NTNU
Norwegian University of
Science and Technology

Design of a Bayesian Recommender System for Tourists Presenting a Solution to the Cold-Start User Problem

Terje Nesbakken Lillegraven
Arnt Christian Wolden

Master of Science in Computer Science
Submission date: June 2010
Supervisor: Helge Langseth, IDI
Co-supervisor: Anders Kofod-Petersen, SINTEF

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

The primary aim of the thesis is to propose and develop a design for a recommender system, that may be used in an application for tourists visiting Trondheim. The designed system should be able to give recommendations to users based on specified needs, and it should also be able to use the current location and possibly other context variables in its reasoning. In addition, the design should provide a solution to the problem of giving accurate recommendations to new users of the system.

Assignment given: 25. January 2010
Supervisor: Helge Langseth, IDI

Abstract

Recommender systems aim to provide users with personalised recommendations of items based on their preferences. Such systems have during the last 15 years been applied in many domains and have enjoyed an increased popularity both in research communities and commerce. In this thesis our overlying aim is to work towards creating a recommender system for tourists visiting Trondheim. We begin this work by addressing the *cold-start user problem*, which is the problem of giving high-quality recommendations to new users who the system has little or no information about. The problem is severe in the tourist domain where the majority of users are cold-start users. To properly address the problem, we present a *systematic literature review* of the recommender system literature identifying *nine* types of solutions to the cold-start user problem. We evaluate the solution types in context of the tourist domain, and find that using *demographic user data* is the best solution in this domain. We include this solution as a part when we propose a design of a location-aware Bayesian recommender system for tourists visiting Trondheim.

Preface

This Master's thesis is the result of 20 weeks of work during our final semester at the Norwegian University of Technology. It has been an educational process, and we have enjoyed gaining deeper knowledge of both recommender systems and systematic literature reviews.

We would like to thank our supervisors, Helge Langseth and Anders Kofod-Petersen, for the constructive feedback, the interesting and clarifying discussions, and for always being in a pleasant mood. We would also like to thank all fellow students in room 108, Fiol, for the good working atmosphere throughout the year, as well as for all the much needed coffee breaks.

Trondheim, 21st June 2010

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we give an introduction to our Master's thesis. First we offer an insight into the motivational factors that inspired us to work with recommender systems in the domain of tourism. We then give a statement of the goals of our thesis in Section 1.2 and summarise our contributions in Section 1.3. In the final section of the chapter we explain the structure of the rest of the thesis.

1.1 Motivation and Background

Recommender systems are systems offering personalised recommendations of items to users. They have since their birth in the early nineties gained popularity, both as a subject in research communities and eventually as a commercial product. Recommender systems are today found on websites with millions of visitors such as amazon.com, YouTube and last.fm where users can receive recommendations for books, videos and music respectively.

Also in the tourist domain there has been growing attention towards systems recommending personalised services for travellers, and many systems for the domain have been proposed. Some are website-based, offering assistance in planning holidays with regard to destination, flights and accommodation, while others are systems for hand-held devices offering on-site information and recommendations of sights, activities, eateries and all sorts of waterholes. For the latter category of systems, new, popular and advanced hand-held devices such as smartphones have opened up new opportunities. Possibilities such as presenting maps where the user's current position and the way to the nearest unseen sight is marked, make it highly attractive to design and create applications for tourists on mobile platforms.

In Trondheim, a company co-founded by NTNU, called Wireless Trondheim, has built a wireless network which is available to all residents and visitors of Trondheim. One goal of the company is to support and help research and innovation by students and professionals on mobile services. The infrastructure offered by Wireless Trondheim is ideal to support a location-aware recommender system for tourists on a walkabout in Trondheim. To the best of our knowledge, no such system exists today.

Because of all these factors, we defined it as the overlying aim of our thesis to begin the work with creating a recommender system for tourists visiting Trondheim.

1.2 Goals

Most recommender systems base their recommendations on the previous feedback given by the target user. A central issue for recommender systems is hence the *cold-start user problem*, which is the problem of recommending items for new users of the system who have no such feedback histories. This problem is particularly important to solve in a recommender system for tourists visiting a city. In any city, most tourists only stay for a short period, and it is even more so in Trondheim where many of the tourists visiting are only staying for one day as part of the *Hurtigruten* boat trip along the coastline of Norway. Poor recommendations may result in valuable time being wasted, and therefore it is crucial for recommender systems for tourists, and perhaps in particular a recommender system for tourists visiting Trondheim, that even the first recommendation given is of high quality.

The analysis presented above lead us to the primary goal of this thesis, which is to:

G1 Find the existing solutions to the *cold-start user problem* presented in the recommender system literature.

We focus both on finding solutions *in general for any domain*, and specifically for recommender systems operating in the *tourist domain*.

We would like to use the solutions found through the work with our first goal as part of the design of a recommender system for tourists visiting Trondheim. As an additional constraint, which also limits the scope of our review of cold-start user problem solutions, we have chosen that the recommender system should use *Bayesian reasoning*. We sometimes refer to such systems as *Bayesian recommender systems* in the remainder of this thesis. The reason for using Bayesian reasoning is two-fold: Bayesian models are popular in current recommender system research; and they have been shown to fit well in recommender systems for the tourist domain. Hence, the second goal of our thesis is to:

G2 Design a Bayesian recommender system for tourists visiting Trondheim.

1.3 Contributions

To reach our first goal (**G1**) of identifying solutions to the cold-start user problem, we have chosen to perform a *systematic literature review*. Systematic literature reviews have their origin in medicine, and offer a way of reviewing literature which raises the probability of finding all relevant literature and helps reviewing the literature in an unbiased manner. However, such reviews require increased effort when compared to unsystematic reviews because of extensive planning and logging procedures. We consider using the systematic method, as part of an artificial intelligence Master's thesis, an additional contribution of this thesis.

On background of what has been presented previously in this chapter, we can summarise our contributions in the following way:

1. We present a review of the recommender system literature identifying solutions to the cold-start user problem:
   (a) in general for Bayesian recommender systems; and
   (b) specifically for Bayesian recommender systems operating in the *tourist domain*.
2. We propose a design of a Bayesian recommender system for tourists visiting Trondheim, which offers a solution to the cold-start user problem.
3. We give an example of how a systematic literature review can be performed in the field of artificial intelligence.
4. We give a discussion and an evaluation of performing systematic literature reviews:
   (a) of artificial intelligence literature; and
   (b) as part of Master's theses.

1.4 Structure of the Thesis

The remainder of this thesis is organised in five chapters.

In Chapter 2 we present theory and background material from topics of relevance to the thesis. In Chapter 3, we present our systematic literature review of recommender system literature, where the focus is on finding solutions to the cold-start user problem. In Chapter 4 we specify the design of a Bayesian recommender system for tourists visiting Trondheim. Chapter 5 includes a discussion of issues related to our work, both with the review and the design. In the last chapter of this thesis, Chapter 6, we conclude our work and present our thoughts on possible paths for further work.

Chapter 2

Theory and Background

In this chapter we will present and review subjects which form the background theory for our Master's thesis. The goal is that readers unfamiliar with the topics can learn what is necessary to understand the material presented in the later chapters of the thesis.

The first topic presented is recommender systems, both in general and specifically for tourists, in Section 2.1 and Section 2.2 respectively. We then move on to present Bayesian reasoning in Section 2.3 and Wireless Trondheim in Section 2.4. Finally we present and explain systematic literature reviews, focusing on how they can be used in computer science, in Section 2.5.

2.1 Recommender Systems

Recommender systems are systems that aim to present personalised recommendations of various items such as movies, websites or research papers to users. The first recommender system (according to Resnick and Varian [1997]), called Tapestry, was presented by Goldberg et al. [1992], and started off a research field which has been gaining much attention since, both in research communities, but also from commercial actors. Nowadays, people surfing the Internet will bump into recommender systems when visiting popular websites like amazon.com or YouTube. For a typical screenshot from a commercial website using a recommender system, see Figure 2.1.

As stated above, the common goal of recommender systems is to provide personalised recommendations of items for users. The most common approach to achieve this is to predict ratings on items that the target user has not seen yet, and use the predicted ratings to produce recommendations. Another approach, called *preference-based filtering* by Adomavicius and Tuzhilin [2005], is to not calculate rating predictions, but to predict a ranking of unseen items relative to how well they are predicted to be received by the target user (see e.g. [Jin et al., 2003]). We will here focus on the prediction of ratings-approach, as this has been the most popular approach so far according to Adomavicius and Tuzhilin [2005].

A common attribute of the vast majority of recommender systems is that they take feedback from users and use this as a source of information about the user

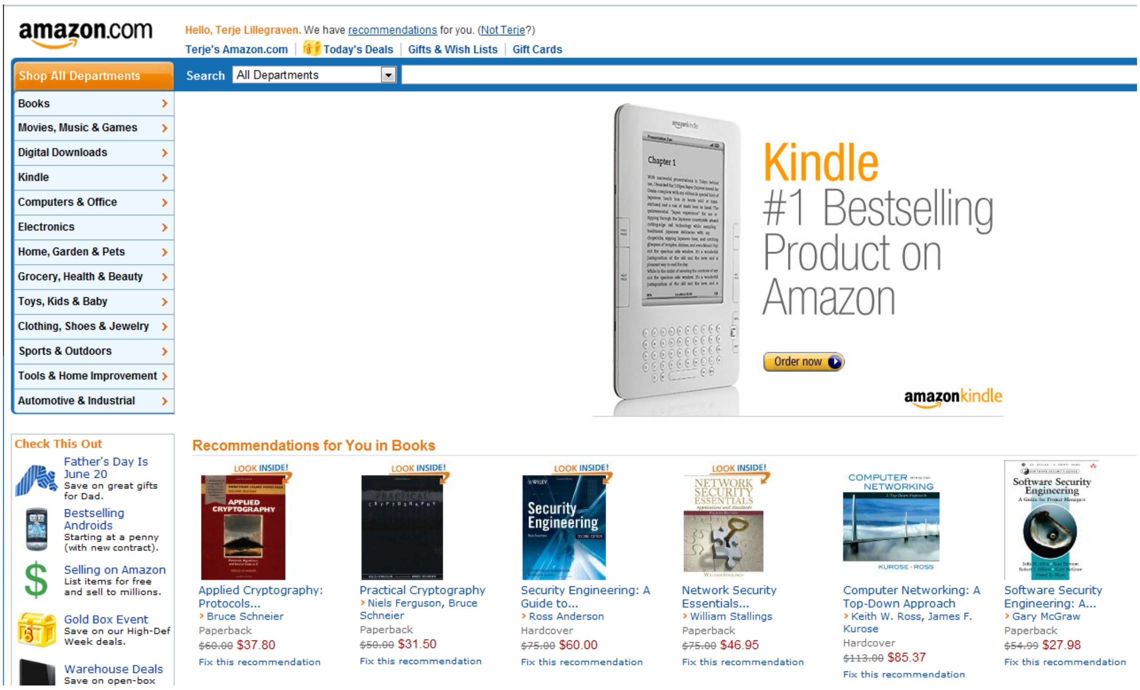

Figure 2.1: A screenshot from amazon.com recommending books for one of the authors.

to personalise recommendations. In many systems this feedback is given *explicitly*, in the form of ratings on items. Most systems use some sort of explicit feedback, but some systems also use *implicit* feedback. Implicit feedback is feedback given indirectly to the system by analysing users' behaviour. A typical example of implicit feedback is viewing time for articles, where the recommender system assumes that an article viewed for a long time is an article that the target user likes. If a user watches an article for a long time, the system registers a positive implicit feedback from the user on that particular article. Oard and Kim [1998] identify three types of implicit feedback that recommender systems can use to gain additional knowledge about users, namely *examination*, *retention* and *reference*.

Most recommender systems reason with two main sets of objects: a set of users, U, and a set of items, I. Each user u is represented by a user profile. The profile can consist of demographic attributes like age, gender and occupation; content-related weights reflecting the taste of the user; or it can simply be limited to the set of historical ratings that a user has given in the past. Similarly, an item i can simply be an object that has received a set of ratings, or it can have a more extensive profile containing facts and content information about the item.

The key to success for recommender systems predicting ratings for the target users, is to create a *utility* function which takes the target user u and an item i as input, and returns a predicted rating from u on i which is as close as possible to the actual rating that the user would give if he had actually experienced the item. This function can then be applied for all unseen items, and the one with the highest predicted rating, $\hat{i}$, can be recommended to the user. This can be mathematically formulated in the following way [Adomavicius and Tuzhilin, 2005]:

$$\hat{i} = \underset{i \in I}{\mathrm{argmax}}\, utility(u, i) \tag{2.1}$$

In the following sections we will first go through different recommendation techniques that are used to create such utility functions, before looking at the cold-start problems which are some of the key issues faced by today's recommender systems.

2.1.1 Types of Recommender Systems

There are several ways of creating personalised recommendations for users, given information and rating histories of users and items. Based on the different techniques used to achieve this we can separate approaches into different types of recommender systems. On the basis of differences in background data, input data and recommendation algorithms, Burke [2002] defines five different types of recommendation techniques, in addition to *hybrid* approaches which mix several techniques.

We will in the following sections thoroughly present the two most popular techniques, *collaborative* and *content-based* filtering, summarise the other techniques identified by Burke, and present ways of creating hybrid approaches. Finally we will take a look at *context-sensitive* recommender systems where other information than just users and items are taken into consideration when generating recommendations.

Collaborative Filtering

Collaborative filtering can be said to be the most successful recommendation technique among those seen until today [Herlocker et al., 2004]. It is based on the idea that recommendations of items for a target user can be generated based on ratings given by the *other users* of a system. These ratings can be used in user-user comparisons in *user-based* collaborative filtering, or in item-item comparisons in *item-based* collaborative filtering. Although item-based collaborative filtering has been gaining growing attention in recommender system literature (see e.g. [Linden et al., 2003], [Sarwar et al., 2001] and [Deshpande and Karypis, 2004]), and even hybrids of the two approaches have been proposed (see [Wang et al., 2006]), we will here focus on user-based collaborative filtering which has the longest traditions [Adomavicius and Tuzhilin, 2005].

User-based collaborative filtering systems go through two steps in order to predict ratings on items for a target user, which are then used to generate recommendations. First, they locate other users that are in some way found to be most similar to the target user, second they use ratings from the most similar users to predict ratings and generate recommendations for the target user.

In order to perform both steps, collaborative filtering systems need a data structure that relates users, items and ratings. This data structure is the User × Item matrix. In the User × Item matrix, each user is represented by a row, and each item is represented by a column. The values of the matrix are the ratings given by users on items. An example of a User × Item matrix is shown in Table 2.1.

	Shrek	Titanic	Matrix	Avatar
Roger	$\emptyset$	2	6	$\emptyset$
Cindy	9	8	7	$\emptyset$
Edgar	9	6	$\emptyset$	$\emptyset$
Helen	$\emptyset$	$\emptyset$	$\emptyset$	7

Table 2.1: An example of a $User \times Item$ matrix for a movie recommender system using ratings between 1 and 10. The rating value $\emptyset$ signifies *not rated*.

In addition to separating user-based and item-based collaborative approaches, we can also separate approaches into two groups based on how they use the User $\times$ Item matrix in their reasoning. The two approaches are called *memory-based* collaborative filtering and *model-based* collaborative filtering and are presented here in the context of user-based collaborative filtering. Both memory-based and model-based collaborative filtering can be applied also in item-based collaborative filtering, but as previously stated, we will here concentrate on the user-based approach.

In *memory-based* collaborative filtering the entire User $\times$ Item matrix is used directly to find similar users and generate recommendations. When the system needs to provide a user with a recommendation, it first uses some heuristic function to measure the similarity of rows (users) in the matrix (step 1 above).

Let $w(a,i)$ be the weight representing the similarity between user a and user i and let j run through the set of items of size m. Let $r_{a,j}$ and $r_{i,j}$ be the ratings given to item j by a and i respectively, and let $\bar{r}_a$ and $\bar{r}_i$ be the respective average ratings given by user a and i. A typical heuristic function used to measure user similarity is then the *Pearson correlation* function, first proposed for use in collaborative filtering by Resnick et al. [1994]:

$$w(a,i) = \frac{\sum_{j=1}^{m}(r_{a,j} - \bar{r}_a)(r_{i,j} - \bar{r}_i)}{\sqrt{\sum_{j=1}^{m}(r_{a,j} - \bar{r}_a)\sum_{j=1}^{m}(r_{i,j} - \bar{r}_i)}} \tag{2.2}$$

Another commonly used metric for measuring user similarity is the *cosine*-metric. This metric is explained in the next section in the context of content-based recommender systems.

The result of the first step of the collaborative filtering algorithm is a ranked set of users, where the ranking is based on similarity to the target user. It is common to pick the K most similar (the K top-most in the ranked list) users, and use this set of so-called *neighbours* to generate recommendations for the target user (step 2). For each item, a predicted rating from the active user is then calculated using the ratings given by the set of neighbours who have rated that particular item.

Let $p_{a,j}$ be the predicted vote for the target user a on item j; $\bar{r}_a$ be the average rating given by target user a; κ be a normalising factor to sum the weights w to unity; n be the number of neighbours in the target user a's neighbourhood; $w(a,i)$ be the measured similarity between user a and user i; $r_{i,j}$ be the rating given by user i on item j; and $\bar{r}_i$ be the average rating of user i. The predicted rating from user a

on item j is then usually calculated using some variation of Equation 2.3, presented by Breese et al. [1998]:

$$p_{a,j} = \bar{r}_a + \kappa \sum_{i=1}^{n} w(a,i)(r_{i,j} - \bar{r}_i) \tag{2.3}$$

The main strengths of memory-based collaborative filtering are the intuitive idea which it is based upon that makes it easy to understand, and its ability to create accurate predictions when sufficient data is available.

In commercial recommender systems (e.g. Amazon.com), the numbers of users and items may reach millions. A pure memory-based system collaborative filtering system needs to compare the target user to all other users in the system, before predicting ratings for all items unseen by the target user. To do this in real-time when the target user requests recommendations may be too computationally expensive, which leads to a scalability problem for memory-based collaborative filtering systems [Sarwar et al., 2000].

The scalability problem of memory-based collaborative filtering has been one of the motivations behind the other main collaborative filtering approach, called *model-based* collaborative filtering. What separates model-based from memory-based collaborative filtering is that instead of using heuristic functions directly on the User $\times$ Item matrix, model-based approaches use the matrix off-line to learn a model which can later be used to make rating predictions [Adomavicius and Tuzhilin, 2005]. The key goals for the model is to allow quick recommendation generation, while achieving as high precision in rating predictions as possible.

Several ways of creating the model of a model-based collaborative filtering approach exist. In the literature we find examples of probabilistic approaches (e.g. [Breese et al., 1998], and more recently [Shani et al., 2005] and [Hofmann, 2004]) and approaches based on machine learning (e.g. [Billsus and Pazzani, 1998]). Because the individual model-based approaches are based on a heterogeneous group of ideas, we will not go into details of any specific modelling technique here.

A well-known issue affecting the performance of collaborative filtering systems is the *sparsity* problem. The problem occurs when it is difficult to measure similarities between users, or items in item-based collaborative filtering, because of the lack of common rating patterns. It is often the case that even long-time users of a recommender system share ratings with only a small percentage of all users, leaving it impossible to compute similarities with the majority of the community. The result is that the recommendation precision suffers.

Content-Based Filtering

Content-based filtering is the second of the two main approaches to generation of recommendations in recommender systems. It has its background in information retrieval [Balabanović and Shoham, 1997], and is based on the idea that users will like items that are similar to items that they have previously expressed liking in.

While collaborative approaches recommend items to the target user if they are liked by similar users, content-based approaches attempt to build a content-based

Feature	Value
Name	McDonald's Trondheim Torg
Address	Kongens Gate 9, 7013 Trondheim
Cuisine	Fast food, burgers
Price level	Low
Clientèle	Kids, youth, families

Table 2.2: Example of a restaurant object tagged with name, address, type of cuisine, price level and clientèle.

user profile which is used to predict the user's opinions on unseen items. In order for predictions to be accurate, the user profiles must correctly represent the taste of the users. The user profile can be inferred implicitly by monitoring the feedback given by users on items, or it can be explicitly specified by the user.

A key for creating a well-working content-based approach is to tag items with content information. This is usually done using feature-value or keyword-frequency pairs. An example of a content-tagged restaurant is given in Table 2.2. In the example, *name*, *address*, *cuisine*, *price level* and *clientèle* are features, and *McDonald's Trondheim Torg*, *Kongens Gate 9, 7013 Trondheim*, *Fast food, burgers*, *Low* and *Kids, youth, families* the respective values for this particular restaurant object. Similarly to user profiles, items also have content profiles which consist of the same features or keywords as the user profiles.

A common way of defining both the user and item profiles are as vectors of weights [Adomavicius and Tuzhilin, 2005]. For a user u we have $\mathbf{w}_u = (w_{u1}, \ldots, w_{un})$, and for an item i we have $\mathbf{w}_i = (w_{i1}, \ldots, w_{in})$ where n is the number of features or keywords used in the system. Dependent on domain of the recommender system, the weights in $\mathbf{w}_u$ reflect users' preferences of features or keywords, and the weights in $\mathbf{w}_i$ reflect items' level of inclusion of the same features or keywords.

Measuring the utility of an item for a user in content-based filtering is often done using heuristic functions. A commonly used heuristic for this purpose is the *cosine similarity metric* which is also used in collaborative filtering for measuring user-user similarities as noted in Section 2.1.1[Salton, 1989]:

$$utility(u, i) = cos(\mathbf{w}_u, \mathbf{w}_i) = \frac{\mathbf{w}_u \cdot \mathbf{w}_i}{||\mathbf{w}_u||_2 \times ||\mathbf{w}_i||_2} \tag{2.4}$$

where u is a user, i is an item, $\vec{w_u}$ is the content weight-vector of user u and $\vec{w_i}$ is the content weight-vector of item i.

Other approaches build models based on underlying training data using machine learning techniques such as Bayesian classifiers, decision trees and artificial neural networks [Adomavicius and Tuzhilin, 2005].

Although content-based filtering can be used in many domains, it is most often used in domains where the values of features for objects can be automatically extracted, such as in recommender systems for text-based items (e.g. research papers or news) [Adomavicius and Tuzhilin, 2005]. For domains like the restaurant domain, the items need to be manually tagged with content information. Such tagging might be comprehensible in domains where the frequency of new items being added is relatively low, e.g. in the restaurant domain, but it soon becomes a problem in other domains where thousands of items may be added every day (e.g. multimedia on the internet). A main strength of pure collaborative filtering systems is that they do not require any tagging of content because recommendations are only based on the ratings data, making the approach suitable for any domain.

Another problem related to tagging of content in content-based filtering, as pointed out by Adomavicius and Tuzhilin [2005], is that two items tagged with the same features are inseparable for content-based recommender systems. This means that two items that have identical features, but are quality-wise very different, must be randomly ordered for recommendation by such systems.

A final issue specific to systems using content-based filtering is that such systems will rarely or never give so-called *serendipitous* recommendations to users [Shardanand and Maes, 1995]. The recommendations will always be based on what the target users have expressed liking in previously, and hence a romantic comedy movie is unlikely to ever be recommended for a user who has never rated anything else than action movies before, regardless of the quality of the movie.

Other Types of Recommendation Techniques

The other types of recommendation techniques identified by Burke [2002] are the *demographic*, *utility-based* and *knowledge-based* techniques. These are techniques that are rarely seen as stand-alone techniques in recommender systems, but are often combined with collaborative filtering or content-based systems in hybrid approaches.

Demographic approaches base their recommendations on personal attributes of the target users. On background of these attributes, users are grouped together in groups that receive similar recommendations. An example of a system using demographic filtering for marketing is presented by Krulwich [1997].

Utility-based approaches defines the user profile as a utility-function. Each user has its own utility function that is used to generate recommendations. The main issue of these systems is how to create the utility function. One example of how this can be done is the utility function in the system Tête-à-Tête presented by Guttmann [1998].

The final recommendation technique identified by Burke [2002] is the *knowledge-based* technique. This technique is based on encoding knowledge of how items meet user needs into data structures that recommender systems can use to infer recommendations. The knowledge can be encoded in different ways, for instance as rules in a rule-based system. An example of a recommender system automatically inferring rules from underlying data is proposed by Mobasher et al. [2001].

Hybrid Approaches

Hybrid approaches combine two or more recommendation techniques in order to overcome the weaknesses of the single methods while also exploiting their strengths.

As part of the study presented by Burke [2002], he presents a thorough survey of existing hybrid approaches and defines seven hybridization methods which are separated by the way they combine techniques:

Weighted
A final utility value of an item for a user is calculated by combining the weighted outputs from several independent recommender techniques.

Switching
Dependent on conditions, one of several recommendation techniques is selected to generate the recommendation.

Mixed
Recommendation techniques independently generate recommendations which are presented together.

Feature combination
Use collaborative information as additional features of items in a content-based approach.

Cascade
Let one recommendation technique filter out a candidate set of items which is refined by a second technique before recommending to user.

Feature augmentation
Use one technique to create additional features or ratings which can be used by second technique to create recommendations.

Meta-level
Let one technique build a model which is used as an input to the second technique which generates recommendations.

Although most combinations of techniques are possible, the most common techniques to combine in a hybrid recommender system are collaborative and content-based techniques. Adomavicius and Tuzhilin [2005] defines four ways that collaborative and content-based methods can be combined:

1. Implement both techniques separately and combine their outputs
2. Use content-based characteristics as input to a collaborative approach
3. Use collaborative characteristics as input to a content-based approach
4. Build a model that combines and unifies both techniques

Context-Aware Recommender Systems

Context-aware recommender systems utilise information about the situation the target user is in to generate more accurate recommendations.

Consider a guy who is usually into science fiction movies, but this time around need a recommendation for a movie which he can watch with his girlfriend who is less than enthusiastic about science fiction. Or similarly, consider a tourist who has spent the last five hours since breakfast walking around in museums, and is eager to find a place nearby to have lunch. Context-independent recommender systems, only considering the user and item dimensions as described previously in this section, will struggle in these situations to recommend suited items because they do not take into consideration the circumstances in which the user is in need of a recommendation. As a solution to this problem, which is identified as one to solve for the upcoming generation of recommender systems by Adomavicius and Tuzhilin [2005], *context-aware* recommender systems have been proposed.

How a recommender system should be context-aware dependens on the domain. In the movie domain it might be most important to incorporate a dimension that is who the target user is watching a movie with (cf. example above). In a tourist domain, it might be more important to take time of day, weather or previous visiting history into consideration. Finally, in a book domain, the recommender system might need to know whether the target user is in need of a book for his professional career, or a book to read on the beach on holiday. In general, however, context-aware recommender systems need to add one or more dimensions to the regular user and item dimensions.

As illustrated with the tourist example described above, context-awareness is something that is highly interesting to apply in recommender systems for the tourist domain. We will discuss this further, and look at some actual context-aware recommender systems for tourists in Section 2.2.

2.1.2 Cold-Start Problems for Recommender Systems

We have already discussed some issues related to specific recommendation techniques in the previous sections. Here, we first give an explanation of so-called *cold* and *warm* situations in recommender systems, before we present the *cold-start* problems that are commonly met in recommender system research.

In recommender system literature, the term *cold* is used about an object in a system, or a whole system, which is new. Being new has certain negative consequences, and to contrast new objects to objects which have been in the system long enough not to suffer these consequences, we use the term *warm* to describe objects or whole systems that are past the initial cold-start phase. As an example of this terminology, we can say that a movie which has received five ratings in a recommender system where the average movie has received 10000 ratings can be said to be *cold*, while another movie in the same system that has received 7000 ratings is a lot *warmer*.

What we refer to as a family of *cold-start* problems, are problems related to objects (users and items) of recommender systems, or the whole system itself, being new. We have chosen to adopt the terminology of Park et al. [2006b], and use the following names and definitions of the cold-start problems:

Cold-start user problem
The problem of giving accurate recommendations to a user who is new to the recommender system.

Cold-start item problem
The problem of recommending items that are new to the system and have not received ratings yet.

Cold-start system problem
The problem of having a new system with only new users and no ratings for items.

The cold-start *user* problem is present both in content-based and collaborative recommender systems. In content-based systems, the lack of ratings given by the target user leaves the system hopeless to generate a content-profile for the user because these need to be extracted using content-profiles of items liked by the user. In user-based collaborative filtering, similarities between users are calculated based on similar rating patterns. If no ratings are given by the target user, no such patterns can be found. Finding solutions to the cold-start user problem is the primary goal of our systematic literature review presented in Chapter 3.

The second of the cold-start problems is the cold-start *item* problem. This problem is most severe in pure collaborative filtering systems where items are recommended because of the ratings given by other users. Items that are new to the system, and thus have not received any ratings, will end up not being recommended to anyone if no additional measures are used to avoid this situation. In content-based recommender systems, this problem is not present as long as new items are tagged with content information when they enter the system.

The final cold-start problem is the cold-start *system* problem. The problem is mostly present in collaborative filtering systems, and includes both the cold-start user and item problems for these systems. The result is an extremely sparse matrix, which means that the problem is closely related to the sparsity problem which was discussed in Section 2.1.1.

As an additional note, it is worth observing the lack of consistency in terminology for the cold-start problems in recommender system literature. The cold-start *user* problem is also referred to as the *new user problem* [Adomavicius and Tuzhilin, 2005], *cold-start problem* [Massa and Bhattacharjee, 2004] and the *new user ramp-up problem* [Burke, 2002]. Similarly, the cold-start *item* problem is referred to as the *first-rater problem* [Melville et al., 2002], *cold-start problem* [Schein et al., 2002], and the *new item problem* [Adomavicius and Tuzhilin, 2005]. We will throughout the remaining chapters and sections of the thesis stick to the terms as we listed them earlier in this section.

2.2 Recommender Systems for Tourists

As previously discussed in this chapter, recommender systems have been used in domains like movies, books, research papers, music and many more. The last decade there has also been major development in recommender systems for the tourism domain [Kabassi, 2010].

In this section we will first present different services offered by recommender systems for tourists, before we take a look at existing recommender systems for tourists for the mobile platform.

2.2.1 Services

Recommender systems for tourists vary in what kind of services they recommend. The different services recommended are summarised in a survey presented by Kabassi [2010]:

- Accommodation
- Food (restaurants and cafés)
- Sights or points of interest
- Flights
- General information
- Map guidance
- Travel packages

Some of the existing recommender system approaches offer support only for single aspects of a trip like flight-booking [Coyle and Cunningham, 2003], restaurants [Burke, 2000] or attractions [Huang and Bian, 2009].

However, most systems offer two or more of these services. One example is *Cyberguide* [Abowd et al., 1997] which offers map guidance as well as general travel information. Two other examples are *Travel Planner* [Chin and Porage, 2001], which offers recommendations on flights, accommodation, food and sights, and *Traveller* [Schiaffino and Amandi, 2009], which offers the same services as *Travel Planner*, but also travel packages.

The systems also vary in the geographic area they recommend services for. Some, like *MastroCARonte* [Console et al., 2003], provide recommendations for wider geographic areas, while others only focus on single cities like Oldenburg (*Sightseeing4U* [Scherp and Boll, 2004]) or Lancaster (*GUIDE* [Cheverst et al., 2000]).

2.2.2 Mobile Recommender Systems for Tourists

Kabassi also points out that there is a line drawn between mobile and non-mobile recommender systems for travellers. Although a few systems have been proposed which offer services for both platforms (e.g. INTRIGUE [Ardissono et al., 2003]), most of the proposed systems are either designed for hand-held devices like smartphones, or website-based solutions used on stationary or laptop computers.

Although website-based systems can be of great help to users when they plan their trips, perhaps the most interesting path of development for recommender systems for tourists, is the development of mobile recommender systems for hand-held devices. In the tourism domain, this development leads to the possibility of assisting users with information at the time they need it the most: when they are actually walking around in unknown locations as tourists.

Of the systems offered thus far for hand-held devices, many are *location-aware* thanks to Global Positioning System (GPS) technology. Examples of such systems are the COMPASS system presented by Setten et al. [2004], and SPETA, presented by García-Crespo et al. [2009]. In recommender systems for tourists, location-awareness offers great opportunities like instant location and map guidance. In addition, the location information itself can be exploited for instance as input to recommender systems to recommend services that are conveniently located given the target user's current position.

2.3 Bayesian Reasoning

In this section we explain topics related to what we refer to as *Bayesian reasoning*, which is reasoning in domains with high levels of uncertainty and where partial beliefs about probabilities are updated as more evidence is observed.

We first present *Bayes' theorem*, also referred to as *Bayes' rule*, which lies at the core of Bayesian reasoning, before we move on to *Bayesian networks* which offer advantages when it comes to modelling in uncertain domains. In the last part of this section we look at different recommender systems presented in the literature which use Bayesian reasoning to generate recommendations for users.

2.3.1 Bayes' Theorem

The term *Bayesian* refers to the English mathematician Thomas Bayes and the theorem which was named after him, *Bayes' theorem*. The theorem explains the relation between a conditional probability and its inverse conditional probability.

If H is a hypothesis and E is the observed evidence for or against H, the theorem states that:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \tag{2.5}$$

Bayes' theorem is helpful in probabilistic reasoning as it provides us with a method to update our beliefs about a hypothesis when new information or evidence becomes available [Jensen and Nielsen, 2007].

2.3.2 Bayesian Networks

Bayesian networks are graphical models commonly used in probabilistic reasoning and artificial intelligence. They were first introduced in the middle of the nineteen eighties by Pearl [1985, 1988] and have become popular tools to model dependency relations between variables in uncertain domains.

A Bayesian network is defined by Jensen and Nielsen [2007] in the following way:

- It consists of a set of variables, represented as nodes, and a set of directed edges between variables.
- For each variable it is defined a finite set of mutually exclusive states.
- The network contains no cycles, hence it is a directed acyclic graph (commonly abbreviated DAG).
- Each variable X, with parents $(Y_1, \ldots, Y_n)$, has attached to it a conditional probability table $P(X|Y_1, \ldots, Y_n)$.

In a Bayesian network, we say that a node X that has a directed link to another node Y, is a *parent* of the latter node Y. This intuitively means that X has a direct influence on Y. In general, the edges of the network encode dependencies between variables. A key assertion, pointed our by Langseth and Nielsen [2009], which is made clear through the encoding, is that variables are conditionally independent of its non-descendants given its parents.

An example of a Bayesian network, presented by Russell and Norvig [2002], is shown in Figure 2.2. In the example network from a simple domain of neighbours, burglars and earthquakes, we have five random variables as nodes, with their respective conditional probability tables. We can see from the topology of the network and the conditional probability tables of each of the nodes, that *Burglary* and *Earthquake* have direct influence on the variable *Alarm*, and that *MaryCalls* and *JohnCalls* are in the next turn directly influenced by the *Alarm*.

Furthermore, we can observe that *MaryCalls* and *JohnCalls* are conditionally independent of all other variables in the network, given *Alarm*. Using abbreviations B, E, A, M and J for the variables respectively, this means that $P(M|B, E, A, J) = P(M|A)$ and similarly for J: $P(J|B, E, A, M) = P(J|A)$. This is because of the aforementioned assertion that variables are conditionally independent of their non-descendants given their parents.

One of the biggest advantages of Bayesian networks is their capability of representing joint probability distributions in a compact way, because of the *chain rule for Bayesian networks* [Jensen and Nielsen, 2007]. Let $\mathbf{X} = (X_1, \ldots, X_n)$ be represented by a Bayesian network and let the function $Parents(X_i)$ denote the set of variables in the network which have a directed edge to variable X_i. Then, according

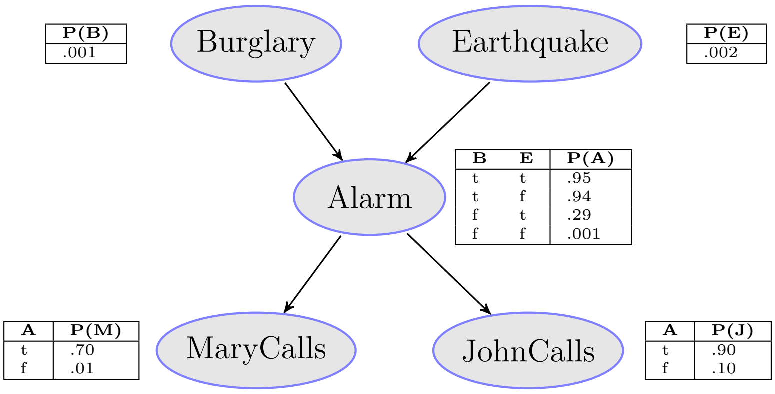

Figure 2.2: An example of a Bayesian network adopted from Russell and Norvig [2002]

to the chain rule for Bayesian networks, the Bayesian network specifies a unique joint probability distribution $P(\mathbf{X})$ in the following way:

$$P(\mathbf{X}) = \prod_{i=1}^{n} P(X_i|Parents(X_i)) \tag{2.6}$$

This compact presentation can be exploited in reasoning under uncertainty.

In addition to the advantage of compact representation of joint probability distributions, Heckerman [1998] points out four advantages of Bayesian networks in combination with statistical techniques when applied in data analysis:

1. Because they encode dependencies between all variables, Bayesian networks readily handle situations with missing data entries.

2. Bayesian networks can be used to learn and discover dependency relationships among variables in a domain, and can hence be used to gain understanding of the domain.

3. Bayesian networks include both causal and probabilistic semantics, and are ideal for combining prior knowledge and data in one graphical model.

4. In combination with statistical methods, Bayesian networks are efficient in avoiding overfitting of data.

2.3.3 Bayesian Reasoning in Recommender Systems

In model-based recommender systems, both collaborative and content-based, some form of Bayesian reasoning is often applied to generate recommendations. We here provide a short overview of how Bayesian reasoning has previously been used in recommender systems.

One example of using Bayesian networks in a model-based recommender system approach based on collaborative filtering is presented by Breese et al. [1998]. They

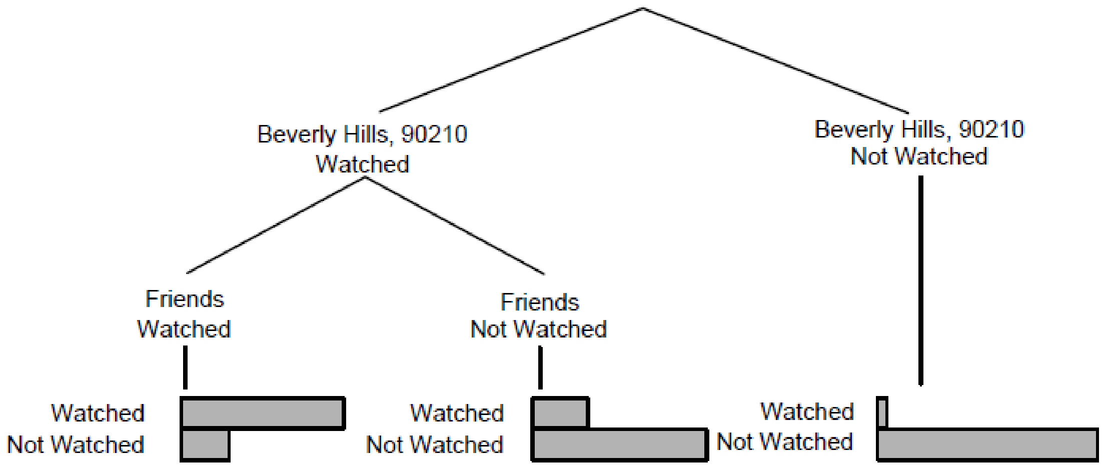

Figure 2.3: Decision tree from recommender system based on Bayesian networks presented by Breese et al. [1998].

use rating data to learn a Bayesian network where each item is represented by a node, and directed arrows between items signify how liking of items influence liking of other items. The network is used to create probabilistic decision trees for each item, where leaf nodes are likelihoods of the target user liking the target item, and intermediary decisions are based on the target user's view on the parent items of the target item from the network. Figure 2.3 shows an example of a decision tree presented by Breese et al. from a TV-series domain.

Other examples of recommender systems using Bayesian reasoning are a recommender system for context-aware music recommendation presented by Park et al. [2006a], and a probabilistic mixture model hybrid approach presented by Popescul et al. [2001] where a *three-way aspect model* realised as a Bayesian network is used to combine content and collaborative data.

Perhaps the most popular way of using Bayesian reasoning in recommender systems is using the naïve Bayes model. This model is a simplified Bayesian network where a class variable is the root node, and attribute variables constitute the leaf nodes. All the attribute variables are conditionally independent given the class variable. This is a naïve simplification in most domains, hence the name of the model. Despite the naïve nature of the model, it has been shown to be surprisingly accurate in many learning applications [Russell and Norvig, 2002].

An example of a recommender system based on a naïve Bayes model is the *Personality Diagnosis* algorithm presented by Pennock et al. [2000]. In this collaborative filtering system the naïve Bayes model is used to diagnose users into groups of similar users based on the opinions they have given on items.

Another example of using naïve Bayes model is presented by Condliff et al. [1999]. In this approach a separate naïve Bayes classifier is trained for each user to classify whether new items will be liked or not by that particular user. All the individual classifiers' parameters are finally combined using a regression model to achieve a collaborative prediction of ratings.

2.4 Wireless Trondheim

Wireless Trondheim, in Norwegian known as Trådløse Trondheim AS, is a company which was originally founded by NTNU, Adresseavisen, City of Trondheim, Sør-Trøndelag County, Sparebank 1 SMN and Trondheim Energi. Their main objectives, which are posted at their website at `http://tradlosetrondheim.no/`, are to:

- Build and deliver wireless internet coverage in Trondheim.
- Research and development of new wireless and mobile services.
- Together with NTNU, to facilitate and accommodate for research and development of new products and services by offering a platform for innovation called Wireless Trondheim Living Lab.

Today Wireless Trondheim's network covers most of the Trondheim city centre. It allows any user with any wireless-compatible device to connect to the network and surf the Internet.

2.5 Systematic Literature Reviews

A systematic literature review is a secondary study synthesizing all the information available from all the available primary studies relevant to a set of determined research questions. Both historically and today, systematic literature reviews are particularly important in medicine, but they are also used in social sciences and, to a lesser extent, computer science. Systematic literature reviews differentiate themselves from unsystematic reviews by using a strict methodological framework with a set of well-defined steps that are conducted in accordance with a predefined protocol. This makes it possible for other researchers to reproduce every step of the review process.

In the first of the following sections about systematic literature reviews, Section 2.5.1, we will present some of the advantages of performing such reviews. We then move on to discuss how systematic literature reviews have previously been used in computer science in Section 2.5.2. Finally, in Section 2.5.3, we present the different stages usually performed in a systematic literature review process.

2.5.1 Advantages of Systematic Literature Reviews

There are several reasons why undertaking a systematic literature review is considered useful both to the researchers performing them and the research community of the field which is target of the review. Here we present some of these reasons.

Most research starts out with a literature review in its initial phase to map out existing solutions. Using a systematic instead of an unsystematic method to perform this review raises chances of discovering work of other researchers which is interesting to build the current research upon. Furthermore, the thorough nature of the systematic literature review makes it a tool which help researchers avoiding bias in their work. As a result, reviews performed using the systematic method gain scientific value.

Another important aspect is that published systematic literature reviews summarise knowledge in the field which they are applied. This makes it possible for other researchers in the community to benefit from the work. As a result of this, the research community can avoid spending resources on duplication of work, and the progression of the field may gain pace. Finally, another beneficial feature of systematic literature reviews is that the extensive overview of the field gained by researchers performing such reviews, makes them able to identify gaps of knowledge and highlight needs for additional research in the field.

2.5.2 Systematic Literature Reviews in Computer Science

Systematic literature reviews in *computer science* are not very widespread, as opposed to the medical field. Some work has been done in the field of software engineering where systematic literature reviews have been presented, as well as a set of guidelines for performing systematic literature reviews, pioneered by Kitchenham [2007]. In the following we point out how performing systematic literature reviews in computer science differs from performing such reviews in medicine.

One major obstacle when performing systematic literature reviews of computer science literature is the lack of information infrastructure. For instance, the medical field has databases like the Cochrane Collaboration[1] that collects thousands of reviews that are up-to-date with all relevant literature. No such database of reviews exist in the field of computer science.

Another problem pointed out by Biolchini et al. [2005] is that it is difficult to do double-blind randomised controlled trials within computer science. For instance, in clinical medicine a professional can administer both placebo drugs and real drugs without being informed about its true effectiveness in order to achieve proper blinding. In software engineering, methods are being applied by professionals well aware of the workings of the methods they apply.

Biolchini et al. also point out that it is more difficult to isolate the effects of a specific method or a technique in computer science, since it often is a part of a life cycle, and interacts with many other technologies. In contrast, testing a drug in a sufficiently big population makes it easier to isolate the effects of the drug.

Similarly as Biolchini et al., Budgen et al. [2006] point out the problem with double-blinding in computer science experiments. Budgen et al. also point out that the effectiveness of a method or technique may be subject to the skill and experience of those utilising it, hence creating bias. Budgen et al. conclude that software engineering methodology is most similar with disciplines on the social end rather than the science end of the spectrum. Kitchenham [2007] takes this into

[1] http://www.cochrane.org

account when she presents a set of guidelines for systematic literature reviews in software engineering, which we describe further in Section 2.5.3.

To the best of our knowledge, no guidelines, such as the ones presented for software engineering by Kitchenham, exist for performing such reviews in the field of artificial intelligence. What we have discovered is that most systematic reviews presented until today that involve artificial intelligence are reviews where the main focus is on finding solutions to medical issues. An example of this is presented in Rajpara et al. [2009] where artificial intelligence techniques are used in diagnosis of melanoma.

2.5.3 Performing Systematic Literature Reviews

Kitchenham identifies three main phases of a systematic literature review process: *planning*, *conducting* and *reporting*. Each of the phases are parted into several steps. In the following parts we go through the phases, and explain the work done in each of the steps performed as part of the phases.

Planning the Review

The first phase of the systematic literature review process is a planning phase which consists of these five steps:

Identification of the need for a review
In this step a phenomenon is identified which requires the researchers to review literature in a thorough and unbiased manner.

Commissioning a review
In some cases one group might identify the need for a review, but lack the time or resources to conduct it. In these cases the organisation in need of the review creates a *commissioning document* which specifies the work that needs to be done.

Specifying the research question(s)
According to Kitchenham, specifying the *research questions* is the most important of all the steps performed as part of the review. The research questions defined in this step will form the basis for the search, data collection and data analysis processes which are performed in later steps of the review process.

Developing a review protocol
To reduce the possibilities of introducing researcher bias, all methods that will be used in later steps of the review process should be specified before commencing the conduction phase of the review. This specification is documented in the *review protocol*.

Evaluating a review protocol
To ensure that methods specified in the protocol are in accordance with the

goals of the review, the protocol should be evaluated by a group of independent experts. Kitchenham recommends that PhD students should present their protocol for the supervisors evaluation and criticism.

Conducting the Review

When the planning of the review is completed and the protocol has been evaluated, the review process goes into the phase which Kitchenham refers to as *conducting the review*. The phase consists of five steps:

Identification of research
The goal of this step is to locate as many primary papers relevant to answering the research questions as possible. To achieve this in an unbiased manner, the search must be performed using a predefined search strategy. In addition, documentation of each step of the search process is needed. In the field of computer science, unlike medical research, no pre-packaged search strategies exist, so these need to be developed.

Selection of primary studies
The goal of this step is to filter out primary studies irrelevant or of low interest to the review from the set of primary studies which were found in the search step. This process is usually performed in several steps, where it is common to start filtering based on abstracts of studies or even titles of studies.

Study quality assessment
When the relevant primary papers are identified, a quality assessment of the remaining studies is conducted. The goal of this step can be to further filter down the set of studies, or to assess the quality of studies included in the review. This assessment will then be utilised in the synthesis and analysis step performed later in the review process.

Data extraction and monitoring
In this step data is collected from each of the studies which are included in the review. This is done using the *data collection form* which was designed during the development of the review protocol. The goal of the step is to collect all data from the studies that is necessary to later answer the research questions.

Data synthesis
The last step of the conducting phase is the synthesising of data in order to answer the research questions. The synthesis is often descriptive, but if the data collected from studies is homogeneous, it might be possible to perform quantitative analyses of the data as well.

Reporting the Review

When the review has been conducted, the review process goes into its final phase, which Kitchenham refers to as *reporting the review*. The phase consists of these three steps:

Specifying dissemination strategy

In this stage a strategy on how the results of the work should be communicated is defined. This strategy might involve journals, press releases, direct communication with stakeholder and other methods to spread the knowledge.

Formatting the main report

In this stage the systematic literature review is reported in a technical report or in the section of a thesis. Most reviews will also be reported in a research paper which is made available for the research community through a conference or a journal.

Evaluating the report

In this final stage of the review process the reported work should be evaluated by experts in the field. If the work is reported in a paper it should be submitted to a journal or a conference where it will be peer-reviewed. For PhD or Master's students it is mandatory that the thesis is reviewed by an independent expert.

Chapter 3

Cold-Start User Problem: A Systematic Literature Review

In Section 2.5, we presented *systematic literature reviews* which are reviews, originally introduced and applied in medicine, where every step of the review process is planned and performed in a systematic manner. In this chapter we present a systematic literature review exploring solutions to the cold-start user problem in recommender systems using Bayesian reasoning.

The chapter is organised as follows: in Section 3.1 we give an introduction of the motivation and goals of our systematic literature review, and present our research questions which form the basis of the review; in Section 3.2 we explain our review method; in Section 3.3 we present the results of our review; in Section 3.4 we analyse our results and discuss answers to our research questions; and in Section 3.5, we identify three additional issues for recommender system research which fall outside the scope of this review. In the last section of this chapter, Section 3.6, we summarise the key findings and observations of the systematic literature review.

3.1 Introduction

When initially analysing potential issues for a recommender system using Bayesian reasoning for tourists visiting Trondheim, we identified a challenge which was of high relevance, namely achieving personalisation in recommender systems where almost all users are cold-start users. This challenge is particularly important to solve in a recommender system for tourists because tourists are not likely to stay in a particular city very long, and bad recommendations may lead them to waste valuable time. As discussed in Section 2.1.2, traditional techniques for recommendation generation in recommender systems like collaborative filtering and content-based filtering share the problem of giving new users personalised recommendations of high quality. Hence, we needed to identify other solutions to the cold-start user problem.

To the best of our knowledge, no systematic review has been performed on how to handle the cold-start user problem in recommender systems. In order to find a reliable solution to the cold-start user problem which we could apply in a recommender system for tourists visiting Trondheim, and to make a significant contribution to the recommender system community, we decided to perform a systematic literature review of recommender system literature to identify existing solutions to the cold-start user problem.

As the first step of our systematic literature review process, we formalised the goals of our review into the following set of research questions:

RQ1 What are the existing solutions to handle the cold-start user problem in recommender systems using Bayesian reasoning?

RQ2 How do the different solutions, found through the work with RQ1, compare to each other with regards to initial user involvement, learning ability and initial service quality?

RQ3 What is the strength of the evidence in support of the different solutions?

RQ4 What implications will these findings have when creating a recommender system for tourists?

3.2 Review Method

In this section we will in detail explain the method used to perform our systematic literature review. To develop our method, we have taken the guidelines for performing systematic literature reviews in software engineering presented by Kitchenham [2007], which were presented in Section 2.5.3, and modified them to fit the artificial intelligence domain and in particular the goals and objectives of our review.

In addition to the guidelines, we have also used the systematic review performed by Dybå and Dingsøyr [2008] as an inspiration, as well as a source of examples of how to perform the different stages of a rigorous review process.

Our systematic literature review process was performed in eight steps:

ST1 Defining research questions

ST2 Defining the systematic literature review protocol

ST3 Search for relevant studies

ST4 Selection of studies

ST5 Quality assessment

ST6 Data collection

ST7 Data synthesis and analysis

ST8 Dissemination

The eight steps will be thoroughly presented and explained in the next parts of this section. For each step we will go through the goal or the purpose of performing the step, and the procedure used to perform the step. The results of ST1-ST4 and ST8 will also be presented in their respective sections, while the results of ST5-ST7 are presented in the *results* section of this chapter, Section 3.3.

3.2.1 ST1: Defining Research Questions

The goal of this process was to form a set of questions that were neither too specific nor too vague, but clearly stated the problems we wished to target in our review. We created a set of four research questions which form the basis of our systematic literature review:

RQ1 What are the existing solutions to handle the cold-start user problem in recommender systems using Bayesian reasoning?

RQ2 How do the different solutions, found through the work with RQ1, compare to each other with regards to initial user involvement, learning ability and initial service quality?

RQ3 What is the strength of the evidence in support of the different solutions?

RQ4 What implications will these findings have when creating a recommender system for tourists?

3.2.2 ST2: Defining the Systematic Literature Review Protocol

The goal of defining the systematic literature review protocol was to specify how to perform each stage of the review process before commencing the work, so that we were unlikely to have any bias that could give a negative influence on the objectivity of the process.

We created an initial protocol that was iteratively refined during the systematic literature review process. For each iteration we reviewed the upcoming stage in the protocol to assure that the procedure we originally specified was still appropriate given the results and the experience gained from the previous stage. By continuously specifying the steps *before* beginning to work with them, the objectivity of the systematic literature review process was preserved.

The final version of the systematic literature review protocol is presented in Appendix A.

3.2.3 ST3: Search for Relevant Studies

In this stage of the review process the goal was to retrieve all the literature relevant to answer our research questions. To to do so, we created a search strategy that specified which sources that were to be searched and how we would perform the search. The search strategy and the results of the search are presented in the following sections.

Search Strategy

Forming the search strategy was a two-step procedure. The first step was to gather a list including all sources likely to provide relevant studies for our review, the second step was to decide how to search the sources.

To create our list of sources we first focused on locating all the relevant on-line digital libraries using lists presented in studies by Kitchenham [2007] and Dybå and Dingsøyr [2008]. In addition to the digital libraries, we also added a set of relevant journals and conference proceedings found on-line that needed to be manually searched. These sources were likely to present relevant studies for our review that would possibly not be found through the digital libraries.

Finally, before defining the search procedures we went through the list and confirmed that all sources were still active and available. The final list of sources and the researcher that was responsible for searching each source is shown in Table 3.1.

Source	Type	URL	Responsible
ACM Digital Library	Digital library	http://portal.acm.org/dl.cfm	Lillegraven
IEEE Xplore	Digital library	http://ieeexplore.ieee.org/Xplore/guesthome.jsp	Lillegraven
ISI Web of Knowledge	Digital library	http://isiknowledge.com	Lillegraven
ScienceDirect	Digital library	http://sciencedirect.com/	Lillegraven
CiteSeerX Beta	Digital library	http://citeseerx.ist.psu.edu/	Lillegraven
SpringerLink	Digital library	http://springerlink.com	Wolden
Wiley Inter Science	Digital library	http://interscience.wiley.com	Wolden
Journal of Artificial Intelligence Research	Journal	http://jair.org	Both
International Joint Conference on Artificial Intelligence	Conference proceedings	http://ijcai.org/	Both
International Conference on Machine Learning	Conference proceedings	http://cs.mcgill.ca/~icml2009/past.html	Wolden
Conference on Uncertainty in Artificial Intelligence	Conference proceedings	http://uai.sis.pitt.edu	Wolden

Table 3.1: Sources used in the search stage of the review and the researcher responsible for performing the search.

	Group 1	Group 2	Group 3	Group 4
Term 1	Cold-start	Recommender systems	User models	Bayes
Term 2	Sparsity	Collaborative filtering	User modeling	Bayesian
Term 3		Content-based filtering	User modelling	
Term 4			User information	
Term 5			User profile	

Table 3.2: The four groups of search terms which we combine in our digital library searches.

After the list of sources was complete, we moved on to defining search terms for the digital libraries, and a procedure for the manual search of the journals and conference proceedings.

The digital libraries provide search engines where users can enter key words to search for relevant studies. Most of the engines also have an *advanced search*-option that allow users to enter boolean search strings. We exploited this feature to form a search string that covered all the different searches we wished to perform for each engine in a single search string.

To create our search string we first formed four groups of key terms as shown in Table 3.2. Each group consists of terms that are either synonyms, different forms of the same word, or terms that have similar or related semantic meanings within the recommender system literature. The four groups are directly related to our first research question (RS1, see Section 3.2.1), each retrieving different sets of research studies: Group 1 finds all studies focusing on the cold-start user problem; Group 2 finds all literature related to recommender systems; Group 3 finds all studies focusing on user modelling; while Group 4 finds all studies presenting systems using Bayesian reasoning. The studies we wished to find through our search were the ones in the *intersection* of the four sets as illustrated with the Venn diagram in Figure 3.1.

To implement the strategy described above in a boolean search string, we used the logical OR-operator within the groups to allow studies to include any of the terms included in each group. The groups themselves were combined with the AND-operator. The effect of the search string was that all studies that included at least one of the terms in each group was found when we performed the search. The structure of the search string, that in practice allowed us to use one search string to perform 60 different searches, is shown below:

$$([G1, T1]\ OR\ [G1, T2])\ AND\ ([G2, T1]\ OR\ [G2, T2]\ OR\ [G2, T3])$$
$$AND\ ([G3, T1]\ OR\ [G3, T2]\ OR\ [G3, T3]\ OR\ [G3, T4]\ OR\ [G3, T5])$$
$$AND\ ([G4, T1]\ OR\ [G4, T2])$$

The manual searches were done by browsing the titles of studies and reading the abstracts when possibly relevant articles were found. Based on reading the abstract, the responsible researcher evaluated if the paper was likely to contribute relevant

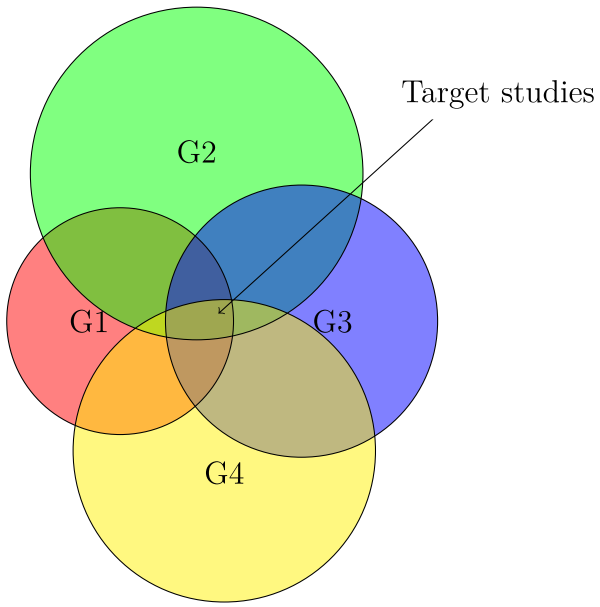

Figure 3.1: Venn diagram showing how the search term groups in Table 3.2 was combined to find studies relevant to our review (the *target studies*).

information for solving our research questions presented in Section 3.2.1. If the study seemed promising in that respect, it was included in further stages.

Additional *New User Problem* Search

While working with the study selection stage of the review process (described in Section 3.2.4) we discovered that the term *new user problem* was a commonly used synonym for what we refer to as the *cold-start user problem*. This discovery lead us to the conclusion that *new user problem* should have been added as an additional term in search term group 1 (see Table 3.2) as a synonym to the cold-start user problem.

To include studies not found in our initial search because of not including the *new user problem* term, we did an additional search through the seven digital libraries in Table 3.1. In this search we combined *new user problem* with the terms from group 2, 3 and 4, similarly to what was explained in Section 3.2.3.

The results of the 'new user problem search' are shown in Table 3.3.

Search Results

The results of our search stage are summarised in Table 3.4.

From the set of 375 studies we removed all duplicate studies and the studies published before 1st of January 2000. In addition we removed studies that presented the same study, but were published in different sources. In these cases we kept the study that was published by the most strictly reviewing publisher (e.g. prefer journal

Source	Number of studies
ACM Digital Library	3
IEEE Xplore	0
ISI Web of Science	1
ScienceDirect - Elsevier	9
CiteSeerX Beta	5
SpringerLink	3
Wiley Inter Science Journal Finder	0
Total number of studies found in search	21

Table 3.3: Studies found from each source in the *new user problem* search.

Source	Number of studies
ACM Digital Library	114
IEEE Xplore	57
ISI Web of Science	30
ScienceDirect - Elsevier	29
CiteSeerX Beta	40
International Joint Conference on Artificial Intelligence	10
Journal of Artificial Intelligence Research	3
SpringerLink	57
Wiley Inter Science Journal Finder	0
International Conference on Machine Learning	7
Conference on Uncertainty in Artificial Intelligence	7
Additional *new user problem* search	21
Total number of studies found in search	**375**

Table 3.4: Studies found from each source, total number of studies found in the *new user problem* search, and the total number of studies found from the search stage of the systematic literature review process.

Criterion identifier	Criterion
IC1	The study's main concern is recommender systems
IC2	The study is a primary study presenting empirical results
IC3	The study focuses on solving the cold-start user problem
IC4	The study focuses on user modelling
IC5	The proposed recommender system uses Bayesian reasoning
QC1	Is there a clear statement of the aims of the research?
QC2	Is the study put into context of other studies and research?

Table 3.5: The inclusion and quality criteria used in the study selection process.

over conference proceedings studies) or the one that was published most recently if published by the same, or similar, publishers.

The final number of studies that was passed on to the study selection process was 320.

3.2.4 ST4: Selection of Studies

The goal of our study selection process was to filter down the set of studies found in the search stage to a set of studies that were thematically relevant for answering our research questions and presented research of high quality. To perform the filtering we defined a set of inclusion criteria and quality screening criteria which are shown in Table 3.5.

The criteria were applied in a three-stage process:

1. Abstract inclusion criteria screening
2. Full-text inclusion criteria screening
3. Full-text quality screening

The process is illustrated in Figure 3.2 where the set containing IC1 and IC2 is referred to as the *Primary Inclusion Criteria*, the set containing IC3, IC4 and IC5 is referred to as the *Secondary Inclusion Criteria* and the set containing QC1 and QC2 is referred to as the *Quality Screening Criteria*.

For each of the stages we created EndNote libraries with the studies that entered the stage, and Google docs spreadsheets where we logged our assessments of the studies. For the first two stages we also reported the disagreement rate for each of the calibration sets (explained below) c_n using the following equation:

$$\text{Disagreement rate}(c_n) = \frac{|disagreeing(c_n)|}{|c_n|} \tag{3.1}$$

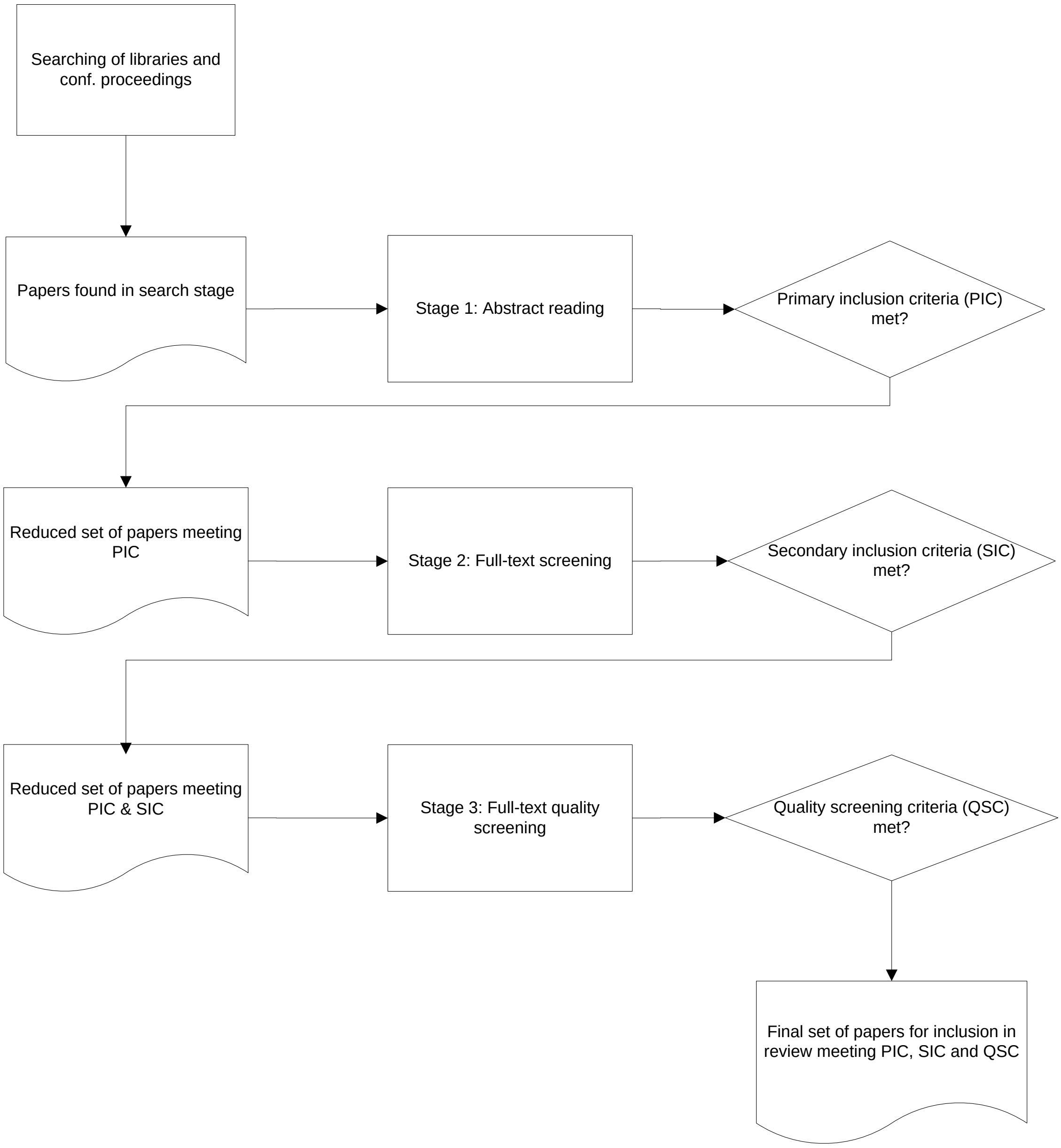

Figure 3.2: The three-stage study selection process.

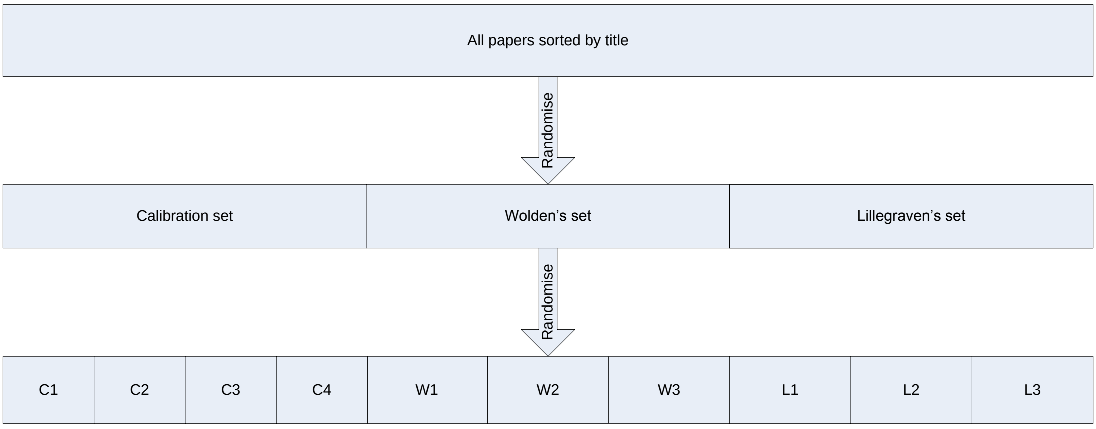

Figure 3.3: Splitting of sets of studies into individual and common sets.

where $disagreeing(c_n)$ returns the set of studies in c_n where the researchers disagreed in whether to accept the paper for further stages.

Detailed descriptions of, and results from, each stage of the selection process are explained in the following sections.

Abstract Inclusion Criteria Screening

In this stage we filtered studies based on reading only the abstracts of the 320 studies that was found in the search phase. Studies were accepted for further stages if the abstract indicated that the first two inclusion criteria (from Table 3.5) were met:

IC1 The study's main concern is recommender systems

IC2 The study is a primary study presenting empirical results

Because of the low detail level of some abstracts, we decided that if we were in doubt whether the study should be accepted or rejected after reading the abstract, we passed the study on to the next stage.

As a result of the limited time, both researchers did not review all abstract. Instead, we first sorted the studies alphabetically by title, then parted the set of studies into three parts: two parts that were divided between the researchers to review independently and one *calibration* set that both researchers reviewed. The point of the calibration set was to align the researchers view of how to evaluate the studies.

Each of the individual sets were once again parted into three, while the calibration set was parted into four. The effect was that we could calibrate, recalibrate, and at the end, control our assessments during the screening process. The splitting procedure and the *assessing and calibration* process are illustrated in Figure 3.3 and Figure 3.4 respectively.

During the calibration and recalibration stages we discussed all studies where assessments were disagreeing until an agreement was reached. The number of initial disagreements was gradually decreasing for each iteration. The first set had a high

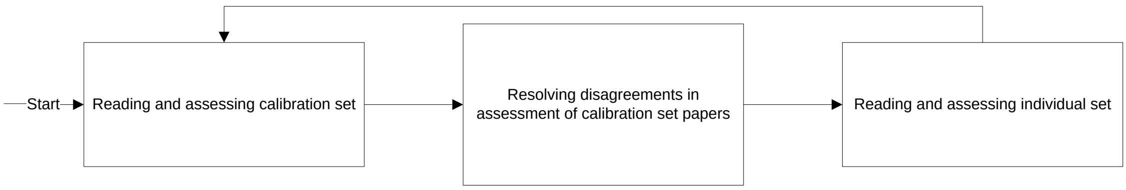

Figure 3.4: *Assessing and calibration* process.

disagreement rate of 42 percent, the second and the third had rates of 27 and 20 percent respectively, while the final set's disagreement rate was only 13 percent.

The final result of the abstract filtering was that 107 studies were rejected, leaving 213 studies that were passed on to the first full-text screening.

Full-Text Inclusion Criteria Screening

In this stage we filtered our studies that failed to meet the following inclusion criteria (from Table 3.5):

IC3 The study focuses on solving the cold-start user problem

IC4 The study focuses on user modelling

IC5 The proposed recommender system uses Bayesian reasoning

Because of lacking detail level in the studies' abstracts, we needed to retrieve full-text exemplars to evaluate whether the criteria were met.

This stage of the selection process was performed in a similar manner to what was presented when explaining the abstract filtering stage. We used the same strategy with partitioning the studies into one calibration set, and two individual sets in order to speed up the process while preserving objective and aligned assessments (see Figure 3.3). The only difference in this stage was that we did not use a fourth calibration set (the control set in the previous stage) so that the calibration set was only parted into three equal-size subsets. We once again performed the process in an iterative manner as illustrated in Figure 3.4.

Similarly to the previous stage, all disagreements were resolved through discussion. In this stage the average initial agreement between the researchers in assessments were higher than in the abstract filtering stage. The disagreement rates for the three calibration sets were 13.6, 13.6 and 4.8 percent for calibration sets c_1, c_2 and c_3 respectively.

The result of this stage was that another 186 studies were rejected, leaving a set of 27 studies that entered the final stage of the study selection process.

Full-Text Quality Screening

Our final stage of the study selection was a quality screening where we filtered out studies that did not meet the following quality criteria:

QC1 Is there a clear statement of the aims of the research?

QC2 Is the study put into context of other studies and research?

In this stage all of the remaining studies were assessed by both researchers independently by answering yes/partly/no to whether each of the criteria were met. After both had finished assessing the studies, we resolved disagreements for each paper and each criterion. Finally, we calculated a sum for each study by giving 1 point for each 'yes', 0.5 points for each 'partly' and 0 points for each 'no'. All studies that scored $Score_{QC1} + Score_{QC2} \geq 1.5$ points were accepted and included in the set of studies used in our review.

Three studies failed to meet our quality screening criteria, leaving a set of 24 studies that were included in the set forming the literature basis for our review.

Before commencing the quality assessment and data collection stages, we gave each of the 24 studies a unique study identifier (S1-S24) after sorting them primarily on year of publication and secondarily on first author's last name. The 24 studies are presented with their identifiers in Table 3.7 which can be found in Section 3.3.1.

3.2.5 ST5: Quality Assessment

To evaluate the strength of the evidence presented by studies included in the review (cf. RQ3 in Section 3.2.1), we assessed the quality of each study using the following 10 criteria:

QC1 Is there a clear statement of the aims of the research?

QC2 Is the study put into context of other studies and research?

QC3 Are system/algorithm design decisions justified?

QC4 Is the test data set reproducible?

QC5 Is the study algorithm reproducible?

QC6 Is the experimental procedure thoroughly explained and reproducible?

QC7 Is it clearly stated in the study which other algorithms the study algorithm(s) have been compared with?

QC8 Are the performance metrics used in the study explained and justified?

QC9 Are the test results thoroughly analysed?

QC10 Does the test evidence support the findings presented?

As the first two criteria had already been assessed for all the studies in the quality screening stage, we only needed to assess the final eight criteria during the full quality assessment stage. Similarly to the quality screening stage, all studies were assessed independently by both researchers, by answering each criteria for each study with yes (1 point), partly (0.5 points) or no (0 points). The assessments were logged in two Google Docs spreadsheets, one per researcher. Finally, disagreements were resolved through discussion until settlement for one grade was found and the final assessments were logged in another spreadsheet.

The final results of the quality assessment are given later in Section 3.3.

Criteria Rationale

The criteria listed above were carefully designed to measure the most relevant aspects when assessing quality of recommender system literature. Here we present a brief rationale for each of the criteria.

The first two criteria (**QC1** and **QC2**) were used as part of study selection process, and are related to good research practice. It is considered good research practice to state the aims of the research, and to put the presented research into context of other research. Only when both criteria are met can readers understand on which assumptions the research is based, and how the research pulls the the field further.

The third criterion (QC3) controls whether the presented system or algorithm is based on ideas that are well analysed. To analyse an approach before performing experiments using it, is also a part of good research practice and is a tool to avoid bias as results and pre-analysis should be coherent.

Criteria QC4, QC5, QC6 and QC8 are related to the reproducibility of data sets, presented algorithms, experimental procedures and performance metrics used in the studies. It should be possible for other researchers to trace and reproduce every step of the work done by studies' authors in order to confirm the validity of the results or to compare their own approaches head-to-head against existing ones.

Criterion QC7 determines whether the studies report which state-of-the-art approaches they compare their own approach to. For readers to be able evaluate the results presented and the effectiveness of the proposed approach, it is important to clearly state which approaches it has been put up against.

Finally, criteria QC9 and QC10 are used in the quality assessment to assess whether all aspects of results are thoroughly analysed, and that the findings are in fact supported by the evidence presented. For readers to be left convinced that the proposed approach indeed has the properties that it claims to have, it is important that aspects of the results that are unexpected are analysed, and that there is a clear connection between the evidence presented and the conclusions of the study.

3.2.6 ST6: Data Collection

The goal of the data collection process was to gather the data necessary to answer research questions RQ1 and RQ2. From each study we collected the data presented in Table 3.6.

Similarly to previous stages, we created a data collection spreadsheet where each data was represented by a column, and each study was represented by a row. After

Data ID	Data
D1	Name of authors
D2	Title
D3	Study identifier
D4	Year of publication
D5	Type of article (e.g. journal, conference proceedings)
D6	Aims, objectives and contributions of study
D7	Term used for cold-start user problem in study
D8	Name of system
D9	Type of recommender system
D10	Cold-start user problem solution
D11	User modelling technique
D12	Experimental design
D13	Test system domain
D14	Test set source
D15	Performance metric used
D16	Findings and conclusions
D17	Reported initial user involvement
D18	Reported learning ability
D19	Reported initial quality of service

Table 3.6: Data collected from studies included in review.

each study was read by both researchers, we had a short meeting where we filled the row representing the study with the extracted data.

The results from the data extraction stage are summarised in Section 3.3.2.

3.2.7 ST7: Data Synthesis and Analysis

We defined the following strategy for how to synthesise and analyse the results of the quality assessment and the data collection stages:

1. Sort the studies included in the review into solution types according to how they solve the cold-start user problem.

2. Answer our research questions presented in Section 3.2.1 with the solution types as a starting point, by consulting the data collection and quality assessment tables.

3. Review the data collection and quality assessment tables to evaluate any additional findings besides the ones directly related to the research questions.

All three stages were performed in meetings involving both researchers where the results from the previous stages formed the basis of the discussions. The discussions led to findings which were logged during the meetings. These findings are presented and discussed in Section 3.4.

3.2.8 ST8: Dissemination

The primary reporting of our review is this chapter of our Master's thesis. In addition to the thesis we intend to submit a research paper to a journal or a conference in order to publish the study for the artificial intelligence, and especially the recommender system, research community.

3.3 Results

This section presents the results of our systematic literature review of recommender system literature focusing on the *cold-start user problem*. We first present an overview of the studies included in the review, in Section 3.3.1, before summarising key points from each of the studies, sorted by solution types, in Section 3.3.2. Finally, in Section 3.3.3, we present our quality assessment of the reviewed articles.

3.3.1 Studies Included in Review

As reported in Section 3.2.4 the result of the study selection process was a set of 24 research studies which were included in our review. The studies are presented in Table 3.7 with their unique identifier, name of authors, the title of the paper and citation. The identifiers will be used for reference in the remainder of this chapter.

Study ID	Authors	Title	Citation
S1	Ko, S. and Lee, J.	User preference mining through collaborative filtering and content based filtering in recommender system	[Ko and Lee, 2002]
S2	Rashid et. al	Getting to know you: learning new user preferences in recommender systems	[Rashid et al., 2002]
S3	Manavoglu et. al	Probabilistic user behavior models	[Manavoglu et al., 2003]
S4	Massa, P. and Bhattacharjee, B.	Using trust in recommender systems: An experimental analysis	[Massa and Bhattacharjee, 2004]
S5	Middleton et. al	Ontological user profiling in recommender systems	[Middleton et al., 2004]
S6	Aimeur, E. and Onana, F. S. M.	Better control on recommender systems	[Aimeur and Mani Onana, 2006]
S7	Park et. al	Naïve filterbots for robust cold-start recommendations	[Park et al., 2006b]
S8	Zigoris, P. and Zhang, Y.	Bayesian adaptive user profiling with explicit & implicit feedback	[Zigoris and Zhang, 2006]
S9	Gao et. al	Personalized Service System Based on Hybrid Filtering for Digital Library	[Gao et al., 2007a]
S10	Lekakos, G. and Giaglis, G.	Probabilistic user behavior models	[Lekakos and Giaglis, 2007]
S11	Pronk et. al	Incorporating user control into recommender systems based on naive bayesian classification	[Pronk et al., 2007]
S12	Ying et. al	A Recommendation Algorithm Combining User Grade-Based Collaborative Filtering and Probabilistic Relational Models	[Gao et al., 2007b]
S13	Ji, L. and Guishi, D.	A New-User Cold-Starting Recommendation Algorithm Based on Normalization of Preference	[Liu and Deng, 2008]
S14	Lam et. al	Addressing cold-start problem in recommendation systems	[Lam et al., 2008]
S15	Victor et. al	Key figure impact in trust-enhanced recommender systems	[Victor et al., 2008]
S16	Agarwal, D. and Chen, B.-C.	Regression-based latent factor models	[Agarwal and Chen, 2009]
S17	Bohnert et. al	Spatial Processes for Recommender Systems	[Bohnert et al., 2009]
S18	Chu, W. and Park, S.-T.	Personalized recommendation on dynamic content using predictive bilinear models	[Chu and Park, 2009]
S19	Maneeroj, S. and Takasu, A.	Hybrid Recommender System Using Latent Features	[Maneeroj and Takasu, 2009]
S20	Park S.-T. and Chu, W.	Pairwise preference regression for cold-start recommendation	[Park and Chu, 2009]
S21	Xue et. al	User language model for collaborative personalized search	[Xue et al., 2009]
S22	Hölbling et. al	PersonalTV	[Hölbling et al., 2010]
S23	Kim et. al	Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation	[Kim et al., 2010]
S24	Wang, H.-F. Wu, C.-T.	A strategy-oriented operation module for recommender systems in E-commerce	[Wang and Wu, 2009]

Table 3.7: The 24 studies selected through the study selection process, forming the literature basis for our systematic literature review.

Solution Type	Papers
Demographic User Data	S9, S10, S12, S14, S16, S18, S20, S24
Ask to Rate	S2, S6
Trusted Users	S4, S15
Specification of Feature Preferences	S11, S22
External User History	S5
Tagging	S23
Geographic Position Data	S17
Fast Learners	S1, S7, S13, S19
Global Model	S3, S8, S21

Table 3.8: Studies included in the review sorted by their type of solution to the cold-start user problem.

3.3.2 Cold-Start User Problem Solutions

In this section we will present the different solutions to the cold-start user problem proposed in the papers included in our review. A classification of the papers based on cold-start user problem solution type is presented in Table 3.8.

We will in each of the upcoming sections present a solution type and summarise the reviewed studies belonging to that particular category of solution types. In the summaries we focus on the idea behind the solution, initial user involvement required by the solution, and performance with regards to the cold-start user problem and learning ability where this is reported by the study.

Demographic User Data

It is a common strategy to gather demographic data in order to gain knowledge about the user. Data that may be collected typically includes age, gender, nationality, marital status, income, educational level and occupation. The idea is that people with a more common background share a more similar taste than someone with a random background, and therefore good recommendations can be made as long as we know the new user's background. A drawback when using demographic data is that it requires user effort initially, something that may be found cumbersome by users.

Gao et al. [2007a] (S9) use meta-information filtering. The technique is a two-step process; First they model the user with regards to item categories. For instance, a person may be a farmer. A book recommendation system may then assume that the person is interested in agricultural literature. In the second step they value the items according to other peoples' rating. In their solution a hierarchical categorisation of all items is needed, and the user needs to enter some basic demographic information (profession, age range, title, department, speciality and research area). The domain in this case is a digital library, which is very well suited for a strict hierarchical ordering of items. The study reports performance gains compared to what they refer to as traditional collaborative filtering.

Some methods create a model of the demographic data through a probabilistic relational model, and use this model as a weighted input together with a traditional collaborative filtering approach to recommend items to the user. The weight is then biased towards heavy dependence on the probabilistic model in the early stages, and gradually becomes more dependent on the collaborative filtering as the user rates more items. The reasoning behind this strategy is that collaborative filtering is more effective than the demographic grouping once we have enough ratings.

Lekakos and Giaglis [2007] (S10) propose that lifestyle information may benefit the recommendation process. In their solution lifestyle data consists of demographics (age, marital status and education) and consumer preferences on eight television genres. The user, of course, needs to provide this information to the system. The attributes are stored binary, and similar users are found by applying the Pearson correlation coefficient. The recommendations to the user are then generated by creating a neighbourhood of similar users, and aggregating their ratings weighted by the lifestyle similarity. They also propose a solution which they call *integrated*, which integrates the lifestyle approach with a pseudo user approach where the pseudo user fills in empty values in the rating matrix. The results show that the lifestyle approach is most effective for cold-start users in sparse environments, and the integrated approach is better in all other conditions. Lekakos and Giaglis propose a combination of different methods in the same recommender system, and the choice of algorithm depends on if the target is recommended by enough people, if the user has recommended enough items, if enough similar items have been rated and the level of sparsity in the User $\times$ Item matrix. The lifestyle and integrated approach are the ones relevant for cold-start situations.

Gao et al. [2007b] (S12) introduces the *user grade function*, which makes the collaborative filtering more efficient. The *user grade function* is a continuous function depending on the amount of ratings done by the user, and the density of users with the approximately same amount of ratings. On the MovieLens data set the user grade collaborative filtering performs slightly better then pure collaborative filtering for cold-start users, but as the user rates more movies, collaborative filtering gives better results. However, the user grade collaborative filtering is between 21 to 26 times as efficient when finding nearest neighbours, making it more scalable. Combined with the probabilistic model it also outperforms collaborative filtering when it comes to performance. They do not, unfortunately, present numbers for cold-start users combining user grade collaborative filtering and probabilistic relational models.

Lam et al. [2008] (S14) train a probabilistic model and depend entirely on that. They try to calculate good recommendations based on the probability of what kind of user type the said user is and the probability of how well liked that item is by those user types. They develop what they call an *aspect model*, based on the works of Hofmann and Marlin. Marlin refers to both in [Marlin, 2004]. They introduce user types as latent variables, which are based on age, gender and job. They then train their user types by an expectation-maximization procedure. When new users enter the system, they get a distribution of latent variables, which is the user types.

The initial user involvement in their experiment consists of entering information of age, gender and occupation, but they do state that further research is needed to establish what attributes needs to be gathered to make good recommendations. The results show that the mean absolute error is very promising, but the authors warn that system has trouble with rejecting bad items.

Agarwal and Chen [2009] (S16) presents *regression-based latent factor models* as a solution in both warm and cold-start situations. Their approach is dependent on demographic and content features, which is used in regression to find latent factors. Users and items start out with profiles that are entirely based on their features, and it will reflect a global feature based profile. When more is known about a user or an item, their profile will deviate from the global one in a refinement process. The latent factors are found by multiplicating the user and item features with a weight matrix. The final latent matrix for a user and an item is estimated as $u'v$, where u' is the transposed latent matrix for the user, and v is the latent matrix for the item, both estimated through regression. As model fitting algorithm, they choose *Monte-Carlo expectation-maximization* as it is less prone to overfitting than comparable algorithms as the number of latent dimensions increases. Their experiments show strong results on the MovieLens and EachMovie data set, as well as on Yahoo! Front Page.

Chu and Park [2009] (S18) use what they refer to as a *predictive bilinear regression framework*. They are especially concerned with dynamic content, in their case news articles, where new items are created at a rapid pace, and old items are less interesting than new items. In their approach they combine user history (consumption and other activities) with user demographics in their user profiles. The demographics are gathered from Yahoo! user profiles. Observational data is stored in a multidimensional matrix, with timestamp as one of the dimensions. Items also gets information profiles, both static data (categories, name, title, etc.) and temporal data (popularity, click-through rate and price). When new items are evaluated for existing users, the users static preferences are highly dominant as the items do not have any temporal information, likewise, when new users enters the system, the recommendations are based on content preferences since no user history is obtainable. As items gets older, temporal information will get more important, and as user history gets more extensive, more recommendations will be made on the basis of it. The predictive bilinear linear regression framework is compared with eight other solutions on Yahoo! Front Page, and performs better than all of its competitors.

Park and Chu [2009] (S20) propose a novel approach using *tensor regression* they call *pairwise preference regression*. Both items and users have their own feature vectors consisting of content information and demographic data. The ratings are used to establish affinities between user features and item features. They use this information to help in cold-start situation, and also take into account user-specific rating bias as a result of personal rating criterias. The initial user involvement is only stating age and gender in this case. They test their approach in three different settings, 1) existing items for new user; 2) new items for existing users; and 3) new items for new users. The solution is compared with random recommendations, most popular, segmented most popular and two versions of the affinity algorithm (called affinity1 and affinity2), which is presented by Nag [2008]. Their solution and segmented most popular have significantly better results than the other solutions in setting 1, and in the two other settings pairwise preference regression is only compared with random and affinity2. In the first experiment segmented most popular performs slightly better then the authors' solution, but the difference is not significant. In the two last experiments, their solution performs significantly better.

Wang and Wu [2009] (S24) approaches the recommender field with a regard for profit consideration. They call their solution *clique-effects collaborative filtering*. They point out that today recommender systems adds value to E-commerce company by easily displaying product information, enhancing cross-selling between users and sustaining user loyalty by providing good service. Their aim is to add a fourth value by adding profit as a performance metric. They group the users into several groups, known as *cliques*, and the cliques is similar to what we would call neighbourhoods in traditional collaborative filtering. The probability of whether a user will purchase a product or not is a convex combination of the probability that the users in the same clique will purchase, and the probability that the users not in the clique will purchase it. Finally they propose two methods for the profit consideration, one for optimising profits, and one to create win-win strategies where the goal is to balance user satisfaction with company profit. Their results show that they perform better in all situations except when new users enter the system. It learns faster than traditional collaborative filtering, and keeps a higher recall while maintaining the precision when it stabilises.

The different solutions using demographic data has several similarities when it comes to what they ask for and what predictive features they accomplish. The different demographics asked for is presented in Table 3.9. When it comes to the amount of demographic attributes that was used by the solution, three papers asked for three [Lekakos and Giaglis, 2007, Lam et al., 2008, Chu and Park, 2009] (S10, S14, S18), two papers asked for four [Agarwal and Chen, 2009, Park and Chu, 2009] (S16, S20), one paper asked for seven [Gao et al., 2007a] (S9) and two did not specify how many were asked for [Gao et al., 2007b, Wang and Wu, 2009] (S12, S24). In other words, most of the solutions asked for less the five demographics. This is not all that surprising when considering the amount of user types available by using the limited amount of categories. Lam et al. [2008] (S14) demonstrates this by pointing out that by using two gender categories, seven age ranges and twenty occupation alternatives, the result is 284 possible user types. They also demonstrate that in their instance it might be more beneficial to not utilise all user types, and they get the most promising results using only six user types.

Ask to Rate

As pointed out by Rashid et al. [2002] (S2), the most direct way of acquiring information for use in personalised recommendations from a new user is to present items for the user to rate. By providing ratings, it becomes possible for collaborative filtering systems to find similar users by comparing the now not-empty row of the

Type of data	Number of Studies	Studies
Age or age-range	6	S9, S10, S14, S16, S18, S20
Occupation	5	S9, S14, S16, S18, S20
Education	1	S10
Marital status	1	S10
Gender	3	S14, S16, S20
Location	3	S16, S18, S20
Did not specify	2	S12, S24

Table 3.9: Studies sorted by which demographics asked for.

new user with other users' rows. In content-based filtering, item-item comparison becomes possible because the system can use the item feature vectors of the newly rated items in order to find similar items. In both types of systems, the effect is better recommendation accuracy for new users. Two of the studies included in our review used some variation of this strategy in order to overcome the cold-start user problem.

Aimeur and Mani Onana [2006] (S6) present a system based on collaborative filtering with a list of trusted contacts. The system, which is tested on movie data from the Epinion website, invites the user to rate some popular products in addition to allowing users to search around freely and contribute more ratings on known products. No experiments are performed to specifically test the performance of the system for new users, rather they test the hypothesis that collaborative filtering using a local network of contacts give more accurate predictions than using all users. Although the results are not commented with respect to cold-start users in the paper, it can be observed that for users with less than 10 given ratings, the mean absolute error is on average lower when using the local contacts instead of all users in the system.

While Aimeur and Mani Onana [2006] (S6) present a system where the *ask new user for ratings*-strategy is implemented, Rashid et al. [2002] (S2) focus on the issue of which items to present to the new user during an initial *ask-session*. Asking the user to go through many pages of items is likely to annoy or frustrate the user, while asking for very few ratings is likely to give the system very little information to base recommendations upon. Additionally, it is important to ask users to rate things they are likely to have an opinion about, but still separate their tastes from other users'. The study focuses on maximising prediction accuracy while minimising the user effort needed, and considers six strategies for presenting items to the new user. To test the strategies, experiments are performed to test how many movies users have seen among the ones presented with different strategies, and prediction accuracy measured as the mean absolute error as a result of presenting the same number of movies, but with different strategies. The data set used is the MovieLens data set. The results of the experiments show that the *Popularity* strategy, presenting movies ranked by number of ratings in a descending order, need to present few movies to the user and has a high prediction accuracy. Another good strategy is the *(log) Popularity*Entropy* strategy which combines the logarithmic popularity score with information gain on items, and rank items in a descending order based on the multiplication of these factors. This strategy requires users to go through more pages of movies, but results in the best prediction accuracy of the presented strategies.

Trusted Users

Two of the studies included in our review propose using trust as a solution to RS weaknesses, and in particular how it can help cold-start users. The idea is that users select other users whom they declare trusted users. This trust-status means that the target user considers the trusted user's opinions to be representative of his own opinions, and that the recommender process should use the trusted users' ratings in generating recommendations for the target user.

Massa and Bhattacharjee [2004] (S4) present an experimental analysis of a data set collected by crawling the epinions.com website. They focus on how the coverage for cold-start users can be significantly increased if users declare trust in other users and trust is allowed to propagate so that if A trusts B, and B trusts C, one can assume that A trusts C to some degree. Metrics for measuring direct and propagated trust are not within scope of the paper. A key result from the experimental analysis presented in the paper, is that while on average only 2.74 users can be compared to cold-start users with the Pearson correlation coefficient, 94.54 users can be reached using trust with one level of trust propagation.

Victor et al. [2008] (S15) point out that cold-start users not only have given few ratings, but they have typically also expressed trust in few users. In order for trust to help cold-start users in getting accurate recommendations, they need to express trust in at least one user. To help cold-start users find trusted users, Victor et al. propose to use key figures like *mavens* (users that have written many reviews), *frequent raters* and *connectors* (users with many (trust) connections to other users). By connecting to (expressing trust in) key figures, cold-start users are shown to significantly increase coverage and maintain a sufficiently high recommendation accuracy. It is also shown that connecting to a key figure is more beneficial to a cold-start user than connecting to a random user.

Specification of Feature Preferences

In two of our review papers the authors propose to mitigate the cold-start user problem by allowing the users to define an initial profile. In the first one Pronk et al. [2007] (S11) describe a content-based approach where the system based on *naive Bayes classification* combines a user-created profile with a learned profile. Initially the system bases its predictions solely on the user-created profile, but as the user interacts with the system, the learned profile gains more and more influence on predictions. After sufficient use, the predictions are solely based on the learned profile unless the user makes a change to his own profile. The user profile in the proposed system is a set of like-degrees for different feature-value pairs. Although the proposed approach has not been tested on real users, it has been tested as a proof of concept with seven collected user stories. The stories are parted into three parts, one for initial profile building, one for new ratings added to training set and one for testing of the system. The results of the tests show that using a user-specified profile initially can indeed mitigate the cold-start user problem, but also that the profiles are quickly learned (classification error rate difference less than 0.05 when more than 40 ratings are available in the training set of the target user).

Similarly to Pronk et al., Hölbling et al. [2010] (S22) use what they call an *implicit* profile (learned) and an *explicit* profile (user-specified) which have weighted influence on the final predictions. Also similar to the Pronk et al. approach, is that the explicit profile is heavily weighted initially, while the implicit profile linearly takes over as more user interaction data is collected by the system. Hölbling et al.'s approach is focused on personalising TV-watching, and uses TV program meta-data as features in their content-based approach. By tokenising the meta-data attached to TV programs and using the resulting tokens as inputs to several statistical classifiers, the approach predicts whether the target user will like or dislike programs. Hölbling et al.'s proposed system is tested on a collected set of 20 users' TV-watching stories. While the experiments do not focus on comparing the prediction performance of the proposed approach to other approaches like collaborative filtering, they focus on how the internal parts of the approach can be tuned to achieve the best performance. The key findings of the experiments are that of the three tested classifier types, the RobinsonFisher classifier produces the best predictions for cold-start users and that the title of programs is the meta-data that raises the prediction accuracy the most, and in particular for cold-start users.

External User History

Middleton et al. [2004] (S5) propose to bootstrap their Quickstep system for recommendation of research papers with an existing ontology built from a publication database and a personnel database. The databases contain information about publications made by the same people that will use the Quickstep system (employees and students at the University of Southampton). Hence, information about interests and preferences of users can be inferred even before the users enter the system for the first time, offering a solution to the cold-start problems. To handle cold-start users, the system uses historical publications by the new user from the publication database and a set of user profiles of similar users to create an initial profile. Experiments on a set of log data gathered by previously deploying Quickstep (not bootstrapped) at the University of Southampton, showed that the initial profiles created for cold-start users on average included 84 percent of the topics in the final profile. However, the experiment also showed that on average more incorrect topics were included in the profile than the number of topics that should have been there (55 percent incorrect to 45 percent correct). It also showed that system struggled with learning new interests.

Tagging of Items

Kim et al. [2010] (S23) propose *collaborative tagging* to solve problems with sparsity and cold-start users. Instead of giving ratings to items in a recommender system, users assign tags or keywords to the items they have used. To measure similarity between users, the proposed approach uses a user-tag frequency matrix to find users that have assigned the same tags frequently. These nearest neighbours to the target user are used to create a *candidate tag set* that contains tags that the target user is likely to apply to items, given the tag preferences of his nearest neighbours. Finally the target user's candidate tag set is used as features in a naïve Bayes classifier which calculates preference probabilities for items given the target user's candidate tag set. The N items with the highest probabilities are recommended for the user.

Because the same tags can be applied to many items, it is likely that the User × Tag matrix will be denser than the User × Item matrix. By exploiting this fact to find similar users to the target user and then to construct the candidate tag set, the proposed approach offers a solution beneficial to cold-start users as the coverage initially is assumed to be increased. To explore this, Kim et al. perform experiments testing the performance of the proposed approach against standard user- and item-based collaborative filtering approaches for cold-start users. The bookmarks data set used in the experiments was gathered through a crawling of the del.icio.us website. The experiments show that the proposed approach performs better, measured by recall, than the other collaborative filtering approaches when the target user has made few (<10) bookmarks, and especially well when the number of bookmarks given is below three. Although the performance was better than the other approaches, it is worth noting that neither of the approaches performed well, as the best recall score, achieved by the proposed approach, for users with less than three bookmarks in the training set, was 1.21 percent.

Global Model

Methods using global models is concerned about finding a global model for all users, and then apply this model to the new users entering the system. Zigoris and Zhang [2006] (S8) point out that average users do not like to answer a lot of questions, or provide explicit feedback on items they have encountered, and at the same time the users want reasonable performance from the moment they start using the system. They suggest two approaches to improve the experience for cold-start users: using implicit feedback from the users to getting to know them by using implicit feedback; and borrowing information from others when creating a new user. By using a Bayesian hierarchical model, they compose the model of two parts. The prior probability, which is generic and based on other users, and the data likelihood, which is based on the data provided by the unique user. When a new user starts using the system, little is known about the new user, and the prior distribution is the major contributor. As the system gets more training data for the new user, the data likelihood becomes more important. Their solution performed well for cold-start users, and strong evidence was provided to support the idea that using a global model as a prior works well to create a reasonable performance from the start. However, it also suggested that using implicit feedback to getting to know a user was not very effective, even combined with explicit feedback.

Manavoglu et al. [2003] (S3) suggest Markov mixture models for generating probabilistic behaviour models. They create a number of user clusters, and the user is distributed among those clusters. The personalized mixture model is presented in Equation 3.2. The $P(A^{next}|H(U), Data, k)$ is a global probability, and can be either a Markov model or a maximum entropy model, and fixed across all users. The $\alpha_{U,k}$ is the individual weight for that particular cluster. When a cold-start user enters the system, the initial weights will be the same as the global ones. Therefore, like in Zigoris and Zhang [2006] (S8), a cold-start user should experience a reasonable non-personalised experience from the start. As users start to use the system, the weights will adjust to fit the unique user in a personalization process. The authors use this framework to predict the next action for a user on the CiteSeer webpage, which is presented by a ranked list. In other words, all possible actions are evaluated in order to find the best one. This is possible due to the limited amount of possible actions. The model shows strong results in their empirical testing, and is able to find user segments based on behaviour.

$$P_U(A^{next}|H(U), Data) = \sum_{k=1}^{N_c} \alpha_{U,k} P(A^{next}|H(U), Data, k) \tag{3.2}$$

Xue et al. [2009] (S21) propose a statistical user language model to integrate an individual model, a group model and a global model. Their domain is predicting what web pages a user would like to visit. The global model is used to smooth the unseen terms in the individual model, and the individual models are then integrated with group clusters through a cluster based language. They also introduce long-term and short-term interests, something that might be very handy since a user might want to plan for the holidays one week, but this is probably not something he would do on a daily basis in a long-term view. Their result shows that using both long-term and short-term interests significantly outperform using either one of them separately. They also show that integrating the individual model with the global model and group model handles the cold-start problem well, and also keeps a slight performance advantage over time, compared to only using individual models, an individual model combined with a group model, or an individual model combined with a global model.

Geographic Position Data

A more domain specific approach to quickly learn the user's preferences is presented by by Bohnert et al. [2009] (S17). By using spatial information and implicit feedback in the museum domain, they are capable of predicting user behaviour. The idea is that in a museum exhibits are not placed at random, but rather carefully planned. They propose that walking distance between exhibits can be interpreted as content distance as well. Another important assumption is that visitors will stay at interesting locations for an extended time, and that the time spent at an exhibit makes it possible to assume the interest level. They develop a Gaussian spatial process model to model associations between locations. They adapt it to suit the environment of recommender systems. To do Bayesian interference on their model, they use slice Gibbs sampling. The system does not require any initial user involvement. They compare their system with a mean model, which expects a visitor to behave according to the average behaviour of all visitors, and a traditional collaborative filtering. The system shows significant improvement compared to the other solutions. However, the visitor needs to see at least one exhibit in order for the system to predict anything better than the mean model.

Fast Learners

What we have chosen to call *fast learner* solutions to the cold-start user problem are solutions that do not require any special input or actions from new users. This contrast them from trust-based solutions where users have to explicitly express trust in a user, or solutions that require the user to give demographic information. Instead fast learners aim to modify filtering algorithms in a way that allows systems to achieve higher recommendation accuracy for new users faster than traditional algorithms. Of the studies included in our review, four approaches use this type of strategy.

Ko and Lee [2002] (S1) present a hybrid system. The system builds a content-based user profile for the target user by monitoring the feedback given on items by the user, and compares this profile to a set of group profiles to find the groups that share preferences with the target user. Items liked by users in the groups with similar preferences are recommended to the target user. This system is helpful to new users because it can find similar users to the target user even if they do not share any rated items.

The approach presented by Ko and Lee is tested on a data set of 200 users and 1600 documents where each user has rated at least 10 documents. In comparison to three other approaches that focused on sparsity and the cold-start user problem the proposed approach performed equally or better both when users are new (given 10 ratings) and when users are becoming warmer (given 80-100 ratings), however, the performances seem to converge as more ratings are available.

Park et al. [2006b] (S7) propose *naïve filterbots* as a solution to handle the cold-start user problem. Filterbots are artificial users that rate most items based on information or features related to the items. In the experiments with movie datasets, seven bots rate movies based on e.g. average ratings from other users, ratings given by critics (e.g. NY Times) and movie genres. Since the bots rate most items in the item set, and they are considered regular users in the user similarity measuring process of collaborative filtering, they allow cold-starting users both to find more similar users and to reach more items than they would without the filterbots. In comparison to pure item- and user-based collaborative filtering approaches, the proposed filterbots-enhanced approach perform better on one dataset, and marginally better on the other two datasets. They also show that item-based collaborative filtering outperforms user-based collaborative filtering in experiments. The filterbots' advantage seems to diminish after a sufficient amount of training data is processed, it then performs similar to non-filterbots solutions.

Liu and Deng [2008] (S13) propose a solution to the cold-start user problem based on normalising ratings in order to more accurately measure similarities between users. By normalising ratings, an agreement between two users on a deviating rating will result in higher similarity score for those two users than for two users agreeing but following the global trends. While the strategy also helps warm users, Liu and Deng show in an experiment that the proposed approach outperforms traditional collaborative filtering on the MovieLens dataset for new users, which are here defined as users with less than 5 ratings. For these users the approach is reaching a mean absolute error score of 0.74 versus 0.80 for the traditional collaborative filtering approach.

In the last of the fast learner-solutions, Maneeroj and Takasu [2009] (S19) propose a hybrid system where content-based pseudo-ratings are generated to fill empty slots in the User $\times$ Item matrix. This means that all neighbours of the target user will have ratings or pseudo-ratings for every item in the system, which again means that predictions can be made for all items as soon as any neighbour to the target user can be found. In the experiments presented, the proposed hybrid approach outperforms two content-based approaches and achieves a mean absolute error of 0.9 when predicting scores for all movies for cold-starting users who have given only two ratings.

3.3.3 Quality Assessment

SID	QC1 Aims	QC2 Context	QC3 Design	QC4 Test data	QC5 Algorithm	QC6 Experiments	QC7 Comparison	QC8 Metrics	QC9 Results	QC10 Findings	Total
S1	1.0	0.5	1.0	0.0	0.5	0.0	0.5	1.0	0.0	1.0	5.5
S2	1.0	1.0	1.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0	9.5
S3	1.0	1.0	1.0	0.0	0.5	1.0	0.0	1.0	1.0	1.0	7.5
S4	1.0	0.5	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	8.5
S5	1.0	1.0	1.0	0.0	0.5	0.5	0.0	1.0	1.0	1.0	7.0
S6	1.0	1.0	1.0	1.0	1.0	1.0	0.0	1.0	0.0	0.5	7.5
S7	1.0	1.0	1.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0	9.5
S8	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.5	1.0	1.0	8.5
S9	1.0	0.5	1.0	1.0	0.5	1.0	0.5	1.0	0.0	0.0	6.5
S10	1.0	1.0	1.0	0.5	0.5	1.0	1.0	1.0	1.0	1.0	9.0
S11	1.0	0.5	1.0	0.0	1.0	0.5	0.0	1.0	1.0	1.0	7.0
S12	1.0	1.0	0.0	1.0	0.5	1.0	1.0	1.0	0.0	0.0	6.5
S13	1.0	1.0	1.0	0.5	1.0	0.5	0.0	1.0	0.0	0.0	6.0
S14	1.0	0.5	1.0	0.5	0.5	0.5	0.0	0.5	0.0	0.5	5.0
S15	1.0	1.0	1.0	1.0	0.5	1.0	0.0	1.0	1.0	1.0	8.5
S16	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	1.0	9.0
S17	1.0	0.5	1.0	0.0	1.0	1.0	1.0	1.0	0.5	1.0	8.0
S18	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	0.5	7.5
S19	1.0	0.5	1.0	0.5	1.0	0.5	0.5	1.0	1.0	1.0	8.0
S20	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	1.0	9.0
S21	1.0	1.0	1.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	8.0
S22	0.5	1.0	1.0	0.0	0.5	0.5	0.0	0.0	1.0	1.0	5.5
S23	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	9.0
S24	1.0	1.0	1.0	0.0	0.5	1.0	0.5	1.0	1.0	0.5	7.5
Total	23.5	20.5	23.0	9.0	19.0	20.0	13.0	21.0	15.5	19.0	Avg=7.65

Table 3.10: Results from quality assessment of studies included in review.

Solution Type	Studies	#Studies	Average
Demographic User Data	S9, S10, S12, S14, S16, S18, S20, S24	8	7.50
Ask to Rate	S2, S6	2	8.50
Trusted Users	S4, S15	2	8.50
Specification of Feature Preferences	S11, S22	2	6.25
External User History	S5	1	7.00
Tagging	S23	1	9.00
Geographic Position Data	S17	1	8.00
Fast Learners	S1, S7, S13, S19	4	7.25
Global Model	S3, S8, S21	3	8.00

Table 3.11: Table showing total quality assessment score of studies averaged within each solution type.

As explained in Section 3.2.5, we performed a quality assessment of each of the papers included in our SLR. Table 3.10 presents the results of the quality assessment for each of the reviewed studies.

As the final filtering of the study selection process required the studies to score at least one and a half points totally on the first two quality criteria (QC1 and QC2) to be included in our review, all the studies included score highly with regards to stating their goals and putting their research into context. The studies in general also average highly on the criteria QC3, QC5, QC6 and QC8, signifying that they do well at justifying their designs and explaining their algorithms, experiments and the metrics used to measure performance.

The studies came out worse with respect to analysing the results of their experiments (QC9) and comparing the results achieved with their approaches with other approaches (Q7). In general, the studies were worst at providing a description of their test set thorough enough for others to reproduce the set (QC4). Although not all studies scored highly with respect to analysing their results and comparing results to other solutions, most findings presented in studies were well supported by evidence (QC10).

All the studies scored between 5.0 and 9.5 out of 10.0 possible points, and the average score was 7.65 points.

Table 3.11 shows the average total scores for each of the solution types.

3.4 Analysis

In this section, we will first analyse our results with respect to the four research questions defined in step ST1 of the review process:

RQ1 What are the existing solutions to handle the cold-start user problem in recommender systems using Bayesian reasoning?

RQ2 How do the different solutions, found through the work with RQ1, compare to each other with regards to initial user involvement, learning ability and initial service quality?

RQ3 What is the strength of the evidence in support of the different solutions?

RQ4 What implications will these findings have when creating a recommender system for tourists?

We analyse each of the research questions in Section 3.4.1, Section 3.4.2, Section 3.4.3 and Section 3.4.4 respectively. In the final part of this section, Section 3.4.5, we point out limitations of our review.

3.4.1 RQ1: Existing Solutions to the Cold-Start User Problem

We have identified several different approaches to the cold-start problem that can be generalised in two categories, those which requires no extra user effort and those which do. The ones which do not require extra user efforts are the following:

Fast learners
These solutions share the common trait that they solve the cold-start problem by speeding up the learning phase.

Global model
Starts off from a global model, which gradually personalises a general model into a personal model.

Geographic position data
Takes advantage of the strict spatial layout in a museum setting to quickly predict user preferences.

A common strength these solution share is the low effort for first time use. The other categories of solutions are those which require extra user effort, namely the following approaches:

Ask to rate
Makes the user rate a number of items before using the system.

Demographics
Asks for demographics in an initial set-up phase, and uses this to find similar users.

External data
Require external data to be analysed to make a user profile.

Trust

Increases the amount of users to compare with by letting cold-start users trust existing users.

Tagging

Stores a User × Tag matrix, and uses that matrix to find neighbours that tag similar. Since the User × Tag matrix is denser than the User × Item matrix it helps cold-start users.

Specification of feature preferences

The user specifies content features he finds interesting, and the system is then able to recommend items that contains those features to the user.

The common positive trait among these solution is the instant personalisation from the first recommendation, but it comes at the expense of a higher user effort.

3.4.2 RQ2: Comparison of Solutions

We have decided to compare the solutions in three dimensions: initial user effort, initial service quality and learning ability, which is in accordance with research question 2. We present our evaluation with regards to the initial user effort as a ranked list, while it was harder to compare the initial service quality and the learning ability in the same manner due to lacking comparability between the different test set-ups and data sets.

Initial User Effort

In Table 3.12 we present our evaluation of the initial user effort in the cold-start solutions. Five stars is the maximum score, and we consider those that have no initial user effort to be the best.

The *fast learners* are the least homogeneous group of solutions, but share the common trait that they do not require any initial involvement. *Global models* also have the benefit of not requiring any initial user effort, and makes the assumption that the global user model probably is quite similar to the local user. Likewise, using *geographical spatial data* is effortless from a user's viewpoint.

When considering those that do require initial user effort, *demographics* is the least obtrusive approach. Solutions using demographics usually only require a few demographic data variables, in the selected studies mostly three or four, and for most people they should be easy to fill in. Unlike solutions *asking to rate* items, there is a known amount of questions asked beforehand and no knowledge of the domain is needed.

The *ask to rate* will also in some instances, like in a book recommender scenario, encounter situations where users with a substantial experience in the domain will struggle to find a sufficient number of items to rate. If we consider the User × Item matrix, where users represent the rows, items represent the columns, and a column consists of ratings for one item, *ask to rate* solutions will typically struggle when the item columns are sparse even for the most popular items. A third possible option may be to let the users choose what items to rate themselves, but this require an increased user effort as the users have to search for the items manually.

Like the *ask to rate* solution, *specification of feature preferences* requires that the user has some initial preferences in the domain, and that the user takes care to add enough preferences to get good recommendations. The effort needed depends on the complexity of the domain. We consider the user effort needed to be somewhat similar between the two alternatives *ask to rate* and *specification of feature preferences*, depending on the complexity of the domain and the sparsity of ratings for popular items.

We consider asking the user to submit an external document to be analysed as a small effort where it is applicable, but the solution makes it necessary to have some sort of user history. This limits the possible domains where it is applicable, and it does not solve the cold-start problem for new users without experience in the domain. Therefore we rank it below solutions mentioned earlier. Middleton et al. [2004] uses published papers as his external data, for a user new to the domain this means actually writing a paper to take advantage of the cold-start solution. Asking people to rate items or making them provide external documents circumvents the cold-start problem by making the users experienced when they take advantage of their prior knowledge of the domain. If a user is new to the domain, this will not work.

Those which require most user effort according to our analysis is tagging and trust-based solutions. Tagging items to kick-start the system requires a creative effort from the user, and this makes it an effort intensive alternative. The nature of the solution also implies that the users should keep on tagging new items during their sessions. We suspect this will be considered a burden for casual users.

Trust-based solutions require that the user rates some items before getting effective beyond traditional collaborative filtering. The user effort required is a search process looking for similar users, which will probably require the new user to go through other users' history to find someone with similar taste. This may very well be too much to ask a regular user looking for a quick recommendation. The process may be simplified if the system can present users that are similar, but less sparse.

Initial Quality of Service and Learning Ability

The initial quality of service and learning ability is not that easy to measure since many studies lack learning progress data. Instead they typically present the mean absolute error, or a similar measurement, for the whole set. This makes it impossible to evaluate the quality of initial recommendations. For instance, an algorithm may produce poor initial recommendations and then later camouflage these with high quality recommendations when the User $\times$ Item matrix gets less sparse. We choose to refrain from ranking solutions by initial quality of service and learning ability in the same way as we ranked them by user effort since there is too little comparable data.

Solution Category	Initial User Effort
Fast learners	⋆⋆⋆⋆⋆
Global model	⋆⋆⋆⋆⋆
Geographic position data	⋆⋆⋆⋆⋆
Ask to rate	⋆⋆⋆
Demographics	⋆⋆⋆⋆
External data	⋆⋆
Trust	⋆
Tagging	⋆
Specification of feature preferences	⋆⋆⋆

Table 3.12: Comparison of effort for new user sorted by different solution categories where more stars means less effort.

The *ask to rate* approach and *external document* approach improves the cold-start situation by taking advantage of prior knowledge in the domain, but how well the system learns will not be affected by asking people to rate as no new changes in the model is necessary. The only paper describing external document is Middleton et al. [2004], and the solution does significantly improve the cold-start situation where it is applicable, but has issues when trying to learn new interests.

The empirical data on initial service quality and learning ability is limited when it comes to *specification of feature preferences*. However, Pronk et al. [2007] shows that the cold-start situation drastically improves in their solution, and it converges with the regular learning algorithm. However, it seems to be an easy training set, and we see little learning in their experiment. Hölbling et al. [2010] demonstrate that the Robinson-Fisher classifier is a good choice both in cold-start and warm-start scenarios.

Studies presenting demographic solutions which provide learning progress data, or cold-start data, with the exception of Wang and Wu [2009], show increased performance in cold-start scenarios. Wang and Wu [2009] show, however, better coverage than traditional collaborative filtering without losing precision. Lekakos and Giaglis [2007] points out that their lifestyle solution is more effective in very sparse data sets when compared to regular Pearson correlation, and becomes more like Pearson correlation when density increases. As a consequence of this the systems where weights are used to differentiate between the learned collaborative model and the initial demographic model seems to be very effective in making accurate predictions.

Zigoris and Zhang [2006] shows that the prior, which is the global model, does help to alleviate the cold-start problem before converging to no-prior, which is a system with no initial global model. It also shows that proper personalisation takes effect after 200 training examples. Since the demographic solutions in practice often is a more fine-grained system, with customised user types instead of the more general, global user type, it is not likely that the global model can outperform demographic solutions when it comes to cold-start users. This means, however, that the solution's initial quality of service for cold-start users is improved while maintaining the performance of traditional approaches when users get warm.

The *fast learners* is a diverse group of solutions, but they share in common that they can not improve the first initial recommendation. However, it is worth noting that they learn quicker than traditional approaches, and both Ko and Lee [2002] and Park et al. [2006b] show that the performance of their solutions gain an early advantage before they converge with the comparison solutions, making them no worse than the traditional approaches without their enhancements. Maneeroj and Takasu [2009] show no significant learning, but takes care to mention that the number of latent factors does not seem to influence the learning ability. Park et al. [2006b] also points out that item-based collaborative filtering outperforms user-based collaborative filtering both with regards to the cold-start problem and later performance.

Both trust-based solution and tagging was hard to evaluate when it comes to performance. The trust-based approaches fail to present any empirical data on how well they handle the cold-start situation when it comes to the precision of the recommendations, only the increased coverage is presented. The only solution using tagging is Kim et al. [2010], and the results do show an improved performance, especially when the number of tags were below three. However, it should be noted that none of the solutions that it was compared against, or the solution itself, had a particularly good recall.

3.4.3 RQ3: Strength of Evidence

This section discusses the strength of the evidence put forward by the studies included in our review. The discussion is based on the results from the quality assessment stage of the systematic literature review. The description of the stage was presented in Section 3.2.5, and the results from the stage can be found in Section 3.3.3.

Analysis of Evidence

The main goals of the quality assessment were to evaluate whether the studies:

- had clear aims and were put into context of other studies (QC1 and QC2).
- based their approach on strong theoretically foundation (QC3).
- presented reproducible designs, experiments and results (QC4, QC5, QC6 and QC8).
- compared their approach to other state-of-the-art approaches, and (QC7).
- presented findings based on thorough analyses. (QC9 and QC10).

As a result of the quality screening done as the final part of our study selection, all the studies included in the review did well with regards to aims and context. The studies were slightly worse at putting their research into context than clearly stating their aims. The studies that failed to get full score for context often explained the general approaches to creating recommender systems and pointed out the problems related to those (e.g. the cold-start user problem), but did not mention how these problems had been attempted solved by other approaches similar or not similar to the proposed approach. By not pointing out how others have tried to solve a problem and the weaknesses of their solutions, it is hard for readers to evaluate whether the proposed approach is based on a better idea than the other approaches.

Almost all of the studies did well in presenting a rationale for the design of their approach. In fact, QC3 was the criterion with the highest total score second only to QC1 which was part of the quality screening. This can be attributed to the nature of the cold-start user problem. Most solutions presented are based on intuitive ideas, like asking the user to rate to fill up the user's row in the User $\times$ Item matrix or using demographic data to place a new user into a group of demographically similar users, that are not difficult to understand nor explain.

When it comes to whether all aspects of the studies were reproducible, the studies fared worse. Creating the best recommender system, both in general for warm situations and in cold-start situations, is usually a matter of providing the most accurate predictions for users. When researchers wish to compare their approach to existing state-of-the-art approaches, they will first need to evaluate which approaches seem to be best among existing solutions by looking at results presented in published studies. This comparison becomes hard if either experimental procedure, performance metric or test data is not properly explained in the studies. Secondly, in order to reimplement an approach to experimentally compare it to own system, the approach algorithm needs to be explained down to a high level of detail. The majority of the reviewed studies (14 out of 24) did provide such detailed presentations of their algorithms, but many left parts of their approaches unexplained.

Of all the criteria related to reproducibility, the studies did worst in providing thorough explanations of their test data sets (QC4). Some of the studies used test sets based on crawling (specified) websites in an unspecified manner leaving it impossible for others to reproduce the test set, and hence the experiments. Others used well-known and available test sets like the MovieLens or EachMovie sets, but randomly, or in a vaguely explained manner, picked out subsets which again leaves the experiments unrepeatable. Yet another group of studies based their tests on publicly unavailable test sets. By leaving the test data sets non-reproducible and vaguely explained, it is hard for other researchers to know how much faith that can be put into the results presented by the study.

Most of the studies included in our review did experimentally compare their approaches to other approaches, but many of these failed either to properly present the comparison approaches or to pick state-of-the-art approaches to compare against. Several of the studies reported experiments where they compared the proposed approach to what was often referred to as *traditional collaborative filtering* or Pearson correlation collaborative filtering. This was done without considering other approaches that had a stronger focus on solving the the cold-start user problem. It would be of great value for the development of recommender systems if more studies presented result sections where proposed approaches were compared head-to-head against the other state-of-the-art approaches performing best with regards to the problem in focus.

Although many studies presented incomplete analyses of their experimental results, the findings and conclusions were in general well founded in the evidence presented. A few of the studies concluded, with questionable correctness, that they had solved the cold-start user problem, but most of the studies presented rather moderate conclusions where they stated that their approaches delivered promising results. As pointed out above, many studies presented very short, or no analyses at all, of their results. This may be due to researchers believing that the theoretical analysis typically presented as part of the introduction of their approach works as a pre-analysis of the results as well, but our opinion is that many of the studies leave questions unanswered.

To summarise the analysis, we were satisfied by how studies presented their aims, put their research into context and explained the theory backing their approaches. We also found that the studies provided evidence that backed conclusions and findings in a satisfactory way, but that this was mostly due to weakly stated conclusions rather than thorough analyses of results. The evidence presented by the studies included in this review was weakest when it came to comparing results to other state-of-the-art approaches, and finally with regards to reproducibility, where the lack of properly presented data sets left experiments hard to repeat.

Evidence and Solution Types

Table 3.11 presented the number and average scores of studies in each of the cold-start user problem solution types found through our review. The table shows that out of the nine categories only three were represented in our review by three or more studies. Out of these, *demographic user data* was clearly the solution type which had been empirically evaluated the most. Eight studies presented results of using this approach to handle the cold-start user problem, and the group of studies averaged on 7.5 points in our quality assessment. Although this score is a little lower than the average for the whole population of studies (7.65), we observed that three of the papers in the demographics-group achieved very high scores of nine points, meaning that this solution type is the type that has been the focus of most high-quality studies.

The second highest number of studies, four, focused on *fast performer*-solutions. Although the average score of these papers was 7.25, which was below average, two of the studies achieved 8.0 and 9.5 points respectively. This shows that the principle of tweaking systems into learning quicker than conventional solutions has potential, but it must be target for more empirical research before it can be considered a proven solution. The same conclusion applies to the *global model*-approach which was the focus of three of the studies in our review. The three studies using this strategy scored 7.5, 8.0 and 8.5 points respectively, resulting in an average score of 8.0 points. This score is well above the total average and hence the *global model* solution type must be considered one of the more proven solutions presented for the cold-start user problem so far.

The remaining six solution types were represented in our review by only one or two studies. Four of these solution types achieved high average scores in the quality assessment (*ask to rate*, *trusted users*, *tagging* and *geographic position data*). Because of the low number of published studies on these solution types they can hardly be considered to be proven, but the high scores achieved by the ones published give good reason to perform further experiments to confirm the potential of these solution types. The solution types *Specification of Feature Preferences* and *External User History* were only represented by two and one study respectively, and the average scores of the studies in these groups were below the total average. These solutions types remain unproven, and we await further research on these types before any conclusion about their quality can be drawn.

In general, most of the solution types can only be considered promising because of the low numbers of studies published performing research on them. Hardly any of the solutions can be considered proven with one possible exception for the *demographic user data* solution which is the solution type with an effect documented by most high-quality studies.

3.4.4 RQ4: Implications for a Recommender System for Tourists

In this section we review the nine solution types identified through the work with **RQ1**, discussed in Section 3.4.1, and evaluate each of them in the context of a recommender system operating in the *tourist domain*.

A regular scenario in the tourist domain is a new user requesting recommendation services for a short time span resulting in every recommendation being important. This means that the system has a limited time available for learning to know new users, and that the initial quality of the recommendation services need to be high. Using the User $\times$ Item matrix as a reference, the tourist attractions are items in the matrix, and they are fairly stable with few new attractions getting added. The users are almost always in a cold-start phase, and few experienced users exist.

To summarise what we find important in a recommender system for tourists, we present the following list:

- Initial quality of service needs to be good for cold-start users.
- Initial user effort should be kept to a minimum.
- The need for further personalisation is small due to the limited user life-cycle.

The need for instant quality makes the *fast learners* and global models not very viable. There is too little time for individual models to get any valuable gain from these methods, and as a result the learning on individual users are not of greater importance. However, there is a need for a solid Bayesian reasoning model that learns well across all users over an extended period of time. *Geographic position* *data* requires a very well-planned city to work, and it is a far stretch to consider it in a real world scenario.

The need for a more instant personalisation makes it desirable with initial user input. Trusting friends, tagging and external data are options that also seem as bad choices because it is hard to trust other peoples' experiences of attractions that users are not familiar with, combined with few experienced users. Tagging also requires some experience, and external data is meant for documents and is not suitable for other domains in its current form. Asking tourists to rate attractions in a city before they have seen them is of course unrealistic. A possible approach is to ask them to rate world known landmarks like The Eiffel Tower and Golden Gate Bridge, however, this can easily turn into a situation where almost everyone rates everything with a high rating, hence becoming little informative. This leaves us with the alternatives of demographic data and specifications of feature preferences.

The success of an approach using demographics depends on the presence of homogeneous subpopulations of users that can be differentiated by collectable demographic data. It is reasonable to believe that knowledge of a tourists age, nationality, education, occupation and income level will make it possible to make enhanced predictions. Specification of preferences requires that the items, or attractions in our case, is possible to classify. The successfulness is also dependent on that the classifications is enough to determine whether this is a good choice for this tourist or not. For instance, a tourist may not be particularly interested in religious buildings, but if the city's main attraction is a cathedral, the tourist may be very disappointed if it is not recommended. Tourist attractions are on the other hand possible to classify. For instance, we have museums that we differentiate in several subcategories, hotels of different standards, landmarks and so on. We have reason to believe through our study that the demographic solution has the greatest fit for the tourist domain and it has a stronger strength of evidence than specification of preferences. However, an interesting idea is to combine preferences with demographics like Lekakos and Giaglis [2007].

Among the different demographic solutions, Lekakos and Giaglis [2007], Agarwal and Chen [2009] and Park and Chu [2009] all get a score of 9.0 in the quality assessment screening. A weakness in the works of Agarwal and Chen with regards to the tourist domain is that they do not provide results for users with less than thirty ratings. Chu and Park [2009] got a slightly lower score, but is tightly related to the solution of Park and Chu, and demonstrates good performance with very sparse users which, even though Chu and Park [2009] focuses dynamic content in their approach, is the most important issue in recommender systems for tourists. We believe that this solution may successfully adapt to the tourist domain.

3.4.5 Limitations of this Review

There are several limitations of these review, most importantly we have might have been to inclusive with regards to the *Bayesian reasoning* criteria (IC5), and the lack of common metrics and experimental designs have made it difficult to draw quantitative conclusions.

A problem with performing systematic literature review in computer science is the lack of a standardised set of key words, unlike the medical research community that has a controlled vocabulary thesaurus in MeSH[1]. As a consequence, even though we tried to be as inclusive as possible, some relevant studies may have been left out from the initial search process. This became very apparent when we in the process discovered that the cold-start user problem often only was referred to as the *new user problem*. In the process we also noticed that the third group of terms in our search string may have been omitted, which could have resulted in additional evidence for existing solutions, or introduced new solutions.

When examining the studies, we discovered that many studies were not concerned about the details on whether the implementation used Bayesian reasoning or not. We chose to be inclusive in those cases, meaning that we passed them on to the next stage. As a result, there is a possibility that some of the solutions found may not be directly applicable using Bayesian reasoning. Also, there might exist studies proposing solutions that are applicable using this type of probabilistic reasoning, but that were not found because the terms *Bayes* or *Bayesian* are not mentioned anywhere in the study. Studies using the term *probabilistic*, but not specifying *Bayes* or *Bayesian*, are examples of studies that may present applicable solutions, but were not included in the results of our search because they do not mention these specific terms.

In the data extraction stage we could not fill in every cell for every study due to lacking information in the studies, also, some of the data was not well documented, and there is a possibility for misunderstandings in the process. To minimise the possibility of those misunderstandings we followed the procedure of Dybå and Dingsøyr [2008], and made all the decisions in consensus meetings.

It was also hard to make a quantitative synthesis of our findings because there is no standardised way of reporting results, nor is there a standardised way of doing experiments. This limits the possibility to rank the results with regards to performance: both precision and recall, and learning ability.

3.5 Additional Issues

Throughout our work with the systematic literature review, we have discovered three issues related to recommender systems research which are not within the scope of our review questions:

- There is a lack of consistent terminology related to the cold-start problems.
- There is no clear definition of when a cold-start ends and an item, a user or a system can be considered warm.
- There is no common method of performing experiments for measuring cold-start user problem performance.

[1]http://www.nlm.nih.gov/mesh/

Term	Number of Studies	Studies
Cold-Start User Problem	12	S3, S5, S7, S10, S12, S13, S14, S16, S18, S20, S21, S23
Cold-Start Problem	7	S4, S6, S8, S11, S15, S19, S22
New User Problem	5	S2, S9, S10, S17, S24
First-Rater Problem	1	S1

Table 3.13: Terms used in our study articles to describe what we refer to as the *cold-start user problem* in the recommender system literature.

We believe it would be beneficial for the progress of recommender system research to address these issues, and therefore present the issues in the following sections along with some of our thoughts on them.

3.5.1 Terminology for the Cold-Start User Problem

Already in the initial phase of our studies of the recommender system literature, we noticed an inconsistent use of terminology related to what we refer to as the cold-start user problem. To illustrate this inconsistency we collected terms used to describe this problem from all the studies included in our review as part of the data collection phase. Table 3.13 shows the results of this work.

Our review discovered that the term *cold-start user problem* was the most frequently used term to describe the problem of giving accurate recommendations to new users without rating histories. The second most used term was *cold-start problem* (not explicitly mentioning *user*), while the third most popular term used for the problem was *new user problem*. Finally, the term *first-rater problem* was used once to denote the aforementioned problem in the reviewed studies.

It is our opinion that the recommender system community should stick to using the term *cold-start user problem* to name this particular cold-start problem. Using the term *cold-start problem* without specifying *user* may confuse readers because the term is also used to describe the problem of recommending *new items* which has not received any reviews yet. For a study using it this way, see e.g. Schein et al. [2002]. The same applies for *first-rater problem*, which is a synonym for the *cold-start item problem*, and should not be used as a term to describe the cold-start user problem. The best alternative to using *cold-start user problem* is the term *new user problem*. We believe the term is equally suited to describe the problem, but because of the lead in popularity held by *cold-start user problem*, and the practicality of relating the cold-start *user* problem to other cold-start problems, *new user problem* should also be abandoned in favour of *cold-start user problem*.

Because of the reasons presented above, we propose to use the term *cold-start problems* to denote the set of problems related to an object (i.e. users or items) of a recommender system , or the whole system, being new. For the separate cold-start problems we recommend to use the same terminology as used by Park et al. [2006b]: *cold-start user problem* to denote the problem of giving accurate recommendations

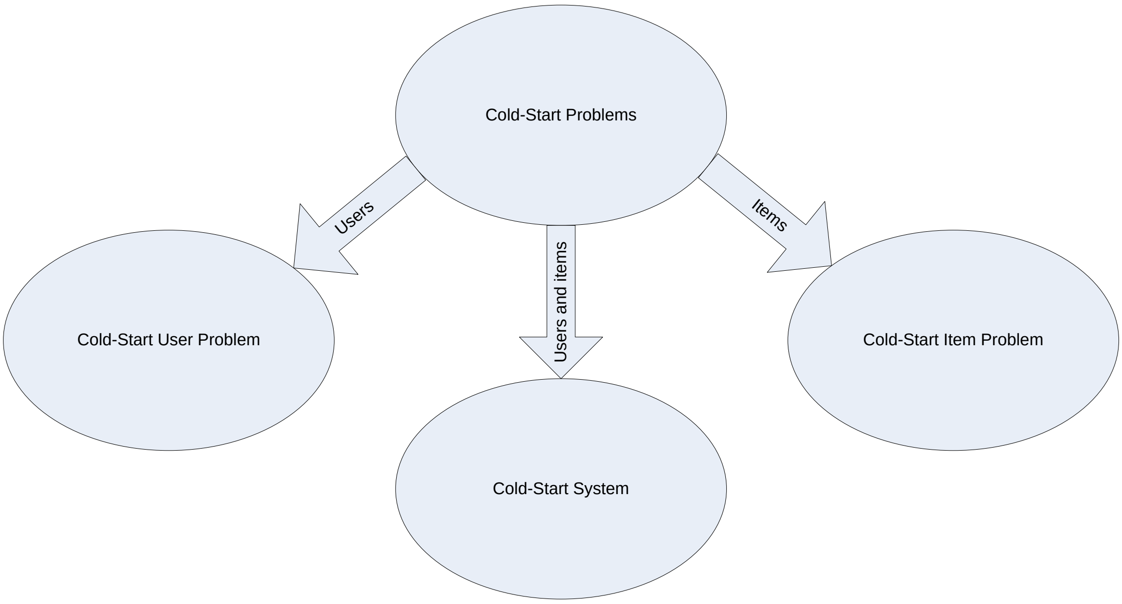

Figure 3.5: Illustration of proposed terminology for the cold-start problems.

to new users without rating histories; *cold-start item problem* to denote the problem of recommending new items which has not received any reviews yet; and finally *cold-start system* to denote the problem of having a new recommender system with no or little data (both users and items). Our proposed cold-start terminology is illustrated in Figure 3.5.

3.5.2 Definition of Cold and Warm Situations

The problem of defining cold and warm users is closely related to the issue of having a common procedure for testing cold-start performance which was discussed in Section 3.5.3.

In order to create such a procedure, we believe the first step must be to make a clear definition of when a user goes from being cold to being warm. This definition may be dependent of domain, but to a limited degree. Regardless of domain, no user will be pleased if he has to receive 15 poor recommendations before relevant ones are presented.

It is not within scope of this review to define when cold becomes warm, but we observe that in the studies included in our review some authors focus on presenting high-quality recommendations for users with no historical ratings at all (e.g. Park and Chu [2009]), others present results from the 10th rating and onwards (e.g. Ko and Lee [2002]), while most authors focus on situations where between one and five ratings are present for the target cold-start user.

3.5.3 Procedure for Cold-Start User Problem Experiments

A problem with the current studies on the cold-start problem is the lack of a common way to test performance. Since there is not a common procedure it is hard to make quantitative comparisons between the different solutions, making the data synthesis more qualitative in nature. Another effect is that many solutions do not provide data for initial quality, nor do they provide data showing how well the prediction ability evolves. To make comparisons easier it would be a great advantage to have a set of common data sets with different sparsity levels, test data that is chronologically ordered so that it is possible to simulate change in interest, a sufficient set of demographics, common metrics and a common way to present progression of prediction ability.

Today EachMovie and MovieLens are by far the most common data sets used to test recommender systems. It would be beneficial to have more data sets with different levels of sparsity to test against. As a common metric mean absolute error (MAE) seems to be the most widely used. A list of interesting demographics are presented in Table 3.9, where we can see that age, gender, occupation and location are especially sought after in many solutions. It is uncertain if those demographics are sought after because of their ability to distinguish users, or out of convenience since they are available in the MovieLens data set.

3.6 Summary

In this systematic literature review we have done a thorough search and study of the recommender system literature to discover the best solutions for the cold-start user problem in recommender systems using Bayesian reasoning.

Our review has identified nine types of solutions to the cold-start problem: using *demographic user data* to group similar users; *ask to rate* which forces users to provide data for the system to reason with; *trusted users* which lets users define other users whom tastes they trust; *specification of feature preferences* which allows users to define which features of items in the domain they like; bootstrapping an *external user history* which kick-starts the users as warm; *tagging* which allows systems to find similarities caught through the semantics of tags on items given by users; *geographic position data* which helps new users by exploring the information in organisation and positioning of items; different *fast learners* which tweak traditional algorithms in order to learn user profiles faster and *global model* approaches which base recommendations for new users on the opinions of all the other users of the system.

We have compared the solution types with respect to initial user effort, initial service quality and learning ability. We found that the clearest difference was in initial user effort, where some of the solution types needed little or no initial contributions from new users while others required users to spend substantial effort in order to overcome the cold-start phase. Due to lacking conformity in experiments and reporting of results found in the reviewed articles, we were not able to draw any clear conclusions about the initial service quality and learning ability of the solutions.

Through our quality assessment, we have also evaluated the strength of evidence for each solution type put forward by the studies included in our review. We conclude that most studies do well at explaining the background of their approaches, but that they often lack details necessary for other researchers to replicate their work. Another observation is that the studies often fail to compare their approaches to other state-of-the-art approaches for solving the cold-start user problem. With regards to solution types, we conclude that *demographic user data* is the solution type which is backed by the strongest evidence. Most of the other solution types can only be considered promising given the presented evidence, and require further experiments in order to become proven solutions to the cold-start user problem.

The final goal of our systematic literature review was to see what implications our findings have for the design of a recommender system for tourists. We conclude that the *demographic user data* solution type is the type that best suits the needs for instant accuracy of recommendations in addition to little user effort of the tourist domain.

Finally, we have pointed out three additional issues for the recommender system community. First, there is a need for a consistent terminology which clearly defines and separates the problems related to cold-start for recommender systems. We propose to use the terms *cold-start user problem*, *cold-start item problem* and *cold-start system* to denote the particular *cold-start problems*. Second and third, we believe the recommender system research related to the cold-start user problem would benefit from having a *clear definition of cold and warm situations* and a *framework for experiments* targeting the problem. We leave these two issues for the recommender system community to address.

Chapter 4

Design of a Recommender System for Tourists

In Chapter 1 we defined two goals for this thesis: to find the existing solutions to the cold-start user problem (**G1**); and to design a location-aware Bayesian recommender system for tourists visiting Trondheim (**G2**).

In the previous chapter we presented a systematic literature review of recommender system literature, which, in accordance with the first goal of our thesis, identified solutions to the cold-start user problem. In this chapter, which builds on the findings of the previous chapter, we present our work with respect to the *second* goal of our thesis, and present a design of a recommender system for tourists visiting Trondheim.

In the first section of this chapter we give a short introduction of the background and goals for the recommender system. In Section 4.2 we present a set of requirements for the recommender system design. Section 4.3 presents an overview of our proposed design, while Section 4.4 goes one level down and allows the reader to inspect the details of the design. In Section 4.5 we explain our thoughts on testing of our design, before we, in light of the requirements, analyse the presented design in Section 4.6. In the last section of this chapter we summarise the most important aspects of what has been presented in this chapter, as well as further work.

4.1 Introduction

The motivation of creating the recommender system was a combination of several factors mentioned earlier in Chapter 1, but the most obvious is the need for good make recommendations to short-time visitors that enters Trondheim while travelling with Hurtigruten[1]. The visitor stays in Trondheim just short of four hours, and is in need of fast and good guidance. We also want to provide services to other tourist segments. Another motivation is to take advantage of the existing infrastructure of Wireless Trondheim, already described in Section 2.4.

[1]http://hurtigruten.co.uk/

Since tourists would benefit from recommendations of different types of services, we had to take this into consideration when designing the system. We also want to make recommendations location-aware, and to make this work we take advantage of GPS technology. We also utilise the increased computational performance in new hand-held devices such as smart phones. This enables us to let the clients compute recommendations offline, and by that removing the need to be online to make the system work. Parallel to our study Wium [2010] has made a location-aware information system for tourists without a proper intelligent recommendation module, and our solution is possible to integrate into his system.

An important aspect is that there is very few experienced users in these settings, and almost all users are cold-start users. This means that traditional collaborative filtering is an inadequate solution. To find a solution to this major problem in our system we carried out an extensive systematic literature review as described in Chapter 3. We have then adapted this solution to fit the tourist domain. This solution takes advantage of demographic data about users and content information on items, and use a predictive bilinear model to provide personalised recommendations to new users. The solution is described in greater detail in Section 4.4.

To summarise, the goals of this part of our thesis are to present a design of a Bayesian recommender system for Trondheim which:

- Provides high initial recommendation quality for cold-start users.
- Requires little initial effort for cold-start users.
- Recommends different types of services.
- Takes context into consideration.
- Is compatible with existing infrastructure provided by Wireless Trondheim.
- Is ready for integration with the tourist information system framework presented by Wium [2010].

4.2 Requirements for Design

In an early stage of the work with our thesis, the main goals for our design, presented in the previous section, were formalised into a list of requirements. They were later reviewed and revived after performing the systematic literature review, when the design process began.

The final list of requirements, presented below, are on a relatively high level, and they do not specify techniques or methods which can be utilised to achieve them. The detailed design which explains the techniques and methods *we* have applied to meet the requirements, is presented in Section 4.4. The requirements that *must* be fulfilled are those with the highest priority; the requirements that *should* be fulfilled is of lower priority, but still should be covered; and the requirements that *will* be fulfilled are those which are planned for the future. The specification of the system is not on the level of detail common in software engineering, but sufficient for our needs.

R1 The system *must* provide personalised recommendations for cold-start users.

R2 The system *must* require little initial effort from cold-start user before being ready to provide recommendation services.

R3 The system *must* handle items tagged with feature information and information used by the service filter.

R4 The system *must* offer recommendations for different services:

(a) Sleeping accommodations

(b) Dining opportunities

(c) Sights and activities

R5 The system *should* take the following contextual parameters into consideration when recommending items:

(a) Location of user

(b) Time of day

(c) User's visit history

R6 The system *should* provide compatibility with the location-aware information system for tourists presented by Wium [2010].

R7 The system *should* support both implicit and explicit feedback from users.

R8 The system *should* provide learning based on Bayesian reasoning.

R9 The system *should* be designed to run on a hand-held device.

After defining what the system should look like, we decided to develop tests to verify that our system would fulfil our goals. An argument for developing the tests before implementing the system is that it clarifies the design of the system further as we have to evaluate the functionality from a user's point of view. We also reduce the possibility bias in the test design. Another motivation for creating the tests was the problems we encountered in our systematic literature review, where many solutions failed to properly test the cold-start performance. Our tests are explained in detail in Section 4.5.

4.3 System Overview

In this section we explain the overview of our solution, as well as explain some of our design choices.

The most important trait a recommender system for tourists needs is an acceptable personalised cold-start performance since most user are new to the system, and they do not have enough time to rate a sufficient amount of items to make traditional collaborative filtering works. Our solution to this is to utilise latent factors derived from the demographic features of the users and the content features from the items.

To achieve this we use the approach explained by Chu and Park [2009] to create latent weights that find the affinity between the different set of features belonging to the items and the users by fitting the weights with observed ratings. The result is a global Bayesian reasoning model where the user's demographics is used as a constant input. Since new items have a set of features attached to them, we also reduce the cold-start item problem much like content-based solutions do.

When classifying our hybrid solution in accordance with the work of Burke [2002] our system is a meta-level hybrid using demographic information to adjust the weights in the model as a result of the ratings for each demographic feature and content feature pair. The model is later used in a content-based environment.

The solution has several nice attributes that are worth mentioning:

- It can be used offline due to the detached local reasoning model.
- Well adjusted weights will provide an improvement to the cold-start user problem.
- The cold-start item problem and sparsity problem are also handled.
- The decentralized approach provides great scalability as most of the computational effort is performed by the hand-held devices.
- Privacy is maintained since the user only needs his own demographic information, which is never passed given to anyone else except the server. There is also no need to pass on location data to the server.
- The reasoning of why a recommendation is recommended can be made transparent since we can trace both the values of the latent weights and the inputs that cause the values.

4.3.1 Item and User Representation

Both users and items are represented as vectors containing their demographical information and content information respectively. The user vector consists of the information stated in Table 4.1, and the item vector consists of the information stated in Table 4.2. All the user vectors are then stored in a User $\times$ Demographic Features Matrix, and all the items are stored in a Item $\times$ Content Features Matrix.

In both instances we can retrieve a single user or item by choosing the appropriate column vector.

4.3.2 Modules of Recommender System

The recommender system is split into two modules, one on the server side and one on the client side. We describe them on a higher level in this section, and in greater detail in Section 4.4.

The Server Module

The role of the server is to collect and store all the information from the different clients, and to update the Bayesian reasoning module and distribute this to all the clients.

The server side module consists of a set of matrices that store all information regarding users, items and ratings in the system. The details of these matrices are explained in greater detail in Section 4.4.1. The server provides interfaces for receiving user demographics, adding items with content information, modifying item information and receiving user ratings on items. The interface for receiving user information is needed when new users enters the system. The server outputs a Bayesian model to the clients and updated lists of items.

The Client Module

The role of the client is to utilise the Bayesian reasoning model to give good recommendations to tourists and to pass on the tourists' ratings to the server so that other clients can benefit from it. The client module resides on the tourists' hand-held devices.

The client consists of a pipeline capable of providing recommendations to new users. The pipeline consists of the following elements:

- An updated list of items containing item information.
- A user profile with demographic features and a list of previously visited items.
- A service filter described in greater detail in Section 4.4.2.
- A Bayesian reasoning model described in Section 4.4.1.
- A selection evaluator described in Section 4.4.1.
- A list of evaluated items.

The most interesting part of the module is the Bayesian reasoning model that is based on the work of Chu and Park [2009]. This was the solution we found to be the best fit for our domain in the systematic literature review presented in Chapter 3. They provided good recommendations for cold-start users in an environment where most of the other users also were cold-start users, not unlike the tourist domain.

User features	
Feature	**Boolean variable**
Age	<25 25-44 45-64 64<
Gender	Male Female
Occupation	Student Science, Education and Academia Customer Service Sales Technology and Engineering Health care Artists Retired Unemployed Other
Type of holiday	One-day stay Regular city break Backpacker
Nationality	Norwegian Other Nordic United Kingdom and Ireland Germany and Benelux Southern and Western Europe Eurasia Central Europe North America Other America China and North-West Asia Other Asia Africa
Budget	Low Medium High

Table 4.1: Demographics that needs to be provided by users.

Item features	
Feature	**Boolean variable**
Night-life	Nightclub Pub Alcoholic beverage
Sightseeing	Art museum Other museum Architectural landmark
Cuisine	Italian cuisine Mexican cuisine Norwegian cuisine Fast-food Asian cuisine
Budget	Low Medium High
Feature	**Other variable**
average rating	continuous between [0.0-1.0]
location	coordinates

Table 4.2: Features attached to items.

The client is also able to receive both implicit and explicit feedback. We explain this in detail in Section 4.4.2.

4.3.3 Stages in the Recommendation Process

To better understand the way the system works we explain the process of what happens when a new user enters the systems for the first time, and how he receive recommendations. The flow of data is illustrated in Figure 4.1, where the box titled *Matrices* resides on the server, while the rest is a part of the client module.

The first step in using the system is accessing the client on a hand-held device. The device collects an updated Bayesian reasoning model from the server, as well as a list of all items and their features. In the figure this is visualised by the arrows departing from the server and into the Bayesian reasoning model and Item list. If the user is a new user he must submit some demographic information to the client application. A list of the information is presented in Table 4.1. The new user profile is then passed on to the server, and the user receives a unique ID.

When the initial set-up phase is completed, the user is ready to receive recommendations. The user can no go offline without losing the recommendation capability. The client application offers three categories of recommendation types: dining, accommodations and sightseeing. Say the user asks for a dining recommendation. All items are then passed on to the service filter of the client as the arrow from Item list to Service filter illustrates. The Service filter then filters out options not

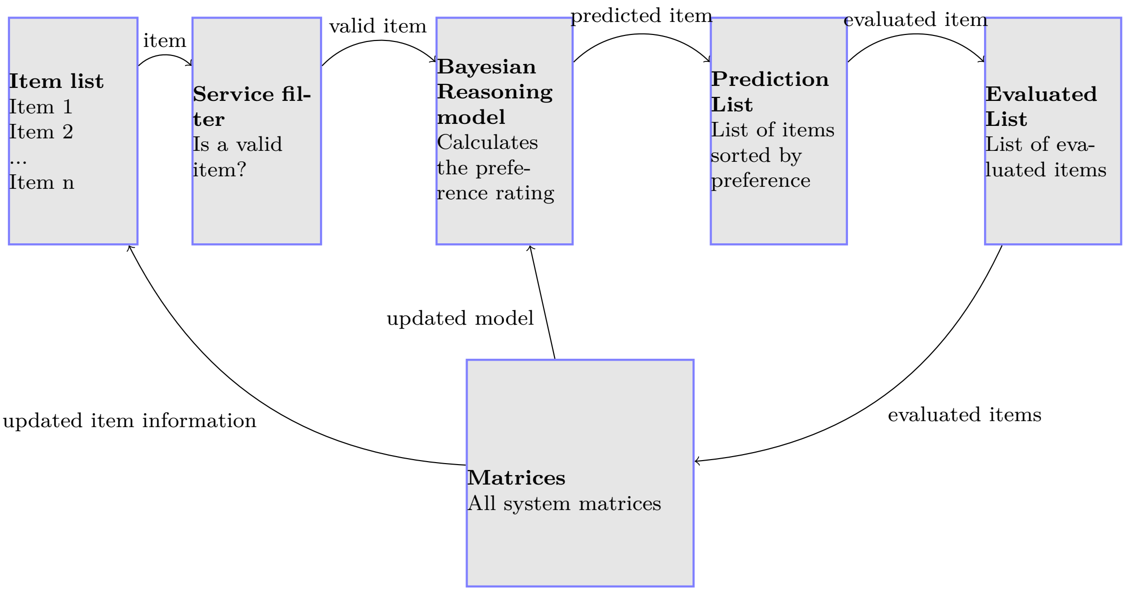

Figure 4.1: An overview of the recommendation process

considered relevant as explained in Section 4.4.2. If the item is considered valid, it is passed on to the Bayesian reasoning model as illustrated with the arrow from Service filter to the Bayesian reasoning model.

The model then returns a probability on how attractive the user will consider the item. All items are then passed on to a list that gets sorted after the highest mean probability. This list is the used to present the items with the strongest recommendations. The system then tracks the user's behaviour to implicitly and explicitly evaluate how the user rates the chosen item. The rating is then stored in a list, illustrated by the arrow from Prediction list to Evaluated list. The details about the feedback in the selection process are explained in Section 4.4.2.

When the user goes online again, the client uploads the ratings given to the server. The server collects all updated ratings, and updates the Bayesian reasoning model in batch mode. The client on the other hand receives the latest reasoning model and item list.

4.4 Design

In this section we present our design in a more thorough manner. We will present the server functionality, as well as each element in the client recommendation pipeline.

4.4.1 The Server

The servers main tasks is to collect and store user, item and rating data, and to provide the clients with updated Bayesian reasoning models. We will explain this in greater detail.

The Data Matrices

All information on the server is stored in a set of matrices. The different matrices are combined in order to create the Bayesian reasoning model that can be used together with item matrices without any server communication. The different matrices are made up of the following data:

User × Demographic Feature Matrix Each user is stored as a vector with the boolean information retrievable from Table 4.1.

Item × Content Feature Matrix Each of the boolean variables in Table 4.2 is stored as a vector for each item.

Item × Service Filter Matrix Each item is stored with some additional information that is used by the service filter, e.g. opening times to avoid recommending unavailable items to users.

User × Item matrix Since this matrix will be very sparse in our case, we store this as a list of lists, where each user stores a list of his ratings as a value pair(item,rating). This design solution is justified from the fact that the algorithm used to create the probabilistic reasoning model never needs to iterate over ratings on a specific item, but instead iterates over ratings from a specific user.

The server receives new ratings from the different clients whenever the clients choose to go online. The server stores the new ratings in the User × Item matrix, and updates the probabilistic model in batch mode. When existing users log on, they update their local probabilistic model and receive updated item matrices.

The two main advantages with decentralising the recommendation process is that it is possible to use the recommender system offline and that the system scales well as most computation is done on local hand-held devices. The system provides an interface to receive demographic information from the client, content features on items and additional information needed by the service filter for handling items. The output from the server which is sent to the hand-held devices is the item matrices and and the recommendation pipeline capable of making recommendations from the item matrices.

Generating the Bayesian Reasoning Model

The server's responsibility concerning the Bayesian reasoning model is to generate it for new and existing users, and to incorporate new information as more ratings enters the system. In this section we explain how the Bayesian reasoning model is generated and mathematics behind it.

The Bayesian model is based on the works of Chu and Park [2009], which was the solution we found to be the most fitting for our domain in the systematic literature review conducted in Chapter 3. However, we omit the dynamic user profile because users of our system will most likely never evolve into experienced users before leaving the city.

The Bayesian model consists of a D dimensional user profile $\mathbf{x}$, a C dimensional item profile $\mathbf{z}$ and latent features $\mathbf{w}_{ab}$ where a and b denotes user and item features. The weights $\mathbf{w}_{ab}$ are created in the same manner as Chu and Park [2009] demonstrate in their study through the following equations, where δ^2 denotes the *prior variance*, $\otimes$ denotes the *tensor product*, σ is the specified *noise level*, $\mathbb{O}$ is the set of *observations*, in our case ratings, and s_{ij} is the *bilinear indicator*:

$$s_{ij} = \sum_{a=1}^{C}\sum_{b=1}^{D} \mathbf{x}_{i,b}\mathbf{z}_{j,a}\mathbf{w}_{a,b} = \mathbf{w}'(\mathbf{z}_j \otimes \mathbf{x}_i) \tag{4.1}$$

$$p(r_{ij}|s_{ij}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(r_{ij}-s_{ij})^2}{2\sigma^2}\right) \tag{4.2}$$

The solution defines s_{ij}, as explained in Equation 4.1, to be the bilinear indicator that is the prior when calculating the probability that the item is a good choice. This can be carried out entirely by matrix operations. The continuous scores are then calculated as explained in Equation 4.2. Chu and Park [2009] also describes binary ratings, but since we want to grade our items we choose to only implement the part of the approach focusing on continuous data.

In order to gain any meaningful results from the system we need train the weights in accordance with the observations made. In this case the observations consists of the existing ratings in the system. If we now specify a standard Gaussian distribution of over $\mathbf{w}$ a priori, shown in Equation 4.3, and use the likelihood of seeing the observed data as explained in Equation 4.4, we can use this to adjust the weights since $p(\mathbf{w}|\mathbb{O}) \propto p(\mathbb{O}|\mathbf{w})p(\mathbf{w})$.

$$p(\mathbf{w}) = \frac{1}{\sqrt{2\pi}\delta} \exp\left(-\frac{\sum\limits_{ab} w_{ab}^2}{2\delta^2}\right) \tag{4.3}$$

$$p(\mathbb{O}|\mathbf{w}) = \prod_{\mathbb{O}\in ij} p(r_{ij}|s_{ij}) \tag{4.4}$$

We can then maximise the joint probability of $p(\mathbb{O}|\mathbf{w})p(\mathbf{w})$ by minimising the negative logarithm of the joint probability as defined in Equation 4.5.

$$\min_{\mathbf{w}}(\mathbf{w}) = \frac{1}{2\delta^2}\sum_{ab} w_{ab}^2 - \sum_{ij} \log p(r_{ij}|s_{ij}) \tag{4.5}$$

To adjust our weights, we use gradient decent, and the gradient $\frac{\partial L(\mathbf{w})}{\partial w_{ab}}$ is shown in Equation 4.6. Equation 4.7 is the logartihmic likelihood function, and the calculation of the first-order derivative is shown in Equation 4.8.

$$\frac{\partial L(\mathbf{w})}{\partial w_{ab}} = \frac{w_{ab}}{\delta^2} - \sum_{ij} \frac{\partial logp(r_{ij}|s_{ij})}{\partial s_{ij}} x_{i,b}z_{j,a} \tag{4.6}$$

$$\log p(r_{ij}|s_{ij}) = -\frac{(r_{ij} - s_{ij})^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma) \tag{4.7}$$

$$\frac{\partial logp(r_{ij}|s_{ij})}{\partial s_{ij}} = \frac{s_{ij} - r_{ij}}{\sigma^2} \tag{4.8}$$

When the process has reached a sufficient level of precision, the Bayesian reasoning model, which is stored as a matrix, is ready to be exported to the clients.

4.4.2 The Client

The three stages in the client recommendation process are the service filter used to make sure irrelevant items are not recommended, the Bayesian reasoning model which predicts user satisfaction and the selection stage where items are ordered and presented to the user.

The Service Filter

The purpose of the service filter is to filter out irrelevant items, as well as avoiding sending tourists to closed attractions.

Since we have to do the filtering process at some stage in any case, we do it before the evaluation process to avoid that irrelevant items are evaluated, and therefore wasting resources. Tourists ask for recommendations among three different services: eating, sightseeing and sleeping. They may also choose to set a distance limitation. An item is allowed to pass on the reasoning model if the item contains any of the features considered relevant for the selected activity.

The relevant features are stored in vectors consisting of boolean values. Since item features also are stored in boolean values, this makes it possible to do the filtering in a single vector multiplication. Let $\mathbf{z}_i$ denote the item feature vector for item i, and $\mathbf{f}_e$ denote the filter for food services. Finally, r is the boolean variable that denotes whether the item is relevant or not. To consider if a item is relevant, we simple use this formula: $r = \mathbf{z}_i \times \mathbf{f}'_e$. In addition we do search on location by calculating distance and comparing it with the user specified radius and filter out items that are already visited by the user.

Using the Bayesian Reasoning Model

The Bayesian reasoning model on the client evaluates all the relevant items that has passed through the service filter. The probability is determined by inserting the item features into the model, as well as the consistent local user features. The system is visualised in Figure 4.2, where x_b denotes the user features, z_a denotes the item features, s_{ij} denotes the bilinear parametric indicator between the item and the user, and the probability distribution is calculated with the formula $p(r_{ij}|s_{ij}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(r_{ij}-s_{ij})^2}{2\sigma^2}\right)$.

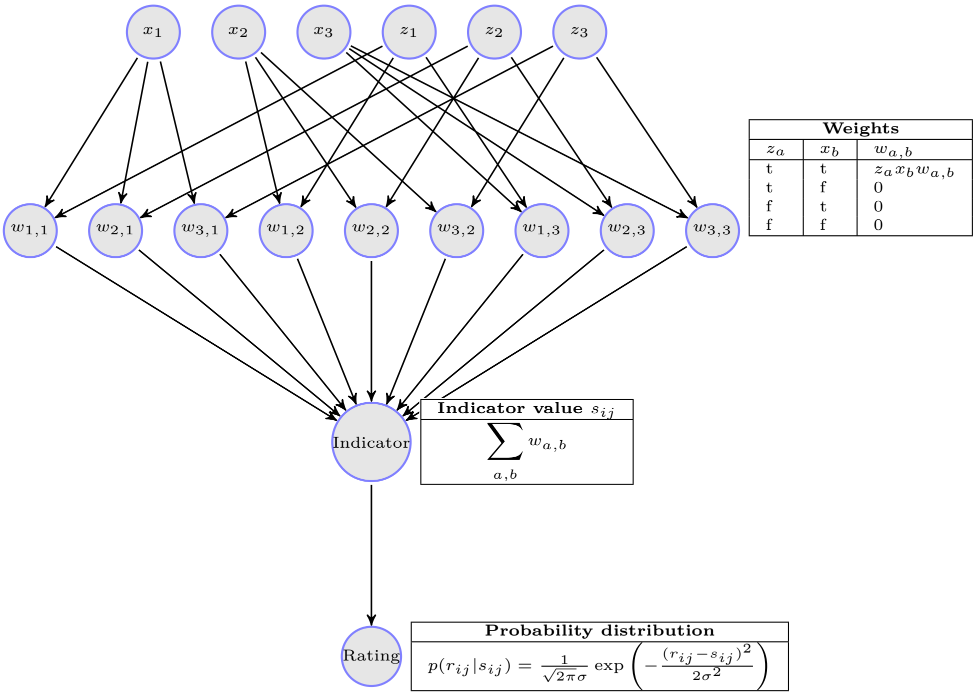

Figure 4.2: A simple graphical explanation of our approach using three user features x_1, x_2 and x_3, and three item features z_1, z_2 and z_3.

The output of the Bayesian reasoning model is a set of items with a Gaussian probability distribution. We sort the list with respect to the highest predicted value. The list is then passed on to the selection stage.

The Selection Process

After receiving the list from the reasoning model, the recommendations are presented on the map as icons. The n most probable recommendations are presented, and they are marked with colour codes where a strong green indicates a strong recommendation, and a soft green indicates a weaker recommendation. The client selects a recommendation by simply physically moving to the location.

To see how popular the recommendation was we have two options, either ask the user to explicitly state how well he liked the recommendation, or implicitly by observing the users actions. Explicit feedback is easy, and only requires the user to give a rating on an item. We have chosen to normalise all feedback to continuous range [0-1], but users give feedback on a discrete scale from [1-5]. Users are tracked when moving, and the system assumes a selection if the user moves to a recommendation. The user is then asked to rate the item. If the user choose not to rate the item, we depend on implicit feedback to evaluate the recommendation.

Implicit Feedback

Because of the location-awareness we have the opportunity to rate user satisfaction on recommendations without asking him explicitly. We have chosen to take advantage of this, and we have made the following rules when it comes to rating by implicit feedback, and it is based only based on common sense since little empirical evidence is available to support our case. We make the assumption that if someone choose not to give feedback on a restaurant experience, it is because it is rather ordinary, meaning not particularly bad or good. If someone visits a restaurant and stays more than twenty minutes, and later choose not to give feedback, we rate the experience with the average score. The same is the case for nightclubs and pubs, where short stays also may indicate that the place was full, or that some friends simply went to another location.

At museums and landmarks we choose another approach. We believe that tourists are more likely to stay longer if they are having a positive experience. We rate their stay with a continuous scores given by using the cumulative distribution for the normal distribution function where $\mu = 0.5$ and $\sigma^2 = 0.2$ and where we scale the time linearly so that 20 minutes = 1.0 for landmarks, and 90 minutes = 1.0 for museums. Values above 1.0 is set as 1.0 and below 0.0 is set as 0.0.

4.5 Testing

As was defined in Chapter 1, the focus and scope of this part of our thesis has been to propose a *design* of a recommender system for tourists visiting Trondheim. This means that integration and testing of the design falls outside the scope. However, in this section we give some thoughts on how the tests can be carried out.

4.5.1 Testing

In order to determine the effectiveness of our approach we want to test our system with regard to following specific criteria:

1. How does the system predict in a cold-start system situation.

2. How does the system prediction ability evolve.

3. How does our system compare against other tried approaches.

4. How does context affect prediction ability.

Unfortunately is carrying out the test beyond the scope of the project, but a test plan is provided in this section.

We would like to test the system on two levels, both quantitative and qualitative. The first is measuring the performance accuracy on a standardised data set to compare our solution with similar solutions available. The other is a case study on real tourists with qualitative feedback from the test group.

Comparative Systems

In order to have a reference on how well the system performs it would be worthwhile to compare its performance against a set of other well-known approaches. More specifically we would like to test against the following systems that are all simple to implement into the existing framework:

Random
: Randomly assigning ratings to items.

Segmented most popular
: Rank the items after popularity within a user segment divided by demographics.

Naïve Bayesian classifier
: The naïve Bayesian classifier is tightly related to our approach, and a comparison of the performance development would therefore be interesting.

Experimental Design and Metrics

To measure the performance we will use mean absolute error across all ratings per update iteration of the probabilistic framework as explained by Equation 4.9, where u is a user, $\mathbb{O}_{u,}$ is the set of ratings belonging to user u that is made by the system in iteration g, n is the total number of ratings, $p_{u,i}$ is a rating prediction for item i to user u and $r_{u,i}$ is the submitted rating for item i to user u. This will make it possible for us to see how the performance of the system evolves for each generation of the system more information is entering the system.

$$MAE_g = \frac{1}{n} \sum_{u}^{users} \sum_{r}^{\mathbb{O}_{u,g}} |p_{u,i} - r_{u,i}| \tag{4.9}$$

The quantitative design will use the standard 100,000 data set from MovieLens. We will do two tests with this data set, the first one being a standard five-fold cross validation test where we compare our solution with the comparative approaches.

In the second test we will randomly remove ratings from users so that all users are left with five ratings. This leaves us with 4715 ratings in total. We then remove additionally 15 random ratings, leaving us with 4700. We will then, by random, add ratings to the system. If the user has not rated anything earlier, he will be added to the system, if an existing user adds a rating, this will be added on the existing user profile. For each 100 ratings we will measure the mean absolute error of the last 100 ratings as explained by Equation 4.9. The procedure is repeated in 47 iterations. The motivation behind the test is to measure cold-start situations in an environment with only cold-start users, as well as measuring the system's ability to learn.

Qualitative Experiment

To measure the qualitative performance of the system, we would like to use visitors that arrives in Trondheim with Hurtigruten. Over a seven day period we would like 10 tourists to use each of the comparative solutions, as well as our own. By that we mean 10 users testing our approach, 10 different users testing the random approach, and so on. At the end of the day we want them to rate the experience on the following criteria:

1. On a scale 1-5(best): How would you describe the quality of the recommendations?
2. On a scale 1-5(best): How would you describe the effort needed to begin using the system?
3. Did you ignore any of the recommendations?
4. Would you consider using such a system in the future (Yes/Maybe/No)?

The reason we want to test it for a period of a full week is to test different context situations, to see if the experience increases over time, and to get a broader range of test users.

The tests are to be carried out when the system is completed.

4.6 Analysis

In this section we analyse our design in context of the goals and requirements for our recommender system for tourists which were specified in Section 4.1 and Section 4.2. We point out the relation between requirements and parts of the design, and explain the choices made in forming the design. Finally, we share our thoughts on the cold-start situation and on the difficulties of using implicit feedback.

4.6.1 Design and Requirements

The requirements are met in the following ways:

R1 The system *must* provide personalised recommendations for cold-start users.

The cold-start user problem is handled by implementing a Bayesian reasoning model that was discovered while conducting the systematic literature review. Ideally, a cold-start user should receive recommendations that appear personalised starting from the first recommendation. To test the real effectiveness we need to test it on a sufficient number of users. A test plan is presented in Section 4.5.

R2 The system *must* require little initial effort from cold-start user before being ready to provide recommendation services.

The user only needs to input six demographical data: age, gender, occupation, type of holiday, nationality and budget. We consider this a small user effort. The system provides an interface with support for demographical data that is easy to extend with new fields. At the moment we use the fields mention in Table 4.1, but since, to the best of our knowledge, a limited amount of research is conducted on what demographic data is most effective in segmenting users into homogeneous populations, these may change as experience is gained.

R3 The system *must* handle items tagged with feature information and information used by the service filter.

The system provides an interface with support for inserting new items with features into the system, as well as updating the existing items with new content information. Also this implementation is flexible since we might want to add additional services later, or discover new content features that add value to the recommendation quality.

R4 The system *must* offer recommendations for different services:

(a) Sleeping accommodations
(b) Dining opportunities
(c) Sights and activities

The system provides this functionality by using filters the removes irrelevant items before they enter the Bayesian reasoning model. New filters can be added as well, making it easy to further expand the set of services.

R5 The system *should* take the following contextual parameters into consideration when recommending items:

(a) Location of user
(b) Time of day
(c) User's visit history

The system is able to reject items that are irrelevant because of contextual issues before they enter the Bayesian reasoning model. Information about context is gathered from the users' hand-held devices, and the service filter is able to calculate whether an option is unavailable.

R6 The system *should* provide compatibility with the mobile information system for tourists presented by Wium [2010].

The system should be ready to be integrated as a part of the information system. However, the implementation process is left as further work.

R7 The system *should* support both implicit and explicit feedback from users.

The system handles both implicit and explicit feedback. The value of implicit feedback is uncertain, but it is probable that the parameters for how implicit feedback should be handled needs to be tuned as more empirical data is collected. The explicit feedback consists of giving a rating between 1 and 5 on a visited item.

R8 The system *should* provide learning based on Bayesian reasoning.

The system uses a Bayesian reasoning model based on the works of Chu and Park [2009].

R9 The system *should* be designed to run on a hand-held device.

We plan to integrate the system the information system provided by Wium [2010] that runs on the Android platform.

4.6.2 The Cold-Start Situation

Even if the cold-start user and cold-start item problems are properly handled, we still have the issue of cold-start system. A way to reduce this problem is to insert pseudo users based on statistics that are available from Statistics Norway[2], however, we still need to do some assumptions when it comes to who would like what. We could also gradually phase out the pseudo users as new users enters the system. There are six different features the user has to specify to start using the system: age, budget, gender, type of holiday, nationality and occupation. For the age, gender and nationality features there are available statistics that we can take advantage of. We assume that gender, nationality and age are independent of each other. When it comes to budget we find it reasonable to believe it depends on age and occupation, and occupation and type of holiday depends on age as well. We then combine ratings from local newspapers and our own prejudices to make a rating algorithm that rates items in accordance with user features. We also add a noise factor into the rating equation. We have implemented this approach based on these assumptions, but the effectiveness is yet to be determined since we have not tested the system yet.

There are some obvious weaknesses in our simple approach, mainly the uncertain assumptions we make in creating the pseudo users, and the biased rating algorithms used to create ratings. It is therefore suggested that these users are phased out when a sufficient amount of real users have entered the system.

4.6.3 Implicit Feedback

The process of implicit feedback can be carried out by simply measuring how long a user stays at a recommended activity. However, there are some flaws with this approach. As an example, if a user is dining at a fast-food chain for a very long time it may be because he is getting poor service and has to wait an extended period to get his order. On the other hand, the user may very well enjoy the stay, and decide

[2]http://www.ssb.no/en/

to order desert as well. Since it often is impossible to differentiate between the two cases by just measuring the length of stay, we can not give a high rating, nor a low rating, based just on the observed behaviour. On the other hand, if a user spends a long period of time at a landmark he probably finds it interesting, in other words, the nature of the service has an effect on the value of implicit learning.

4.7 Summary

In this chapter we have presented a location-aware Bayesian recommender system for tourists visiting Trondheim. We have focused on making a system that creates good recommendations even for cold-start user and requires little user effort to start using. We have placed no emphasis on personal learning since a tourist will not be able to utilise it due to the short user time, but a stronger emphasis on the global learning valid for all users since this helps us provide new users with stronger initial recommendations.

To achieve our goals we used the findings from Chapter 3 to find a solution capable of making good recommendations even in domains with a high level of cold-start users. We adjusted the design to fit the tourist domain, and made the system possible to integrate with an existing system provided by Wium [2010]. This makes the recommender system capable of making recommendations on a set of services, and it is possible to use implicit feedback from the users as well.

We have split the system in two modules, client and server, and made it possible for the client application to work offline. This increases scalability drastically. We have also developed tests that we would like to carry out once our solution is fully implemented.

Chapter 5

Discussion

In Chapter 1 we introduced the two main goals of our Master's thesis:

G1 Find the existing solutions to the *cold-start user problem* presented in the recommender system literature.

G2 Design a Bayesian recommender system for tourists visiting Trondheim.

In the first two sections of this chapter, Section 5.1 and Section 5.2, we will go back to these goals. We will in the light of what has been presented in Chapter 3 and Chapter 4 discuss whether we have succeeded in reaching our goals.

Readers should note that we have already performed detailed analyses of the results of our work with the two goals in Section 3.4 and Section 4.6 respectively. Therefore, the discussion presented here does not go into details of the previously presented results, but argues on a high level whether we have succeeded in reaching our goals.

In the final section of this chapter, Section 5.3 we discuss our experiences with performing a systematic literature review focusing on two aspects:

1. Performing a systematic literature review *of artificial intelligence literature.*

2. Performing a systematic literature review *as part of a Master's thesis.*

5.1 G1: Solutions to the Cold-Start User Problem

In this section we focus on whether we have succeeded in reaching our first goal of finding the existing solutions to the cold-start user problem for recommender systems using Bayesian reasoning. As was stated in Section 1.2, this goal can be parted into two sub-goals: finding existing solutions to the cold-start user problem *in general for any domain*; and specifically for recommender systems operating in the *the tourist domain.* We use this ordering when we now discuss whether each of the sub-goals have been reached.

Through our systematic literature review, we have identified *nine* different solution types to the cold-start user problem in recommender systems using Bayesian reasoning. However, a question that should be asked is whether any solution types exist which we have *not* discovered in our review process.

An issue which could have influenced the results of our review is *researcher bias.* In our case a problem could have arisen for example if any of the researchers had preferences for one of the solution types before the review process begun, or developed such preferences during the work. Researcher bias could have resulted in relevant primary studies not being found or being rejected during the search and selection stages, when they should have been passed on to the next stages. In our systematic literature review we guarded ourselves against researcher bias by developing the *review protocol* which fully specified all procedures before they were used.

Another essential part of finding the existing solutions to the cold-start user problem was having a *search strategy* that was able to retrieve all relevant studies. The first part of the strategy was the set of sources to be searched through. We can not exclude that sources exist where we could have discovered relevant studies that are not included in our review. However, we do believe that our list includes all sources where recommender system literature is likely to be published. Hence, the vast majority of relevant studies for our review should have been available through these sources.

The second part of the search strategy were the specific procedures used to search the various sources. When manually searching through journals and conference proceedings, the responsible researcher browsed titles, and if necessary, abstracts to find relevant studies. In this search we had an inclusive policy which included studies when we were in doubt if they were of relevance to the review. We believe that this procedure was satisfactory, and retrieved all the relevant studies from those sources thanks to the aforementioned inclusive policy.

In the digital libraries we performed advanced searches using a search string combining four groups of search terms as was specified in Section 3.2.3. We have identified two possible weaknesses in this procedure which are related to the *third* and *fourth* group of search terms.

The third group of search terms is the *user modelling*-group. We included this set of terms to ensure that the studies retrieved in the search somehow built a profile or model of the user to achieve personalisation of services. We have later realised that almost all studies use some kind of user profile or model in its reasoning. Sometimes this is limited only to the historical ratings given by the user, but studies using this sort of profile may also have contributed solutions or solution types to our review. However, the *user modelling*-group includes five variations of the user modelling-term and we believe that most of the studies relevant for our review do include at least one of the different terms in the group.

As we pointed out in Section 3.4.5, studies using probabilistic approaches might not mention the word *Bayes* or *Bayesian* explicitly. These were the terms required by group *four* to be included in studies for them to be found during our search of digital libraries. We acknowledge that some probabilistic solutions which fall within the limits of our *Bayesian reasoning*-constraint may have been missed in the search stage due to this weakness.

That being said, we believe that the latter issue is independent of *solution types*. What separates the solution types is not related to how systems implement their *reasoning*, but is rather related to ideas of how extra data such as tags, demographic information or trust-relations can be included to mitigate the cold-start user problem. Because of this observation we consider it unlikely that we have missed out on important solution *types* even though single studies may have been missed. Solution types documented with several studies such as the *demographics* and *fast learner* types are likely to have been found regardless of the constraint put on the search by the *Bayesian*-group of search terms.

The final stage where solution types could have been missed is the *study selection* stage which was described in Section 3.2.4. Similarly to the previous stages, also the procedures used in this stage were carefully specified before we began executing them in order to avoid biased assessments. We also let the assessments be guided by an inclusive policy, similarly to what we did in the manual search of sources. This meant that studies which left the researcher in doubt were always passed on to the next stage. Considering that we used predefined procedures, and also were inclusive in assessments, we believe that chances are low that any solution types were incorrectly filtered out in this stage of the review process.

After mapping the existing solutions to the cold-start user problem independent of domain and classifying them into *solution types*, we turned to our second sub-goal which was to evaluate the solutions found in the context of Bayesian recommender systems for tourists. To reach this sub-goal we analysed the nine solution types focusing on the special requirements of high initial recommendation service quality and little initial user effort. This analysis, which was presented in Section 3.4.4, concluded that using *demographic user data* was the best solution type of the nine. However, it should be evaluated to which degree this conclusion holds.

First, it is important to stress that the conclusion is only valid within the scope of our review. Similarly to the domain-independent problem discussed previously in this section, we can only conclude that the *demographics*-solution is the preferred one, given that we are focusing on recommender systems for tourists which use Bayesian reasoning.

Neither can we exclude that there might be some criteria for evaluating a recommender system for tourist that we have not taken into consideration. Our analysis of solutions for tourists includes the aforementioned special requirements for the tourist domain, but these are based on our own evaluation of the tourist domain, not scientific sources. There might exist such sources, or at least gathered statistics from the tourist industry, which offer additional requirements that should be used when evaluating how to best create recommender systems for tourists.

Even though we have in this section identified two possible weaknesses of our search strategy, we believe they are minor issues which did not significantly influence the outcome of our systematic literature review. This means that we consider it unlikely that many solution types for the cold-start user problem exist which we have not discovered. We also conclude that it would be interesting to more formally investigate requirements for recommender systems operating in the tourist domain. If such requirements are found, it should be tested that the solution type we have found to best in the domain, using *demographic user data*, is in accordance with these requirements.

5.2 G2: Design of a Recommender System for Tourists Visiting Trondheim

In this section we focus on how we fulfilled the goal of designing a Bayesian recommender system for tourists visiting Trondheim. For a deeper analysis of the design choices and how these goals were met in detail, we refer to Section 4.6.

When designing the recommender system for tourists we had to take into consideration factors such as context dependencies, need of good initial experience and few learning opportunities. These restraints makes traditional collaborative filtering challenging, since it depends on a sufficient amount of training data to provide good recommendations, and is typically used without context parameters. Another approach is content-based filtering, but this also has limitations like important tags may be highly subjective to user opinion, no possibilities of providing serendipitous findings, and questionable level of initial user effort if we want to avoid the cold-start user problem.

The quest for this answer made us carry out a systematic literature review where we found Bayesian demographical solutions to be the best fit. The reasoning for this is the low user effort required, possibilities of hybrid solutions, and acceptable initial user experience. Even though a case study is needed to evaluate the effectiveness of our approach, it does provide an instant recommendation capability, takes advantage of previous demographically similar users experience, is able to determine if activities are unavailable due to context restraints, and provides an easy to expand set of services. We also propose to reduce the cold-start system problem by algorithmically adding pseudo users.

An important note on the tourist domain is that the possibilities of learning to know a person is very limited given the typical stay, especially in city breaks that are typically shorter than charter holidays. This makes it less important to provide a system that gives improved personalised recommendations for users after a substantial amount of ratings, but it makes it important that the Bayesian reasoning model evolves successfully over a long period of time in order to provide cold-start users a high initial quality of service. This is why we have chosen a solution without learning on the client side and a dynamic solution that evolves over time on the server side, which also makes it possible to make the recommender system work offline without privacy concerns.

5.3 Performing Systematic Literature Reviews

In this section we will discuss different aspects of performing a systematic literature review. We will first discuss how well the systematic literature review method is suited to review artificial intelligence literature, before we discuss our view on performing systematic literature reviews as part of Master's theses.

Both discussions are primarily based on our own experiences and insight gathered throughout the work with this thesis, and are hence qualitative in nature.

5.3.1 Reviewing Artificial Intelligence Literature

The time and effort needed for a systematic literature review far exceeds what is needed in unsystematic reviews, and the question researchers need to ask themselves before commencing such a review is whether the problem at hand calls for such a review. Note that the term *problem* spans widely in this context, and that one such problem might be to identify areas of research not yet covered with sufficient evidence e.g. in machine learning.

In a systematic review, more than one researcher is desirable to avoid researcher bias. If a researcher does the review solitary, he should take extra steps to ensure good quality, either by discussing the included and excluded papers with an expert panel, or by applying a test-retest approach. In most cases researchers end up with duplication of work. Also, the whole process needs to be properly planned in order to make sure all relevant literature is collected and again to avoid bias in the conducting phase. The nature of search also makes it highly probable that much time will be used on screening of papers of low relevancy to the problem.

However, this also highlights some of beneficial features of systematic literature reviews. As pointed out by Kitchenham [2007], too much of the research in computer science is performed without properly evaluating what has already been done in the field. The value of having good reviews which other researchers can use as a solid starting point for their research, must not be underestimated. Using a systematic framework to perform a review is the best way of ensuring that the review will be of high value to the research community.

The results of a systematic review may be of great value to a research community, but an additional positive trait is that performing a systematic review in itself is a way of getting valuable and deep insight to a domain for the researchers performing it. In our case we experienced that performing each of the stages from the abstract screening to the reporting stage lead to an increased understanding of the cold-start user problem in particular, but also many other topics related to recommender systems.

What must be weighted against the effort needed to perform a systematic literature review, is the value of the results and evidence produced by such reviews, in addition to the value of performing the review for the research team. In order for a systematic review to be justified, our opinion is that the problem which is target for the review should be a problem which is important to solve in order for the research domain to progress. In medicine, such situations regularly arise because humans may be affected by unwanted side-effects of treatments and drugs. When problems arise in computer science generally, and perhaps artificial intelligence specifically, we should always ask ourselves whether the problem justifies the effort needed to perform a systematic literature review.

If the decision is made to perform a systematic literature review, it is crucial that the review questions are carefully designed. The review questions will indirectly or directly set the scope of the project. Setting the scope too wide will usually result in a large increase in effort needed to perform the review because the result of an increased scope is reflected in a wider search, which again results in more studies returned which need to be assessed in the various stages of the review process. However, if the scope is too narrow, the result may be a review of limited value because the problem is rarely met in the domain or is generally not considered to be of high importance.

In our review, we chose to set a relatively narrow scope. This was because of the limited time we had available to perform our review. The result of narrowing the scope was that we had to define two additional constraints on the population of systems which we wanted to study in search for solutions to the cold-start user problem. In our case, it was necessary to add these constraints, but we believe that our review could have been of even higher value to the recommender system community if we had the time available to perform our review without those constraints. However, it is important to note that the value of the review as a learning process for the authors of this thesis, would probably not have been significantly higher if the constraints were dropped.

When sharing our experiences with systematic literature reviews in artificial intelligence, we would like to emphasis the amount of knowledge we gained through the process. Both specific knowledge centred around the research questions, but also more general knowledge of the recommender system domain. The more general knowledge acquired helped us when we designed the system, and also helped us better understand the more specific cold-start user problem.

We also experienced the struggle that followed due to the lacking standardisation of keywords and experiments in the artificial intelligence field. Even though that was frustrating and took a toll on the morale, it did make us more aware of the value of following stricter scientific methods in research. We value the outcome of the review as high, as the results found could be implemented fast and we knew it would work right away.

We conclude that researchers considering to use a systematic literature review to solve a computer science problem or even more specifically, a problem within artificial intelligence, should analyse the importance and severity of the problem, and the potential value of the review both to the research community and the research team themselves, before choosing such a rigorous method. If the decision is made to perform a systematic literature review, the researchers should put considerable effort into setting the scope neither too wide nor too narrow in order to secure that the review will be possible to perform within budgeted hours and be of high value to the research community.

5.3.2 Systematic Literature Review as Part of a Master's Thesis

The guidelines for systematic literature reviews in software engineering presented by Kitchenham [2007] state that such reviews are appropriate for *software engineering researchers, including PhD students*. Because of the many advantages of systematic literature reviews, we decided to perform one as part of our Master's thesis. We here discuss whether systematic literature reviews are suited to be performed not only by professional researchers and PhD students, but also by Master's degree students.

Whenever a research team decides to perform a systematic literature review, they are likely to gain knowledge of the domain while performing the stages of the review. However, professional researchers and PhD students are likely to initially have a better insight into the relevant domain than students on the Master's level. Kitchenham does not give any particular reasons why the guidelines presented are aimed at professionals and PhD students, but we believe that one of the main reasons is that the planning of a review becomes easier with more knowledge about the domain.

According to the guidelines, a full review protocol should be defined initially before the search for relevant literature begins. This means that research questions, search strategy, inclusion and exclusion criteria, quality assessment criteria and data to be collected in the data collection phase should be specified during the initial planning phase. We attempted to use this approach, but found it challenging to specify procedures for the later stages of the review at that point. As a result, the number of adjustments in the protocol increased at the later stages. It should be noted that iterative refinement of each stage before executing is mentioned as a valid strategy by Kitchenham [2007].

Another reason for aiming it on PhD students and professionals is that it is a time consuming process, and will inevitably take up a large portion of the total time available. Both the process of making the protocol and learning the methodology require some initial effort, but especially the selection phase, quality assessment and data collection has the potential of consuming months of effort. Hence, if a systematic literature review is chosen as a tool, one must expect that a substantial part of the thesis will be the outcome of the review.

However, as we pointed out in Section 5.3.1, performing a systematic literature review is almost guaranteed to give valuable learning and insight to the domain which is studied. For most Master's theses, the main goal is to perform a piece of research which drives the research field further, but in many cases a Master's thesis can also lead to a continued research career, e.g. as a PhD student. In these cases, performing a systematic literature may be an ideal way of both contributing to the research field through the thesis itself, while personally achieving thorough knowledge of the field which will be of great benefit for the further research.

As was discussed in Section 5.3.1, we became aware during our review process that the scope of our review was probably too narrow to fully exploit the effort put into it. This illustrates one of the weaknesses of the method in general, and especially when used by students on lower levels than PhD. If any of the stages are not planned or performed in an ideal manner, it soon becomes costly or unmanageable, both in work hours and morale, to go back and redo one or two stages. In our case we did a minor error in our initial step when we defined our research questions. When we realised this while screening the primary studies, it was already too late to go back and redo our work because of the limited time allowed for our thesis. We believe that the problem of exact planning and performance of steps is a general weaknesses of the systematic literature review method, as it is to many rigid frameworks based on initial planning, but also one that is all the more visible when such reviews are performed by researchers less knowledgeable in the research domain and with little experience from performing systematic reviews.

Chapter 6

Conclusion

In this thesis we have first, through a systematic literature review of recommender system literature, identified *nine* types of solutions to the cold-start user problem in recommender systems using Bayesian reasoning.

We have evaluated the solution types with respect to initial user involvement, learning ability and initial quality of service, and found that the the most obvious differences between the types were in how much effort they required from users in order to overcome the cold-start user problem. We also evaluated the strength of evidence put forward for each of the types, and concluded that given the evidence, most of the solution types should for now be considered promising solutions requiring further research.

As part of the systematic literature review, the nine solution types were also evaluated in the context of recommender systems applied in the *tourist domain*, where users are in need of recommendations of high quality from the very first use. We found that using *demographic user data* is the best of the solution types for this domain because it does not demand much effort from users, and offers instant personalisation of services.

The key findings from the systematic literature review were then utilised when we presented the design of a location-aware Bayesian recommender system for tourists visiting Trondheim.

The proposed design uses historical rating data from users to build a Bayesian model where latent variables reflect correlations between demographic information about users and content information about items. When the model is used in the recommendation process for a target user, it calculates a probability for each relevant item which reflects the chance that the target user will enjoy that particular item.

Because the only input needed from new users before they can receive recommendations is six types of demographic data, the design readily handles the cold-start user problem at the same time as it does not require much initial user effort. To further personalise the user experience, the design offers *context-awareness* as it exploits variables such as current user location, time of day and previous visiting history to filter out items which are not of interest to the target user.

6.1 Further Work

Our primary goal for further work is to complete the integration of our system into the mobile information system framework provided by Wium [2010]. When this is completed we wish to test the complete system with a double-blind case study as explained in Section 4.5.

When the system has been used for a sufficient period of time, we would also be interested in analysing the latent variables computed in the Bayesian model by the system. By studying these variables we can discover which demographic information that is most valuable for segmenting tourists into groups. This could lead to new insight which can be used to further improve the recommendation quality of the system.

A third path for further research is to explore how we can utilise context-awareness and location-awareness to a higher degree. For instance, the system could itself initiate recommendation of items on the fly as a user takes a walk in the city, or use weather-awareness to recommend a certain café that is popular when the sun is shining. By utilising an extra context dimension in the system we could calculate the correlations between different items in different conditions for different demographics. A possible challenge with this approach that must be further researched is how the increase in sparsity, as an extra dimension is added, will affect recommendation quality.

The need for further studies on how *implicit feedback* can improve recommendations for mobile recommendation systems is also apparent. If it is possible to reliably use implicit feedback to reason about users' preferences on items, it could increase both recommendation quality and usability of recommender systems. We consider this an interesting path for further research.

In process of evaluating the different options on how to handle the cold-start user problem we had to make assumptions on how much effort a new user is willing to go through in order to reap the benefits of personalisation. We would like to further investigate this in order to find the ideal amount of initial interaction between user and system which: allows the system to achieve high initial precision on recommendations; and does not deflect users from using the system.

A final path we point our for further research, is to define a standard for experiments measuring the performance of recommender systems on cold-start users. The lack of common experimental procedures makes it difficult for the research community to review and compare different studies, as we pointed out in Section 3.5.3.

Appendix A

SLR Protocol

This is the full version of the systematic literature review protocol described in Section 3.2. The protocol was originally named *Protocol for a Systematic Literature Review of Recommender System Literature Focusing on the Cold-Start User Problem* and was written and used by this thesis' authors to guide the systematic literature review process.

A.1 Background

This systematic literature review is part of our Master's thesis, which is being written throughout the spring 2010 semester at NTNU (Norwegian University of Science and Technology). The main goal of the thesis is to create a recommendation system prototype, helping tourists visiting Trondheim. The system will be using information about the user's location as well as a dynamic profile of the user, to recommend restaurants, sights, hotels, transport and possibly other things of interest to the user. The system will use Bayesian reasoning to generate recommendations.

A common issue in designing recommendation systems is how to solve the *cold-start user problem*. The cold-start user problem is the problem of reasoning about the preferences of a new user who has provided very little, or no, information or feedback for the system to use in its reasoning. Tourists are likely not to stay in a city for very long, and it will be crucial for a tourist recommendation system to provide recommendations of relevance to the user from the moment the user starts moving around in the city.

The first objective of this systematic literature review is to search artificial intelligence literature in a structured way to find out how other researchers have solved the cold-start user problem in similar applications. The second objective is to explore the solutions and evaluate the research and evidence backing them, in order to find out which solution we wish to adopt in our tourist recommendation system.

Source	Responsible
ACM Digital Library	Lillegraven
IEEE Xplore	Lillegraven
ISI Web of Knowledge	Lillegraven
ScienceDirect	Lillegraven
CiteSeerX Beta	Lillegraven
International Joint Conference on Artificial Intelligence	Both
Journal of Artificial Intelligence Research	Both
SpringerLink	Wolden
Wiley Inter Science Journal Finder	Wolden
International Conference on Machine Learning	Wolden
Conference on Uncertainty in Artificial Intelligence	Wolden

Table A.1: Digital libraries and Conferences, and the person responsible for the search.

A.2 Research Questions

RQ1 What are the existing solutions to handle the cold-start user problem in recommender systems using Bayesian reasoning?

RQ2 How do the different solutions, found through the work with RQ1, compare to each other with regards to initial user involvement, learning ability and initial service quality?

RQ3 What is the strength of the evidence in support of the different solutions?

RQ4 What implications will these findings have when creating a recommender system for tourists?

A.3 Search Process

The search strategy involve searching through digital libraries, journals and conference proceedings in a systematic manner. The sources that will be searched and the responsible researcher is shown in Table A.1. The list of sources has been gathered through studying articles about computer science systematic literature reviews [Dybå and Dingsøyr, 2008, Kitchenham, 2007].

Based on our research questions, we have produced a set of search terms. We have put these terms into four groups, as shown in Table A.2. Each group consists of terms that are either synonyms or can be said to have related semantic meanings.

We do full text searches of the digital libraries using combinations of terms from the four groups. Each search consists of one term from each group combined by the logical 'AND' operator. This gives us a set of 60 search combinations, which we include in one search string, by using the logical 'OR' operator. This is illustrated in the formula below.

	Group 1	Group 2	Group 3	Group 4
Term 1	Cold-start	Recommender systems	User models	Bayes
Term 2	Sparsity	Collaborative filtering	User modeling	Bayesian
Term 3		Content-based filtering	User modelling	
Term 4			User information	
Term 5			User profile	

Table A.2: The four groups of search terms which we combine in our digital library searches.

$$([G1, T1]\ OR\ [G1, T2])\ AND\ ([G2, T1]\ OR\ [G2, T2]\ OR\ [G2, T3])$$

$$AND\ ([G3, T1]\ OR\ [G3, T2]\ OR\ [G3, T3]\ OR\ [G3, T4]\ OR\ [G3, T5])$$

$$AND\ ([G4, T1]\ OR\ [G4, T2])$$

For the *ISI Web of Science* library, where full text search is not available, we use topic search where we combine *cold-start* and *sparsity* with *recommender system*. These search strings are more general than the search string we use in the other libraries. This will hopefully compensate for the lack of full text search, and retrieve all texts of interest in the library.

In addition to digital libraries, we will search specific conference proceedings manually (see Table A.1). In this process the responsible researcher will look primarily at titles of texts, but will use abstracts when studying the title leaves the researcher in doubt if the study should be included in further stages.

All the texts found through our search phase will be included in an *EndNote* library which we use as a starting point for out filtering process. Finally, before starting the filtering process, we remove duplicate texts and texts that were published before 1st of January 2000.

If we find several versions of the same article (or about the same study subject), e.g. one conference proceeding, and one journal article, we keep the one that was most recently published which is assumed to be the most complete text.

A.4 Study selection process

After searching through the sources as specified in Section A.3, we will filter down the set of studies in a three-stage process. The first two stages are thematic inclusion criteria filtering stages determining the relevance of studies and are explained in the remainder of this section. The final stage is a quality screening which is described in Section A.5.

A.4.1 Inclusion Criteria Filtering

We will first filter out studies by reading abstracts to discover whether studies meet our first two inclusion criteria:

1. The study's main concern is recommender systems

2. The study is a primary study presenting empirical results

After reading abstracts we will retrieve full-text exemplars of all remaining studies to do a screening of the articles to discover if they meet the following more specific inclusion criteria:

3. The study focuses on solving the cold-start user problem

4. The study focuses on user modelling

5. The proposed recommender system uses Bayesian reasoning

All studies that remain after this screening stage will be passed on to the quality assessment stage described in the next section.

A.5 Quality Assessment

To assess the quality of each primary study, we have formulated a set of quality evaluation questions. For each study, the responsible researcher answers each quality evaluation question by answering Y, P or N. Y (yes) means that the study fully answered the question, P (partly) means that study partly answered the question, while N (no) means that the study failed to answer the question. Y gives 1 point, P gives 0.5 while N gives 0 points.

The first stage of the quality assessment will be a quality screening where we filter out studies that lack fundamental research information. The following screening criteria will be used for this purpose:

1. Is there a clear statement of the aims of the research?

2. Is the study put into context of other studies and research?

All studies that pass the quality screening will be included in the final set of research studies included in the systematic literature review.

After the quality screening we will do a full quality assessment of the remaining studies. For each study we calculate a total quality score by taking the sum of all the single question scores (including the screening questions).

3. Are system/algorithm design decisions justified?

4. Is the test data set reproducible?

5. Is the study algorithm reproducible?
6. Is the experimental procedure thoroughly explained and reproducible?
7. Is it clearly stated in the study which other algorithms the study algorithm(s) have been compared with?
8. Are the performance metrics used in the study explained and justified?
9. Are the test results thoroughly analysed?
10. Does the test evidence support the findings presented?

Both researchers will read through all studies, and give their assessment of each criteria. Disagreements will be resolved through discussion. The final quality assessment scores will be recorded in a table where each study has a row and the assessment criteria each have a column. The full quality assessment will be done simultaneously with the data collection stage which is described in the next section.

A.6 Data Collection

The following data will be extracted from the studies included in the review:

1. Name of author(s)
2. Title
3. Study identifier
4. Year of publication
5. Type of article
6. Aims, objectives and contributions of study
7. Term used for cold-start user problem in study
8. Name of system
9. Type of recommender system
10. Cold-start user problem solution
11. User modelling technique
12. Experimental design
13. Test system domain
14. Test set source

15. Performance metric used
16. Findings and conclusions
17. Reported initial user involvement
18. Reported learning ability
19. Reported initial quality of service

To collect the data, we will use a table where each study is represented with a row, and each data type is represented as a column.

For each paper in the final set of study to be included in the review, the researchers will:

1. Separately read the study and fill in the quality assessment form (as described in the previous section)
2. In common resolve any disagreements in the quality assessment criteria
3. Together fill out the study's row in the data collection matrix

A.7 Data Analysis

After collecting data from, and assessing quality of, the studies included in the review, we will perform an analysis based on the resulting quality assessment and data collection tables.

We will perform our analysis in three stages:

1. Sort the studies included in the review into solution types according to how they solve the cold-start user problem.
2. Answer our research questions (given in Section A.2) with the solution types as a starting point, by consulting the data collection and quality assessment tables.
3. Review the data collection and quality assessment tables to evaluate any additional findings besides the ones directly related to the research questions.

All three stages will be performed by both researchers in meetings. Findings from each stage will be recorded at the end of each session.

A.8 Dissemination

The results of the review will form the basis in our further work with creating a recommender system for tourist. We will summarise our results, the analysis and our findings in our Master's Thesis. In addition, we believe that our findings will be of interest to the recommender system research community. Therefore we intend to write a research paper describing our work, results and findings for submission to a conference or a scientific journal.

Bibliography

Gregory Abowd, Christopher Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: A mobile context-aware tour guide. *Wireless Networks*, 3(5): 421–433, October 1997. ISSN 10220038. doi: 10.1023/A:1019194325861.

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

Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 19–28, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9. doi: http://doi.acm.org/10.1145/1557019.1557029.

Esma Aimeur and Flavien Serge Mani Onana. Better control on recommender systems. In *CEC-EEE '06: Proceedings of the The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services*, page 38, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2511-3. doi: http://dx.doi.org/10.1109/CEC-EEE.2006.17.

Liliana Ardissono, Anna Goy, Giovanna Petrone, Marino Segnan, and Pietro Torasso. Intrigue: Personalized recommendation of tourist attractions for desktop and handset devices. *Applied Artificial Intelligence*, 17(8):687–714, 2003. doi: http://dx.doi.org/10.1080/713827254.

Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/245108.245124.

Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pages 46–54, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8.

Jorge Biolchini, Paula Gomes Mian, Ana Candida Cruz Natali, and Guilherme Horta Travassos. Systematic review in software engineering. Technical report, Universidade Federal do Rio de Janeiro, 2005.

Fabian Bohnert, Daniel F. Schmidt, and Ingrid Zukerman. Spatial processes for recommender systems. In *IJCAI'09: Proceedings of the 21st international jont conference on*

Artifical intelligence, pages 2022–2027, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, San Francisco, 1998. Morgan Kaufmann.

David Budgen, Stuart Charters, Mark Turner, Pearl Brereton, Barbara Kitchenham, and Stephen Linkman. Investigating the applicability of the evidence-based paradigm to software engineering. In *WISER '06: Proceedings of the 2006 international workshop on Workshop on interdisciplinary software engineering research*, pages 7–14, New York, NY, USA, 2006. ACM. ISBN 1-59593-409-X. doi: http://doi.acm.org/10.1145/1137661.1137665.

Robin Burke. Knowledge-based recommender systems. In *Encyclopedia of Library and Information Systems*, volume 69, 2000.

Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002. ISSN 0924-1868. doi: http://dx.doi.org/10.1023/A:1021240730564.

Keith Cheverst, Nigel Davies, Keith Mitchell, and Adrian Friday. Experiences of developing and deploying a context-aware tourist guide: the guide project. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 20–31, New York, NY, USA, 2000. ACM. ISBN 1-58113-197-6. doi: http://doi.acm.org/10.1145/345910.345916.

David N. Chin and Asanga Porage. Acquiring user preferences for product customization. In *UM '01: Proceedings of the 8th International Conference on User Modeling 2001*, pages 95–104, London, UK, 2001. Springer-Verlag. ISBN 3-540-42325-7.

Wei Chu and Seung-Taek Park. Personalized recommendation on dynamic content using predictive bilinear models. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 691–700, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-487-4. doi: http://doi.acm.org/10.1145/1526709.1526802.

Michelle Keim Condliff, David D. Lewis, and David Madigan. Bayesian mixed-effects models for recommender systems. In *In ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

Luca Console, Ilaria Torre, Ilaria Lombardi, Sara Gioria, and Valentina Surano. Personalized and adaptive services on board a car: An application for tourist information. *J. Intell. Inf. Syst.*, 21(3):249–284, 2003. ISSN 0925-9902. doi: http://dx.doi.org/10.1023/A:1025506816422.

Lorcan Coyle and Pádraig Cunningham. Exploiting re-ranking information in a case-based personal travel assistent. In *In Workshop on Mixed-Initiative Case-Based Reasoning at the 5th International Conference on Case-Based Reasoning*. Springer, 2003.

Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004. ISSN 1046-8188. doi: http://doi.acm.org/10.1145/963770.963776.

Tore Dybå and Torgeir Dingsøyr. Empirical studies of agile software development: A systematic review. *Inf. Softw. Technol.*, 50(9-10):833–859, 2008. ISSN 0950-5849. doi: http://dx.doi.org/10.1016/j.infsof.2008.01.006.

Fengrong Gao, Chunxiao Xing, Xiaoyong Du, and Shan Wang. Personalized service system based on hybrid filtering for digital library. *Tsinghua Science & Technology*, 12(1):1 – 8, 2007a. ISSN 1007-0214. doi: DOI:10.1016/S1007-0214(07)70001-9.

Ying Gao, Hong Qi, Jie Liu, and Dayou Liu. A recommendation algorithm combining user grade-based collaborative filtering and probabilistic relational models. In *FSKD '07: Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery*, pages 67–71, Washington, DC, USA, 2007b. IEEE Computer Society. ISBN 0-7695-2874-0. doi: http://dx.doi.org/10.1109/FSKD.2007.113.

Angel García-Crespo, Javier Chamizo, Ismael Rivera, Myriam Mencke, Ricardo Colomo-Palacios, and Juan Miguel Gómez-Berbís. Speta: Social pervasive e-tourism advisor. *Telematics and Informatics*, 26(3):306 – 315, 2009. ISSN 0736-5853. doi: 10.1016/j.tele.2008.11.008. Mobile and wireless communications: Technologies, applications, business models and diffusion.

David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/138859.138867.

Robert H. Guttmann. Merchant differentiation through integrative negotiation in agent-mediated electronic commerce. Master's thesis, School of Architecture and Planning, Program in Media Arts and Sciences, Massachusetts Institute of Technology, 1998.

David Heckerman. A tutorial on learning with Bayesian networks. In *Proceedings of the NATO Advanced Study Institute on Learning in graphical models*, pages 301–354, Norwell, MA, USA, 1998. Kluwer Academic Publishers.

Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004. ISSN 1046-8188. doi: http://doi.acm.org/10.1145/963770.963772.

Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004. ISSN 1046-8188. doi: http://doi.acm.org/10.1145/963770.963774.

Günther Hölbling, Michael Pleschgatternig, and Harald Kosch. Personaltv. *Multimedia Tools Appl.*, 46(2-3):259–288, 2010. ISSN 1380-7501. doi: http://dx.doi.org/10.1007/s11042-009-0352-2.

Yuxia Huang and Ling Bian. A Bayesian network and analytic hierarchy process based personalized recommendations for tourist attractions over the internet. *Expert Systems* *with Applications*, 36(1):933 – 943, 2009. ISSN 0957-4174. doi: DOI:10.1016/j.eswa.2007.10.019.

Finn V. Jensen and Thomas D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer Publishing Company, Incorporated, 2007. ISBN 9780387682815.

Rong Jin, Luo Si, and Chengxiang Zhai. Preference-based graphic models for collaborative filtering. In *In Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence*, pages 329–336, 2003.

Katerina Kabassi. Personalizing recommendations for tourists. *Telematics and Informatics*, 27(1):51–66, 2010.

Heung-Nam Kim, Ae-Ttie Ji, Inay Ha, and Geun-Sik Jo. Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. *Electronic Commerce Research and Applications*, 9(1):73 – 83, 2010. ISSN 1567-4223. doi: DOI:10.1016/j.elerap.2009.08.004. Special Issue: Social Networks and Web 2.0.

B. A. Kitchenham. Guidelines for performing systematic literature reviews in software engineering version 2.3. Technical Report EBSE-2007-01, Keele University and University of Durham, 2007.

Su-Jeong Ko and Jung-Hyun Lee. User preference mining through collaborative filtering and content based filtering in recommender system. In *EC-WEB '02: Proceedings of the Third International Conference on E-Commerce and Web Technologies*, pages 244–253, London, UK, 2002. Springer-Verlag. ISBN 3-540-44137-9.

Bruce Krulwich. Lifestyle finder: Intelligent user profiling using large-scale demographic data. *AI Magazine*, 18(2):37–45, 1997.

Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. Addressing cold-start problem in recommendation systems. In *ICUIMC '08: Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 208–211, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-993-7. doi: http://doi.acm.org/10.1145/1352793.1352837.

Helge Langseth and Thomas D. Nielsen. A latent model for collaborative filtering. Technical Report 09-003, Department of Computer Science, Aalborg University, 2009.

George Lekakos and George M. Giaglis. A hybrid approach for improving predictive accuracy of collaborative filtering algorithms. *User Modeling and User-Adapted Interaction*, 17(1-2):5–40, 2007. ISSN 0924-1868. doi: http://dx.doi.org/10.1007/s11257-006-9019-0.

Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003. ISSN 1089-7801. doi: http://dx.doi.org/10.1109/MIC.2003.1167344.

Ji Liu and Guishi Deng. A new-user cold-starting recommendation algorithm based on normalization of preference. In *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on*, pages 1 –4, 12-14 2008. doi: 10.1109/WiCom.2008.2141.

Eren Manavoglu, Dmitry Pavlov, and C. Lee Giles. Probabilistic user behavior models. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 203, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1978-4.

Saranya Maneeroj and Atsuhiro Takasu. Hybrid recommender system using latent features. In *WAINA '09: Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops*, pages 661–666, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-0-7695-3639-2. doi: http://dx.doi.org/10.1109/WAINA.2009.122.

Benjamin Marlin. Collaborative filtering: A machine learning perspective. Technical report, University of Toronto, 2004.

Paolo Massa and Bobby Bhattacharjee. Using trust in recommender systems: An experimental analysis. In *In Proceedings of iTrust2004 International Conference*, pages 221–235, 2004.

Prem Melville, Raymod J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth national conference on Artificial intelligence*, pages 187–192, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence. ISBN 0-262-51129-0.

Stuart E. Middleton, Nigel R. Shadbolt, and David C. De Roure. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54–88, 2004. ISSN 1046-8188. doi: http://doi.acm.org/10.1145/963770.963773.

Bamshad Mobasher, Honghua Dai, Tao Luo, and Miki Nakagawa. Effective personalization based on association rule discovery from web usage data. In *WIDM '01: Proceedings of the 3rd international workshop on Web information and data management*, pages 9–15, New York, NY, USA, 2001. ACM. ISBN 1-58113-444-4. doi: http://doi.acm.org/10.1145/502932.502935.

Biswadeep Nag. Vibes: A platform-centric approach to building recommender systems. *IEEE Data Eng. Bull.*, 31(2):23–31, 2008.

Douglas Oard and Jinmook Kim. Implicit feedback for recommender systems. In *in Proceedings of the AAAI Workshop on Recommender Systems*, pages 81–83, 1998.

Han-Saem Park, Ji-Oh Yoo, and Sung-Bae Cho. A context-aware music recommendation system using fuzzy Bayesian networks with utility theory. In Lipo Wang, Licheng Jiao, Guanming Shi, Xue Li, and Jing Liu, editors, *Fuzzy Systems and Knowledge Discovery*, volume 4223, chapter 121, pages 970–979. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006a. ISBN 978-3-540-45916-3. doi: 10.1007/11881599_121.

Seung-Taek Park and Wei Chu. Pairwise preference regression for cold-start recommendation. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 21–28, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-435-5. doi: http://doi.acm.org/10.1145/1639714.1639720.

Seung-Taek Park, David Pennock, Omid Madani, Nathan Good, and Dennis DeCoste. Naïve filterbots for robust cold-start recommendations. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 699–705, New York, NY, USA, 2006b. ACM. ISBN 1-59593-339-5. doi: http://doi.acm.org/10.1145/1150402.1150490.

Judea Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th Conference of the Cognitive Science Society, University of California, Irvine*, pages 329–334, August 1985.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 1558604790.

David Pennock, Eric Horvitz, Steve Lawrence, and C Lee Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *In Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 473–480. Morgan Kaufmann, 2000.

Alexandrin Popescul, Lyle H. Ungar, David M. Pennock, and Steve Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 437–444, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-800-1.

Verus Pronk, Wim Verhaegh, Adolf Proidl, and Marco Tiemann. Incorporating user control into recommender systems based on naive Bayesian classification. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 73–80, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-730–8. doi: http://doi.acm.org/10.1145/1297231.1297244.

S. M. Rajpara, A. P. Botello, J. Townend, and A. D. Ormerod. Systematic review of dermoscopy and digital dermoscopy/ artificial intelligence for the diagnosis of melanoma. *British Journal of Dermatology*, 161(3):591–604, 2009. ISSN 1365-2133. doi: 10.1111/j.1365-2133.2009.09093.x.

Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. Getting to know you: learning new user preferences in recommender systems. In *IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces*, pages 127–134, New York, NY, USA, 2002. ACM. ISBN 1-58113-459-2. doi: http://doi.acm.org/10.1145/502716.502737.

Paul Resnick and Hal R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, 1997. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/245108.245121.

Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994. ACM. ISBN 0-89791-689-1. doi: http://doi.acm.org/10.1145/192844.192905.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, December 2002. ISBN 0137903952.

Gerard Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989. ISBN 0-201-12227-8.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, USA, 2001. ACM. ISBN 1-58113-348-0. doi: http://doi.acm.org/10.1145/371920.372071.

Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John T. Riedl. Application of dimensionality reduction in recommender system – a case study. In *IN ACM WEBKDD WORKSHOP*. ACM, 2000.

Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260, New York, NY, USA, 2002. ACM. ISBN 1-58113-561-0. doi: http://doi.acm.org/10.1145/564376.564421.

Ansgar Scherp and Susanne Boll. Generic support for personalized mobile multimedia tourist applications. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 178–179, New York, NY, USA, 2004. ACM. ISBN 1-58113-893-8. doi: http://doi.acm.org/10.1145/1027527.1027566.

Silvia Schiaffino and Analía Amandi. Building an expert travel agent as a software agent. *Expert Syst. Appl.*, 36(2):1291–1299, 2009. ISSN 0957-4174. doi: http://dx.doi.org/10.1016/j.eswa.2007.11.032.

Mark Van Setten, Stanislav Pokraev, Johan Koolwaaij, and Telematica Instituut. Context-aware recommendations in the mobile tourist application compass. In *In Nejdl, W. & De Bra, P. (Eds.). AH 2004, LNCS 3137*, pages 235–244. Springer-Verlag, 2004.

Guy Shani, David Heckerman, and Ronen I. Brafman. An mdp-based recommender system. *J. Mach. Learn. Res.*, 6:1265–1295, 2005. ISSN 1532-4435.

Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating "word of mouth". In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-84705-1. doi: http://doi.acm.org/10.1145/223904.223931.

Patricia Victor, Chris Cornelis, Martine De Cock, and Ankur M. Teredesai. Key figure impact in trust-enhanced recommender systems. *AI Commun.*, 21(2-3):127–143, 2008. ISSN 0921-7126.

Hsaio-Fan Wang and Cheng-Ting Wu. A strategy-oriented operation module for recommender systems in e-commerce. In *AIC'09: Proceedings of the 9th WSEAS international conference on Applied informatics and communications*, pages 78–83, Stevens

Point, Wisconsin, USA, 2009. World Scientific and Engineering Academy and Society (WSEAS). ISBN 978-960-474-107-6.

Jun Wang, Arjen P. de Vries, and Marcel J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508, New York, NY, USA, 2006. ACM. ISBN 1-59593-369-7. doi: http://doi.acm.org/10.1145/1148170.1148257.

Magnar Wium. Design and evaluation of an personalized mobile tourist application. Master's thesis, Norwegian University of Science and Technology, 2010.

Gui-Rong Xue, Jie Han, Yong Yu, and Qiang Yang. User language model for collaborative personalized search. *ACM Trans. Inf. Syst.*, 27(2):1–28, 2009. ISSN 1046-8188. doi: http://doi.acm.org/10.1145/1462198.1462203.

Philip Zigoris and Yi Zhang. Bayesian adaptive user profiling with explicit & implicit feedback. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 397–404, New York, NY, USA, 2006. ACM. ISBN 1-59593-433-2. doi: http://doi.acm.org/10.1145/1183614.1183672.