwhile, a value of 0 implies that both $r_{u,i}$ and $r_{v,i}$ are equal to the median.

A measure of dissimilarity $\text{dis}(u, v)$ between users u and v can now be expressed by the following equation:

$$\text{dis}(u, v) = \frac{1}{|I_u \cap I_v|} \sum_{i \in I_u \cap I_v} \text{sqd}(r_{u,i}, r_{v,i}) \cdot r_p(r_{u,i}, r_{v,i}), \tag{4.4}$$

where I_u and I_v denote the sets of items rated by users u and v, respectively. The corresponding similarity measure between users u and v is defined as follows:

$$\text{sim}(u, v) = \frac{1}{\text{dis}(u, v) + 1}. \tag{4.5}$$

4.3.2 Predicted ratings from user and item biases

In highly sparse datasets, the target user may have very few neighbors. When this happens, ratings predicted using Eq. 2.1 are influenced by users with low similarity to the target. This is problematic because users who are weakly correlated with the target user may cause the recommender to yield inaccurate predictions. This phenomenon can also be explained as overfitting caused by the normalization factor in Eq. 2.1 when only low-similarity neighbors are present. Moreover, when the target user has no neighbors, the rating for that specific user-item combination cannot be predicted. In these cases, the naive approach involves reporting the global average or the average of all the ratings by the target user [10]. As an alternative, the global, item, and user averages are considered to compute the predicted rating when the number of neighbors found is insufficient.

In the proposed method, The user bias (the deviation between the user rating average and total average) is calculated and added to the item average. Formally, the predicted rating $\hat{r}_{u,i}$ of user u for item i is calculated as follows:

$$\hat{r}_{u,i} = \bar{r}_i + (\bar{r}_u - \bar{r}) \tag{4.6}$$

where $\bar{r}_i$ denotes the average rating for item i by all users who rated it, $\bar{r}_u$ is the average of all items that user u rated, and $\bar{r}$ is the overall average of all observed ratings. The term $\bar{r}_u - \bar{r}$ represents user bias. A user has positive bias if the user has an inclination to rate items favorably. Conversely, when the user's ratings tend to be critical, the user exhibits a negative bias. Following this approach, the ratings of all users for the target item are considered in equal proportion but are regulated by the preferences of the target user, encoded in their rating trends.

A similar idea was proposed by Koren [9], who observed that a baseline estimate for a user-item pair $b_{u,i}$ can be calculated as follows:

$$b_{u,i} = \bar{r} + b_u + b_i \tag{4.7}$$

where b_u and b_i indicate the observed deviations from the average for user u and item i, respectively. To estimate b_u and b_i, Koren proposes solving the following regularized least-squares optimization problem:

$$\min_{b_u, b_i} \sum_{r_{u,i} \in R} (r_{u,i} - \bar{r} - b_u - b_i)^2 + \lambda \left(\sum_u b_u^2 + \sum_i b_i^2 \right), \tag{4.8}$$

where the first sum runs over the elements present in the sparse user-item matrix R. The second expression is a regularization term to avoid overfitting on the optimization procedure, and λ is the regularization parameter that sets how much the flexibility of the model will be penalized. λ is often chosen by testing a range of values and selecting the one that gives a good approximate solution.

The optimization of Eq. 4.8 must be performed over a sparse matrix, typically using approximate methods such as stochastic gradient descent (SGD). Thus, the optimal values of the biases b_u and b_i are updated in steps, following the slope (gradient) of the objective function downward, until a minimum is reached. This is not the case for the proposed method in Eq. 4.6, where the biases are approximated by the explicit formulas $b_u = \bar{r}_u - \bar{r}$ and $b_i = \bar{r}_i - \bar{r}$. The experiments in Section 4.4 provide evidence that both approaches yield comparable results when applied to the SHB proposal.

4.3.3 Proposed recommender system

Subsections 4.3.1 and 4.3.2 introduce two methods to make a rating prediction, each of which performs best under different situations. To choose the optimal method, a switching hybrid method is introduced. The first recommender R1 (neighborhood-based) yields better predictions if a set of representative neighbors can be found in the dataset. Meanwhile, the second recommender R2 (based on rating biases) can more adequately capture the underlying patterns encoded by the global statistics, which prove useful if a better-customized prediction is not available. Consequently, a switching criterion that selects R2 if the number of neighbors is below a threshold value γ, and R1 otherwise is proposed. The recommendations of the switching hybrid recommender are as follows:

$$\hat{r}_{u,i}^{(\mathrm{h})} = \begin{cases} \hat{r}_{u,i}^{(1)}, & \text{if } |N(u,i)| \geq \gamma \\ \hat{r}_{u,i}^{(2)}, & \text{otherwise} \end{cases} \tag{4.9}$$

where $\hat{r}_{u,i}^{(1)}$ denotes the recommendation of R1, calculated using the formula in Eq. 2.2. The recommendation of R2, $\hat{r}_{u,i}^{(2)}$, is computed using Eq. 4.6. The value γ was determined heuristically from the analysis of diverse datasets and set to 10 in the experiments.

As a final step, a hyperparameter α is introduced. This parameter is used to combine the predicted rating from the switching hybrid with that of a standard recommender using JAC. This yields the final prediction of the proposed SHB as follows:

$$\hat{r}_{u,i} = \alpha \cdot \hat{r}_{u,i}^{(\mathrm{h})} + (1-\alpha) \left(\bar{r}_u + \frac{\sum\limits_{v \in N(u,i)} \mathrm{JAC}(u,v) \cdot (r_{v,i} - \bar{r}_v)}{\sum\limits_{v \in N(u,i)} \mathrm{JAC}(u,v)} \right). \tag{4.10}$$

The following is an intuitive, step-by-step description of the proposed algorithm, which is also illustrated in Fig. 4.1.

1. Compute the prediction $\hat{r}_{u,i}^{(1)}$ using Eq. 2.2 and the similarity measure defined in Eq. 4.5.

2. Compute the prediction $\hat{r}_{u,i}^{(2)}$ as described in Eq. 4.6.

3. If $|N(u,i)|$ (total neighbors of u for item i) is greater than or equal to 10, set $\hat{r}_{u,i}^{(\mathrm{h})}$ equal to $\hat{r}_{u,i}^{(1)}$, otherwise set it to $\hat{r}_{u,i}^{(2)}$.

4. Calculate an additional rating prediction using Eq. 2.2 with JAC as the similarity metric.

5. Compute the final rating $\hat{r}_{u,i}$ as the linear combination (weighted by hyperparameter α) of the ratings calculated in the previous two steps, as shown in Eq. 4.10.

4.3.4 Complexity and scalability analysis of the recommender system

To consider the computational complexity of the proposed recommender system, the Big O notation is used. This expresses the asymptotic performance bounds that an algorithm which implements the recommender can be expected to achieve.

The amount of computations required by the proposed method depends on the number of users and items in the dataset. As illustrated in Fig. 4.1, the prediction of rating scores requires the calculation of the user similarity matrix from the user-item matrix. This is the most computationally expensive operation in the recommender,

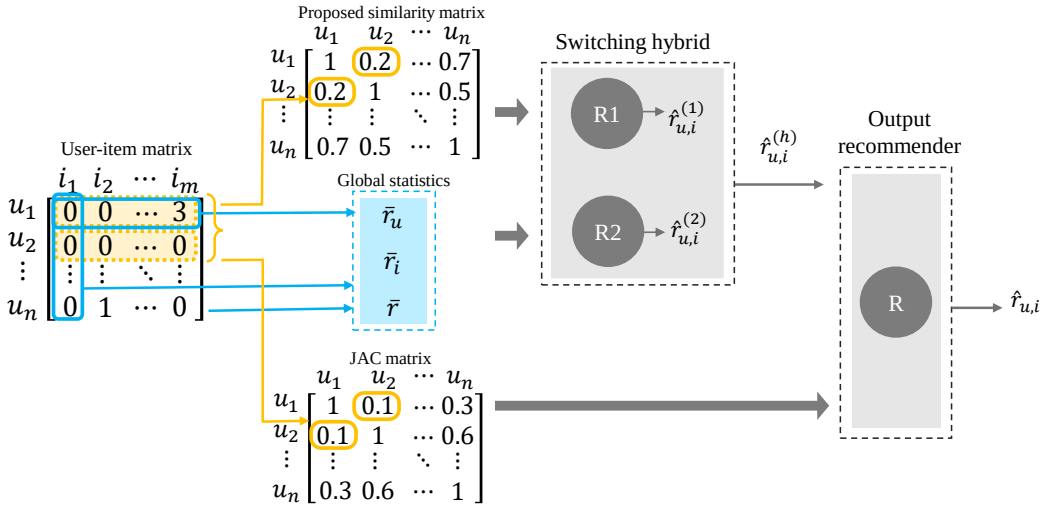

Figure 4.1: Overview of the proposed recommender system. The sparse user-item matrix is used to estimate the global statistics and similarities between users. These serve as inputs to a nested pair of hybrid recommenders.

and thus it determines its time and space complexities, which scale non-linearly with the number of users and items. Algorithm 1 summarizes this procedure, and shows that the total number of similarity calculations needed is $n\frac{(n-1)}{2}$, due to its symmetric nature. Additionally, computing the similarity between two users requires operations between at most m items (in the extreme case where both users have rated all items). The similarity matrix has a total of $n\frac{(n-1)}{2}$ unique entries; therefore, in terms of the

Algorithm 1 Similarity matrix calculation for an $n \times m$ user-item matrix R

Input: user-item matrix R
Output: $n \times n$ user similarity matrix

```
for i ← 1 to n do
    S[i,i] ← 1
    for j ← i + 1 to n do
        S[i,j] ← similarity(i,j)
        S[j,i] ← S[i,j]
    end for
end for
return S
```

dataset size, the time and space complexities associated with generating and storing the user similarity matrix are $O(|U|^2 \cdot |I_u|)$ and $O(|U|^2)$ respectively. It should be noted that these complexities also apply to other commonly used similarities, such as PCC, COS or JAC.

The high time complexity involved in the computation of user similarity matrices makes their online update impractical for situations such as e-commerce systems, in which the number of users and items can exceed tens or hundreds of millions. This poor scalability problem is a known challenge that affects not only the proposed method, but is inherent to neighborhood-based methods in general [67], [68]. Several works have attempted to address the scalability problem in CF [69], [70], [71]. While it is beyond the scope of this study to conduct an extensive analysis of these techniques, it is worth mentioning that common applications need not update the similarity matrix online. Given that its calculation could take hours or even days to complete for large real-world datasets, it is common to pre-compute it offline and assume it remains constant throughout a subsequent, low-latency online phase which computes the rating predictions. Additionally, the memory requirements to store the pre-computed similarity matrix can be significantly reduced if, instead of storing it entirely, only the tuples of users with a similarity larger than a predetermined threshold are kept.

4.4 Experiments

4.4.1 Procedure

In this section, the experiments to test the performance of the proposed SHB recommender are presented. The results are compared with those of five different

neighborhood-based approaches, two MF methods, and the global average as a baseline. Each neighborhood-based approach uses a different similarity measure. PCC, COS, and JAC, as well as the hybrid variations PCC+JAC and COS+JAC were selected because of their widespread use in practical applications. Other related approaches are not included in the experiments because of their drawbacks detailed in Section 4.2. In particular, they might require extensive dataset-dependent parameter tuning to reach their optimal results or additional user or item features to complement their predictions. Furthermore, some of the most complex approaches do not provide available implementations that can scale to the size of the datasets used in these experiments.

The tests are conducted using five real-world public datasets and one synthetic dataset. This work considers the application of RS to predict user preferences (in the form of ratings); therefore, their accuracy is measured by comparing the predicted rating against the ground truth. The accuracy is reported in terms of the RMSE and MAE. These metrics are defined as follows [2]:

$$\text{RMSE} = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} (r_{u,i} - \hat{r}_{u,i})^2}, \tag{4.11}$$

and

$$\text{MAE} = \frac{1}{|T|} \sum_{(u,i) \in T} |r_{u,i} - \hat{r}_{u,i}|, \tag{4.12}$$

where T denotes the test set, $r_{u,i}$ the ground truth, and $\hat{r}_{u,i}$ the predicted rating. The tests are performed using the k-fold cross-validation method. The reported results are the average of the cross-validation trials, in which smaller values of RMSE and MAE indicate better predictive accuracy. The plots presented are calculated with error bands showing 99% confidence intervals.

The two error metrics, RMSE and MAE, each provide a different piece of information which may be more useful depending on the use case for the recommender. The squared component in the RMSE indicates large errors, which makes it more suitable in applications where even a few large deviations are undesired. In contrast, MAE is a more impartial indicator of the typical error and tends to be more robust to outliers. Another approach used to evaluate RS involves considering the task as a classification problem. When the recommendation task is not to predict ratings, but to generate a list of interesting items, then the score is only used to determine if the item should be recommended or not. In this case, metrics such as precision and recall are more suitable. If the order of the recommended list is also important, then metrics such as normalized discounted cumulative gain and mean reciprocal rank are useful to evaluate both the adequacy of the selected recommendations and their ranking.

4.4.2 Evaluation datasets

The real-world datasets used in the experiments are summarized in Table 4.2. Different types of datasets were used to verify the generalization power of the results. All real-world datasets are publicly available for research use and vary in terms of the total number of users, items, ratings, sparsity, and the type of items being rated. MovieLens 100k and MovieLens 1M [72] contain the ratings of users who evaluated at least 20 movies. These users were randomly sampled from the complete MovieLens dataset, which contains 26,000,000 ratings from 270,000 users and 45,000 movies. Epinions [73] and Book-crossing [74] data were collected using a crawler that browsed over epinions.com and bookcrossing.com, respectively. The Epinions dataset comprises user ratings from various consumer items, whereas the Book-crossing dataset

Table 4.2: Characteristics of the real-world datasets used in the experiments

Dataset	Total users	Total items	Total ratings	Rating scale	Sparsity	Items category	Source
MovieLens 100k	610	9,724	100,836	[0.5, 5]	0.983	Movies	MovieLens website
MovieLens 1M	6,040	3,706	1,000,209	[1, 5]	0.9553	Movies	MovieLens website
Epinions	40,163	139,738	664,823	[1, 5]	0.9995	Consumer items	Epinions website
Book-crossing	105,283	340,556	1,149,780	[0, 10]	0.9999	Books	Book-crossing website
Jester (dataset 3)	54,905	140	1,842,370	[-10, 10]	0.7603	Jokes	Jester joke recommender system

comprises book ratings by members of an online book club. The complete Jester [75] dataset contains 6.5 million anonymous ratings of jokes that were collected from users of the Jester joke recommender system. Consequently, the Jester (dataset 3) is used in the experiments. This dataset is similar to the MovieLens datasets, containing a subset of the complete dataset after some jokes were removed and only users who rated 36 or more jokes were included. In the case of the Epinions and Book-crossing datasets, a similar filtering was applied to retain only the data for users who rated at least 20 items.

Next, the generation of the synthetic datasets is described in detail. To generate data for the synthetic dataset, the following assumptions were considered:

- Users can be classified into types, where a type is a cluster of users who share similar tastes and interests. CF operates under the same assumption, identifying users in the same cluster, and calculating ratings based on the proximity of users within the cluster.
- The probability of a user being included in a particular user type is the same for all types. This assumption might not hold in real-world settings because some types might be more popular than others. However, the probability was set to be uniform to simplify the data generation process.

- All elements of the full matrix have the same probability of being missing in the sparse matrix. This assumption will not reproduce the known biases in the rating data. For example, users may be more likely to rate items for which they have strong opinions or some users may provide many more ratings than others [2].

The ratings in the synthetic dataset are the output of a random process modeled by a categorical distribution with probabilities for each possible score given for each pair of user type (U_T) and item (i). These probabilities were decided from a separate random process such that the histogram of the ratings for the (U_T, i)-pair will approximate a gaussian with a randomly selected mean. In addition, they are chosen to ensure that the full dataset has a specified mean and standard deviation. The parameters used to generate the synthetic data are listed in Table 4.3.

The advantages of having a synthetic dataset are that it allows us to freely evaluate the proposed method under different conditions, such as different levels of sparsity or types of users, while having complete knowledge regarding the underlying patterns in the data as well as the ground truth for the full user-item matrix. In particular, this dataset allowed us to study the performance of different RS as sparsity increases. The same analysis using a subset of real-world datasets would have added the risk of bias of the data by the subsampling scheme.

4.4.3 Results

The first analysis focuses on the proposed similarity metric of Eq. 4.5 and compares its performance with that of other conventional similarity measures, along with the global

Table 4.3: Parameters used to generate the synthetic data

Parameter	**Value**
Number of users	500
Number of movies	10,000
Rating scale	[1, 5]
Number of user types	5
Target mean for generated data	3.8
Target standard deviation for generated data	1.5
Sparsity	0.983

mean as a baseline. To focus exclusively on the impact of the choice of similarity, the rating predictions using Eq. 2.1 were computed. Therefore, this first analysis does not consider the proposed recommender SHB in its entirety but instead investigates the effects of the proposed similarity (SIM) on its own. Figs. 4.2 and 4.3 shows different plots of the RMSE and MAE for different values of k-NN. All similarities attained their highest RMSE and MAE values when the number of neighbors was small; as the number of neighbors increased, the performance improved asymptotically. Moreover, excluding the Book-crossing dataset, the proposed similarity (referred to as SIM in the figure) achieved the best performance, with the lowest RMSE and MAE values, indicating that the semantic information exploited by the method can efficiently discover rating patterns among like-minded users. Furthermore, the proposed method exhibits a more consistent behavior across datasets than other similarity metrics.

The second part of the analysis considered the proposed recommender SHB based on Eq. 4.10. Its performance is compared with those of other CF methods. In addition, two widely used MF recommenders, FunkMF and SVD++, are included in the comparison. Although these MF approaches are known to have better performance than neighborhood-based methods, this is achieved at the cost of interpretability and a higher maintenance cost. Note that the purpose of including this comparison is not to

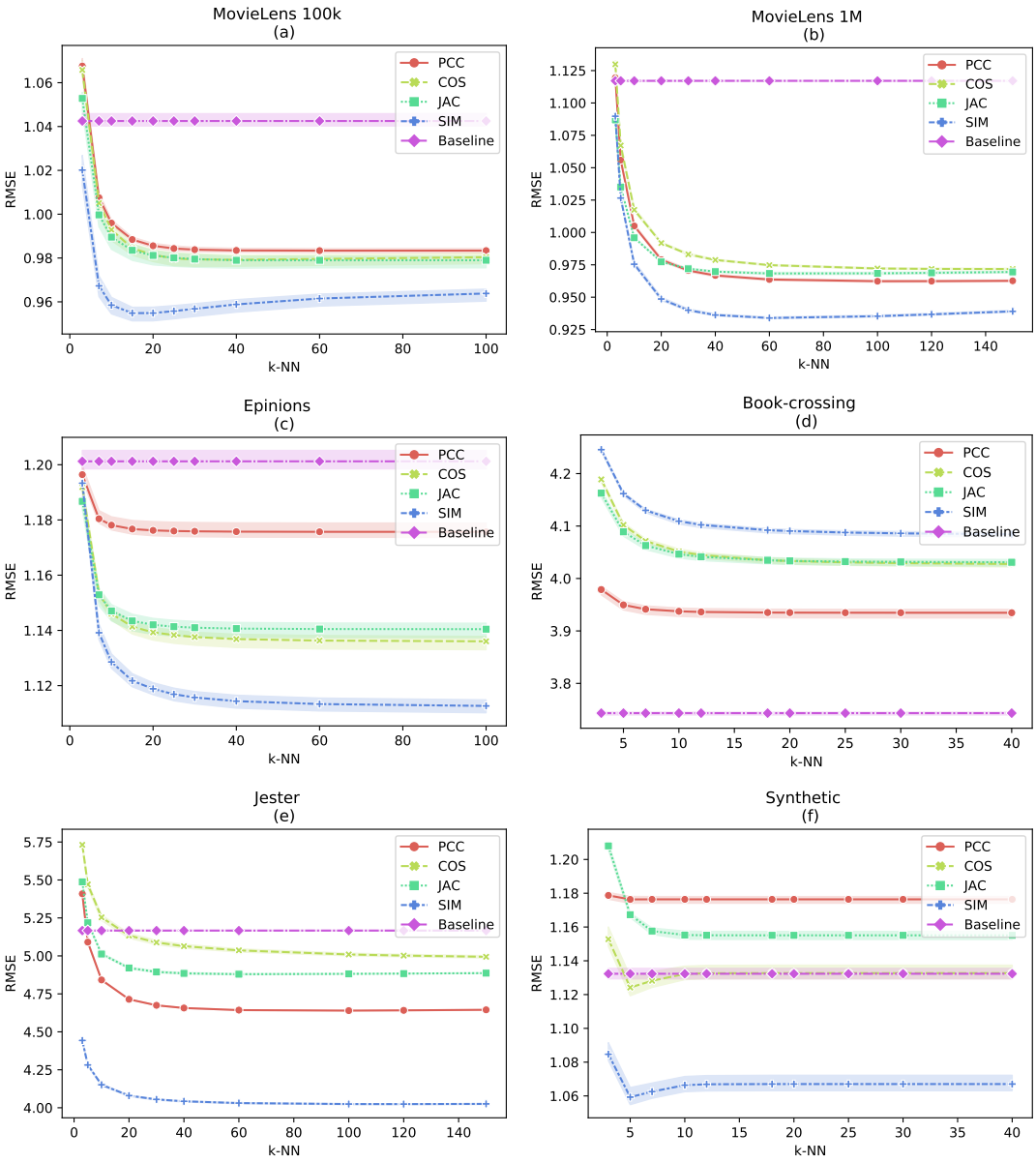

Figure 4.2: Recommendation performance of various neighborhood-based methods without rating normalization and using different similarity measures. The global mean is also shown as a baseline. RMSE values presented for different k-NN values. The proposed similarity (SIM in the plots) displays better general performance than that of other methods.

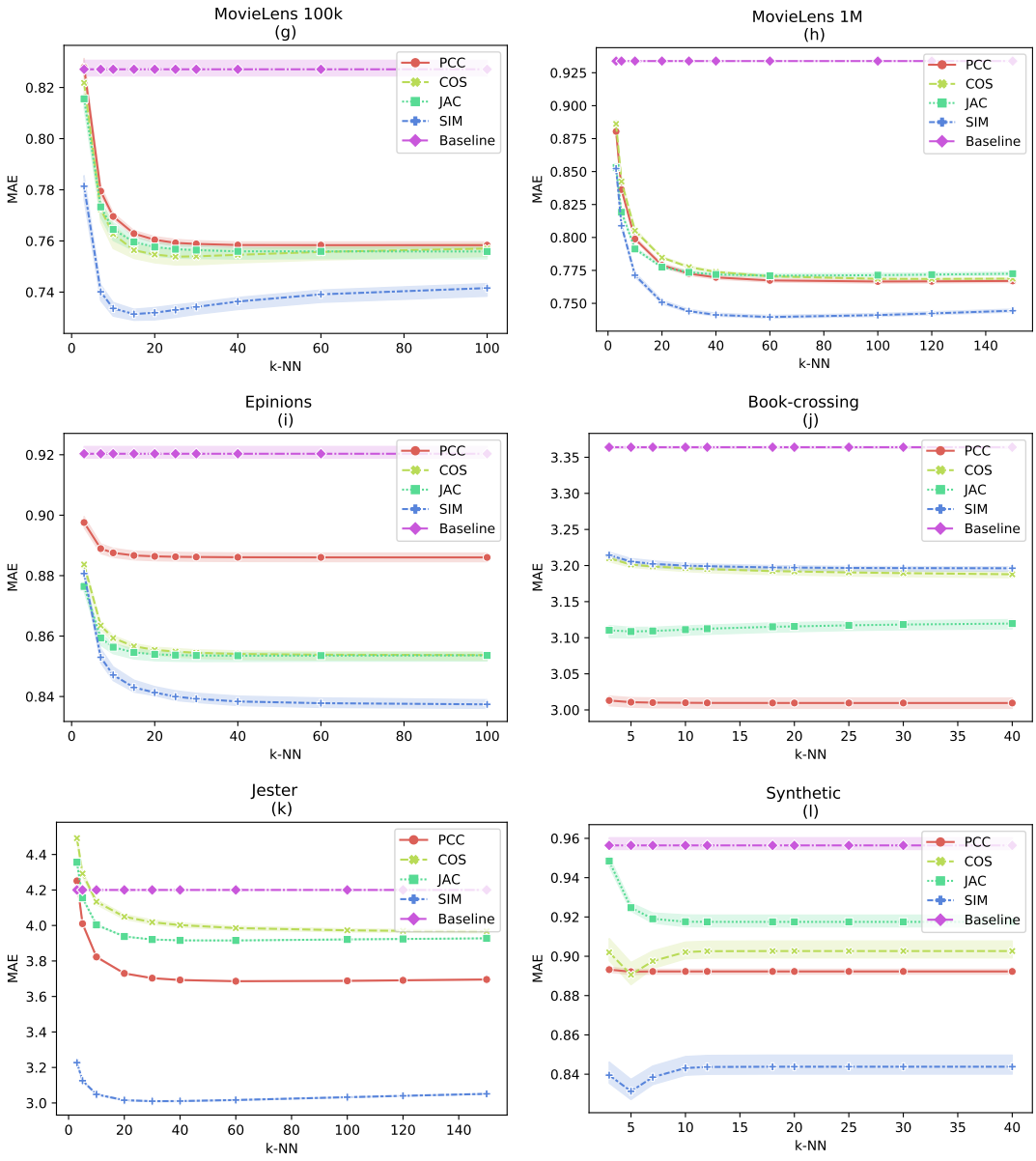

Figure 4.3: Recommendation performance of various neighborhood-based methods without rating normalization and using different similarity measures. The global mean is also shown as a baseline. MAE values are presented for different k-NN values. The proposed similarity (SIM in the plots) displays better general performance than that of other methods.

suggest that neighborhood-based models can outperform MF approaches. In many cases, both approaches complement each other in large-scale systems. Rather, the reason to examine these differences is that in some particular applications, a small loss in performance can be outweighed by other practical benefits, such as better interpretability or flexibility to add new users and items. Tables 4.4 and 4.5 summarize the RMSE and MAE values for the different CF methods applied to all datasets. The lowest values for each row are marked in bold. In addition, to emphasize the distinction between neighborhood-based and MF methods, Tables 4.4 and 4.5 also indicate in bold the best result among the neighborhood-based models even if they are outperformed by the MF ones (in this case, the MF value, also in bold, is marked with a star). The bottom rows of the tables show the average error across all datasets.

Table 4.6 presents the scores from statistical significance tests between the proposed method and the other neighborhood-based approaches from Tables 4.4 and 4.5.

From Tables 4.4 and 4.5, it can confirmed that while the MF methods tend to have the lowest error, the proposed recommender achieves comparable levels of performance. Furthermore, it appears to generalize better to different datasets, achieving the lowest total average error among all neighborhood-based approaches. It also outperforms other neighborhood-based methods on datasets with the highest sparsity. On datasets of relatively higher density, several CF methods, including the proposed method, seem to reach a minimum error level. One interesting result that can be observed is that FunkMF and SVD++ significantly outperformed the other techniques when applied to the synthetic dataset. This is explained by noting that the synthetic dataset does, in fact, consist of a few separately defined user types and therefore naturally lends itself to the low-dimensional representation of the MF approaches.

Table 4.4: Performance comparison in terms of RMSE for different CF methods, including two MF approaches, and all datasets. The performance of the proposed method (SHB) is comparable to that of SVD++ and FunkMF. The lowest errors among the neighborhood-based methods for each row are marked in bold. If an MF method outperforms all neighborhood-based ones, this is also marked in bold for reference and is differentiated by a star label.

Dataset	SVD++	FunkMF	SHB	PCC	COS	JAC	PCC+JAC	COS+JAC
MovieLens 100k	⋆ **0.87132**	0.88380	**0.88499**	0.90945	0.90969	0.90480	0.89881	0.90195
MovieLens 1M	⋆ **0.86610**	0.87769	0.90870	0.93835	0.95754	0.91347	**0.90779**	0.90919
Epinions	1.05481	⋆ **1.05153**	**1.06776**	1.15395	1.11452	1.11403	1.12221	1.11236
Book-crossing	3.51305	⋆ **3.34099**	3.41898	3.66195	3.63262	**3.41508**	**3.41508**	**3.41508**
Jester	4.44688	4.38814	4.13157	**4.10789**	4.21123	4.23024	4.19367	4.23952
Synthetic	⋆ **0.87565**	0.88042	**1.07592**	1.17417	1.13532	1.13384	1.13299	1.13321
Total average	1.93797	⋆ **1.90376**	**1.91465**	1.99096	1.99348	1.95191	1.94509	1.95188

Table 4.5: Performance comparison in terms of MAE for different CF methods, including two MF approaches, and all datasets. The performance of the proposed method (SHB) is comparable to that of SVD++ and FunkMF. The lowest errors among the neighborhood-based methods for each row are marked in bold. If an MF method outperforms all neighborhood-based ones, this is also marked in bold for reference and is differentiated by a star label.

Dataset	SVD++	FunkMF	SHB	PCC	COS	JAC	PCC+JAC	COS+JAC
MovieLens 100k	⋆ **0.66764**	0.67983	0.67609	0.69384	0.69761	0.69111	**0.68607**	0.68961
MovieLens 1M	⋆ **0.67555**	0.68915	0.71463	0.74034	0.76169	0.71610	**0.71282**	0.71521
Epinions	⋆ **0.80733**	0.81160	**0.81814**	0.87326	0.83742	0.85888	0.87149	0.85902
Book-crossing	2.61654	⋆ **2.56925**	**2.74326**	2.83552	2.82705	2.75208	2.75208	2.75208
Jester	3.27225	3.25326	3.10847	**3.05888**	3.15597	3.19095	3.16102	3.20591
Synthetic	⋆ **0.70849**	0.71490	**0.89065**	0.89773	0.90589	0.95271	0.95313	0.95229
Total average	1.45796	⋆ **1.45299**	**1.49187**	1.51659	1.53093	1.52697	1.52276	1.52902

The third part of the analysis considers again the performance of the proposed SHB recommender compared with that of other CF methods, focusing on the error metrics when different numbers of nearest neighbors are considered. The results are shown in Figs. 4.4 and 4.5. The proposed method achieves the lowest or close to the lowest errors in all datasets. The RMSE and MAE values for SHB stabilized when

Table 4.6: Significance tests for the proposed SHB recommender with respect to the neighborhood-based approaches in Tables 4.4 and 4.5.

	PCC		**COS**		**JAC**		**PCC+JAC**		**COS+JAC**	
Dataset (RMSE)	t-value	p-value	t-value	p-value	t-value	p-value	t-value	p-value	t-value	p-value
MovieLens 100k	-19.213	0.00270	-41.747	0.00057	-37.475	0.00071	-33.227	0.00090	-42.827	0.00054
MovieLens 1M	-61.440	0.00026	-58.387	0.00029	-29.315	0.00116	11.656	0.00728	-10.320	0.00926
Epinions	-107.867	0.00009	-79.119	0.00016	-335.519	0.00001	-357.852	0.00001	-465.074	0.00000
Book-crossing	-274.130	0.00001	-253.404	0.00002	3.764	0.06389	3.764	0.06389	3.764	0.06389
Jester	25.086	0.00159	-97.378	0.00011	-200.655	0.00002	-100.403	0.00010	-216.848	0.00002
Synthetic	-29.136	0.00118	-77.265	0.00017	-52.531	0.00036	-54.154	0.00034	-59.010	0.00029
Dataset (MAE)	t-value	p-value	t-value	p-value	t-value	p-value	t-value	p-value	t-value	p-value
MovieLens 100k	-10.869	0.00836	-33.272	0.00090	-22.972	0.00189	-32.972	0.00092	-34.793	0.00083
MovieLens 1M	-53.800	0.00035	-76.076	0.00017	-10.904	0.00831	155.968	0.00004	-12.037	0.00683
Epinions	-102.336	0.00010	-21.923	0.00207	-146.196	0.00005	-154.499	0.00004	-124.185	0.00006
Book-crossing	-84.434	0.00014	-390.777	0.00001	-14.747	0.00457	-14.747	0.00457	-14.747	0.00457
Jester	102.893	0.00009	-80.833	0.00015	-343.237	0.00001	-125.263	0.00006	-354.511	0.00001
Synthetic	-2.492	0.13030	-18.985	0.00276	-72.604	0.00019	-68.250	0.00021	-80.041	0.00016

approximately 20 neighbors were considered, implying that the proposed method is well suited to datasets that are highly sparse. As for other methods, PCC and COS both benefit from being combined with JAC; this result is consistent with the findings in [13]. When considering the MovieLens 1M and Jester (dataset 3) datasets, these hybrid methods perform similarly to the proposal. This can be explained by noting that both datasets have relatively low sparsity. Therefore, more high-quality neighbors with more ratings can be found, and more reliable PCC and COS scores can be obtained.

The results for the Book-crossing dataset exhibit the best performance for all methods that include a JAC component. Furthermore, the performance appears to be independent of the choice of k-NN for all of them except SHB. The proposal also converges to this value, reaching it for a k-NN of approximately 20. This behavior may be due to the high sparsity of the Book-crossing dataset (99.96%). The intersections between user vectors are very limited, thus restricting the precision of all similarity calculations. In this situation, the intersection of rated movies considered by JAC is the most reliable measure of similarity. This demonstrates that in general terms, JAC

focuses on unique aspects of the data and therefore tends to complement the other similarity measures.

Although other similarities attempt to identify like-minded users from their ratings, JAC only considers whether the users rated an item or not, irrespective of the score given. Although JAC discards information exploited by other similarity measures, it still retains a strong implicit signal of preference because users do not choose at random the items to rate. It may be argued that the likelihood of a user rating an item depends, in part, on their perceived investment and expected return when consuming the item. For example, watching a movie requires investing a few hours; thus, users may be more inclined to watch those movies that they believe they will like. The act of rating a movie that the users must have watched is, therefore, not entirely random because the movie had to be selected first, and this selection bears an implicit signal of preference.

Another observation is that for some datasets, SHB reaches an optimal performance and then slowly deteriorates as k-NN increases. This trend can be explained by the effects of weakly correlated neighbors being added in the rating prediction. In the extreme case where all other users are treated as neighbors, even as the closest user has a large weight, a high number of low-similarity neighbors would introduce an appreciable noise, which ultimately affects the recommender performance. This phenomenon, along with the previously observed stabilization of the error from approximately 20 k-NNs, indicates that SHB only requires a small number of neighbors to attain its best predictions.

Next, the effects of the proposed method shown in Subsection 4.3.2 are analyzed by comparing the accuracy of two variations of SHB. The first variation uses the

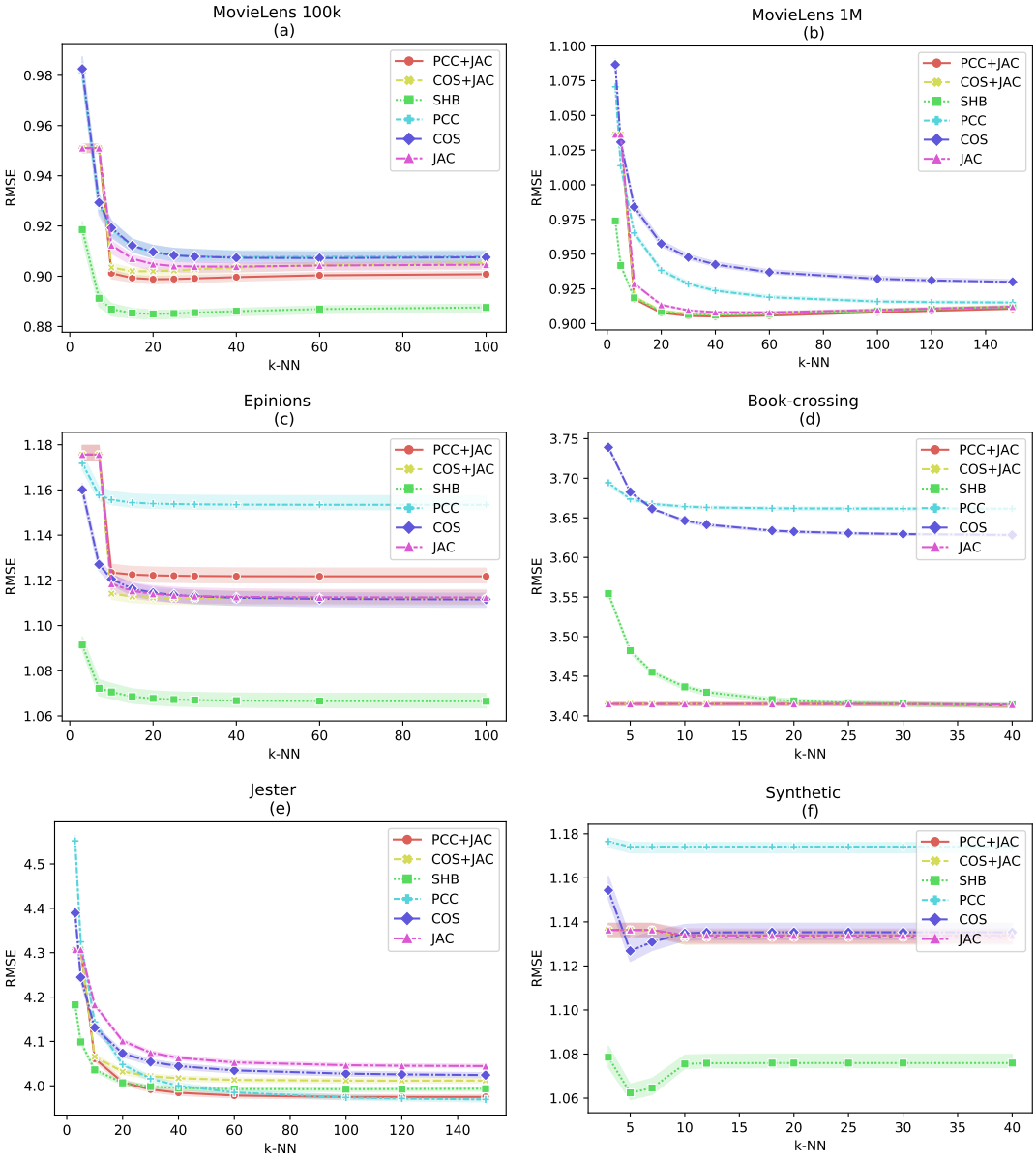

Figure 4.4: Recommendation performance of various CF approaches. RMSE values are listed for different k-NNs. The error levels attained by SHB are among the lowest for all datasets.

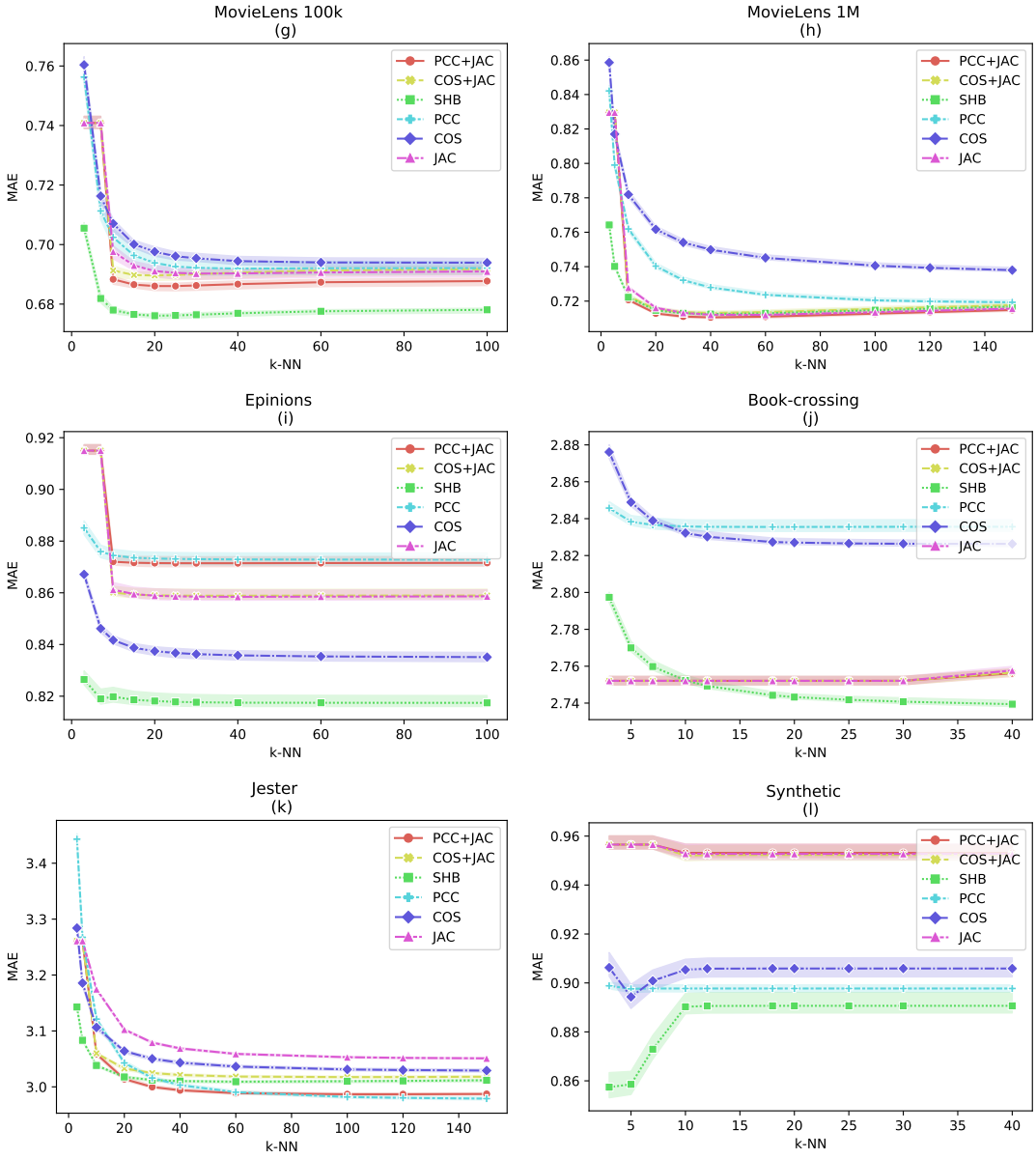

Figure 4.5: Recommendation performance of various CF approaches. MAE values are listed for different k-NNs. The error levels attained by SHB are among the lowest for all datasets.

Table 4.7: Performance of two variations of SHB. The first variation relies on the proposed closed-form formula to compute the global statistics, and the second variation (SGD) solves the optimization problem of Eq. 4.8 using SGD. Both approaches yield comparable performance.

	RMSE		MAE	
Dataset	**Proposal**	**SGD**	**Proposal**	**SGD**
MovieLens 100k	**0.884997**	0.893841	**0.676092**	0.683599
MovieLens 1M	**0.906104**	0.906288	**0.712491**	0.712610
Epinions	**1.066525**	1.076724	**0.817426**	0.830775
Book-crossing	**1.107421**	1.120157	**0.859504**	0.870760
Jester	4.035909	**3.038308**	4.035909	**3.038308**
Synthetic	**1.075514**	1.090935	**0.890268**	0.912781

simplified predictions of Eq. 4.6 when the switching criterion is met. For the second variation, an alternative implementation replaces these predictions with those from the method proposed by [9], as shown in Eq. 4.8. The results, summarized in Table 4.7, indicate that both approaches yield comparable performance when used in conjunction with the proposed SHB recommender.

The performance of SHB is now considered with the contribution of JAC being set to zero ($\alpha = 1$) and compare it with the other CF methods when they are applied to the synthetic dataset generated using different levels of sparsity. The results are shown in Fig. 4.6. The first observation is that the RMSE for PCC and COS exhibits an approximately concave behavior with a pronounced decrease in performance as sparsity increases up to a specific value. Counterintuitively, the performance appears to improve for higher sparsities. The reason for this is that if the dataset becomes too sparse, there will be several cases where no neighbors are found. In this situation, the second term of Eq. 2.2 is taken to be zero, and thus, the predicted rating becomes the user mean. This is also the reason for the convergence of all methods to the same value as sparsity approaches 1. Although the performance of SHB also decreases as the dataset becomes more sparse, it retains some degree of accuracy even at very

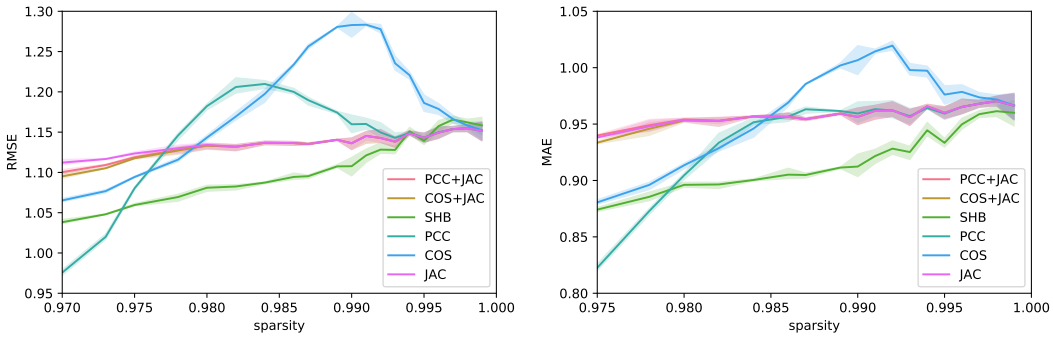

Figure 4.6: Performance of different CF methods at different sparsity levels, where SHB's hyperparameter α was set to zero (no JAC contribution). The accuracy of SHB remains low even at high sparsity levels.

high sparsity levels until it reaches the user mean when there are no neighboring users in the dataset. Other methods that include a JAC contribution also exhibit this trend; however, the proposed method outperforms them at lower sparsities. A possible interpretation is that the information being exploited by SHB is more meaningful than that used by the other CF methods.

Finally, the influence of JAC on the performance of the different CF approaches was considered. Figs. 4.7 and 4.8 shows the RMSE and MAE for different values of the hyperparameter α. The results show that including JAC leads to better predictions for all methods and datasets with the exception of COS+JAC applied to the Epinions dataset, in which case the MAE is the smallest without the contribution of JAC. It can be observed that the proposed SHB method exhibits convex behavior, implying the existence of an optimal value of α. This indicates that while one term in Eq. 4.10 overestimates the rating, the other must be underestimating it and, therefore, provides further evidence that each term focuses on different aspects of the data.

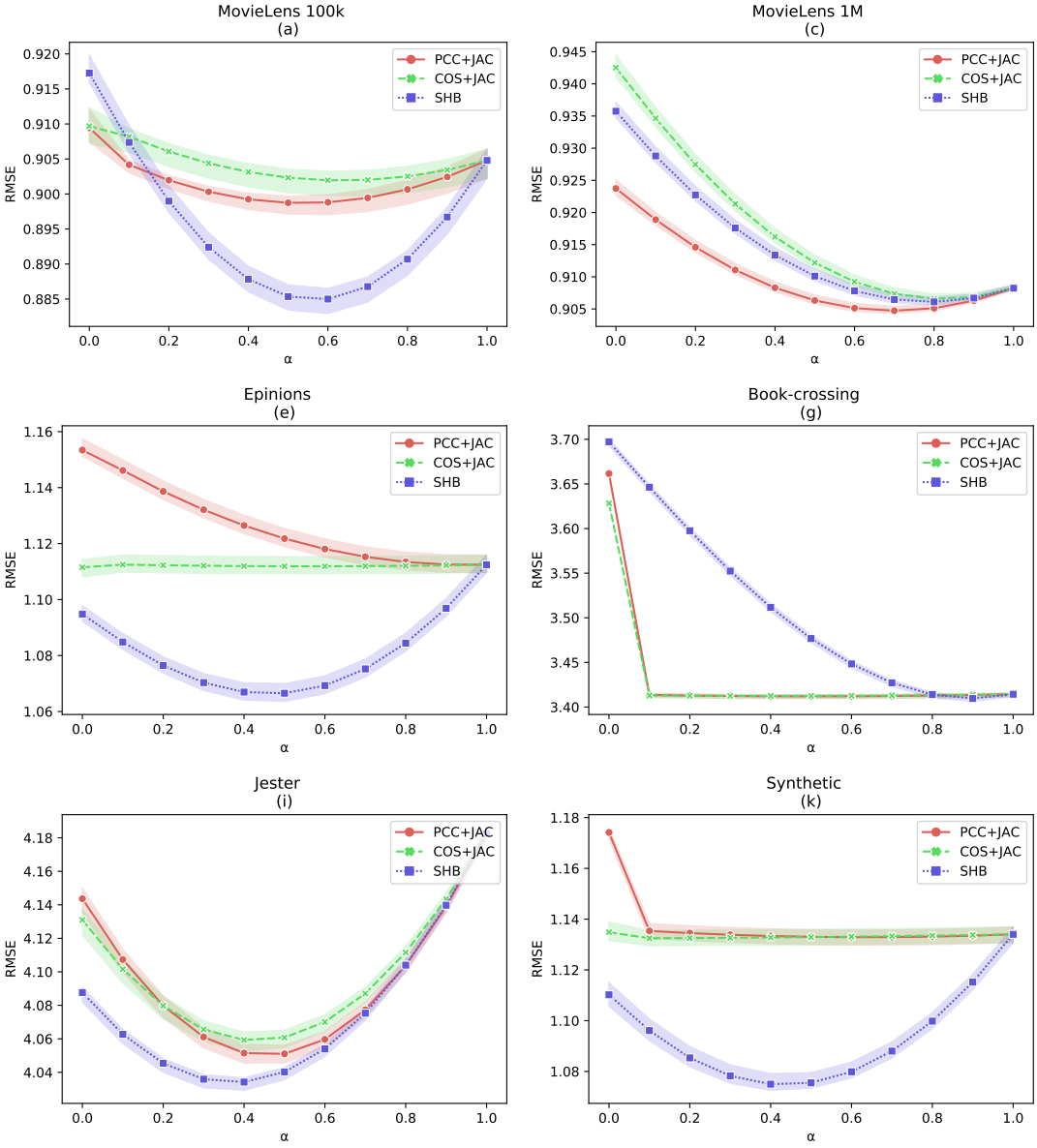

Figure 4.7: Recommendation performance of different CF approaches. RMSE are reported over different values for the hyperparameter α. The combination with JAC consistently leads to an improvement in performance.

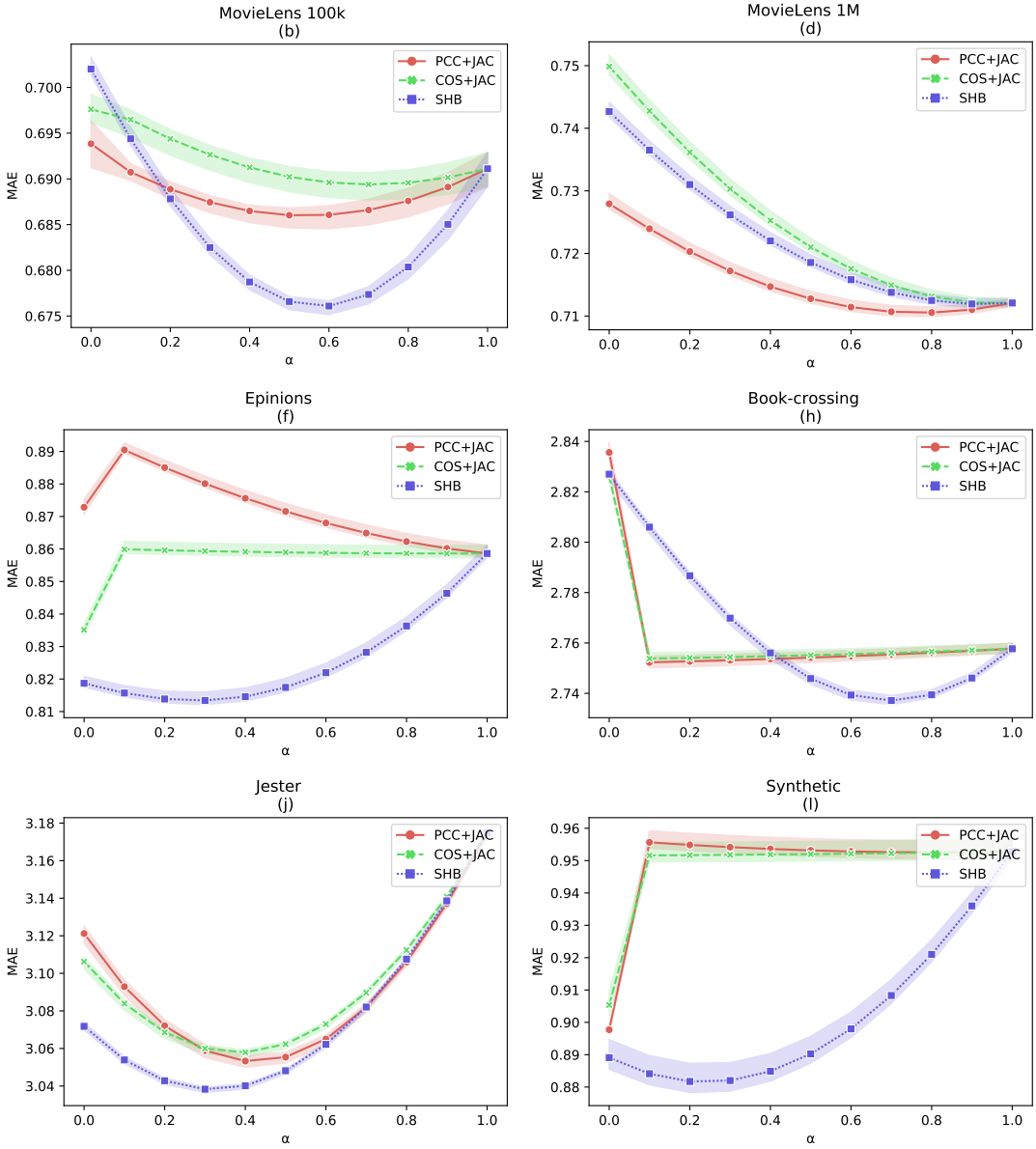

Figure 4.8: Recommendation performance of different CF approaches. MAE are reported over different values for the hyperparameter α. The combination with JAC consistently leads to an improvement in performance.

4.5 Conclusions

A new similarity measure designed to overcome the drawbacks of conventional similarities was proposed. The proposed method considers the semantic nuances of ratings, which quantify the degree of resemblance between user tastes as represented on an ordinal scale. This makes it suitable for applications in which measuring a user's degree of indifference toward an item is important. In addition, an RS based on an ensemble of a switching hybrid and a CF method with JAC was introduced. The switching hybrid relies on the predictions made either using a CF technique that uses the proposed similarity measure or from the global statistics of the dataset. The approach is designed to retain the simplicity and explainability aspects which make neighborhood-based approaches a mainstream technique used in real-world applications.

The predictive accuracy of SHB was evaluated using the RMSE and MAE, and its performance was compared with that of other CF approaches. Experiments were conducted using five real-world datasets and a synthetic dataset. The results indicate that the error when using the proposed method is either lower or matches that of the best predictions obtained from conventional similarity measures. In particular, SHB consistently outperformed the other methods when applied to datasets with high sparsity. It achieved its best performance using only a few neighbors, making it more scalable. However, a small amount of overfitting was observed if k-NN was set excessively high, thus negatively affecting the quality of the predictions. The analysis also confirms that combining individual similarity measures with JAC can consistently improve their performance. It was observed that JAC encodes a strong signal of implicit preferences that complements the aspects found by other similarities.

Chapter 5

Conclusions

This dissertation introduced two general practical matrix completion problems and presented two methods that estimate unobserved values in data matrices based on CF assumptions. The first method, discussed in Chapter 3, estimates a complete row vector that represents a product described in terms of its main attributes. The estimation is computed as the weighted average by similarity of known products based on previously collected survey data and user online behavior. The second method, discussed in Chapter 4, predicts the elements of a matrix that correspond to the ratings that a user might give to each of the movies that the user has not rated. The predictions are calculated with an ensemble of a switching hybrid and a CF method with JAC, and are based on the user's closeness to other users with similar viewing and rating patterns.

Furthermore we proposed two custom similarity measures that were designed based on each domain's specific assumptions, and compared their performance against standard similarity measures. The first similarity measure relies on the total number of items bought as an implicit measure of preference. The proposed similarity measure assigns a larger similarity score to products that were bought in equal amounts as they

are assumed to be consistent with the user tastes, and a smaller one as the difference in purchase amounts increases. The second proposed similarity measure also considers the semantic meaning of the matrix elements, in this case, the ratings. Ratings in this context, are meant to reflect the degree of indifference towards an item on an ordinal scale. Thus the proposed similarity is a measure that considers the difference between ratings and how extreme these differences are.

The results obtained indicate that both proposed similarity measures outperform standard metrics in terms of MAE and RSME. It was also possible to observe the benefit from defining custom similarity measures by considering the semantic meaning of the concept being compared, on the specific domain that they are applied.

The method discussed in Chapter 3, was limited in terms of the kind of target products used, which consisted mainly of daily use products, such as food and beverages. For this reason, it is unknown whether the proposal can remain effective for product catalogues of a different kind, such as cars or electronic equipment which are not purchased frequently and in high numbers. Further, the dataset does not fully sample all types of customers. It is limited to Japan, and biased towards the age and social groups that tend to make online purchases. Nevertheless, the results show the practical applicability of the proposal in the case study. Moreover, both proposals have the disadvantage of requiring high time and computational complexity to calculate their respective similarity matrices. This poor scalability problem however, is a known challenge that affects not only our proposed methods, but is inherent to neighborhood-based methods in general. A common approach to this problem is to pre-compute the similarity matrices once, and to store only the tuples of the matrices with a similarity larger than a predetermined threshold.

Some potential areas for future research include the extension to account for products with unique features in the proposal described in Chapter 3, or improvements to the switching criteria in the method described in Chapter 4, for example, by considering the confidence intervals. Moreover, the proposal in Chapter 4 may be applied and evaluated from an item-based perspective. Item-based approaches are built comparing the similarities of items instead of users. These methods recommend items based on other items previously rated by a user, as opposed to recommending items that other similar users rated. This approach is popular because items characteristics tend to be more constant than human preferences that change over time, so item-based approaches tend to be more stable.

Furthermore, because of the importance of designing effective similarity measures, another promising line of research is similarity learning. Where the objective is to learn a similarity function based on domain data. A model trained with domain data might be able to characterize the complex relationships that describe more accurately human behaviors such as purchasing or rating items.

References

[1] Luong Trung Nguyen, Junhan Kim, and Byonghyo Shim. “Low-Rank Matrix Completion: A Contemporary Survey”. In: *IEEE Access* 7 (2019), pp. 94215–94237.

[2] Lior Rokach et al. *Recommender Systems Handbook*. Springer US, 2015.

[3] Upendra Shardanand and Pattie Maes. “Social Information Filtering: Algorithms for Automating “Word of Mouth””. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’95. Denver, Colorado, USA: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 210–217.

[4] David Goldberg et al. “Using Collaborative Filtering to Weave an Information Tapestry”. In: *Communications of the ACM* 35.12 (Dec. 1992), pp. 61–70.

[5] Paul Resnick et al. “GroupLens: An Open Architecture for Collaborative Filtering of Netnews”. In: *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*. CSCW ’94. Chapel Hill, North Carolina, USA: Association for Computing Machinery, 1994, pp. 175–186.

[6] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. “Restricted Boltzmann Machines for Collaborative Filtering”. In: *Proceedings of the 24th International Conference on Machine Learning*. ICML ’07. Corvalis, Oregon, USA: Association for Computing Machinery, 2007, pp. 791–798.

[7] Robert Bell, Yehuda Koren, and Chris Volinsky. “Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems”. In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’07. San Jose, California, USA: Association for Computing Machinery, 2007, pp. 95–104.

[8] Simon Funk. *Netflix Update: Try This at Home*. 2006.

[9] Yehuda Koren. “Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model”. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’08. 2008, pp. 426–434.

[10] Charu Aggarwal. *Recommender Systems: The Textbook*. 1st ed. Springer Publishing Company, Incorporated, 2016.

[11] Michel Deza and Elena Deza. *Encyclopedia of Distances*. 3rd ed. Springer-Verlag Berlin Heidelberg, 2014.

[12] Guibing Guo, Jie Zhang, and Neil Yorke Smith. “A Novel Bayesian Similarity Measure for Recommender Systems”. In: *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*. IJCAI ’13. Beijing, China: AAAI Press, 2013, pp. 2619–2625.

[13] Laurent Candillier, Frank Meyer, and Françoise Fessant. “Designing Specific Weighted Similarity Measures to Improve Collaborative Filtering Systems”. In: *Advances in Data Mining. Medical Applications, E-Commerce, Marketing, and Theoretical Aspects*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 242–255.

[14] Deepa Anand and Kamal Bharadwaj. “Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities”. In: *Expert Systems with Applications* 38.5 (2011), pp. 5101–5109.

[15] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. “A Novel Evidence-Based Bayesian Similarity Measure for Recommender Systems”. In: *ACM Transactions on the Web* 10.2 (May 2016).

[16] Achraf Gazdar. “A new Similarity Measure for Collaborative Filtering based Recommender Systems”. In: *Knowledge-Based Systems* (Sept. 2019).

[17] Z. Tan and L. He. “An Efficient Similarity Measure for User-Based Collaborative Filtering Recommender Systems Inspired by the Physical Resonance Principle”. In: *IEEE Access* 5 (2017), pp. 27211–27228.

[18] Lamis Al Hassanieh et al. “Similarity measures for collaborative filtering recommender systems”. In: *2018 IEEE Middle East and North Africa Communications Conference* (Apr. 2018), pp. 1–5.

[19] Guineng Zheng et al. “OpenTag: Open Attribute Value Extraction from Product Profiles”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’18. London, United Kingdom: Association for Computing Machinery, 2018, pp. 1049–1058.

[20] Rayid Ghani et al. “Text Mining for Product Attribute Extraction”. In: *SIGKDD Explorations Newsletter* 8.1 (2006), pp. 41–48.

[21] Santosh Raju, Prasad Pingali, and Vasudeva Varma. “An Unsupervised Approach to Product Attribute Extraction”. In: *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*. ECIR ’09. Toulouse, France: Springer-Verlag, Apr. 2009, pp. 796–800.

[22] Ajinkya More. “Attribute Extraction from Product Titles in eCommerce”. In: *Enterprise Intelligence Workshop*. KDD ’16. Aug. 2016.

[23] Sowmya Vajjala et al. *Practical Natural Language Processing: A Pragmatic Approach to Processing and Analyzing Language Data*. O’Reilly Media, Incorporated, 2020.

[24] Zornitsa Kozareva et al. “Recognizing Salient Entities in Shopping Queries”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany, 2016, pp. 107–111.

[25] Guillaume Lample et al. “Neural Architectures for Named Entity Recognition”. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 260–270.

[26] Xuezhe Ma and Eduard Hovy. “End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1064–1074.

[27] Eleanor Rosch. “Principles of Categorization”. In: *Cognition and Categorization*. Hillsdale, NJ: Erlbaum, 1978, pp. 27–48.

[28] Christopher Brooks and Nancy Montanez. “Improved Annotation of the Blogosphere via Autotagging and Hierarchical Clustering”. In: *Proceedings of the 15th International Conference on World Wide Web*. WWW ’06. Edinburgh, Scotland: Association for Computing Machinery, May 2006, pp. 625–632.

[29] Scott Golder and Bernardo Huberman. “The Structure of Collaborative Tagging Systems”. In: *Journal of Information Science* 32 (Sept. 2005).

[30] Marieke Guy and Emma Tonkin. “Folksonomies: Tidying up Tags?” In: *D-Lib Magazine* 12.1 (Jan. 2006).

[31] Sarah Hayman. “Folksonomies and Tagging: New developments in social bookmarking”. In: *Proceedings of Ark Group Conference: Developing and Improving Classification Schemes*. Ark Group. Sydney, June 2007.

[32] Margaret Kipp and Grant Campbell. “Patterns and Inconsistencies in Collaborative Tagging Systems : An Examination of Tagging Practices”. In: *Proceedings of the American Society for Information Science and Technology* 43 (Oct. 2007).

[33] Katharina Probst et al. “Extracting and Using Attribute-Value Pairs from Product Descriptions on the Web”. In: *Web to Social Web: Discovering and Deploying User and Content Profiles*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 41–60.

[34] Lidong Bing, Tak Lam Wong, and Wai Lam. “Unsupervised Extraction of Popular Product Attributes from E-Commerce Web Sites by Considering Customer Reviews”. In: *ACM Transactions on Internet Technology* 16.2 (Apr. 2016).

[35] Sudheer Kovelamudi et al. “Domain Independent Model for Product Attribute Extraction from User Reviews using Wikipedia”. In: *Proceedings of the 5th International Joint Conference on Natural Language Processing*. Nov. 2011, pp. 1408–1412.

[36] Feilong Tang et al. “Aspect based fine-grained sentiment analysis for online reviews”. In: *Information Sciences* 488 (Mar. 2019).

[37] Lei Zhang and Bing Liu. “Sentiment Analysis and Opinion Mining”. In: *Encyclopedia of Machine Learning and Data Mining*. Boston, MA: Springer US, 2017, pp. 1152–1161.

[38] Jesse Vig, Shilad Sen, and John Riedl. “The Tag Genome: Encoding Community Knowledge to Support Novel Interaction”. In: *ACM Transactions on Interactive Intelligent Systems* 2.3 (Sept. 2012).

[39] Douglas Eck et al. “Automatic Generation of Social Tags for Music Recommendation”. In: *Proceedings of the 20th International Conference on Neural Information Processing Systems*. NIPS ’07. Vancouver, British Columbia, Canada: Curran Associates Inc., 2007, pp. 385–392.

[40] Adrian Ulges et al. “A System That Learns to Tag Videos by Watching Youtube”. In: *Computer Vision Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 415–424.

[41] Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. “Social Tag Prediction”. In: *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’08. Singapore, Singapore: Association for Computing Machinery, 2008, pp. 531–538.

[42] Claude Sammut and Geoffrey Webb. “Encyclopedia of Machine Learning”. In: Boston, MA: Springer US, 2010, pp. 600–601.

[43] Gary Klein, Judith Orasanu, and Roberta Calderwood. *Decision Making in Action: Models and Methods*. Cognition and Literacy Series. Ablex, 1993.

[44] Daniel Kahneman and Thomas Gilovich. *Heuristics and Biases: The Psychology of Intuitive Judgment*. Cambridge University Press, 2002.

[45] Alexander Chernev, Ulf Böckenholt, and Joseph Goodman. "Choice overload: A conceptual review and meta-analysis". In: *Journal of Consumer Psychology* 25.2 (2015), pp. 333–358.

[46] Michael Solomon. *Consumer Behavior: Buying, Having, and Being*. Pearson, 2017.

[47] Robin Burke. "Hybrid Web Recommender Systems". In: *The Adaptive Web: Methods and Strategies of Web Personalization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 377–408.

[48] Gordon Foxall. *Understanding Consumer Choice*. 1st ed. Palgrave Macmillan UK, 2005.

[49] Shun Cai and Yunjie Xu. "Effects of outcome, process and shopping enjoyment on online consumer behaviour". In: *Electronic Commerce Research and Applications* 5 (Dec. 2006), pp. 272–281.

[50] Ben Schafer, Joseph Konstan, and John Riedl. "Recommender Systems in E-Commerce". In: *Proceedings of the 1st ACM Conference on Electronic Commerce*. EC '99. Denver, Colorado, USA: Association for Computing Machinery, 1999, pp. 158–166.

[51] Jian Wei et al. "Collaborative filtering and deep learning based recommendation system for cold start items". In: *Expert Systems with Applications* 69 (2017), pp. 29–39.

[52] Fengkun Liu and Hong Joo Lee. "Use of social network information to enhance collaborative filtering performance". In: *Expert Systems with Applications* 37.7 (2010), pp. 4772–4778.

[53] Robert Bell and Yehuda Koren. "Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights". In: *Seventh IEEE International Conference on Data Mining*. Oct. 2007, pp. 43–52.

[54] Heng Luo et al. "A collaborative filtering framework based on both local user similarity and global user similarity". In: *Machine Learning* 72 (Sept. 2008), pp. 231–245.

[55] Andrew Schein et al. “Methods and Metrics for Cold-Start Recommendations”. In: *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’02. Tampere, Finland: Association for Computing Machinery, 2002, pp. 253–260.

[56] Jesús Bobadilla et al. “A collaborative filtering approach to mitigate the new user cold start problem”. In: *Knowledge-Based Systems* 26 (2012), pp. 225–238.

[57] Hyung Jun Ahn. “A New Similarity Measure for Collaborative Filtering to Alleviate the New User Cold-Starting Problem”. In: *Information Sciences* 178.1 (Jan. 2008), pp. 37–51.

[58] Haifeng Liu et al. “A new user similarity model to improve the accuracy of collaborative filtering”. In: *Knowledge-Based Systems* 56 (2014), pp. 156–166.

[59] Alan Said, Brijnesh Jain, and Sahin Albayrak. “Analyzing Weighting Schemes in Collaborative Filtering: Cold Start, Post Cold Start and Power Users”. In: *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. SAC ’12. Trento, Italy: Association for Computing Machinery, 2012, pp. 2035–2040.

[60] Ramesh Laveti et al. “A Hybrid Recommender System Using Weighted Ensemble Similarity Metrics and Digital Filters”. In: *IEEE 23rd International Conference on High Performance Computing Workshops*. 2016, pp. 32–38.

[61] John Breese, David Heckerman, and Carl Kadie. “Empirical Analysis of Predictive Algorithms for Collaborative Filtering”. In: *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*. UAI’ 98. Madison, Wisconsin: Morgan Kaufmann Publishers Inc., 1998, pp. 43–52.

[62] Jun Wang, Arjen Vries, and Marcel Reinders. “Unifying User-Based and Item-Based Collaborative Filtering Approaches by Similarity Fusion”. In: *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’06. Seattle, Washington, USA: Association for Computing Machinery, 2006, pp. 501–508.

[63] Jiyong Zhang and Pearl Pu. “A Recursive Prediction Algorithm for Collaborative Filtering Recommender Systems”. In: *Proceedings of the 2007 ACM Conference on Recommender Systems*. RecSys ’07. Minneapolis, MN, USA: Association for Computing Machinery, 2007, pp. 57–64.

[64] Aline Bessa et al. *Alleviating the Sparsity Problem in Recommender Systems by Exploring Underlying User Communities*. Jan. 2012.

[65] Bilal Hawashin et al. “An efficient hybrid similarity measure based on user interests for recommender systems”. In: *Expert Systems* (Aug. 2019).

[66] Ron Garland. “The mid-point on a rating scale: Is it desirable”. In: *Marketing Bulletin* (1991), pp. 66–70.

[67] Kai Yu et al. “Instance Selection Techniques for Memory-Based Collaborative Filtering”. In: *Proceedings of the Second SIAM International Conference on Data Mining*. Apr. 2002.

[68] Abhinandan Das et al. “Google News Personalization: Scalable Online Collaborative Filtering”. In: *Proceedings of the 16th International Conference on World Wide Web*. WWW ’07. Banff, Alberta, Canada: Association for Computing Machinery, 2007, pp. 271–280.

[69] Badrul Sarwar et al. “Recommender Systems for Large-scale E-Commerce scalable neighborhood formation using clustering”. In: *5th International Conference on Computer and Information Technology* 50 (Jan. 2002).

[70] Gui-Rong Xue et al. “Scalable Collaborative Filtering Using Cluster-Based Smoothing”. In: *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’05. Salvador, Brazil: Association for Computing Machinery, 2005, pp. 114–21.

[71] Gábor Takács et al. “Scalable Collaborative Filtering Approaches for Large Recommender Systems”. In: *Journal of Machine Learning Research* 10 (June 2009), pp. 623–656.

[72] Maxwell Harper and Joseph Konstan. “The MovieLens Datasets: History and Context”. In: *ACM Transactions on Intelligent Systems* 5.4 (Dec. 2015).

[73] Paolo Massa et al. “Trustlet, Open Research on Trust Metrics.” In: *Scalable Computing: Practice and Experience* 9 (Jan. 2008).

[74] Cai-Nicolas Ziegler et al. “Improving Recommendation Lists through Topic Diversification”. In: *Proceedings of the 14th International Conference on World Wide Web*. WWW ’05. New York, NY, USA: Association for Computing Machinery, 2005, pp. 22–32.

[75] Kenneth Goldberg et al. “Eigentaste: A Constant Time Collaborative Filtering Algorithm”. In: *Information Retrieval* 4 (July 2001), pp. 133–151.

Appendix A

The Author's Publications

Journal papers

- Patricia Ortal and Masato Edahiro, "Switching Hybrid Method Based on User Similarity and Global Statistics for Collaborative Filtering". In: *IEEE Access*, vol. 8, pp. 213401-213415, 2020.
- Patricia Ortal and Masato Edahiro, "Similarity Measure for Product Attribute Estimation". In: *IEEE Access*, vol. 8, pp. 179073-179082, 2020.

International Conferences

- Patricia Ortal, Shinpei Kato and Masato Edahiro, "Real-Time Visualization of Moving Objects". In: *2015 IEEE 3rd International Conference on Cyber-Physical Systems, Networks, and Applications*, Hong Kong, 2015, pp. 60-65.