The influence of social filtering in recommender systems

Nick Dekkers 3693406

1 Introduction

Recommender systems have become more and more intertwined in our everyday usage of the web. Think about the recommended videos on Youtube or the recommended items on Amazon. These systems collect information on the preferences of its users for a certain set of items, like movies, songs, books, travel destinations, and many more. This information can be gathered in an explicit way (by collecting ratings of users) or in an implicit way [38][21][53] (by monitoring users' behavior, such as songs heard or books read). The aim of a recommender system is to reduce information overload by retrieving the most relevant information and services from a huge amount of data. In order to do so recommender systems may use demographic features of its users (like age, nationality, gender). With the popularity of services like Facebook, social information, like followers, followed, twits and posts can be used to gather information. More recently a growing trend towards the use of information from the Internet of things has started. Information like GPS location, RFID or health signals.

Recommender systems try to balance factors like accuracy, novelty, dispersity and stability in the recommendations. They use several different sources of information for providing users with predictions and recommendations of items. The most common methods are collaborative filtering methods. These methods are often used together with other filtering techniques, like content-based or social ones. Collaborative filtering is very intuitive because it is based on the human decision making process. It does not only consider its own experiences, but it also takes into account the experiences of a large group of acquaintances.

Since the implementation of recommender systems in the Internet has increased, the systems are used in a lot of areas [56]. Most research is done on movie recommendation studies [18][81]. However, a lot of literature is also focused on different topics, like music [38][52][78], television [84][6], applications in markets [22], web search [50], books [53][23], documents [72][61][60][62], e-learning [86][13], e-commerce [31][19], and others.

Throughout the years recommender systems have been evolving in several ways. Hybrid techniques have been applied on recommender systems by merging current techniques to get the advantages of both [15]. Apart from hybrid techniques, the design of recommender systems has been evolving alongside the Internet in three phases. In the first phase the systems were mainly based on the information from the web. The second phase incorporated social information into the design. Lastly, in the more recent third phase information from the internet of things was added.

The remainder of this article is structured as follows: In Section 2 the foundations of recommender systems will be explained, looking at the considerations for recommender systems. So which methods, algorithms and models can be used based on the information from the web (the first phase). Section 3 shows the use of social information in recommender systems for making recommendations based on trust, reputation and credibility (the second phase). Section 4 focuses on the trends in recommender systems. In this section techniques based on location-aware recommendation systems will be described. Finally, Section 5 will summarize the history of recommender systems and will show areas for future research.

2 Foundation of recommender systems

In this section a general description will be given of the foundations of recommender systems, like taxonomies, algorithms, methods, filtering approaches, databases, etc. After that the cold-start problem will be explained. This problem describes the difficulty in recommending on a small set of data. Next, the kNN algorithm will be described. This is the algorithm that is most used in recommender systems based on collaborative filtering. Finally, similarity measures will be given for comparing users and items.

2.1 Fundamentals

When designing a recommender system several considerations have to be made on the following aspects:

- The type of data available in the database (ratings, features and content for the items that can be ranked, social information, location-aware information).
- The filtering algorithm used (demographic, content-based, social-based, context-aware, hybrid).
- The model chosen (based on direct use of data: memory-based, or a model generated using such data: model-based).
- The employed techniques: probabilistic approaches, Bayesian networks, nearest neighbors algorithm; bio-inspired algorithms like neural networks, genetic algorithms; fuzzy model, singular value decomposition to reduce sparsity levels, etc.
- The sparsity level of the database and the designed scalability.
- The performance of the system based on time and memory consumance.
- The objective sought (e.g. predictions and top N recommendations).
- The desired quality of the results, like novelty coverage and precision.

2.1.1 Databases

In order to do research on recommender systems there has to be a public database in order to investigate techniques, methods and algorithms. By using these databases research can replicate experiments and validate techniques or even improve on current techniques. In Table 1 examples of public databases are shown. From these databases Last.Fm and Delicious are the only databases that use implicit ratings and social information.

2.1.2 Filtering algorithms

The internal functions of a recommender system are determined by the filtering algorithm used. The filtering algorithms can be divided in [2][17]: (1) content-based filtering, (2) demographic filtering, (3) collaborative filtering[69] and (4) hybrid filtering[15].

	Without social information						With social information		
	MovieLens 1M	Movielens 10M	Netflix	Jester	EachMovie	Book-crossing	ML	Last.Fm	Delicious
Ratings	1 million	10 million	100 million	4.1 million	2.8 million	1.1 million	855598	92834	104833
Users	6040	71567	480189	73421	72916	278858	2113	1892	1867
Items	3592	10681	17770	100	1628	271379	10153	17632	69226
Range	{1,...,5}	{1,...,5}	{1,...,5}	-10, 10	[0,1]	{1,...,10}	{1,...,5}	Implicit	Implicit
Tags	N/A	N/A	N/A	N/A	N/A	N/A	13222	11946	53388
Tags assignment	N/A	N/A	N/A	N/A	N/A	N/A	47957	186479	437593
Friends relations	N/A	N/A	N/A	N/A	N/A	N/A	N/A	25434	15328
Items	Movies	Movies	Movies	Jokes	Movies	Books	Movies	Music	URL's

Table 1: Most often used memory-based recommender systems public databases.

Demographic filtering Demographic filtering is the most basic filtering method. It is used to identify the types of users that like certain objects. It uses information like age, gender, education, etc. Items are recommended to a user based on the ratings given by other users that have a lot of common demographic features. The principle is that users with common personal attributes also have common preferences. Even though this filtering approach does not yield comparable results to the other filtering approaches, it still performs better than randomly guessing. It is better to use this filtering method in a hybrid approach with one of the other filtering methods to increase the accuracy.

Content-based filtering Recommendation system using content-based filtering approaches recommend items to the user that are similar to the ones the user preferred in the past. The rating of an item is based on the ratings assigned by the user to items that are similar to the new item. Content-based filtering has its roots in information retrieval and information filtering research. Because of the early advancements in these fields and the importance of text-based applications, many content-based systems focus on recommending items containing textual information, such as documents or websites. The content of the items considered is therefore represented by keywords. For example, a document is represented by the 100 most important words, where the importance of a word is determined by some weighting measure. The most well known and often used measure in information retrieval is TF-IDF, short for term frequency-inverse document frequency. The measure TF-IDF [63] is the product of two statistics, term frequency and inverse document frequency. There are various ways to define these statistics. In case of the term frequency $tf(t, d)$, taking the raw frequency is easiest.

Count the number of times term t occurs in document d and that will be the value. By denoting this value as $f_{t,d}$ this scheme gives the value $tf(t,d) = f_{t,d}$. Other ways do determine the value of term frequency are [45]: (a) boolean frequencies: $tf(t,d) = 1$ if t occurs in the document and zero otherwise, (b) logarithmically scaled frequency: $tf(t,d) = 1 + log f_{t,d}$ or zero if $f_{t,d}$ is zero, and (c) augmented frequency: $tf(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{max\{f_{t',d}:t' \in d\}}$, which is the raw frequency divided by the maximum raw frequency of any term in the document. The inverse document frequency $idf(t,D)$ is a measure of how much information a term provides in all the documents. The formula for this value is: $idf(t,D) = log\frac{N}{|\{d \in D:t \in d\}|}$. This formula divides the total number of documents N by the number of documents containing the term $|\{d \in D : t \in d\}|$, and then takes the logarithm of that quotient. The tf-idf value is than calculated by taking the product of these statistics, so $tdidf(t,d,D) = tf(t,d) \cdot idf(t,D)$.

Besides the information retrieval techniques, other techniques for content-based recommendation have been used, like the naive Bayesian classifier [58] which basically computes a prediction of a rating based on the chance the new item has of falling in a certain rating class. This chance is based on the features of the new item which are compared with the features of items that are already rated. Other techniques are machine learning techniques, such as clustering [68] or decision trees [27]. Clustering techniques work by grouping those users together who have similar preferences. Once a cluster is made, predictions for items, for a single user in this cluster, are given by taking an average of the ratings of the other users in this cluster for that particular item. From this, recommendations can be presented to the user based on the estimated ratings for the items. Decision trees can also be used for recommending items. To use a decision tree in a content-based recommendation process, first the decision tree has to be build for each user. Building this decision tree is done by using the content of the items already rated by the user. The features of the items are used to build a model of the user's preferences and this information gain is used as the splitting criteria. For a new item a value is given by traversing the constructed tree using the content information of the item and assigning the value at the end of the traversed path. All techniques mentioned above are based on a model, learned from the underlying data, by using statistical and machine learning techniques.

Using content-based filtering also has its limitations. There are three main limitations when using content-based filtering, namely:

- Limited content analysis: content-based systems are limited by the features that are explicitly associated with the items that are recommended. To get a sufficient set of features, it should be possible for the content to be parsed automatically, or the features should be assigned manually. Even though automatic parsing is easy for text features, this is not the case for features of images or videos. On top of that, manual assignment is not practical due to the limitations of resources. Another problem arising in limited content analysis is that two different items that are represented by the same features are indistinguishable to the system.

- Overspecialization: If a system only recommends items which score highly against a user's profile, the user is limited to only being recommended items that are similar to the items already rated by the user. So the range of items is limited. In order to cope with this limitation some sort of randomness has to be added. A second problem is the diversity. Users should not only be recommended items that are too similar too the items the user has already seen, because then the user is never recommended something new.

- The new user problem: this is the most difficult problem to solve in recommender system and more details on this problem will be explained in section 2.2. The limitation is that the user has to have rated a sufficient number of items. When the user has only rated a small number of items, the system will not really understand the user's preferences, which will lead to less reliable ratings. Which will ultimately result in new users not getting very accurate recommendations.

Collaborative filtering Collaborative filtering is the process of filtering or evaluating items using the opinions of other people. This technique gets its roots from something humans have been doing for years, namely sharing opinions with each other. If a lot of people in your surrounding environment like a certain product, you might be tempted to get it as well. On the other hand, when a lot of people dislike a product, you might not want to get it. People are even able to distinguish opinions based on their experience with other people, learning whose opinion to listen to and whose not to listen to.

With the help of computers and the web the amount of opinions on items has increased drastically. Not only can a lot more opinions be used, but this information can also be processed very fast in real time. So it is not only possible to determine what a large community thinks, but a personalized view can also be created for items by using the opinions that are most appropriate for a user.

In order to use collaborative filtering, ratings are important. A user has to have given a value to a certain item. These values and their meaning can be represented in a matrix. A ratings matrix is a table where each row represents a user and each column represents an item. A matrix cell can either contain a value, which is the rating given by the user to that specific item, or a blank, meaning the user has not rated the item.

In a collaborative filtering system a rating can be presented in several forms:

- Scalar ratings: either numerical ratings, such as a star rating from 1-5, or an ordinal rating, where the user has to choose between strongly disagree, disagree, neutral, agree or strongly agree.
- Binary ratings: ratings with only two choices, like bad/good.
- Unary ratings: a rating indicating whether a user bought an item or rate it positively.

These ratings can be gathered explicitly by asking users to rate items or implicitly by looking at the users' actions.

Collaborative filtering started being researched when the text repositories shifted from having purely formal content to a mixture of formal and informal content. Content-based techniques became more inadequate to help users find the items they were most interested in. There were two solutions to this problem, either wait for the artificial intelligence to improve, or start using human judgment.

The usage of collaborative filtering system has changed with the adaptation of the web. At first collaborative filtering systems were designed for providing users with information explicitly. Users visited websites for the main purpose of getting recommendations. Nowadays, these systems are used extensively behind the scenes. Websites adapt the content by choosing which news to provide to the users. There is only a limited amount of space in which the providers can show information and collaborative filtering systems can help determine what information a user would like to see.

Collaborative filtering systems can be used for a variety of tasks. Some of the user tasks for which collaborative filtering can be used are:

- Finding new items a user might like. Nowadays there is a huge information overload and it is hard to find interesting items. Collaborative filtering helps tackle this problem by presenting the user with just a few interesting items to choose from.

- Advising certain items. When a user has found a particular item of interest, the user might want to know whether the item is good or bad. Collaborative filtering can therefore present the opinions of other users in the community to the user.

- Finding users a person might like. A user with a similar taste might provide good recommendations to the current user. Focusing on these users is just as important as focusing on the items itself, because this can create a network from which predictions and recommendations can be retrieved for the current user.

- Helping a group of people find something they all like. Collaborative filtering can look at all the group members and recommend the items which might be liked by the entire group the most.

Apart from user tasks a collaborative filtering system can also perform tasks from a functional point of view. The most important ones begin recommending items and predicting the value of a given item. When a collaborative filtering system recommends items it can show a number of items to a user based on the usefulness of the items to the user. By predicting the ratings a user might give to unrated items and ranking them from highest to lowest, the system can provide the list of most useful items. Predicting values is a more complicated task than just recommending items, because a value might have to be given to a fairly new item, which has not received many ratings yet. Giving the correct rating is much more complicated than just recommending the item as an interesting item to look into.

Collaborative filtering has been introduced to improve the recommendation process at the point where content-based filtering could not perform well enough. In contrast to content-based filtering, which assumes that items with similar features are rated similarly, collaborative filtering assumes that people with a similar taste also rate similarly. These two filtering techniques have been seen as two complimentary techniques. Where content-based filtering can determine the relevance of items without the use of ratings, collaborative filtering needs the ratings in order to present the user with a prediction or recommendation. On the other hand collaborative filtering does not need the content of items to make predictions. If the content of an item cannot be extracted, the features of the item cannot be used in a content-based filtering system. Collaborative filtering does allow the evaluation of those features, because users provide this evaluation through ratings. One of the limitations of content-based filtering was over-specialization. An item that does not contain the exact same features will not be recommended, even though there could be a high degree of similarity. Collaborative filtering can deal with this limitation and can therefore give more unexpected recommendations.

The core algorithms of a collaborative filtering system can be divided in non-probabilistic and probabilistic algorithms. When looking at non-probabilistic algorithms, the most used algorithms are nearest neighbor algorithms. The nearest neighbor algorithms can in turn be divided in user-based and item-based nearest neighbor. More on the nearest neighbor algorithm can be found in section 2.3. Other non-probabilistic techniques would be dimensionality reduction techniques like matrix factorization, which will be discussed in section 2.1.4. The probabilistic algorithms mainly just calculate the probability that a user would assign a certain rating to an item. The most used algorithms for this are based on Bayesian networks.

Because a collaborative filtering system works with ratings of users and therefore has information about the users apart from the standard information, these systems have to deal with privacy issues. The information a user provides to the system has to be stored in a secure location. Users must trust that the providers of the systems will not use their ratings or preferences for other purposes except providing ratings and recommendations. So these systems have to have a good security system to protect all the information. This field will always provide challenges because even well secured systems might become exploited to reveal personal information.

Apart from the privacy issues a collaborative system also has to deal with trust issues. Collaborative filtering systems use ratings as a base for their recommendations, but it is not guarantied that all users rate items honestly. When a producer wants to make its own items more appealing it might provide very positive ratings for them and rate similar items from other producers negatively. A collaborative filtering system has to be able to distinguish between normal rating behavior and these so called shilling attacks. This is yet another challenge collaborative filtering has to deal with.

Hybrid filtering In a hybrid recommender system two or more recommendation techniques are combined. In most systems collaborative filtering is combined with another technique. By combining techniques the system is able to minimize the limitations of the techniques when used separately. There are several forms of hybrid recommender systems and in this section some of these hybrid techniques are discussed.

The first way of combining recommendation techniques is in a weighted fashion. In such a weighted hybrid recommender system the score of an item is computed by taking the score of each individual technique into account. One way to do this is to give each technique an equal weight at the start and adjust this as more information about the prediction quality becomes available. This technique is easy to implement and is pretty straightforward. It is also easy to adjust the system after the evaluation step. By using this technique the relative value of the individual techniques is the same for all items considered. In practice this is not the case, as collaborative filtering systems perform worse for items that have only a few ratings and content-based filtering systems perform less when the features of items become more difficult to extract.

Another way to combine techniques is to use switching. In such a hybrid system a criterion is used to switch between the different techniques. For example, the system starts with a content-based recommendation method until it cannot make good recommendations anymore. At that point the system switches to a collaborative filtering method. The collaborative filtering method might provide recommendations which are not similar based on content. This switching technique cannot overcome all limitations of the individual filtering methods. It cannot overcome the cold-start new-user problem, which will be discussed in section 2.2. Apart from the new-user problem, this hybrid system also adds another parameter to the process, namely the switching criterion. The biggest benefit of this hybrid method is that it can deal with some of the weaknesses of the individual techniques by using the strength of another technique. If a content-based method cannot make good recommendations because the content of an item cannot be appropriately evaluated, a collaborative method can use the ratings provide by users for that particular item to still give a good recommendation in the end.

When a lot of predictions have to made a mixed hybrid might be useful.

In a mixed hybrid system the recommendations from all individual techniques are shown together. A content-based method shows the top recommendations based on the content of the items, whereas a collaborative method running simultaneously shows the top recommendations based on other users' ratings. By running different filtering methods, this hybrid method is able to deal with the cold-start new-item problem. A new item cannot be easily recommended by a collaborative system using ratings, because the item has not received a sufficient amount of ratings yet. If the features of a new item can be extracted, a content-based system might recommend this new item and since a mixed hybrid system presents the top recommendations of all individual filtering methods the item will still be recommended. This type of hybrid recommender system can also deal with over-specialization, one of the limitations of content-based filtering, because it runs a collaborative filtering method which does not need the content of an item to give good recommendations. So mixed hybrid systems are able to deal with the limitations that the filtering methods cannot deal with when used individually. Even though it can deal with several limitations the filtering methods have when used separately, the new-user problem can still not be overcome using this technique.

Feature combination is yet another way of combining different filtering methods into a hybrid recommendation system. Looking at a combination of content-based and collaborative filtering, feature combination used the information retrieved for collaborative filtering as input features fro a content-based filtering method. In this way, it considers the collaborative data but does not necessarily rely on it. Feature combination based hybrid recommendater systems only use those features that can help in the recommendation process of the second method.

The previous hybrid recommendation method have all used the filtering methods in a simultaneous manner. Cascading is a form of hybrid recommender system that uses the filtering methods in a sequential manner. This technique first creates a candidate set of relevant items by using one of the filtering methods and then uses the second filtering method to refine this candidate set even further. Cascading avoids using the second filtering method on items that do not fit the user's interest based on the techniques employed in the first filtering method. This technique is more efficient than the hybrid methods already discussed because in the second step only those items are used which still need more discrimination. This is only a subset of all items. Apart from using less data through the entire process, cascading is also not influenced by a technique performing bad as this technique only refines the results of the technique that does present good recommendation.

Feature augmentation is another way to create a hybrid recommender system using a sequential application of the individual filtering methods. In such a hybrid system the first filtering method is used to produce ratings or classification of the items. This information is then used as additional information for the second filtering method. This differs from the feature combination discussed earlier. In feature combination raw data for different sources is combined, whereas in feature augmentation the results of the first filtering methods are used as input.

It is not only possible to use the output of a filtering method as new input material for the next filtering method, but it is also possible to use the model that was created in the first filtering method as the input. This is what meta-level hybrid recommender systems do. In the first filtering method a model is created which represents a user's interest and this model is used as input for the second filtering method. A collaborative filtering method in particular can benefit a lot from using a model, because it can operate more easily on a model than just raw rating data.

So there are several ways to create a hybrid recommender system. The systems discussed here show a clear distinction between hybrids where the order of the individual filtering methods does not make a difference, such as a weighting, switching, mixed or feature combination hybrid. Performance-wise a content-based/collaborative filtering hybrid system does not differ from a collaborative/content-based filtering hybrid system. In a feature augmentation, cascade or meta-level hybrid system this is not the case. Here the output from one filtering method is used as the input for the next filtering method. If the order of the methods change, so does the entire process. Since each method creates different output, the input for the second method is also not the same, influencing the entire process. So not only the filtering methods considered can determine what kind of hybrid recommender system is created, also the order of the filtering methods used can result in different hybrids. Here only content-based filtering and collaborative filtering have been discussed as combinations, but demographic filtering could also be incorporated in a hybrid recommender system.

2.1.3 Recommendation methods

The recommendation methods can be divided into two categories, namely memory-based methods and model-based methods.

The memory-based methods [2][17] are methods that only use the ratings of items a user has presented and also uses ratings generated before the referral process, meaning that it always updates its results and it always uses the entire database. These kind of methods typically use similarity measures to obtain the distance between two users or items. The advantages of using these kind of methods are that the quality of the predictions is pretty good and it is a relatively simple algorithm to implement in many situations. Updating the database is also very easy, since it uses the entire database when making predictions. Using the entire database also has a huge disadvantage. Because the database has to be in memory and is used every time a prediction is made, the performance is very slow. On top of that if a user has no items in common with all the people who have rated the target item, it can not make a prediction for this item due to a lack of information.

Model-based methods [2] use the information in the recommender system to build a model which is based on the dataset of ratings. Meaning that some information is extracted from the entire dataset and is used as a model to make recommendations without having to use the entire dataset every time a prediction is made. The Bayesian classifiers, clustering and decision tree algorithms are examples of model-based methods. The advantages of using model-based methods is that the models resulting from the algorithms used are much smaller than the actual dataset itself. Which affects the scalability of the system. The prediction speed is also faster, because only a small sample of the dataset is considered instead of the entire dataset. This also has its disadvantages. These systems are inflexible when it comes to adding new data to the system. Every time new data is added, the model has to be updated and this is a very time consuming task. Also, because it does not use all the information available, it is possible that the system using model-based methods might not give as accurate predictions as a memory-based systems.

2.1.4 Sparsity and scalability

High levels of sparsity is one of the problems recommender databases face. In order to reduce these sparsity problems in the databases, dimensionality reduction techniques can be applied [66]. The reduction methods are based on a process called Matrix Factorization [35][43][44]. The idea behind matrix factorization is to find two (or more) matrices such that when those matrices are multiplied the result will be the original matrix. Matrix factorization can be used to discover latent features underlying the interactions between different kind of entities, like predicting ratings in collaborative filtering. In a recommendation system the group of users and items can be represented as a matrix containing the ratings a user has given to an item and some blanks (the missing ratings to be predicted). The idea behind using matrix factorization to solve the problem of filling the blanks in a database is that there should be some latent features that determine how a user rates an item. By discovering these features the system should be able to predict a rating with respect to a certain user and a certain item, because the features associated with the user should match the features associated with the item. Matrix factorization works well for processing large databases and it provides scalable approaches [76]. Other techniques are Latent Semantic Index, Singular Value Decomposition and combinations of the two [77][87][16]. Although singular value decomposition provides good prediction results, the process is computationally very expensive. It can only be used offline when the preference information does not change.

2.1.5 Prediction quality and evaluation

One of the most important aspects of a recommender system is the quality of the predictions. In order to improve the prediction quality of a system, clustering techniques can be used. This can also reduce the cold-start problem when it is applied to hybrid filtering. In these hybrid recommendation systems it is typical to use clusters of items [74]. Another method that is used to improve the quality of the predictions is bi-clustering, which is the clustering for both users and items [26]. Clustering is also used in recommender systems based on social information. In these systems the clustering improves several areas, such as tagging, explicit social links and explicit trust information. The final part of a recommender system is the evaluation of the system. The evaluation of recommender systems is done through typical information retrieval metrics, like Mean Absolute Error, Precision and Recall. The MAE is a quantity that measures how close predictions are to the evaluated outcome. It can be calculated with equation 1, where f_i is the prediction and y_i is the real value.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i| \tag{1}$$

Precision and recall are closely related terms that describe the classification quality of a system. Terms like true positives, true negatives, false positives and false negatives are used to define these values. These terms compare the results of the classifier considered with the actual value. The terms positive and negative refer to the classifier's prediction and the terms true and false refer to whether the prediction is the same as the actual value. Table 2 illustrates this. Precision is calculated according to equation 2 and is also called the positive predictive value. Recall is calculated according to equation 3 and is also referred to as the true positive rate or sensitivity.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

		Actual value	
		Positive	Negative
Predicted value	Positive	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	True negative (TN)

Table 2: Confusion matrix

2.2 The cold-start problem

The cold-start problem [69][2] is a problem that arises when it is not possible to make reliable recommendations due to a lack of initial ratings. The cold-start problem can be divided in three types: new community, new item and new user. The last one of these three types is the most difficult one to solve.

The new community problem [70][37] occurs when a recommender system is starting up in a community. At that point in the system has to cope with the difficulty of obtaining a sufficient of data, for example ratings, to make reliable recommendations. Some ways to tackle this problem are to encourage users tot give ratings through different means or to take collaborative filtering based recommendations their is sufficient data, so when there are enough users and ratings.

The new item problem [57] occurs when new items are added to the recommender systems. These items do not have initial ratings and might not be taken into account in the recommendation process. This means that the item will never become known the community of users and will therefore never receive ratings. This leads to a vicious circle in which items in the recommender system are never used in the recommendation process. A common approach to solve this problem is to have several users be responsible for rating each new item in the system.

The new user problem [64] is the most difficult problem to solve in recommender systems. A new user entering a recommender system has not provided the system with any ratings yet. Therefore personalized recommendations based on memory-based collaborative filtering cannot be given. On top of that, users may think that, when they enter a couple of ratings, the system can already offer personalized recommendations. Usually this is not the case, since the number of ratings provided by the user is not yet sufficient enough to give very reliable recommendations. This may result in users feeling that the recommender system does not provide the service that is expected and they will stop using the system.

To tackle the new user problem, it is common to turn to additional information, apart from ratings, in order to be able to give recommendations based on the data available for each user. The hybrid type of recommender systems are mostly effected by the new user cold-start problem. Ahn [4] present a new similarity measure which is composed of three factors of similarity, proximity, impact and popularity. The similarity between users is based on the similarity between ratings of the users based on those three factors of similarity, utilizing domain specific interpretations of user ratings on products. The tests done by the authors show that this measure performs well for cold-start conditions. Bobadilla et al. [11] present another new similarity measure to account for the lack of additional information of new users. The metric consists of a linear combination of a group of simple similarity measures in order to obtain a global similarity measure between pairs of users. In this linear combination it is necessary to assign weights to the different similarity measures to present the importance of each measure.

The test done on this similarity measure show very promising results, as it runs much faster than other measures and it also outperforms on accuracy.

2.3 The kNN algorithm

The most used algorithm in collaborative filtering recommender systems is the kNN algorithm. The main reason for using this algorithm is that it is a fairly simple algorithm and it has a reasonable accuracy. It is a very simple straightforward implementation with good-quality predictions, but the similarity measures encounter processing problems and cold-start situations [70] due to the high level of sparsity in recommender system databases [43].

Another problem that arises when using the kNN algorithm is the low scalability problem [43]. As the size of the databases increase, the generation of a neighborhood becomes too slow. For every new user in the database similarity measures must be processed. The item to item version of kNN reduces the scalability problem [67]. Instead of calculating neighbors for user, the neighbors for items are calculated and the top n similarity values are stored. For a certain amount of time this stored information is used for providing predictions and recommendations to users. Even though the stored information does not have ratings from previous storage, for items outdated information is less sensitive than for users.

In collaborative filtering calculating the similarity between users (or items) accurately and precise is done by generating metrics. A series of metrics have been used [2][17]: the Pearson correlation, cosine, constraint Pearson correlation and mean squared differences. Similarity measures will be discussed in section 2.4. Other metrics have been created to fit the constraints and peculiarities that recommender systems have [12]. To give more importance to more relevant items and users, the relevance concept has been introduced [10]. Metrics have also been specifically created to deal with the cold-start situations [4][11].

The kNN algorithm uses some of theses similarity measures. Similarity measures compute the similarity between users x and y based on the respective ratings both users give to items. Whereas the item to item kNN does the same for the similarity between items i and j.

A formal approach for the kNN algorithm can be found in [9]. To get a basic understanding of the kNN algorithm an example will be given. Making recommendations is based on the following three steps: (1) When a similarity measure is selected, produce the set of k neighbors for user a. These neighbors are the nearest k users to a. (2) In order to obtain the prediction of item i, one of the following aggregation approaches is used: average, weighted sum and deviation from mean. (3) Choose the n items which match the preferences of the user best. These will be the top-n recommendations. Figure 1 shows a simple example of the user to user kNN. In this example k is equal to 3, the similarity measure is 1 (mean squared differences) and the aggregation approach used is the average.

When using item to item kNN the following steps are done: (1) First

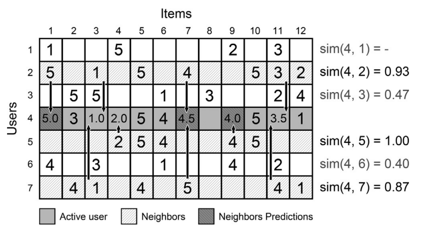


Figure 1: User to user kNN algorithm example.

determine the set of item neighbors for each item present in the database. (2) For each item which is not ranked by the user, calculate the prediction value based on the ratings of the neighbors of the item. (3) Select the top-n recommendations for the user. This procedure is faster than the user to user kNN version because the first step can already be done beforehand.

Both of the approaches (user to user and item to item) can also be combined [80]. This is done in such a way that the positive aspects of both approaches are combined in the hybrid approach. This fusion is done in a probabilistic fusion framework. This new framework has improved the prediction quality and it is able to deal with the sparsity problem.

2.4 Similarity measures

Similarity is the measure of how much alike two data objects are. Similarity in data mining is described as a distance with dimensions representing the features of the data objects. When the distance between objects is small the objects have a high degree of similarity, whereas a high distance between objects means a low degree of similarity. Care should be taken when calculating the distance on features that are unrelated. The values of each feature have to be normalized in order to prevent dominance of a single feature. Similarity has a value in the range 0 to 1, where a value of one means that two objects are identical and a value of zero means the objects are completely unidentical. In recommender systems the similarity between users and items can be measured. The ratings of all items by users can be used (user to user) or all users who have rated an item can be compared (item to item).

The kNN algorithm uses traditional similarity measures of statistical origin to determine the nearest neighbors. All these similarity metrics need are the votes made by the users on the items. Some of the most common used similarity measures will be discussed in this section, like Euclidean distance, Manhattan distance, cosine distance and Pearson correlation.

The most common use of distance is the Euclidean distance. When data is dense or continuous, this is the best proximity measure. The Euclidean distance between two points is the length of the line segment connecting these points. The distance of between points $p = (p_1, p_2, p_3, ...)$ and $q = (q_1, q_2, q_3, ...)$ is given by the Pythagorean formula:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{4}$$

The Manhattan distance is a metric in which the distance between two points is the sum of the absolute differences of their coordinates. It is the absolute sum of differences between all features of the object. The Manhattan distance between two point is measured along the axes at right angles. The distance of between points $p = (p_1, p_2, p_3, ...)$ and $q = (q_1, q_2, q_3, ...)$ is given by the formula:

$$d = \sum_{i=1}^{n} |x_i - y_i| \tag{5}$$

Another well know similarity measure is the cosine similarity metric. The cosine similarity measure finds the normalized dot product between two attributes. By determining the cosine similarity, the cosine of the angle between the two objects is calculated. When the vectors representing the object in an n-dimensional space overlap completely the angle is 0° and the cosine of this angle is equal to 1, whereas a similarity of 0 is represented by an angle of 90°. One of the reasons for the popularity of cosine similarity is that it is very efficient to evaluate, especially for sparse vectors. The cosine of two vectors is derived by using the Euclidean dot product:

$$a \cdot b = ||a||||b||cos(\theta) \tag{6}$$

Given two vectors A and B, the similarity is calculated by the following formula:

$$sim(A,B) = cos(\theta) = \frac{A \cdot B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{7}$$

The final similarity measure is the Pearson correlation [2]. This correlation is a measure of the linear correlation between two variables. The values for this measure range from -1 to 1, where 1 is complete positive correlation and -1 is complete negative correlation. The value 0 denotes no correlation. This correlation coefficient is used to measure the linear dependence between the two variables. The similarity between two users x an y is measure as follows:

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{x,s} - \overline{r}_x)(r_{y,s} - \overline{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \overline{r}_x)^2}\sqrt{\sum_{s \in S_{xy}} (r_{y,s} - \overline{r}_y)^2}} \tag{8}$$

In this equation, S_{xy} denotes the set of items that are rate by both users x and y, so $S_{xy} = \{s \in S | r_{x,s} \neq \emptyset \wedge r_{y,s} \neq \emptyset\}$, where $r_{x,s}$ and $r_{y,s}$ represent the rating of users x and y for item s respectively.

3 Social information

Developments in the web through social media like Facebook have resulted in recommender systems incorporating social information into their design (e.g. followers and followed, friends lists, post). This new form of information [78] has improved recommender systems. By adding social information the sparsity problem in memory-based recommender systems has been improved. Social information reinforces memory-based information by assuming that users that are connected through a network of trust have higher similarity in their interest in items than non-connected users.

Social information is used for three main objectives: (a) to improve the quality of prediction and recommendations [18][5], (b) to propose or generate new recommender systems [41][75], (c) to explain the relationship between social information and collaborative filtering [30][59].

When working with social information in recommender systems trust and reputation have an important role [54]. Approaches for generating trust and reputation measurements are done on two field, user trust and item trust. In user trust the credibility of a user is either calculated through explicit information from other users [85][40] or it is calculated through implicit information obtained through the social network [48]. For item trust the credibility of the item is either calculated through users' feedback [33] or by studying how users work with the item [20].

By assigning labels to items, users can enable the recommender system with extra information. The information then stored as a triple containing information of the user, the item and a tag. This information space is called a folksonomy. Folksonomies can be used to create tag recommendation systems [46] or to expand the recommendation process by using tags [25].

By allowing users to add content (e.g. comments, ratings, labels) and enabling them to create social relationship links, content-based filtering has become far more important. This contextual information has lead to an increase in accuracy of predictions and recommendations made by the system.

3.1 Social filtering

As stated before, social information can be gathered explicitly or implicitly through a network by using the information the user generates [59]. Both explicit and implicit information sources can be combined to generate recommendations. Research using social information in recommender systems is focused on three objectives.

Most of the research is focused on improving the recommendations by referring to the extra information gained from the social information. Woerndl and Groh [82] have added a social model to the collaborative filtering process. This model is based on the fact that a person in a group, who is seeking advice, is greatly influenced by the opinion of the leader of the group or other group members. Through empirical studies they have showed that adding social context does perform better than a standard collaborative filtering approach. In their research Arazy et al. [5] have build a framework using online social networks and electronic tools. Using this framework they have tackled the cold-start problem and have increased the accuracy of the recommendations. Fenkung and Hong [42] have also done research on the effect of adding social network information to a standard collaborative filtering process. Instead of picking just the nearest neighbors they also consider the relationship of these neighbors with the user by looking at the social network. They performed four experiment using different approaches: collaborative filtering with nearest neighbors, collaborative filtering with friends, collaborative filtering with nearest neighbors and friends and collaborative filtering with nearest neighbors and amplifying friends' preferences. The results show that the hybrid methods using not just nearest neighbors, but also considering friends perform better than both methods used separately.

Another research area focuses not on improving existing recommender systems, but create new systems based on social information or enable existing recommender systems to use social information. Siersdorfer and Sergei [75] have build a recommender systems based on folksonomies. The model they used captured dependencies between users, items, tags and other social information by representing the information in a vector space model. The results showed that when using folksonomies, the information present in just the folsonomies (e.g. user, item, tag) is not sufficient. Other social information, like contact and favorites have proven to be of great importance in the recommendation process. In their research Li and Cheng [39] propose a system for recommending blog articles that uses a combination of trust, social relationship and semantic analysis. The trust indicates the trustworthiness and reliability of the target, where the social relationship score indicates the social intimacy in the blog social network. The semantic analysis just indicates the how similar separate blog articles are. They give each of these aspects a score and use these separate scores to give the final recommendation. They have showed that using these factors in a recommender system yield good results in recommending blog articles.

A third research area is to explain the relationship between social information and collaborative processes. This research does not focus on creating or improving recommender systems, but specifically focuses on a more abstract level aiming to create principles. Bonhard [14] explains that the relationship between recommender and recommendee have a significant impact on decision-making. The research goals were to not only get a better understanding of the way people seek advice, but also to apply this understanding in a recommender system test. They wanted to find out how this influenced the decision-making process. Hossain and Fazio [30] present a study in which the connection between social networks and a collaborative process is explored. They look at academics' network position and its effect on their collaborative networks. Two types of networks of collaboration are considered. One is on citations and the other is on co-authorship. By using these types of networks and using this as a network position, they have developed a social networking that uses the academics as nodes instead of the paper that are published. The result have shown that using the academic's network position through the citations does not yield good results to predict future collaborative links between academics. When using social information, trust plays an important role. In their research Golbeck and Kuter [28] present a study based on several trust inference algorithms. They state that most algorithms that use trust do not take into account the changes in trust between people. Because trust is changing a lot, so does the social network. By presenting several types of trust inference algorithms they want to answer several questions on trust and change. The results of their experiment provide insight into which of the algorithms considered are most suitable for certain applications.

3.2 Content-based filtering

In section 2.1.2 content-based filtering has already been discussed, but with social information the usage of this filtering method has increased drastically. New content is added to the items' attributes by adding social information (e.g. tags, comments, opinion). This extra information has produced new information spaces (folksonomies). Using those information spaces, new research on recommender system can be classified in two categories: (1) tag recommendation systems, and (2) using tags in the recommendation process:

1. Recommender systems based on tags provide personalized item recommendation through the most representative tags. Jächke et al [32] present two tag recommendation algorithms. The first algorithm is an adaptation of user-based collaborative filtering and the second one is a graph-based recommender algorithm. They evaluate and compare both of these algorithms with other non-personalized methods, showing that these new personalized algorithms outperform the non-personalized methods. In the Marinho and Schmidt-Thieme [46] research the collaborative filtering method is cast upon the tag recommendation problem. By comparing the collaborative tagging to several simpler tag recommenders they have shown that collaborative filtering based on a user-tag profile matrix is a significant improvement over the simpler tag recommendation methods. This research shows that users with a similar tag vocabulary also tag the same way. This states the importance of personalized tag recommendation systems.

2. The goal of methods using tags in the recommendation process is to improve the capacity of current recommender systems. Tso-Sutter et al

[79] discuss a method which allows tags to be incorporated in standard collaborative filtering methods. They have also found an approach to reduce the three dimensional correlation between users, items and tags by first using their tag extension method and than fusing the results. In their experiment they show that this fusion method outperforms other basic models, especially with the incorporation of tags. Gedikli and Jannach [25] use tags to express which features of an item users like. From this point of view users not only add a tag to an item, but also add a preference or rating to this tag. Since user ratings are sparse in commercial recommender applications, they try to infer the user's opinion based on the tags the user add to an item. Evaluation of this method has shown that it performs slightly better than other tag-based recommender systems.

4 Internet of things

Looking at the evolution of recommender systems there is a tendency to gather data in different ways and from different resources. This trend is the same as the evolution of the web, which can be classified in three stages: (1) at first recommender systems only used demographic information and content-based information included in the items which was only gathered explicitly. (2) With the web evolving with social information, recommender systems also added this social information to the process. This information can be gathered in several ways, as described in section 3. Users also aid the inclusion of this information through tags, comments, etc. (3) With the current Internet of things, context-aware information, such as geographic information or health signals can be added to the recommender systems.

Context-aware recommender systems [3][1] use extra contextual information, such as time, location and wireless sensor information [24]. This contextual information can be gathered explicitly, implicitly, using data mining or with combinations of these method (hybrids). Mobile applications already use geographic information, which enables geographic recommender systems better known as location-aware recommender systems. For these geographic recommender systems [55] recommendations are given based on the location of the user.

This section first gives an introduction of important concepts incorporated in this new field of recommender system research, such as Internet of things, privacy preservation and shilling attacks. Next, research on the location-aware recommender systems will be discussed. This is the first step of recommender systems into the Internet of things. These recommender systems are the start of a very promising research field.

4.1 Introduction of concepts

There is a clear trend towards the collection of implicit information instead of the traditional collection of information through explicit ways like ratings. Gathering information implicitly can easily be applied to several everyday situations, such as food purchased, access to sports facilities, use of public transport systems and access to learning resources.

This incorporation of implicit information on everyday activities of users allows the recommender systems to use more data, which can be used in the collaborative filtering process. By gathering information through the Internet of things, privacy and security issues become more important.

Privacy is an important issue in recommender systems [7] because these systems contain information on millions of users. In order to preserve the privacy in recommender systems, a level of uncertainty must be added to the prediction [51]. Differential privacy can be used to achieve this. This method constrains the computation in such a way that it is not possible to find the records on which the prediction was based. By using these algorithms to produce uncertainty a trade-off between accuracy and privacy has to be made. Privacy can also be preserved if companies start combining their data [34]. The more social and contextual information is used in recommender systems, the more important privacy becomes.

Recommender sytesms are used in various fields including electronic commerce. Producers may find it profitable to shill recommender systems by lying to the system in order to make their product look much better than the products of the competitors. Because of this recommender systems can experience shilling attacks [36], which generate positive ratings for a certain type of product of a producer while negatively rating products from competitors. Current recommender systems are still vulnerable to shilling attacks [65].

Recently, knowledge based filtering is becoming an important field of research in recommender systems. Knowledge recommender systems use knowledge about users and items to generate predictions by reasoning what items meet the requirements of the user. The recommendations are based on the user's needs and preferences. These models are based on knowledge structures such as queries, constraints and social knowledge.

Incorporating different types of information (explicit ratings, social relations, locations) has produced hybrid methods for recommender systems. Once memory-based, social and context-based methods have become better at their prediction accuracy, the evolution of recommender systems will steer towards producing recommender systems with hybrid methods. Research based on only a single source of information has shown that the improvements for predictions and recommendations are much less than when several algorithms are combined with their respective data types. Simultaneously incorporating memory-based, social and context-based information is becoming more important.

To unify the considered concepts figure 2 shows the original taxonomy for recommender systems. The core of this taxonomy focuses on: (1) the target of data: user or item; (2) mode of acquisition: explicit or implicit; and (3) information levels: memory, content, social or context.

4.2 Geographic recommender sytems

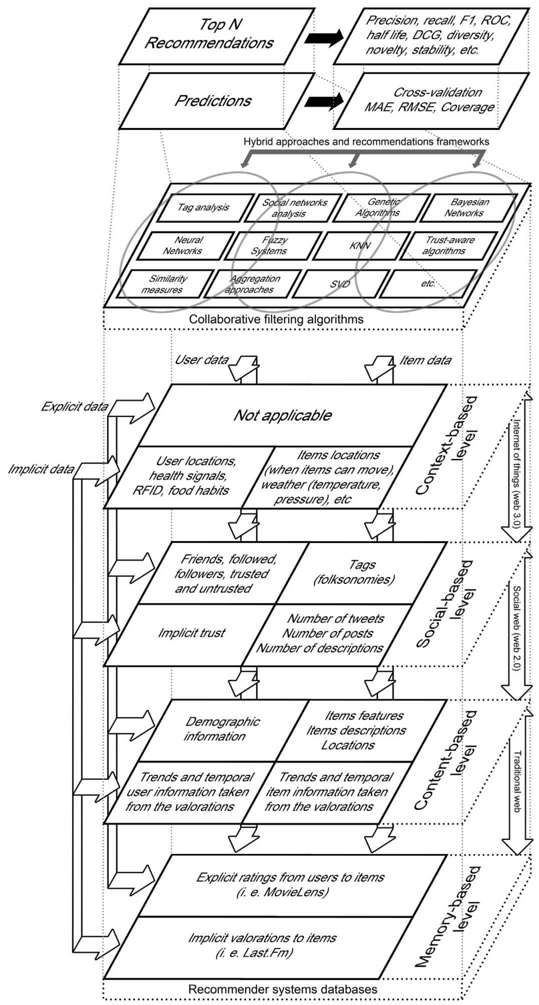
The increased use of mobile devices has lead to an increase of location-aware systems by using the GPS functions. More of these systems are created and this will lead to location-aware collaborative filtering and location aware recommender systems, which will be called geographic collaborative filtering and geographic recommender systems.

The geographic information can be used for both items and users. This leads to a classification of geographic recommender systems in the following categories:

- RS: these are the traditional recommender systems in which ratings and recommendations are made without the use of the geographic information of either category.

- RS+G: these are the traditional recommender systems which also use the geographic information of the items. These systems cannot be truly regarded as geographic recommender systems because the geographic information is not used in the recommendation process.

- GRS: this the group of geographic recommender systems that will be used most in the future. In these systems ratings are given in the



Top N Recommendations
Precision, recall, F1, ROC, half life, DCG, diversity, novelty, stability, etc.
Predictions
Cross-validation MAE, RMSE, Coverage
Hybrid approaches and recommendations frameworks
Tag analysis
Social networks analysis
Genetic Algorithms
Bayesian Networks
Neural Networks
Fuzzy Systems
KNN
Trust-aware algorithms
Similarity measures
Aggregation approaches
SVD
etc.
Collaborative filtering algorithms
User data
Item data
Explicit data
Implicit data
Not applicable
User locations, health signals, RFID, food habits
Items locations (when items can move), weather (temperature, pressure), etc
Context-based level
Internet of things (web 3.0)
Friends, followed, followers, trusted and untrusted
Tags (folksonomies)
Implicit trust
Number of tweets Number of posts Number of descriptions
Social-based level
Social web (web 2.0)
Demographic information
Items features Items descriptions Locations
Trends and temporal user information taken from the valorations
Trends and temporal item information taken from the valorations
Content-based level
Traditional web
Explicit ratings from users to items (i. e. MovieLens)
Implicit valorations to items (i. e. Last.Fm)
Memory-based level
Recommender systems databases

Figure 2: Recommender system taxonomy

traditional way and the recommendations are based on the geographic location of the user to whom the recommendation is to be made. A good example is a recommender system for restaurants. The users give ratings to restaurants in various ways but the distance to the restaurant at the time of voting is not considered when giving ratings. When looking at recommendations, a user does not only want restaurants recommended according to the rating, but also according to the distance between the current position of the user and the restaurants. This same principle can be applied to several other public places like pubs, sports clubs, supermarkets, etc.

The remainder of this part will be about current research on the GRS-type of geographic recommender systems. Due to a lack of public databases that include ratings and geographic positions capable of being combined in a recommender system, the research on geographic information based recommender systems has not progressed as far as the other fields already discussed. A true GRS-type of geographic recommender system has not been created yet.

Marintez et al. [47] and Biuk-Aghai et al. [8] are some examples of research in the RS+G group. More closely related to the GRS group is the research of Schlieder [71]. A new approach is proposed for modeling the collaborative semantics of geographic folksonomies. This is done by allowing so called multi-object tagging, giving a single tag to a larger group of objects, like a group of photographs.

Wan-Shiou et al. [83] propose a recommender system that uses both content-based information and geographic information to recommend vendor offers and promotions. The idea is that the mechanism analyses the user's history and position to rank vendor information according to the match and the preferences of the user. The experimental results show that the hybrid method of content-based and geographic information performs better than both methods separately.

The research closest to a GRS-type is from Matyas and Schlieder [49]. They show a collaborative system that could be placed between a traditional recommender system and a geographic recommender system. Ratings are based on the photos users download from the web and the photos users have uploaded. The uploaded photos also had a GPS address associated to them. Then a search of k-neighborhoods is carried out on this data. Even though it takes geographic information into account, it is not a true GRS-type of geographic recommender system because it does not use the user's position in the recommendation process.

5 Literature summary and research question

Recommender systems prove to be a useful tool for addressing the problem of finding the interesting information from the enormous amount of information on the web. They can be used in several different fields, such as movies, websites, music, etc. The goal of the system is to provide users with just a small subset of the information that is relevant to them. Recommender systems have been developed in three stages. The first recommender systems produced recommendations based on (1) demographic data collected from the user, (2) content-based data from the items a user has purchased or seen, and (3) collaborative data collected from the preferences of a user through ratings. In the second phase of recommender systems, social information was added (e.g. friends, followers, etc.). The third and current generation of recommender systems uses the Internet of things. In these systems the data is collected through several devices and sensors (e.g. health signals, location, RFID, etc.).

At first recommender systems mainly focused on improving the accuracy of the individual filtering methods as far as possible. At the point where the individual methods could not be significantly improved anymore hybrid systems were created. These systems consisted of a combination of one or more filtering methods. The main hybrid systems were collaborative-demographic filtering or collaborative-content filtering. After these developments and with the increase of social media, algorithms were developed to incorporated this social information into the hybrid recommender systems. Finally, with the rise of mobile devices and the Internet of things, location information was added to recommendation algorithms.

Future research could focus on improving the current algorithms and methods by increasing the prediction quality of the recommender systems. Apart from that, new research can also be done on creating more hybrids using the newer techniques like social information or location information, creating hybrids that use data from different sources of information. As discussed in section 4 there are still a lot of possibilities in getting the maximal potential out of the devices and sensors in the Internet of things. Another important aspect for recommender systems is the security and privacy of the data. Developing better security systems for recommender systems will always be an important area of research.

This research focus on the second wave of recommender systems, the social recommender systems. This study will examine the impact of the social network in social recommender systems. Therefore the research question will be: Does adding a social network to the recommendation process have a positive effect on the accuracy of the recommender system?

6 Experiment

The goal of this experiment is to test what the influence of a social network is when it is added to the recommendation process. In order to test this, a recommender system is built incorporating not only standard recommendation techniques like collaborative filtering, but also social filtering. The performance of these techniques is measured by calculating the accuracy they achieve. After analyzing this performance, the influence of the social network can be discovered.

6.1 The database

One of the main aspects of a recommender system is the database it uses to give recommendations. For this research the CiaoDVDs database[29] is used for the recommender system. This database consists of files with movie rating, review ratings and friendship relationships. The review ratings will not be considered because the goal is to present users who rated movies with a list of new movies. The important files are the files containing the movie ratings and the friendship relationships. The file containing the movie ratings consists of approximately 72700 entries, each entry consisting of a user ID, movie ID, genre ID, review ID, movie rating and date. The file containing information on the friendship relationships consists of 40000 entries. Here, all entries consist of two user IDs and a trust value. The information from these two files will be used as input for the recommender system.

For this research not all users that are in this database can be used. The total number of users in the database consists of three types of users: users who have only rated movies, users who have only rated reviews and users that have done neither one of the two. The users considered in the recommender system are those users that have not only rated at least one movie, but also have friends. The latter of these can be obtained from the friendship relationship file. Since social filtering is the main point of this research, users that do not have a social network will not give any useful information and will therefore not be considered. After all these requirements are met, the correct database of users and movies is created. The database of movies consists of all the movies in the original database.

6.2 The method

With a useful database at hand, the analysis of the influence of a social network on the recommendation process can begin. In order to give a good estimate of this influence, a baseline has to be created to compare the results of the other algorithms too. Five different types of algorithms are used for testing the influence of the social network. The baseline for this research is created by using two of these five algorithms. The other three algorithms are used to gather information for answering the research question.

6.2.1 The baseline algorithms

The two baseline algorithms are based on only one source of information. One of the algorithms is based on the standard techniques used in the first wave of recommender systems. The standard techniques consist of collaborative, content-based and demographic filtering. In this research the decision was made to use collaborative filtering as the filtering method. This decision was based on the fact that collaborative filtering performs better than demographic filtering and the information that was present in the database did not provide enough information on the content of the movies to use content-based filtering. The only content available was the genre of the movie, which is not enough information to create an accurate content-based filtering recommender system. So the collaborative filtering method is used based on the ratings users have given to certain movies. This method uses the well known kNN algorithm to create a neighborhood of users based on a similarity measure. The similarity measure used is the mean squared difference[73]. This technique applies to two users. For each movie that both of the users have rated, the difference between their ratings is squared. These squares are summed up and the sum is divided by the number of movies rated by both users. The resulting value is tested against a certain threshold L to see if the users are similar. In their paper the authors show that with the threshold L set to a value of 2.0 the system had the best performance. After running tests on the recommender system used for this research, it could also be concluded that the value of 2 was optimal. If it did not find any users using this threshold, it had to be increased to more than 50 in order to find other users to put in the user's neighborhood. Increasing L this much only occurred a few times. Furthermore, doing this would not result in a good neighborhood of neighbors that are similar to the user. Therefore the decision was made to only look at the value of L being equal to 2. If this mean squared difference will result in a neighborhood with more than K nearest neighbors, only the K nearest neighbors will be selected as the neighborhood for the user. Based on this neighborhood the list of predictions will be generated for the current user. This list will be created by taking a weighted average of all the users that have rated a movie, where the weights represent the similarity between two users. This weight is calculated for each user k in the neighborhood of the user i. The following formula is used:

$$w(i,k) = \frac{L - D(i,k)}{L} \tag{9}$$

Here, $D(i,k)$ denotes the mean squared difference between two users. By assigning these weights to the neighbors, the rating of a neighbor that has a lot in common with the current user will have a larger impact on the final rating the system would give for a specific movie.

The second baseline algorithm is a rather simple algorithm that only uses the friendship relationships as its source of information. The neighborhood consists only of the friends of a user. In order to see how well the social network performs on its own the recommendations for the user will only depend on the ratings of the friends of the user. Seeing as how friends are generally chosen by the user because of matching interests, this should lead to good results. People are, after all, more tempted to start watching a movie when a lot of friends recommend it. Because in this database all friends are trusted equally well, the rating for a specific movie is created by just taking the average of all friends that have rated the movie.

6.2.2 Extensions of the baseline algorithm

As stated earlier, three more algorithms have been used to test the influence of the social network. With these two baseline algorithms the performance of the separate methods can be compared, but nowadays most recommender systems use hybrid recommendation methods. In this research three different types of hybrid methods are considered.

The first one is an extension of the baseline algorithm using the friendship relationship. Even in real life, people don't trust all their friends equally well. There are always a couple of friends whose advice you would follow much easier than other friends'. Since in the baseline algorithm all friends are considered as equally trustworthy, this extension provides a slightly different approach. Collaborative filtering is added as an enhancement to the algorithm. Instead of simply taking the average of all ratings as the value of the new rating, weights are first calculated for each friend. These weights are calculated in the same way as is done in the other baseline algorithm, but instead of only looking at the k nearest neighbors, all friends are still considered to be the neighborhood. In this situation it can occur that the friends have not rated the movies that the current user has rated, because of the sparseness of database with social information. In this case a rating is predicted for these movies based on the genre of the movie. The average of all other movies with the same genre is calculated as the rating the friend would give to the movie. Using this rating a similarity value can be calculated using the same technique as before. If it is also not possible to use this way of calculating a similarity, the similarity is just set to 1, which is the base value. By adding this enhancement the friends that also have a more similar way of rating (and therefore have more similar interests) are considered as more trustworthy than other friends. In the end not the normal average, but a weighted average is used to calculate the new rating for each movie that the user's friends have rated, but the user has not.

Such an extension can also be done the other way around. After generating the neighborhood using the kNN algorithm, the neighborhood is refined by giving the neighbors that are also in the user's friend list a weight of 1. That way the opinion of friends is considered as being more important than the opinions of strangers. Again, this can be linked to the real life. People would rather believe their friends than a person they don't even know. This change is added after the creation of the neighborhood. The rating for a movie is then created by taking the weighted average of all the users in the neighborhood who rated that movie. By changing the weight of a friend to the 1, it is easy to see that the opinion of a friend will have a higher impact on the new rating.

The final hybrid algorithm used in this research is a weighted hybrid of both baseline algorithms. It first creates the list of movie predictions for both the collaborative filtering algorithm and the friendship algorithm and after that it takes a weighted value of both algorithms. So if the collaborative filtering counts for 20% then the friendship filtering will count for 80%. This way it will always sum up to 100%. This method only works if a movie is recommended by both algorithms. This is not the case for all movies. For those movies that are recommended by both algorithms, this weighted method is used to calculate the rating, and otherwise the rating is used which is given by the baseline algorithm. So the resulting set of movies is a combination of weighted and non-weighted movie ratings. This is due to the fact that the database is sparse. The more ratings the database has, the more weighted movie ratings will be presented.

6.3 Training and testing the system

To get results on the performance of all the algorithms mentioned above, sample sets are taken from the database. These sample sets are created by first taking one third of the users in the database. These users are memorized by the system and then all the friends of these users, which are not already in the sample set, are added to it. So the end result will be two sets of users. One set containing only the users in consideration and one with all those users and their friends. Since the algorithms use machine learning techniques, parameters have to be trained to get the best results. A total of three sample sets are created for the purpose of training and testing the system. Since a memory-based approach is used in the current setup, only a single sample set is used for training the parameters, which are K for the kNN algorithm and λ for the weights in the weighted hybrid method. The other two sample sets are used for testing the system. By already implementing three sample sets, the system can easily be extended with model-based approaches. In the case of model-based approaches two sample sets are used for training, one to create a model and the other to find the optimal parameters for this model. The final sample set can then be used for testing the system.

In order for the system to learn what the optimal values are for the parameters, an evaluation criterion has to be set. The accuracy of the system depends on the ratings the system gives to the movies that a user has already seen. If the predicted ratings are just as high as the ratings the user has given to the movies himself, then the system has correctly predicted the value for a specific movie. This is measured by recommending movies to each user in a sample. During this process a certain movie is removed from the list of movies that the user has rated. Afterwards it is checked if the movie has been added to the list of recommended movies. In this case it would be enough if the movie is in the list and has the same rating as the user had given it. Taking the top n movies from the list of recommendation did not work for this database, because of the amount of movies that had a recommended rating of 5. The removed movie was of course a movie the user gave a high rating (preferably 5) and with a high number of movies with a recommended rating of 5, this removed movie might not be in the top n. This would be a random process and therefore would not give a good estimate of the accuracy of the system. Therefore the decision was made to just look at the movies that have the same rating as the removed movie. If it would be in that list, the system has accurately predicted the rating for that specific movie and would therefore also give accurate recommendations.

Now that an evaluation criterion has been set, the system can be trained and tested based on its accuracy. Since the weighted hybrid algorithm is the only algorithm that uses both parameters, the optimal values of the parameters are set to those values that result in the highest accuracy for this algorithm. The training is done by increasing K from 1 to 10 and for each K increasing the values of λ from 0.0 to 1.0 by taking steps of 0.1. The results of this training will be discussed in section 7. With the optimal values for the parameters, the system is tested on the two remaining sample sets. For each test the accuracy is calculated for each of the five algorithms.

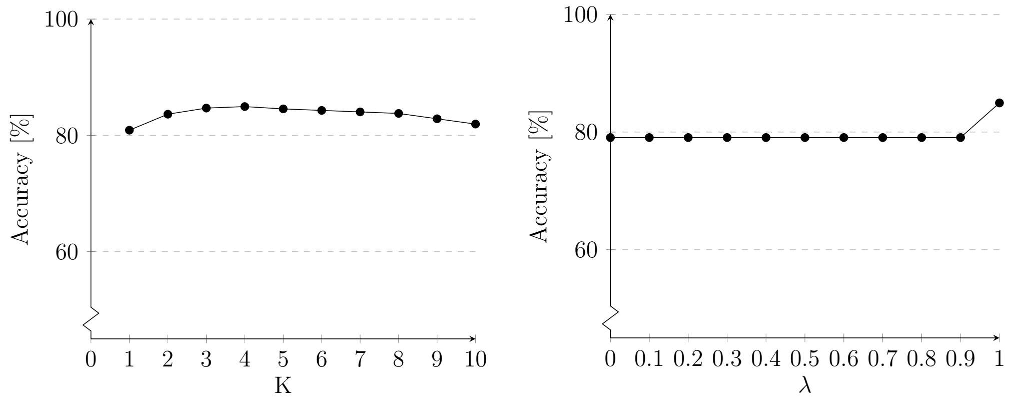
7 Results

The results of this experiment are divided in two parts. First the results of the training will be presented and using these results, the result of testing the system will follow. As mentioned in section 6.3 the system is trained on one of the sample sets to get results from which we can deduce the optimal value for K and λ. The weighted hybrid method is used for the purpose of training, since it is the only algorithm that uses both these parameters. The accuracy results are shown in table 3.

Looking at the values in the table, it is clear that the best overall value of K is equal to 3 since it produces the highest accuracy overall. However, this is not the value that actually gives the highest accuracy. The highest accuracy is achieved when $K = 4$ and $\lambda = 1.0$. For these values the patterns for both the accuracy for K and λ are shown in figure 3a and 3b respectively. These patterns clearly show that $K = 4$ and $\lambda = 1.0$ are the optimal values. Moreover, the pattern in figure 3b roughly holds for all values of K. So changing K does not change the results of the optimal value of λ. As a matter of fact, for all values of K the algorithm performed best for $\lambda = 1.0$. From these result we can already conclude that the weighted hybrid algorithm will not use the ratings from a user's friends if the movie is already present in the list of predicted movies presented by the collaborative filtering algorithm. This is due to the fact that in most cases the friends of a user have not rated enough movies. The list presented by the collaborative filtering algorithm contains a lot more movies. It often happens that the removed movie is not rated by any of the user's friends, but is rated by one of the nearest neighbors. This is due to the fact that the database is sparse.

		The value of λ										
		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
The value of K	1	78,01	78,01	78,01	78,01	78,01	78,01	78,01	78,01	78,01	78,01	80,89
	2	79,18	79,18	79,18	79,18	79,18	79,18	79,18	79,18	79,18	79,32	83,64
	3	79,19	79,19	79,19	79,19	79,19	79,19	79,19	79,19	79,19	79,32	84,69
	4	79,06	79,06	79,06	79,06	79,06	79,06	79,06	79,06	79,06	79,19	84,95
	5	78,66	78,66	78,66	78,66	78,66	78,66	78,66	78,66	78,66	78,80	84,55
	6	78,14	77,88	77,88	77,88	77,88	77,88	77,88	77,88	77,88	78,01	84,29
	7	77,62	77,35	77,35	77,35	77,35	77,35	77,35	77,35	77,35	77,49	84,03
	8	77,09	76,70	76,70	76,70	76,70	76,70	76,70	76,70	76,70	76,83	83,77
	9	76,31	75,79	75,79	75,79	75,79	75,79	75,79	75,79	75,79	75,92	82,85
	10	75,39	74,87	74,87	74,87	74,87	74,87	74,87	74,87	75,00	75,13	81,94

Table 3: Accuracy (%) for the training of K and λ



(a) Accuracy pattern for K, with $\lambda = 1$ (b) Accuracy pattern for λ, with K = 4

Figure 3: Result of the best value for K and λ.

Using the optimal values obtained from the training sample, the system is tested on two different sample sets. For both of these sample sets the accuracy of all five algorithms was computed. This not only gives an estimate of the performance of the system, but also shows what the influence of the social network is on the recommendation process. The results of these tests can be found in table 4.

Algorithm	Test #1	Test #2
Collaborative filtering	90,75	79,68
Enhanced collaborative filtering	90,75	79,68
Trust filtering	60,59	67,98
Enhanced trust filtering	67,98	68,75
Weighted hybrid filtering	87,61	78,79

Table 4: Accuracy (%) results

Even though the percentages in both tests differ quite a bit for some algorithms, both tests show the same result patterns. When looking at the baseline algorithms it is very clear that social filtering does not perform very well on its own. When compared to a standard filtering method like the collaborative filtering, it is outperformed by it in both cases. A user's friends do not necessarily have to have a similar rating pattern, whereas for the collaborative filtering such a neighborhood was specifically created. When looking at the removed movie, this neighborhood would therefore only consist of users with a similar rating as the user had given to it. It is no guarantee that this is also the case for the user's friends.

The enhancements seem to only work in favor of the collaborative filtering method, because the performance increases only when it is added to the recommendation process. Also, the enhanced collaborative filtering showed no increase in accuracy, whereas the enhanced trust filtering did. This means that the rating patterns from a user's friends are not enough alike for the friends to be considered close neighbors. That explains why there is no change in accuracy between the collaborative filtering based algorithms. What is also interesting to see is that the weighted hybrid filtering does not perform as well as the collaborative filtering method. Since the value of λ is set to 1.0, it would mean that, when a movie is recommended by both baseline algorithms, it would only consider the recommendation from the collaborative filtering method. That means the only explanation for the decrease in performance lies with those movies that are only recommended by the friendship algorithm. This can only happen when the removed movie was only rated by friends and was not rated as high as the user himself had rated it. This only validates the results of the other enhancements.

8 Discussion

For this research we wanted to find out if adding a social network to the recommendation process would have a positive effect on the accuracy of the recommender system. We have created a recommender system that uses the CiaoDVDs database to present movie recommendations to the users. In order to find out what the influence of the social network is, several algorithms were created and for each of these the accuracy was calculated. These algorithms consist of two baseline algorithms and three extensions of these baseline algorithms. Since these algorithms use machine learning techniques the system had to be trained before it could be tested with the most optimal parameters. The results that were obtained from the training and testing showed that a recommender system does not benefit from adding the social network to the recommendation process.

The data obtained from training the system already proved this statement. The weighted hybrid filtering algorithm was used for training both parameters of the system. The parameter λ was used to assign weights to the output of each of the baseline algorithms, with $\lambda = 1$ meaning only the output of the collaborative filtering is considered and the other way around for $\lambda = 0$. The result of the training showed that the system performs best with λ set to 1, meaning it only considers collaborative filtering. This shows that the neighborhood created in this filtering method is much closer to the user than the friends are. Therefore it can be concluded that a neighborhood consisting of friends is less similar than a specifically created neighborhood. Considering that not all of a person's friends have similar interests, it is only normal that the neighborhood created for collaborative filtering performs better.

The results from both tests only confirm that social filtering does not add much to the process. When looking at only the baseline algorithms, it is clear that social filtering does not perform as well as collaborative filtering. With a difference of over 30% in the first test and 10% in the second test, it can be concluded that, on its own, social filtering does not perform well enough. Looking at the accuracies of the enhanced algorithms, it can clearly be stated that adding social filtering does not yield better results. In both tests the accuracy did not increase at all. On the other hand, adding collaborative filtering did yield better results. This was shown by an increase in accuracy in both tests. This actually shows that a social network can be enhanced to show better results. Overall it can be concluded that adding a social network to the recommendation process does not increase the performance of the recommender system and therefore does not have a positive influence on the recommendation process.

Even though this research shows that, at this point, social filtering does not perform as well as standard techniques, it might still be useful as an extra source of information, because of the results obtained from enhancing the social network. This enhancement showed an increase in performance of the system. Instead of using the entire social network after enhancing it, in future research it might be better to look only at the part of the social network that contains friends that are similar to the user. Secondly, the database used in this research did not contain a lot of social information. Users only had a few friends, who in turn only rated a couple of movies. If the database would have been less sparse, the system might have shown better results with respect to the social filtering. This sparsity problem is one of the main problems for recommender systems. So for future research on social filtering it is highly recommended a database is available that is less sparse. Other possibilities for future research would be using different methods in different parts of the recommendation process. For this database the collaborative filtering method had to be used because of the lack of content-based information, but using content-based filtering might also give different results. This also holds for the machine learning algorithm. Instead of using a memory-based algorithm like kNN, which is shown to work well in collaborative filtering, model-based algorithms might present different results. The final recommendation for future research would be to change the evaluation criterion. Instead of looking at the predicted rating of just one movie, it might also be interesting to look at the difference between the ratings that are predicted and the ratings the user had given to movies. So we have seen that social filtering will not perform well on its own, but when the social network is enhanced it could potentially give good results for social filtering.

References

[1] Sofiane Abbar, Mokrane Bouzeghoub, and Stéphane Lopez. Context-aware recommender systems: A service-oriented approach. In *VLDB PersDB workshop*, pages 1–6, 2009.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[3] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.

[4] Hyung Jun Ahn. A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, 178(1):37–51, 2008.

[5] Ofer Arazy, Nanda Kumar, and Bracha Shapira. Improving social recommender systems. *IT Professional Magazine*, 11(4):38, 2009.

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

[7] Alper Bilge and Huseyin Polat. An improved privacy-preserving dwt-based collaborative filtering scheme. *Expert Systems with Applications*, 39(3):3841–3854, 2012.

[8] Robert Parviz Biuk-Aghai, Simon Fong, and Yain-Whar Si. Design of a recommender system for mobile tourism multimedia selection. In *Internet Multimedia Services Architecture and Applications, 2008. IMSAA 2008. 2nd International Conference on*, pages 1–6. IEEE, 2008.

[9] Jesus Bobadilla, Antonio Hernando, Fernando Ortega, and Jesus Bernal. A framework for collaborative filtering recommender systems. *Expert Systems with Applications*, 38(12):14609–14623, 2011.

[10] Jesús Bobadilla, Antonio Hernando, Fernando Ortega, and Abraham Gutiérrez. Collaborative filtering based on significances. *Information Sciences*, 185(1):1–17, 2012.

[11] JesúS Bobadilla, Fernando Ortega, Antonio Hernando, and JesúS Bernal. A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-Based Systems*, 26:225–238, 2012.

[12] Jesús Bobadilla, Francisco Serradilla, and Jesus Bernal. A new collaborative filtering metric that improves the behavior of recommender systems. *Knowledge-Based Systems*, 23(6):520–528, 2010.

[13] JESUS Bobadilla, Francisco Serradilla, Antonio Hernando, et al. Collaborative filtering adapted to recommender systems of e-learning. *Knowledge-Based Systems*, 22(4):261–265, 2009.

[14] Philip Bonhard. Who do trust? combining recommender systems and social networking for better advice. In *Proceedings of the Wokshop Beyond Personalization 2005, in conjunction with the International Conference on Intelligent User Interfaces IUI05*, pages 89–90. Citeseer, 2005.

[15] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[16] Fidel Cacheda, Víctor Carneiro, Diego Fernández, and Vreixo Formoso. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web (TWEB)*, 5(1):2, 2011.

[17] Laurent Candillier, Frank Meyer, and Marc Boullé. Comparing state-of-the-art collaborative filtering systems. In *Machine Learning and Data Mining in Pattern Recognition*, pages 548–562. Springer, 2007.

[18] Walter Carrer-Neto, María Luisa Hernández-Alcaraz, Rafael Valencia-García, and Francisco García-Sánchez. Social knowledge-based recommender system. application to the movies domain. *Expert Systems with applications*, 39(12):10990–11000, 2012.

[19] Jose Jesus Castro-Schez, Raul Miguel, David Vallejo, and Lorenzo Manuel López-López. A highly adaptive recommender system based on fuzzy logic for b2c e-commerce portals. *Expert Systems with Applications*, 38(3):2441–2454, 2011.

[20] Jinhyung Cho, Kwiseok Kwon, and Yongtae Park. Q-rater: A collaborative reputation system based on source credibility theory. *Expert Systems with Applications*, 36(2):3751–3760, 2009.

[21] Keunho Choi, Donghee Yoo, Gunwoo Kim, and Yongmoo Suh. A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis. *Electronic Commerce Research and Applications*, 11(4):309–317, 2012.

[22] Enrique Costa-Montenegro, Ana Belén Barragáns-Martínez, and Marta Rey-López. Which app? a recommender system of applications in markets: Implementation of the service for monitoring users interaction. *Expert systems with applications*, 39(10):9367–9375, 2012.

[23] Rubén González Crespo, Oscar Sanjuán Martínez, Juan Manuel Cueva Lovelle, B Cristina Pelayo García-Bustelo, José Emilio Labra Gayo, and Patricia Ordoñez De Pablos. Recommendation system based on user interaction data applied to intelligent electronic books. *Computers in Human Behavior*, 27(4):1445–1449, 2011.

[24] Damianos Gavalas and Michael Kenteris. A web-based pervasive recommendation system for mobile tourist guides. *Personal and Ubiquitous Computing*, 15(7):759–770, 2011.

[25] Fatih Gedikli and Dietmar Jannach. Rating items by rating tags. In *Proceedings of the 2010 Workshop on Recommender Systems and the Social Web at ACM RecSys*, pages 25–32, 2010.

[26] Thomas George and Srujana Merugu. A scalable collaborative filtering framework based on co-clustering. In *Data Mining, Fifth IEEE International Conference on*, pages 4–pp. IEEE, 2005.

[27] Amir Gershman, Amnon Meisels, Karl-Heinz Lüke, Lior Rokach, Alon Schclar, and Arnon Sturm. A decision tree based recommender system. In *IICS*, pages 170–179. Citeseer, 2010.

[28] Jennifer Golbeck and Ugur Kuter. The ripple effect: change in trust and its impact over a social network. In *Computing with Social Trust*, pages 169–181. Springer, 2009.

[29] G. Guo, J. Zhang, D. Thalmann, and N. Yorke-Smith. Etaf: An extended trust antecedents framework for trust prediction. In *Proceedings of the 2014 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2014.

[30] Liaquat Hossain and Daniel Fazio. The social networks of collaborative process. *The Journal of High Technology Management Research*, 20(2):119–130, 2009.

[31] Zan Huang, Daniel Zeng, and Hsinchun Chen. A comparison of collaborative-filtering recommendation algorithms for e-commerce. *IEEE Intelligent Systems*, (5):68–78, 2007.

[32] Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *Knowledge Discovery in Databases: PKDD 2007*, pages 506–514. Springer, 2007.

[33] Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision support systems*, 43(2):618–644, 2007.

[34] Cihan Kaleli and Huseyin Polat. Privacy-preserving som-based recommendations on horizontally distributed data. *Knowledge-Based Systems*, 33:124–135, 2012.

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

[36] Shyong K Lam and John Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web*, pages 393–402. ACM, 2004.

[37] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 208–211. ACM, 2008.

[38] Seok Kee Lee, Yoon Ho Cho, and Soung Hie Kim. Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations. *Information Sciences*, 180(11):2142–2155, 2010.

[39] Yung-Ming Li and Ching-Wen Chen. A synthetical approach for blog recommendation: Combining trust, social relation, and semantic analysis. *Expert Systems with Applications*, 36(3):6536–6547, 2009.

[40] Yung-Ming Li and Chien-Pang Kao. Trepps: a trust-based recommender system for peer production services. *Expert systems with applications*, 36(2):3263–3277, 2009.

[41] Yung-Ming Li, Tzu-Fong Liao, and Cheng-Yang Lai. A social recommender mechanism for improving knowledge sharing in online forums. *Information Processing & Management*, 48(5):978–994, 2012.

[42] Fengkun Liu and Hong Joo Lee. Use of social network information to enhance collaborative filtering performance. *Expert systems with applications*, 37(7):4772–4778, 2010.

[43] Xin Luo, Yunni Xia, and Qingsheng Zhu. Incremental collaborative filtering recommender based on regularized matrix factorization. *Knowledge-Based Systems*, 27:271–280, 2012.

[44] Xin Luo, Yunni Xia, and Qingsheng Zhu. Applying the learning rate adaptation to the matrix factorization based collaborative filtering. *Knowledge-Based Systems*, 37:154–164, 2013.

[45] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. Scoring, term weighting and the vector space model. *Introduction to Information Retrieval*, 100:2–4, 2008.

[46] Leandro Balby Marinho and Lars Schmidt-Thieme. Collaborative tag recommendations. In *Data Analysis, Machine Learning and Applications*, pages 533–540. Springer, 2008.

[47] Luis Martinez, Rosa M Rodriguez, and Macarena Espinilla. Reja: A georeferenced hybrid recommender system for restaurants. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 03*, pages 187–190. IEEE Computer Society, 2009.

[48] Paolo Massa and Paolo Avesani. Trust-aware collaborative filtering for recommender systems. In *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, pages 492–508. Springer, 2004.

[49] Christian Matyas and Christoph Schlieder. A spatial user similarity measure for geographic recommender systems. In *GeoSpatial Semantics*, pages 122–139. Springer, 2009.

[50] Kevin McNally, Michael P OMahony, Maurice Coyle, Peter Briggs, and Barry Smyth. A case study of collaboration and reputation in social web search. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(1):4, 2011.

[51] Frank McSherry and Ilya Mironov. Differentially private recommender systems: building privacy into the net. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 627–636. ACM, 2009.

[52] Alexandros Nanopoulos, Dimitrios Rafailidis, Panagiotis Symeonidis, and Yannis Manolopoulos. Musicbox: Personalized music recommendation based on cubic analysis of social tags. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(2):407–412, 2010.

[53] Edward Rolando Núñez-Valdéz, Juan Manuel Cueva Lovelle, Oscar Sanjuán Martínez, Vicente García-Díaz, Patricia Ordoñez de Pablos, and Carlos Enrique Montenegro Marín. Implicit feedback techniques on recommender systems applied to electronic books. *Computers in Human Behavior*, 28(4):1186–1193, 2012.

[54] John O'Donovan and Barry Smyth. Trust in recommender systems. In *Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174. ACM, 2005.

[55] Kenta Oku, Rika Kotera, and Kazutoshi Sumiya. Geographical recommender system based on interaction between map operation and category selection. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 71–74. ACM, 2010.

[56] Deuk Hee Park, Hyea Kyeong Kim, Il Young Choi, and Jae Kyeong Kim. A literature review and classification of recommender systems research. *Expert Systems with Applications*, 39(11):10059–10072, 2012.

[57] Seung-Taek Park and Wei Chu. Pairwise preference regression for cold-start recommendation. In *Proceedings of the third ACM conference on Recommender systems*, pages 21–28. ACM, 2009.

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

[59] Saverio Perugini, Marcos André Gonçalves, and Edward A Fox. Recommender systems research: A connection-centric survey. *Journal of Intelligent Information Systems*, 23(2):107–143, 2004.

[60] Carlos Porcel and Enrique Herrera-Viedma. Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries. *Knowledge-Based Systems*, 23(1):32–39, 2010.

[61] Carlos Porcel, Juan Manuel Moreno, and Enrique Herrera-Viedma. A multi-disciplinar recommender system to advice research resources in university digital libraries. *Expert Systems with Applications*, 36(10):12520–12528, 2009.

[62] Carlos Porcel, A Tejeda-Lorente, MA Martínez, and Enrique Herrera-Viedma. A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. *Information Sciences*, 184(1):1–19, 2012.

[63] Juan Ramos. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, 2003.

[64] Al Mamunur Rashid, George Karypis, and John Riedl. Learning preferences of new users in recommender systems: an information theoretic approach. *ACM SIGKDD Explorations Newsletter*, 10(2):90–100, 2008.

[65] Sanjog Ray and Ambuj Mahanti. Strategies for effective shilling attacks against recommender systems. In *Privacy, Security, and Trust in KDD*, pages 111–125. Springer, 2009.

[66] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

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

[68] Badrul M Sarwar, George Karypis, Joseph Konstan, and John Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the fifth international conference on computer and information technology*, volume 1. Citeseer, 2002.

[69] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.

[70] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM, 2002.

[71] Christoph Schlieder. Modeling collaborative semantics with a geographic recommender. In *Advances in Conceptual Modeling–Foundations and Applications*, pages 338–347. Springer, 2007.

[72] Jesus Serrano-Guerrero, Enrique Herrera-Viedma, Jose A Olivas, Andres Cerezo, and Francisco P Romero. A google wave-based fuzzy recommender system to disseminate information in university digital libraries 2.0. *Information Sciences*, 181(9):1503–1516, 2011.

[73] Upendra Shardanand. *Social information filtering for music recommendation.* PhD thesis, Citeseer, 1994.

[74] Subhash K Shinde and Uday Kulkarni. Hybrid personalized recommender system using centering-bunching based clustering algorithm. *Expert Systems with Applications*, 39(1):1381–1387, 2012.

[75] Stefan Siersdorfer and Sergej Sizov. Social recommender systems for web 2.0 folksonomies. In *Proceedings of the 20th ACM conference on Hypertext and hypermedia*, pages 261–270. ACM, 2009.

[76] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Scalable collaborative filtering approaches for large recommender systems. *The Journal of Machine Learning Research*, 10:623–656, 2009.

[77] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Scalable collaborative filtering approaches for large recommender systems. *The Journal of Machine Learning Research*, 10:623–656, 2009.

[78] Shulong Tan, Jiajun Bu, Chun Chen, and Xiaofei He. Using rich social media information for music recommendation via hypergraph model. In *Social media modeling and computing*, pages 213–237. Springer, 2011.

[79] Karen HL Tso-Sutter, Leandro Balby Marinho, and Lars Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *Proceedings of the 2008 ACM symposium on Applied computing*, pages 1995–1999. ACM, 2008.

[80] Jun Wang, Arjen P De Vries, and Marcel JT 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*, pages 501–508. ACM, 2006.

[81] Pinata Winoto and Tiffany Y Tang. The role of user mood in movie recommendations. *Expert Systems with Applications*, 37(8):6086–6092, 2010.

[82] Wolfgang Woerndl and Georg Groh. Utilizing physical and social context to improve recommender systems. In *Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Workshops*, pages 123–128. IEEE Computer Society, 2007.

[83] Wan-Shiou Yang, Hung-Chi Cheng, and Jia-Ben Dia. A location-aware recommender system for mobile shopping environments. *Expert Systems with Applications*, 34(1):437–445, 2008.

[84] Zhiwen Yu, Xingshe Zhou, Yanbin Hao, and Jianhua Gu. Tv program recommendation for multiple viewers based on user profile merging. *User modeling and user-adapted interaction*, 16(1):63–82, 2006.

[85] Weiwei Yuan, Donghai Guan, Young-Koo Lee, Sungyoung Lee, and Sung Jin Hur. Improved trust-aware recommender system using small-worldness of trust networks. *Knowledge-Based Systems*, 23(3):232–238, 2010.

[86] Osmar R Zaíane. Building a recommender agent for e-learning systems. In *Computers in Education, 2002. Proceedings. International Conference on*, pages 55–59. IEEE, 2002.

[87] Sheng Zhang, Weihong Wang, James Ford, Fillia Makedon, and Justin Pearlman. Using singular value decomposition approximation for collaborative filtering. In *E-Commerce Technology, 2005. CEC 2005. Seventh IEEE International Conference on*, pages 257–264. IEEE, 2005.