Acknowledgments

Resumen

Abstract

List of Figures

List of Tables

Contents

Chapter 1

Introduction

It is a reality that the World Wide Web in recent years, is growing exponentially, which means the presence of millions of users on Web sites, Web applications, Web systems, etc. []. There is a wide variety of Web systems, where we have different users interacting with them. These users have different goals when using these Web systems. For example do a search in Google [] of particular topic, make a reservation for a room in a luxury resort, check your bank account or simply view your status on your Facebook account [].This variation of users represents a complex diversity as individuals []. This diversity lies in different skills, interests, preferences and ways of thinking, learning and knowledge [].For this reason users need different ways to interact with the information presented by the great variety of Web systems that exist.

When we intend to customize any element in Web system, we need to know the user's personal information. This information is a collection of needs, characteristics, feelings, tastes, etc. This information is required to be able to form the representation of knowledge about users. This is what is known as user modeling (UM).

A user modeling can be as simple as a profile systems where is basic knowledge of users. Also can be as complex as represent its characteristics, needs, interests, ways to feel. In order to understand specific users. The main goal of user modeling is to represent aspects of the real world of the user's in autonomous automatically way.

In this document we present a user modeling in the context of Web-based interactive evolutionary computation.

Interactive evolutionary computation (IEC) is a branch of evolutionary computation where users become a part of the evolutionary process by replacing the fitness function; evaluating individuals of a population based on their personal preferences[13]. These evaluations are subjective according to the user point of view based on their perceptions, interests and desires.

Normally such systems require users to evaluate large amounts of individuals iteratively, causing them to lose interest for participate by fatigue that is generated[13]. Nowadays some of these systems are migrating to Web technologies looking for volunteers users to collaborate in the evaluations for distribute the load and lower the fatigue. Having Web- based interactive evolutionary systems open the possibility for linked to social platforms in order to involve the largest number possible of users to assist in the evaluation of individuals produced by these systems applications.

Chapter 2

State of the art

Some 1986's Dawkins's research was the pioneer of a significant addition to the 1990s IEC algorithms research works[][Dawkins 1986].

There is two key research approach about his field:

Creative Approach: The Artificial Life (AL) was the base of creative approach. AL uses complex algorithms for biological life models emulation. To perform this task, it is needed to include some of the different techniques starting from right image treatment. Good graphic creation as well as a great music and quality sounds, [Sims 1991b], [Sims 1991c], [Sims 1994], [Dawkins 1986], [Disz 1997], [Unemi 2000]and [Unemi 2003].

Humanized technology approach: The concept of humanized technology approach comes from the approach that is focused on the IEC algorithms interface, this is the research of interaction between humans and computer systems. The main goal of this was to reduce the user's fatigue and to promote the inputs and outputs of algorithms to improve the efficiency of them. IEC has made his own way in practical fields such as engineering, education,etc., [Parmee 1993], [Ventrella 1994a], [Takagi 1996], [Poli 1997], [Parmee 1998] and [Takagi 1998].

Computer graphics (CG) The Biomorph of Dawkins was the first IEC research, from this research comes to many motivated works mostly about the Selfish Gene, come of these works are: [Ochoa 1998], [Mccormack 1993], and [Smith 2003].

In Dawkins work a conventional recursive algorithm was used as a baseline maintaining the main target of trees with an L-system Lindenmayer.

Some 1986's Dawkins's research was the pioneer of the significant addition to the 1990s IEC algorithms research works[Dawkins 1986].

There is two key research approach about his field:

Creative Approach: The Artificial Life (AL) was the base of creative approach. AL uses complex algorithms for biological life models emulation. In order to perform this task it is needed to include some of the different techniques starting from right image treatment , good graphic creation as well as a great music and quality sounds, [Sims 1991b], [Sims 1991c], [Sims 1994], [Dawkins 1986], [Disz 1997], [Unemi 2000]and [Unemi 2003].

Humanized technology approach: The concept of humanized technology approach comes from the approach that is focused on the IEC algorithms interface, this is the research of interaction between humans and computer systems. The main goal of this was to reduce the user's fatigue and to promote the inputs and outputs of algorithms to improve the efficiency of them. IEC has made his own way in practical fields such as engineering, education,etc., [Parmee 1993], [Ventrella 1994a], [Takagi 1996], [Poli 1997], [Parmee 1998] and [Takagi 1998].

Computer graphics (CG) The Biomorph of Dawkins was the first IEC research, from this research comes to many motivated works mostly about the Selfish Gene, come of these works are: [Ochoa 1998], [Mccormack 1993], and [Smith 2003].

In Dawkins work, a conventional recursive algorithm was used as a baseline maintaining the main target of trees with an L-system (Lindenmayer). This same L-system was the base for another experiment to create 2-D CG forms insects from a system called Blind Watchmaker who used L-system angles from L-system output intuitively selected; the creation was called biomorphs. These creations reach his target with the multiple selections of the users based on their preferences; all these selections acted like a natural adaptation filter.

We can find plenty of applications and works for fractal generation [Sims 1991a] and [Sims 1992], [Baluja 1993] and [Baluja 1994], [Lund 1995], or [Angeline 1996],[Raynal 1999] and [Lutton 2003], for rendering in tridimensional, [Todd 1991],[Broughton 1997], [Das 1994] and [Tam 2002], for generation of virtual creatures, [Sims 1994], [Rowland 2000], or aerodynamic surface design (wings), [NGuyen 1993], [NGuyen 1994] and [NGuyen 1997].

We can discover more than one additional way to use this research in the artistic field with several applications of IEC who are used for cartoon face construction and animations matters, like Mutator [Todd 1991], [Todd 1994] and[Todd 1999] or [Bentley 1999a].

The genetic programming (GP) applications offers a category called Interactive Genetic Programming (IGP) with many examples of successful application in tridimensional artwork for artistic animations or construction using mathematical equations as CAVE [Das 1994], [Papka 1996] and [Disz 1997], [Sims 1991], [Sims 1991a], [Sims 1992],[Sims 1993] and [Min 2004]. As this work consequence, Panspermia orPrimordial Dance was created.

Imagen

Imagen

The artistic field is only the first step of a great IEC implementation; it is important to mention another relevant projects called Galapagos, [Sims 1997], and SBART, [Unemi 2000]. The IEC application Galapagos Project is the exhibit in Tokio Multimedia Museum, (NTT Intercommunication Center) and this project originates engaging images to all visitors based on L-systems.

Imagen

There are created after one selection, to get a good solution through multiple repetitions. This action is performed with Genetic Programming (GP), after the calculation of each pixel value using trees of equations combining logarithm, maximum, and minimum, sine, root, cosine, exponential arithmetic operators. AnimationLab is found as an outstanding work who offer figures that can run or walk working with the user to receive more opportunities to be picked. A particular characteristic of all of the figures is that the figures extremities Mentioning open source works, we can find SBART as an IGP [Unemi 2000] tool to create graphics. SBART allow to users to evaluate 20 two-dimensional images, subsequently twenty new image has direction and angles.

Imagen

There are many examples for this field application as [McKenna 1990], [Ventrella 1994a], [Ventrella 1994b], or [Ventrella 1995], [Lim 1999] and [Lim 2000]. One of the Interactive Evolutionary Programming (IEP) artistic application was created by [Angeline 1996], as a fractal generation where the system allows the evolution of animations for the ones who were selected from the user, the application initially show only 10 animations to rate.

Music and sound

It is important to know how IEC was implemented in music generation, with several applications in this field. We will start mentioning the pioneer application

GENJAM, [Biles 1994], [Biles 1996] or [Biles 1999] and [Biles 2000]. Some other attractive works are Sonomorph, [Nelson 1993] and[Nelson 1995], or SBEAT, [Unemi 2003], [Horowitz 1994], [Onisawa 2000], [Tokui 2000] and [Fels 2002]. It is possible to hear a part of the music songs of these previously mentioned works broadcasted in the radio station WDYN. (100.1, New York, USA, WEBPage:http://www.wdyn.net/).

The IEC algorithms are the base for the functionality of the music generation systems, a visual representation of this is given in the below figure:

Imagen

Table 2.1: Comparison of context-aware recommender systems.

Application	Contextual Factor	Domain	Paradigm	Device
CoMoLE	**Time, available time, place, device, level of knowledge, learning style.**	**E-learning**	**Pre-filtering**	**Mobiles, PC, laptop.**
Moma-System	**Location, time.**	**E-commerce**	**Post-filtering**	**PC, laptop.**
UbiquITO	**Season, time, temperature.**	**Tourism**	**Post-filtering**	**Mobiles**
ReRex	**Distance of the point of interest, temperature, weather, season, weekend, companion, travel goal, transport.**	**Tourism**	**Model-based**	**Mobiles**
LifeTrack	**Location, time, day of the week, traifc noise(level), temperature, weather.**	**Music**	**Post-filtering**	**PC, Mobiles.**
CARS	**Location and season.**	**Restaurants**	**Post-filtering**	**PC, laptop.**
InCarMusic	**Driving style, road type, landscape, sleepiness, traffic conditions, mood weather and natural phenomena.**	**Music**	**Model-based**	**Mobiles**
REJA	**Location.**	**Restaurants**	**Pre-filtering and Post-filtering**	**PC, laptop, mobiles.**
CiberGuide	**Location, time, weather.**	**Tourism**	**Post-filtering**	**Mobiles**
PECITAS	**Location, routes.**	**Transport**	**Post-filtering**	**Mobiles**
LARS	**Tourists location and time.**	**Tourist packages**	**Post-filtering**	**Mobiles**
I'm feeling LoCo	**Location, transportation.**	**Tourism**	**Model-based**	**Mobiles**
MOPSI	**Location**	**Tourism and transport**	**Post-filtering**	**Mobiles**

Chapter 3

Background

This chapter present the fundamental concepts related this work. The formal definitions referring to fuzzy systems, contextual factors and recommender system techniques used in the proposed method.

3.1 User Modeling

User modeling can be represented as the technique of building a model of the user to personalize a system. The user model is commonly created as the user is working with the system. An example is an educational application that teaches students an individual skill: given the rules and knowledge in the user model, the difficulty level of the exercises in the form is altered as the user progresses. Formally definition of user modeling according to McTear (1993, p. 158): " user modeling is the process of gathering information about the users of a computer system and of using the information to provide services or information adapted to the specific requirements of individual users (or groups of users)". The purpose of the user model is to have a module containing the operations that are needed to personalize the system, and the user profile, which includes the personal data of the user (Mohamad et al., 2013). System personalization over user modeling is related to the research field of adaptive systems; this subject is beyond the scope of this research work. Focus on the human user, user modeling is a very cross-disciplinary research topic, comprehending the domains of artificial intelligence, computer science, and social science. Ideas have been coopted from an extensive range of subdomains, such as humancomputer interaction, elearning, information science, social computing, machine learning, data mining, cognitive science, and so on (Kay, et al., 2012; Kobsa, 2001). There is interest in user modeling from both a scientific and commercial perspective (Razmerita, 2009).

3.1.1 Application Domains for user modeling

Amount Research and implementation exist in this domain in which personalization and user modeling plays an important role. This section presents several works of these domains. To understand this topic, the different objects are divided into three general categories: * supporting a user during a task. * giving a user a specific personalized experience. * training and educating a user. The categories especially differ in the kind of user data that is used. For each domain, the general purpose of the domain and the more accurate purpose of the user model are discussed.

3.1.2 User models for providing task support

Task support systems are f systems that help a user during a task by either supporting the user perform the task or by completely taking over this task (Nurmi et al., 2007; Brun et al., 2010). For instance, an application that automatically categorizes the incoming emails of the user. The goal of the user model in these requests is to promote the efficiency of interactions with the user, to simplify these interactions and to make complex systems more usable (Razmerita, 2009; Fischer, 2001). To perform this personalization, data is collected through observations of the user. This information is related to the users goals and needs, but especially to the task that the user currently is accomplished, like the users task knowledge and background. Much research has been done in this domain, but because many separate research projects are focusing on an exact task or subject (Sannes, 2011), it is hard to make generalizations or to establish one delimited investigation topic. Commonly discussed research subjects are Decision Support Systems, Adaptive Hypermedia, and Adaptive Ubiquitous Systems, each having his or her own specific domain and way of personalization.

3.1.3 Decision support systems

Decision support systems are systems that support a user with making a decision in a complex, professional environment (Nurmi et al., 2007). For example, a system used at a pharmacy for automatically checking valid combinations of medicine. The method can be used to help the pharmacist in prescribing the right combinations and to give information for making a decision when a problem occurs. The purpose of the user model in decision support systems is to present the user with the right and appropriate information, giving different feedback or applying various decision steps according to the characteristics of the user.The data that is used is often associated with the users task and background knowledge. The adaptation takes place by adapting the amount and the content of the feedback provided by the system.

Decision support systems are traditionally ruled or logic-based systems, in which all the relevant information is represented in a knowledge base. This means that the content of the user model itself is also highly dependent on the way the rules and knowledge are represented.

3.1.4 Adaptive Hypermedia

Adaptive hypermedia system is a system that grant users to browse freely information network, structured by nodes and links, to retrieve items of information (Nurmi et al., 2007; Deepa et al., 2012). For instance an internet website application. The goal of the user model is to make the interface and structure of the system dynamic. This enables the application to adapt to the user and to make it easier for the user to search for and retrieve relevant information. The data used in the user model is related to the users abilities, knowledge, and goals in the application. The adaptation happens by adjusting the structure and the presentation style to the expected needs of the user. For example, by enhancing web search: promoting pages that might better correspond to the users characteristics, on the other hand by giving navigation support, through highlighting certain components of a page (Razmerita et al., 2012).

3.1.5 Adaptive ubiquitous systems

Ubiquitous systems are concerned with data handling applications integrated into everyday objects and activities (Nurmi et al., 2007). For example the smart energy meter, recording the energy usage in a household through small devices distributed in a house, supporting the user with managing this energy usage(Hargreaves et al., 2010). * The main purpose of the user model is to improve the system, facilitate the users preferences and thus make the overall use easier. * Because the personalization can take place in every situation and location, the data is focused on the user state and context. For example to enable the contextualization to a current environmental change. *The adaptation takes place by changing the behavior of, and the feedback given by the whole system. These objects can be inferred by looking at the properties of the objects in the user profile, or by looking at the objects in other user profiles that are similar to the user (Kobsa, 2001; Kay et al., 2012). Because of the predominantly commercial goal of these systems, the adaptations often take place in a very intrusive way, to make sure the user notices the change. Most recommender systems used on the Internet, which means that some typical technical difficulties are associated with these kinds of user models. First, the user profile is often only saved during the users visit, which means that fast and efficient adaption is necessary. Recommender systems often become more precise when the user spends more time on the system. Second, the systems structure is often split up in a client and on a server side, where the client side solely gathers user information and sends it to the server, where the actual computation takes place.

3.1.6 User models for providing a personal experience.

User models for providing the user with a personal experience have the goal to improve the user experience while using the system. This kind of user modeling is especially focused on more commercial fields, such as e-commerce, marketing, and computer games, and became popular with the rise of the Internet. The information that is used by the user in this main domain is mostly focused on the information that defines the user, such as the users preferences and interests. Since this data is regularly delicate, privacy is a big issue (Toch et al., 2012). While in other domains the privacy of the user data is also important, in this area it is even a greater topic of discussion because the incentive of the application developers is frequently contradictory to the incentive of the actual user, considering gaining and sharing the users personal information. For instance, user profiles are often shared among diverse components of the same application, or even with different applications (Brun et al., 2010; Karam et al., 2012), which presents additional weaknesses and possible undesirable information sharing. Ensuring personal data is not open to all people, in addition to defining strict privacy policies, is thus essential in these user models. Some investigation in this domain are.

3.1.7 Recommender Systems and User Adaptive Computer Games.

Recommender systems are concerned with presenting the user with relevant information and suggestions. They are commonly used on the Internet, for example on websites such as Facebook, to provide the user with personalized news, targeted advertisements and possibly new friends (Brun et al., 2010). The purpose of the user model is to give the system with information that is assumed to be important for the user. The information that is stored for this goal is associated with the preferences of the user to certain objects, like products, music or people. To benefit a classification of these objects, the interaction history of the user is stored, or the user is explicitly asked to rate certain objects. The content of the system is eventually adjusted by showing the recently inferred objects. In these senses, objects can be inferred by looking at the attributes of the objects in the user profile, or by looking at the objects in other user profiles that are related to the user (Kobsa, 2001; Kay et al., 2012). As a result of the predominantly commercial target of these systems, the adaptations often take place in a very invasive way, to make sure the user notices the change. Most recommender systems are based on the Internet, which means that some typical technical difficulties are associated with these kinds of user models. First, the user profile is often only stored while the users visit, which means that fast and efficient adaption is important. Recommender systems usually become more precise when the user spends more time with the system. Second, the systems architecture is usually client-server, where the client side gathers user information and sends it to the server, where the actual process takes place.

3.1.8 User Adaptive Computer Games.

User Adaptive computer games are games that focus on increasing the perceived value by providing a strongly individualized experience (Brisson, 2012). For example is a firstperson shooter that adapts the performance of the enemy according to the shooting accuracy of the player.

The fundamental idea of the user model is to identify or classify the user, so the appropriate adjustment is made in the computer game. The information that is used addresses the preferences and progress of the user, such as the users current difficulty level or even the employed strategy. This data is usually obtained through the interactions of the user with the game, and therefor first should be translated and formalized before it can be used to interpret conclusions on a higher level. The adaptation that takes place in the game concerns changing the content and role of the game, such as the game difficulty, the behavior of nonplayer characters or even the background music (Bakkes, et al., 2012) .

Because of the emphasis on the user, user adaptive computer games have relatively a lot of processing power available for personalization. In this sense the user adaptive computer games domain is a very interesting research domain.

3.1.9 User models for educational purposes.

Educational systems are systems developed with a teaching reason. They are commonly applied in elearning, where electronic media and Information technologies are used for education. However, in most educational systems, user modeling and adaptation plays a minor role. Content is presented, and only simple things such as the students progress in the course are registered. By adding personalization to these applications, the learning value can notably increase, ensuring that every learner achieves and reaches the highest standards possible (Heller et al., 2006). Also, the experience of the teacher or supervisor can be increased through personalization, for instance through inferring and employing the preferred teaching style. However, here we consider the student as the user to which the system will be personalized. Thus, it is preferable that the data stored in the user profile be interpretable by humans. When looking at the time of adjustment in educational systems, we can make a clear difference between adaptation while the student is doing an exercise, which we will refer to as online adaptation, and adaptation that takes place afterward, which we will refer to as offline adaptation. The most important investigation domains that do utilize considerable user modeling constructions are Intelligent Tutoring Systems and Adaptive Educational Games.

3.1.10 Intelligent Tutoring Systems.

Intelligent Tutoring Systems (ITS) are systems that provide students automated stepbystep instruction as the students complete training tasks and/or work on exercises. An ITS has the purpose to complement or even replace the human teacher. For example a system for teaching students how to program, with the ability to automatically detect common mistakes (ElsomCook, 1993).

The particular goal of the user model is to select educational activities and strategies and in addition delivered individual feedback that is most relevant to the users level of knowledge (Kobsa, 2001; McTear, 1993). The user information that is stored for this purpose is the students state, knowledge and level of achievement. This data is exclusively observed over the actions and results of the student, such as the answers the student gives. After observing this information, it is used to infer higher level properties, such as the students learning style and other preferences.

Traditionally, just like decision support systems, information technology systems are knowledge based systems, using formalized domain knowledge and rules to drive the user adjustment (adaptation). For instance stereotypes are widely used in information technology systems(Kay, 2000) and represent a set of default attributes that often cooccur in users or in a certain group of people. The different stereotypes that have been build differ in granularity of detail and complexity.

3.1.11 Adaptive Educational Games.

Adaptive Educational Games (AEGs) are complicated educational games that combine ideas from several investigations areas, to increase the students learning experience (Peeters, et al., 2012a). These are especially based on serious games: computer games with an educational approach, where things are taught to students by using a playful idea (Korteling, et al., 2011; Johnson, et al., 2005). For instance an AEG is a training application for fire fighters, letting the fire fighters train their skills and knowledge in a safe on a virtual environment.

The objective of the user model in an AEGs is to optimize the learning process and outcome. The user information is considered with the advance and knowledge of the student, but also with the students mental and cognitive characteristics. The gained data can be used to adjust the content, presentation, and system behavior to the students need, for example, by adjusting the content, tone, or amount of presented feedback. Adaptive computer games have a lot of processing power available for personalization, making a complex and interesting domain for user modeling.

3.1.12 Methods for user modeling.

In the user modeling topic, researchers have proposed more general design methods and frameworks to guide the developers in the process of user modeling. These general methods are useful in research projects, where the knowledge can be reused to adjust the user model to the systems characteristics. Also in commercial applications, these general methods have proven to be useful (Brun et al., 2010), because they make it easier and more feasible to implement personalization into a system. In early work, the process of user modeling was mostly based on the intuition and experience of the developer or researcher. In recent work, the techniques of user modeling were essentially based on the intuition and expertise of the developer or researcher. As the user modeling research field evolved, there has been put much effort in creating a general way for designing and constructing a user model, by basing decisions on more empirical grounds and by defining methods applicable to the whole field (Kobsa, 2001; Durrani, 1997).

Frameworks, methodologies, and architectures have been developed, defining the strict process, restrictions and choices on how to design and build a user model. In the early days of user modeling, the focus was put on developing one method applicable to the user modeling field as a whole. However, user modeling is a very cross-disciplinary research subject. Therefore, throughout the decades, the user modeling area of research has been influenced by the important research topics and trends of their time. For example, when information technologies became a major subject in the early nineties, user modeling methods were also mostly focused on the application of stereotypes, knowledge bases, and logic to define a user model. With the rise of the Internet, the objectives of the user modeling field change to Web-oriented applications and all the specific problems that arise with this. Thus this connection, also the general user modeling methods that were developed, were focused on the popular research domains of their time (Kay et al., 2012). The main approaches to user modeling did not change, but the specific fillingin of the user model, such as which technology to apply, did change. In this sense development of user modeling as a whole, most researchers eventually agreed that one method to solve all problems is not possible (McTear, 1993; Kobsa, 2001). Instead, a broad range of generic user modeling methods has been developed (Fischer, 2001); each of which supports only a few of the very different manifestations of personalization. In the rest of this section, the general user modeling architecture and the most interesting general and domain specific methods will be shortly discussed.

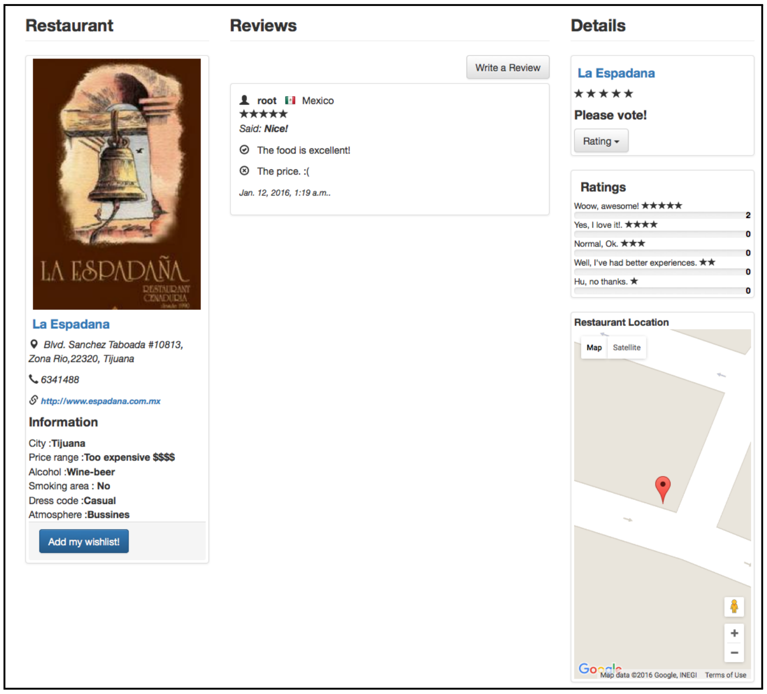


Figure 3.1: User interface of the restaurant model.

3.2 Interactive Evolutionary Computation.

3.3 Fuzzy Logic.

$$\mu_A \cap \mu_B = T(\mu_A(x), \mu_B(x)) \tag{3.1}$$

3.4 Gamification.

Chapter 4

Proposed method

4.1 Data models

The data models was implemented in the DBMS with PostgreSQL database. All the information in the context-aware recommender system was managed in a scheme of a relational database. Each model is referring in its section in order to have a better comprehension of each one.

4.1.1 Restaurant model

An effective on-line recommender system must be based upon an understanding of consumer preferences and successfully mapping potential products into the consumers preferences[2]. Pan and Fesenmaier[34] argued that this can be achieved through the understanding of how consumers describe in their own language a product, a place, and the experience when they are consuming the product or visiting the place.

Traditionally, choosing a restaurant has been considered as rational behavior where a number of attributes contribute to the overall usefulness of a restaurant. For example: food type, service quality, atmosphere of the restaurant, and availability of information about a restaurant, plays an important role at different stages in consumers desitions making[5]. While food quality and food type have been perceived as the most important variables for consumers restaurant selection, situational and contextual factors have been found to be important also. Due to this in Kivela[26] identifies 4 types of restaurants: 1) fine dining/gourmet, 2) theme/atmosphere, 3) family/popular, and 4) convenience/fast-food; and Auty[5] identifies 4 types of dining out occasions: 1) namely celebration, 2) social occasion, 3) convenience/quick meal, and 4) business meal.

Taking in account the context, the restaurant model proposed for context-aware recommender system was definded with 55 attributes about the restaurants features. An exploration about contents of models of others works were compared to define the suitable information into the model. Therefore, the restaurant model is a binary vector with the following contextual attributes: 1) price range, 2) payment type, 3) alcohol type, 4) smoking area, 5) dress code, 6) parking type, 7) installations type,

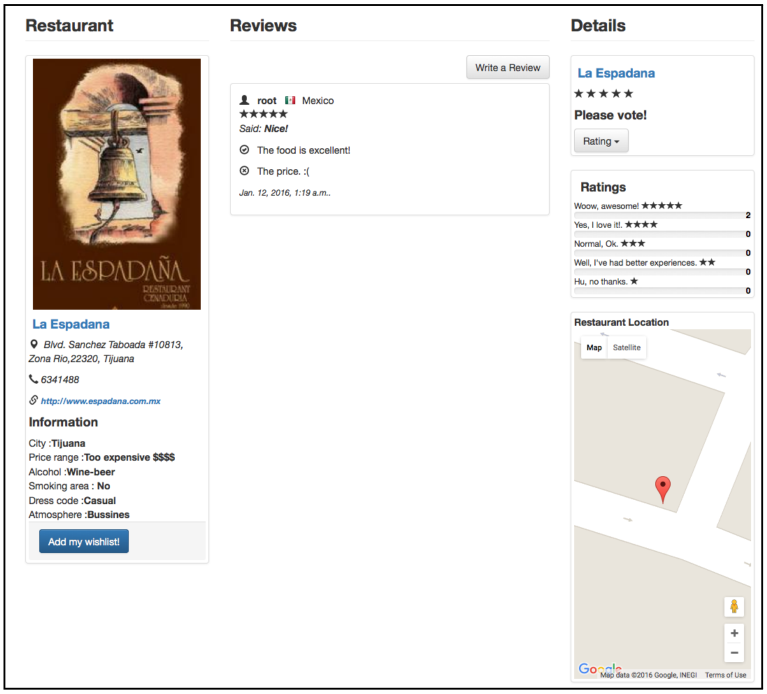


Figure 4.1: User interface of the restaurant model.

8) atmosphere type, and 9) cuisine type. An example of restaurant model in the context-aware recommender system is depicted in figure 4.1 with some domain values of the context represented by a binary vector where 1 means that the restaurant has the property that corresponds to the position value. Additionally, the restaurant model contains contextual information such as users's reviews, ratings average, and geographycal location.

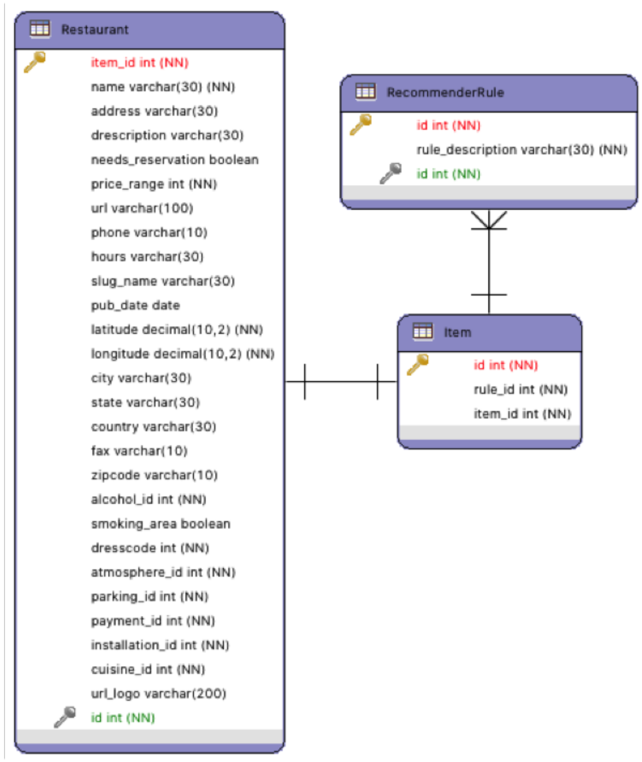


Figure 4.2: The data model of restaurant.

Data model

The data model in postgreSQL is depicted in the figure 4.2, the model contains the restaurant entity and its attributes. The restaurant entity is related to *Item entity* in a *"one-to-one"* relation that at the same time is related to the *RecommenderRule entity* which specifies some restrictions for item recommendations. A large view of all the entities related is depicted in the whole scheme refered in figure 4.5.

Some related entity corresponds to the proposed catalogues, that are defined as following:

- **Installations:** garden, terrace, indoor, outdoor.
- **Atmosphere:** relax, familiar, friends, bussines, romantic.
- **Parking:** no parking, free parking, valet parking.
- **Payment:** credit/debit card, cash.
- **Smoking area:** yes, no.
- **Price:** cheap, regular, expensive, too expensive.
- **Dresscode:** casual, informal, formal.
- **Alcohol:**no alcohol, wine-beer.
- textbfCuisine: japanese, chinese, italian, argentinean, cantonese, mandarin, mongolian, arabic, greek, spanish, brasilian, swiss, szechuan, asian, international, steak grill,vegetarian, natural/healthy/light, traditional mexican, tacos, mediterranean, middle eastern, american/fast food, gourmet, pizza, bar/beer, tapas cafe/bar, french, birds, seafood.

The cuisines were defined according the food variety of restaurants in Tijuana, there are 30 kinds of cuisines defined in the system.

The smoking area is the unique attribute with boolean value, it defines if a restaurant has a smoking area into its installation.

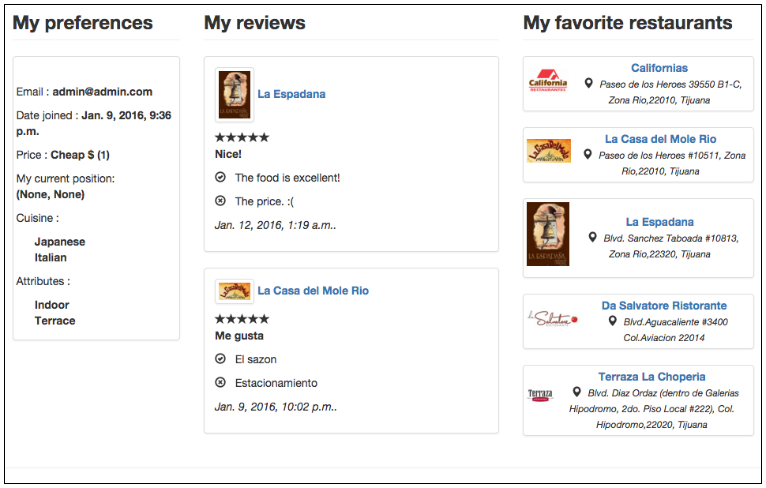


Figure 4.3: Example of user interface for user profile.

4.1.2 User model

The user's profile is derived from the ratings matrix. Let $U = [u_1, u_2, ...u_n]$ the set of all users and $I = [i_1, i_2,...i_n]$ the set of all items, if R represent the ratings matrix, an element $R_{u,i}$ represents a users rating $u \in U$ in a item $i \in I$. The unknown ratings are denoted as $\neq$. The matrix R can be decomposed into rows vectors, the row vector is denoted as $\overrightarrow{r_u}$=$[R_{u,1}...R_{u,|I|}]$ for every $u \in U$. Therefore, each row vector represents the ratings of a particular user over the items. Also denote a set of items rated by a certain user u is denoted as $I_u = i \in I|\forall i : R_{u,i} \neq \emptyset$. This set of items rated represents the user preferences where for each domain element $R_{u,i} \in [1 - 5]$ represents the intensity of the user interest for the item.

In context-aware recommender system, user profile has contextual information such as: 1) price range, 2) current location, 3) cuisine types, 4) attributes or features of restaurants that the user want, 5) the reviews posted, and 6) the favorite restaurants list. The user profile is stored in database and it is available for queries request, and it can be changed by users many times in a session. The information used to recommendations is the last one register stored. The user interface is represented in figure 4.3.

Data model

The user's data model in postgreSQL is represented in the figure 4.4, the model involves the entities: *User, UserProfile, and Friends. UserProfile entity* provides the contextual information of user, *User entity* is the default model defined in the system and is related to userProfile for suplies valuable information. The *Friends entity* represents the social aspect into the userProfile, Friends involves the users related to the current user taking in account the preferences of each other.

The user profile entity is related with 3 catalogues: price and cuisine are the same that in restaurant model, attribute groups corresponds to restaurant model mentioned (section 4.1.1). A total of 55 attributes(or features) could be contained in user profile, this information is used such as contextual information also. The domain values of the related catalogues are following:

- **Price:** cheap, regular, expensive, too expensive.

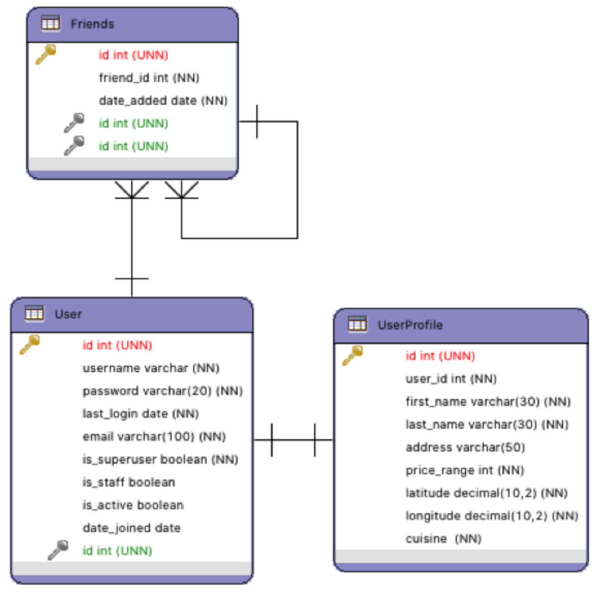


Figure 4.4: The data model of user profile.

- **Cuisine:** japanese, chinese, italian, argentinean, cantonese, mandarin, mongolian, arabic, greek, spanish, brasilian, swiss, szechuan, asian, international, steak grill,vegetarian, natural/healthy/light, traditional mexican, tacos, mediterranean, middle eastern, american/fast food, gourmet, pizza, bar/beer, tapas cafe/bar, french, birds, seafood.

- **Attribute groups:** Installations, atmosphere, parking, payment, smoking area, dresscode, alcohol.

4.1.3 Relational data model

A complete database relational scheme is represented in the figure 4.5. This model involves the whole database for context-aware recommender system, as well as the entities and relations among them.

The context is modeled as a relational database, each user context is a new register into data table to store user contexts.

Contextual information is also stored in the entities: *Reviews, CurrentLocation, DistancePoi and Ratings.* For instance, *Reviews entity* describes the users opinion about visited restaurants, this information contributes to have additional information about recent preferences of diners.

CurrentLocation entity stores the geographical position of user to get a *"nearby recommendation"*, the system locates restaurants around 2 kilometers from the user position. The position is changed frequently, in this manner, it allows the adaptation for each particular situation. *Distance Poi entity* stores the distances (kilometers) between the user and restaurants, this information is used to calculate *"nearby recommendation"*, each recommended restaurant ought be over the threshold defined. Finally, *Rating entity* represents the user preferences in a vector of scores, ratings could be increased in time and the user's preferences patterns could be changed in time also.

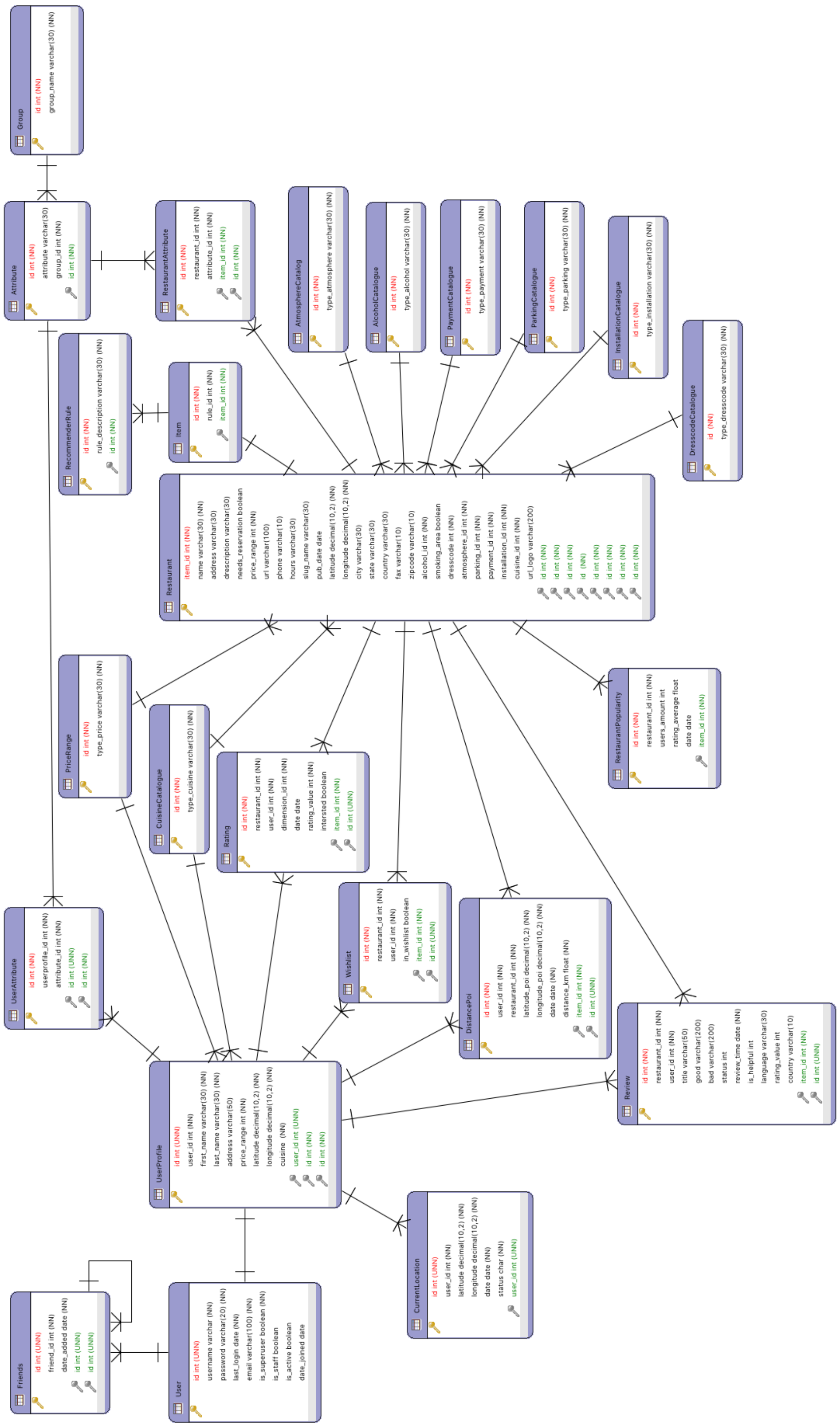


Figure 4.5: The relational database of context-aware recommender system.

4.2 Expert recomendation

Fuzzy logic is a methodology that provides a simple way to obtain conclusions of linguistic data. Is based on the traditional process of how a person makes decisions based in linguistic information.

Fuzzy logic is a computational intelligence technique that allows to use information with a high degree of inaccuracy; this is the difference with the conventional logic that only uses concrete and accurately information [46].

In this work, fuzzy logic is used to model fuzzy variables that highligh in the popularity of a restaurant. The context-aware recommender system has implemented a Fuzzy Inference System that represents the expert recommendation.

The expert Fuzzy Inference System generates recommendations when the recommendation techniques (collaborative filtering, content-based) are not getting results because of the cold start problem.

The Fuzzy Inference System proposed has 3 **input variables** (such as in previous work realized[22]): 1)*rating* is an average of ratings that has a particular restaurant inside the user community; the domain of variable is 0 to 5 and contains 2 membership functions labeled as *low* and *high*(figure 4.9a), 2)*price* represents the kind of price that has a particular restaurant; the domain of variable is 0 to 5 and contains 2 membership functions labeled as *low* and *high* (figure 4.9b), and 3)*votes* is used to measure how many items have been rated by the current user, i.e., the participation of the user, if the user has rated few items (less than 10) is not sufficient to make accurate predictions(figure 4.9c), the domain of variable is 0 to 10 and contains 2 membership functions labeled as *insufficient* and *sufficient*.

The **output variable** is *recommendation*, represents a weight for each restaurant proposed by the expert considering the inputs mentioned above, the domain of variable is 0 to 5 and contains 3 membership functions labeled as *low*, *medium* and *high* (figure 4.10c).

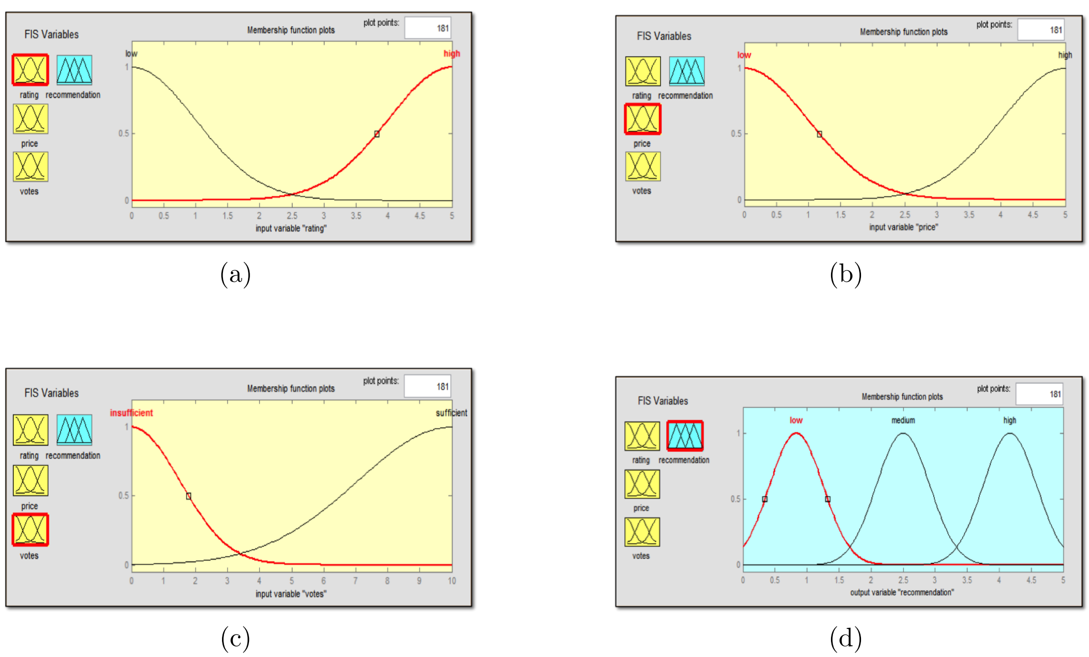


Figure 4.6: The Gaussian membership functions of the expert system.

The proposed Fuzzy Inference System(figure 4.7) represents the users experience and their knowledge about restaurants. This factors are considered important for users that visiting a restaurant. This information is recovered of user profile and restaurant profile; and the system uses this information to get weights that influence in the final recommendations. The Fuzzy Inference System uses 5 inference rules that involve the variables of inputs and output.The input variables determine the recommendation activation; each input variable contains labels as *low* and *high* that also correspond to memberships functions of Gaussian type. For the output variable *recommendation* the labels *low*, *medium*, and *high* are used with membership functions Gaussian type also. The rules are:

1. *If **rating** is high and **price** is low then **recommendation** is high.*
2. *If **rating** is high and **votes** is sufficient then **recommendation** is high.*
3. *If **rating** is high and **votes** is insufficient then **recommendation** is medium.*
4. *If **rating** is low and **price** is high and then **recommendation** is low.*
5. *If **rating** is low and **votes** is insufficient then **recommendation** is low.*

4.3 Fuzzy Inference System to assing weights

The main goal of this Fuzzy Inference System is to define weights for each recommendation list. The recommendation technique is based in the amount of available

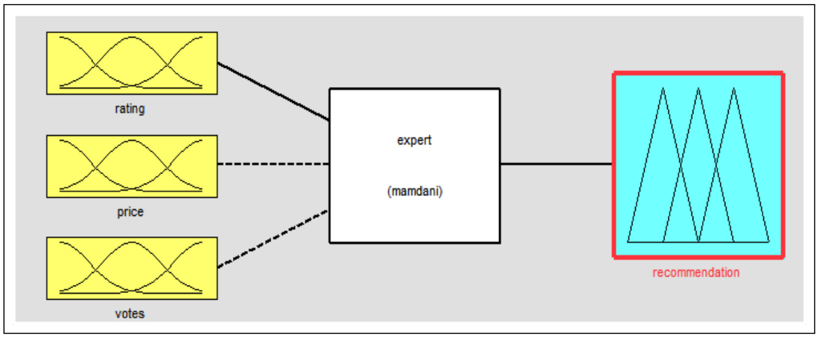


Figure 4.7: Fuzzy Inference System of expert.

information stored, so each technique utilizes this information to provide a list of restaurants as well as a weight for each one, therefore, these is used for recommendations if its weight is upper the threshold. The Fuzzy Inference System has inputs and outputs to infer each list's weight, its variables are depicted in figure 4.8. There are 3 membership functions for inputs and 3 for outputs. The input variables are: *userSimilarity, restaurantSimilarity and Participation* and are depicted in figure 4.9. The (4.9.a) and(4.9.b) are in a range from 0 to 1 to define the similarity average among users and restaurants, respectively. The figure (4.9.c) has a range from 0 to 15 to represent the ratings of the user(participation). This information is stored in the Popularity entity (see figure 4.5).

By other side, the output variables are: *Expert, RestaurantProfile and Correlation*, these are depicted in figure 4.10. The figure (4.10.a) represents the weight for expert recommendation list, figure (4.10.b) represents the weight of the content-based list and figure (4.10.c) represents the weight of collaborative recommendation list, their

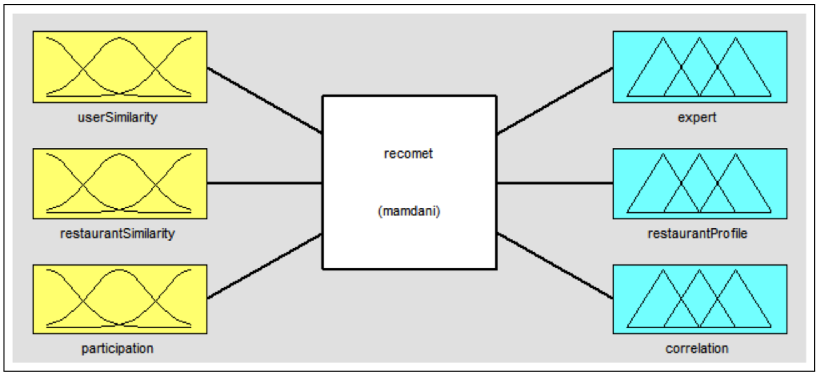
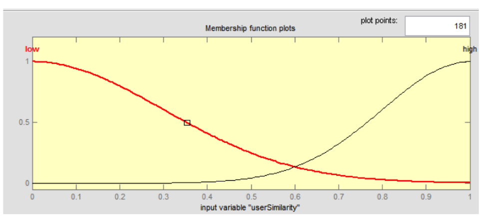
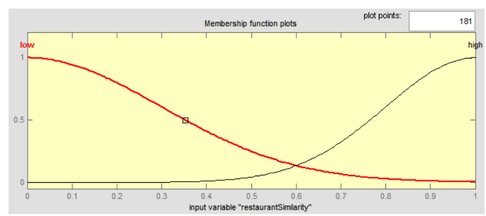


Figure 4.8: Fuzzy Inference System to assign weights.

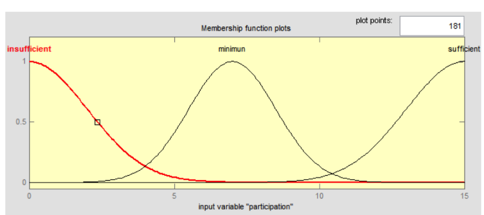
membership functions are in a range from 0 to 1 to get the value. Taking in account



(a)



(b)

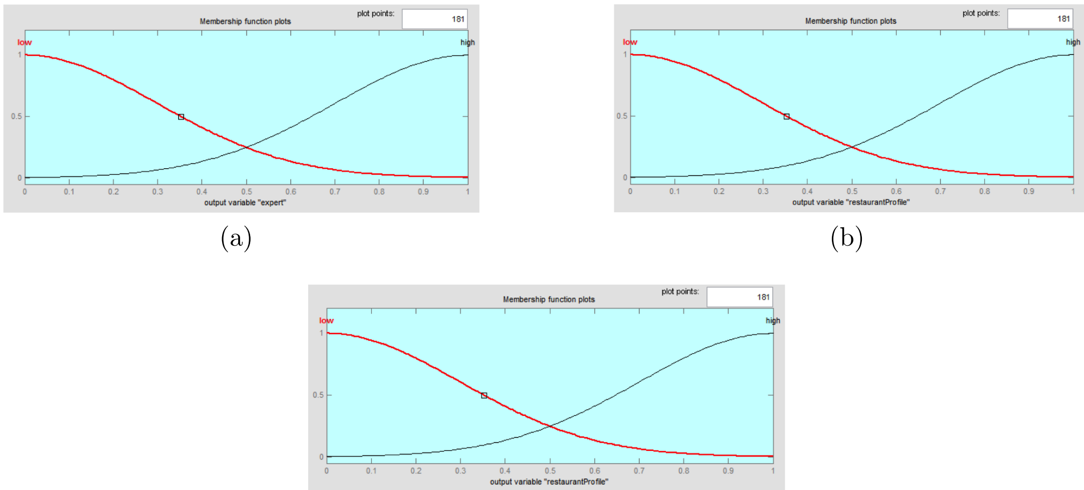


(c)

Figure 4.9: The Gaussian membership functions of input variables.

the input variables, the rules utilized to infer the output values are following:

1. *If* ***userSimilarity*** *is low and* ***restaurantSimilarity*** *is low and* ***participation*** *is insufficient then* ***expert*** *is high,* ***restaurantProfile*** *is low,* ***correlation*** *is low.*



(a) (b)

(c)

Figure 4.10: The Gaussian membership functions of output variables.

2. *If* ***userSimilarity*** *is low and* ***restaurantSimilarity*** *is low and* ***participation*** *is sufficient then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

3. *If* ***userSimilarity*** *is low and* ***restaurantSimilarity*** *is low and* ***participation*** *is minimun then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

4. *If* ***userSimilarity*** *is low and* ***restaurantSimilarity*** *is high and* ***participation*** *is insufficient then* ***expert*** *is low,* ***restaurantProfile*** *is high,* ***correlation*** *is low.*

5. *If* ***userSimilarity*** *is low and* ***restaurantSimilarity*** *is high and* ***participa-***

tion *is minimun then* ***expert*** *is low,* ***restaurantProfile*** *i s high,* ***correlation*** *is low.*

6. *If* ***userSimilarity*** *is low and* ***restaurantSimilarity*** *is high and* ***participation*** *is sufficient then* ***expert*** *is low,* ***restaurantProfile*** *is high,* ***correlation*** *is low.*

7. *If* ***userSimilarity*** *is high and* ***restaurantSimilarity*** *is low and* ***participation*** *is insufficient then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

8. *If* ***userSimilarity*** *is high and* ***restaurantSimilarity*** *is low and* ***participation*** *is minimun then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

9. *If* ***userSimilarity*** *is high and* ***restaurantSimilarity*** *is low and* ***participation*** *is sufficient then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

10. *If* ***userSimilarity*** *is high and* ***restaurantSimilarity*** *is high and* ***particicipation*** *is insufficient then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

11. *If* ***userSimilarity*** *is high and* ***restaurantSimilarity*** *is high and* ***participa-***

tion *is sufficient then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

12. *If* ***userSimilarity*** *is high and* ***restaurantSimilarity*** *is high and* ***participation*** *is minimun then* ***expert*** *is low,* ***restaurantProfile*** *is low,* ***correlation*** *is high.*

4.4 Contextual Recommendation

The interface of the system(figure 4.11) allows to collect contextual information such as type of price, restaurant's attributes, type of cuisine and geographical location. The context- aware recommender system uses pre-filtering paradigm, then the contextual information is used for adjust the final recommendations list. For example, geographical location is used to get restaurants around 2 kilometers of distance, next, the list of nearby restaurants is displayed for the user. If context-aware recommender system considers another attributes as type of price and type of cuisine preferred by the user, the system gets restaurants matched in the context especified by the user in this time. In the attributes box, the user can chose any preference about what things are importants to select a restaurant. The features are collected from the dataset of Tijuana restaurants. In the cuisine box, the user choosen his/her favorite cuisine, it can be one or more cuisines such as in attributes also.

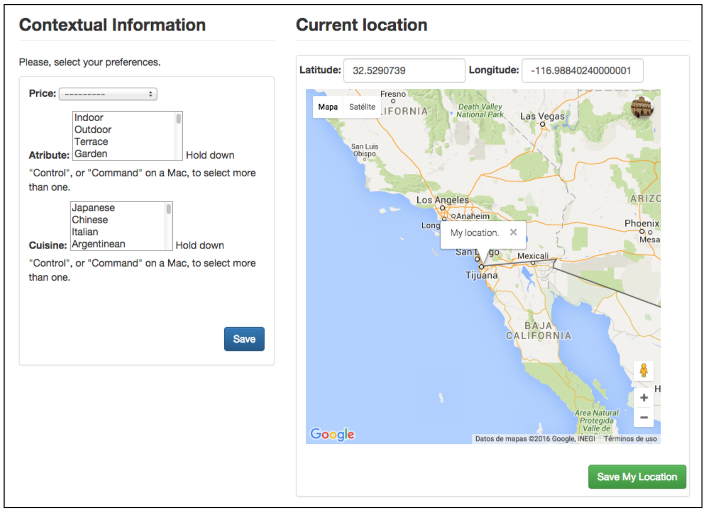


Figure 4.11: System interface to collect contextual information.

The context changes constantly, indeed, the users migh change it many times such as them wish. After the post-filtering, the system displays the recommended restaurants according the information provides by the user. The context-aware recommender system contains 4 techniques to display recommendations. The interface in figure 4.12 shows recommendations: *1) Expert, 2) Content- based, 3) Collaborative filtering and 4) Nearby.* Each one was explained above, except the nearby recommendations. For nearby recommendations the system calculates the approximate distance between the current geographical location of the user and the available restaurants in the area. The threshold is 2 kilometers around the user position to determine what restaurants will be recommended. The geographical position is obtained throught Google maps.

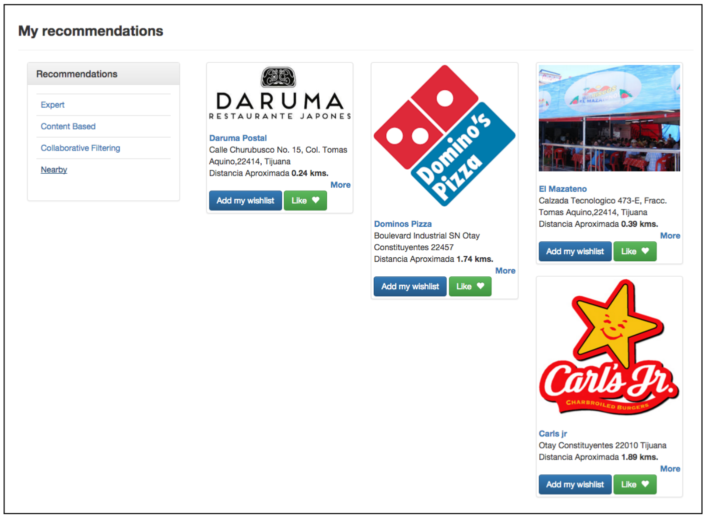


Figure 4.12: System inferface of recommendations for the user.

4.5 Architecture

The architecture for poposed method is depicted in the figure 4.13. In the first part, the three techniques of recommendations are suplied by the rating matrix to obtain the recommendation list. Ratings matrix makes that Fuzzy Inference System can obtain the inputs values to calculate the output value. The Content-based utilizes the rating matrix and user profiles to compare the similarity among the restaurants through cosine similarity. The collaborative filtering is based in rating matrix (user profiles) to predict ratings for restaurants using Pearson correlation to get the K neighbors.

The second part shows the recommendation lists for the user getting of each algorithm. Subsequently, the recommendation lists are reduced when filter context is

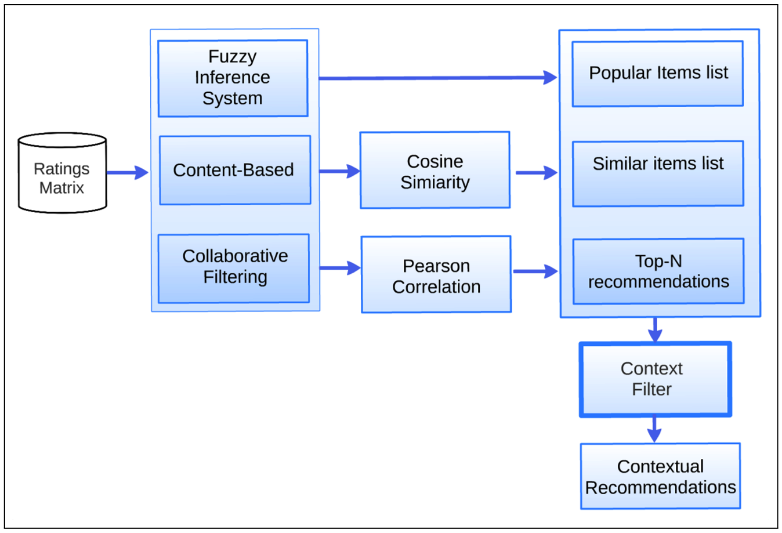


Figure 4.13: Context-aware recommender system architecture.

applied, i.e., the recommendations are adjusted for the user current context. Finally, the contextual recommendations list is displayed in the user interface (figure 4.12).

Chapter 5

Experiments and Results

In this chapter the results of the experiments described in chapter 4 are presented.

5.1 EvoDrawings01

In the experiment EvoDrawings01 the following data are generated, this data is presented in Table 5.1.

These data are the total of nodes and relations generated in EvoDrawings01.

The total number of active users who participated voluntarily meners are pre-

Table 5.1: Data generated in graph based user modeling.

	Data
Nodes	595
Relations	2220

Table 5.2: Total number of volunteers active users.

	Users
Total number of users	54

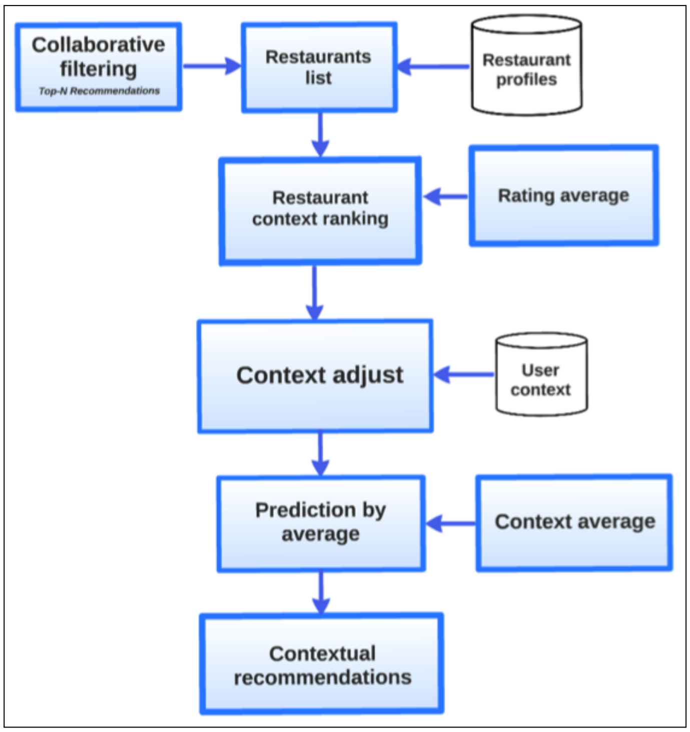


Figure 5.1: The Post-Filtering architecture for Tijuana restaurants.

sented in Table 5.2.

In order to validate the proposed approach, data about restaurant preferences of users in different contexts was collected. The study subjects were students enrolled in a computer engineer major, a masters program and professors of the Tijuana Institute of Technology. A total of **50 users** answered a questionnaire; the questions were about their preferences for nearby restaurants and the technology used by them.

The *questionnaire* consisted of **8 questions** and they rate restaurants from a list of 40 restaurants. Each restaurant chosen was rated 6 times one per context considered, a matrix rating with *1,422 ratings* were collected. The questions are shown in the table **??**.

The user's answers from question 1 to question 6 are represented in the figure 5.2. **Figure 5.2a** represents the percentage of surveyed students and teachers; **figure 5.2b** the percentage of the element that users consider the most important to visit a restaurant; **figure 5.2c** represents the preferences of devices when are using Internet for restaurant recommendations; **figure 5.2d** represents the percentage of operating system that often used, **figure 5.2e** shows the percentage of users that use the Internet to search restaurants in Tijuana; and **figure 5.2f**, shows the percentage of users that would like using a restaurant recommender system of Tijuana. For questions 7 and 8 only the top-ten restaurants are shown, without/with the contextual situation. In figure 5.3a, the favorite restaurant is **Daruma**(178 votes), whereas in figure 5.3b, **Daruma** does not appear in the top-ten. When considering the context *midweek*, the favorite restaurant was **Carls Jr.**, which appears in both graphs; this restaurant was also the most voted in the different contexts. Contextual recommendations of post-filtering approach depends of context *midweek* or *weekend*, which is the day when the restaurants were rated. Subsequently, the result of the query is refined according to the user context; the 6 contexts mentioned correspond

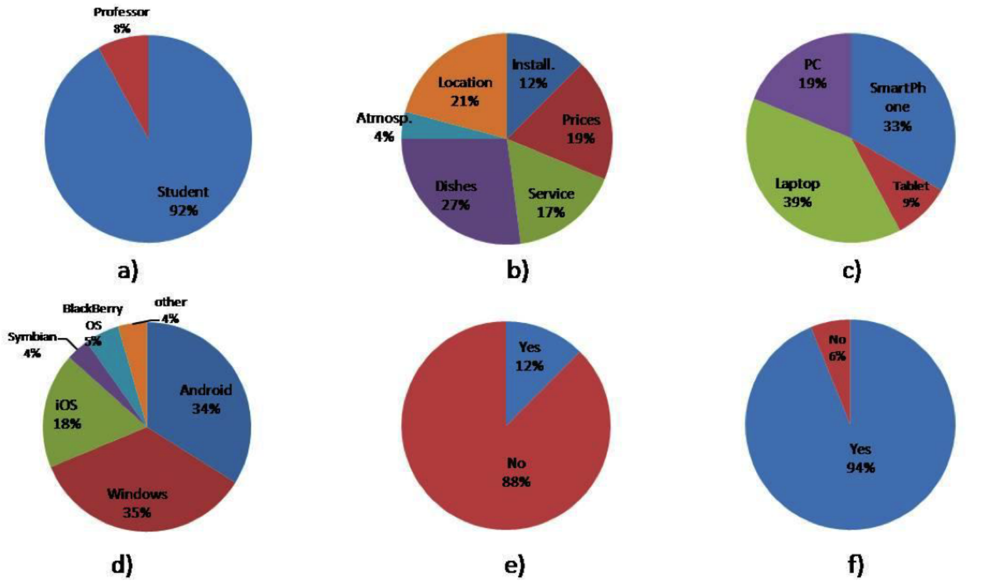


Figure 5.2: The chart shows the users preferences for questions from 1 to 6.

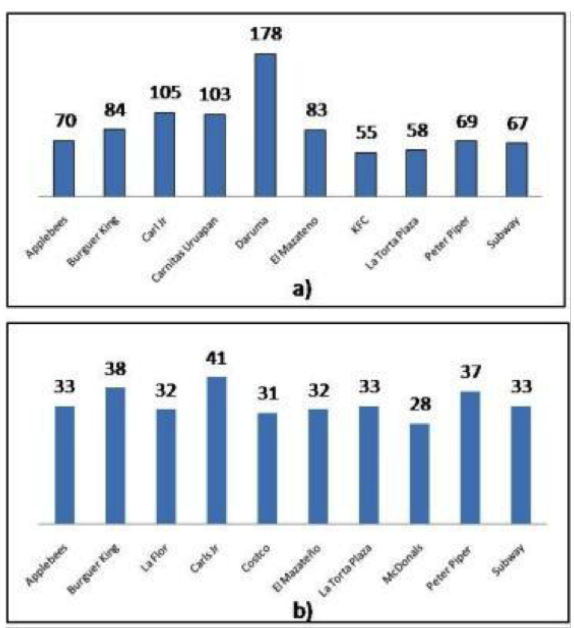


Figure 5.3: The chart shows the users preferences for questions 7 and 8.

Table 5.3: Contextual factors considered in the questionnaire.

Contextual Factor	Context
Day	1.Midweek(Monday, Thuesday,Wednesday and Thursday) 2.Weekend(Friday,Saturday and Sunday)
Place	1.School 2. Home 3.Work

to combinations of contextual factors shown in table 5.3. The dataset was explicitly collected from **50 users** whom answered questionnaire (see table **??**). A total of 172 predictions was made for different users and the error **MAE=0.5859** when the context **midweek** for current user was considered. The observation for this result is that using a small dataset the performance of the method proposed is limited. By other hand, having only one contextual factor does not improve the accuracy of the recommendations in this domain.

5.2 MovieLens dataset

GroupLens Research has collected and made available rating data sets from the MovieLens web site (http://movielens.org).

The data sets were collected over various periods of time, depending on the size of the set.

- **MovieLens 1M Dataset**: Stable benchmark dataset, 1 million ratings from

6000 users on 4000 movies. Released 2/2003.

Downloaded from *http://grouplens.org/datasets/movielens/1m/*.

- **MovieLens 10M Dataset**: Stable benchmark dataset, 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Released 1/2009.

  Downloaded from *http://grouplens.org/datasets/movielens/10m/*.

The recommender system proposed for MovieLens uses post-filtering and time segmentation. Time in recommender systems is used as a contextual factor in the research reviewed [9], [8], [28], and [25], results vary according the techniques that were done.

In [25] the pre-filtering approach was used, time was divided in time intervals and the size of time intervals is directly proportional to the distance from initiating the historical information to the current user context. In [28] a tracking model of user behavior over the life-time of data is proposed, in order to exploit the relevant components of all data instances , while discarding only what is modeled as being irrelevant.

In [9] it is shown that the time division is beneficial and its performance depends on the items selection method and influence of contextual variables in item ratings. In [8] the user profile is segmented into micro-profiles corresponding to a particular

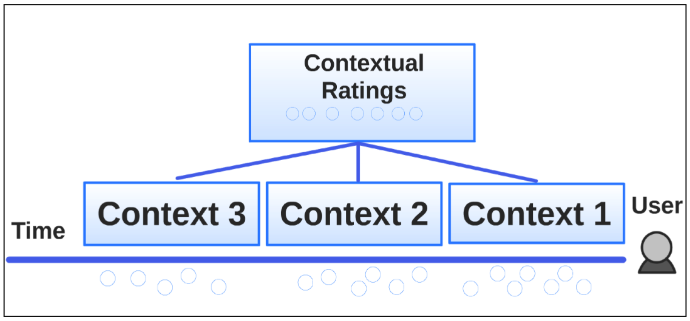


Figure 5.4: Time segmentation of contexts based on current user context.

context, each context represents a time span in which recommendations for users are derived.

This experiment implements fuzzy logic on time segmentation, in order to improve user satisfaction by providing recommendations based to context and recents user preferences without discarding tastes in the past, as they include important information for the recommender system proposed. The first phase is division of three time segments based on the current context of the user is performed such as in is depicted in figure 5.4.

In recommender system, the first step is get the current user context (user-application interaction), from this information three contexts (figure 5.4) will be obtained that representing a time segment of three months each one, in total the algorithm considers all the ratings users did during nine months prior the current context. Subsequently, ratings are classified by contexts and reused as contextual rating matrix being, a ratings matrix for each context.

The size of matrix depends of users participation during the last nine months. One of the aims is to identify the user behavior through recent information, in order to, for instance, know whether the user changes ratings constantly; whether usually assign high, low or mixed ratings; whether user likes to see different items or whether have a favorite category.

Recommender systems use the collaborative filtering algorithm in order to find relevant items for the user [43]. User's profiles are used for determine the similarity between users calculated with Pearson correlation. The similarity between users can provide valuable information as long as user participation is enough (less than 10 ratings). The next step is to obtain recommendations list (Top-N), three contextual lists are the outputs of collaborative filtering algorithm and contain items with user's predictions for each context.

Popularity's prediction considers other variables: 1) users participation in respect of an item in the context and, 2) the rating's average that users have given for item in the same context.

A Fuzzy Inference System (FIS) uses these parameters to assign a weight within a scale from 1 to 5 (prediction value). These recommendations are used when the ratings matrix is sparse, a popularity prediction is done.

Finally, the system gets the recommendations list for each user in different contexts. The recommendation process of pre-filtering is depicted in figure 5.5. The dataset

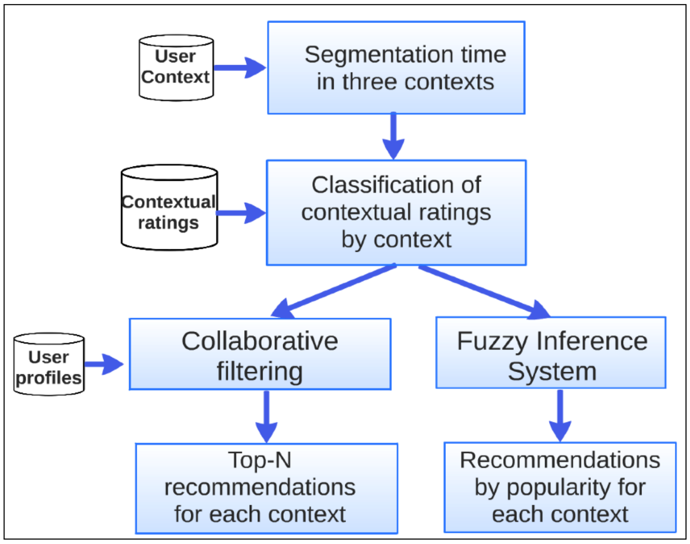


Figure 5.5: Pre-filtering process for context-aware recommender system.

used to test the algorithm was MovieLens(100000 ratings) with 943 users and 1,682 movies. The ratings were collected in a period of 2 years. MovieLens is not a contextual dataset, however, the timestamp was used to determine the rating time, i.e., in this way it was noted the day to know whether the rating time was in weekday or weekend. In this terms, the context was used. Then, the time for each context was divided in 3 months each one, this span covers 9 months before the user's current context.

The neighbors in each context are considered to recommend movies in that context only. An average of predictions are considered for add a movie to the top-N list of contextualized recommendations.The result in table 5.4 shows the error in three contexts. The error increase in context 3, in this context the ratings matrix is a

Table 5.4: Results of comparison by contexts in MovieLens dataset.

Context	# Preditions	MAE
1	12235	0.28
2	21049	0.24
3	1075	0.38

little bit sparse; the error is justifiable because user has less participations.

5.3 Tripadvisor dataset

The dataset used to evaluate the algorithm was TripAdvisor in two versions downloaded [47], this datasets was used in [49], [48] to evaluate the performance of context-aware recommender systems.

The first dataset contains 4669 contextual ratings, 1202 users and 1890 hotels; the second dataset contains 14175 contextual ratings, 2731 users and 2269 hotels. Data were collected of reviews online in tripadvisor.com. There is only one context: type of trip (family, friends, bussines, romantic and relax).

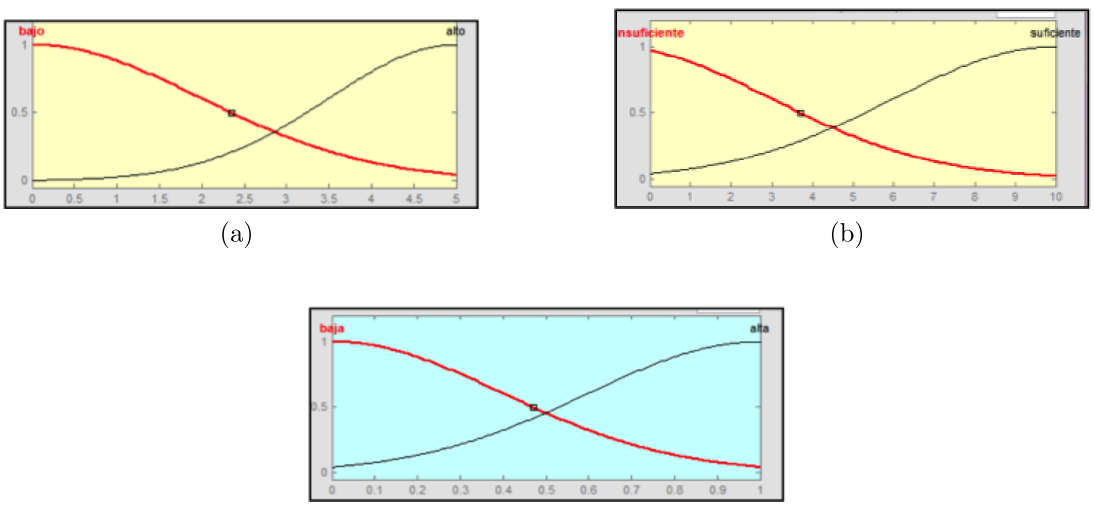
The proposed method consists of three algorithms to recommend: Fuzzy Inference System, collaborative filtering and content-based. Each one uses rating matrix to get recommendations.

The context-aware recommender system uses the post-filtering paradigm[2] for adjust recommendations in context. The recommendation by popularity is through the

Fuzzy Inference System depicted in figure 5.7, the Fuzzy Inference System contains the variables that are involved in the process to recommend in a human interaction, this process is the same that the recommender system does.

The output represents how matter each item into the users community, i.e. if it was a popular item for users.

The FIS has Gaussians membership functions and are depicted in figure 5.6. The



(a) (b) (c)

Figure 5.6: Gaussian Membership functions in the input are: a) RatingAverage, b) UserParticipation, and an output: c) Recommendation.

Fuzzy Inference System uses fuzzy rules to infer the inputs and output (a numeric value) that represents the weight of the recommendation. The rules are following:

1. *If* ***RatingAverage*** *is low and* ***UserParticipation*** *is insufficient then* ***recommendation*** *is low.*

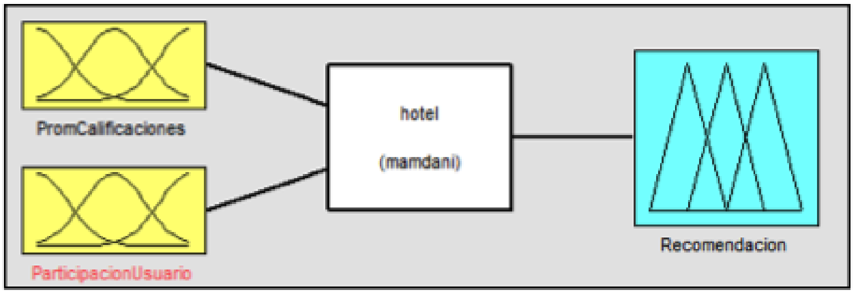


Figure 5.7: Fuzzy Inference System.

2. *If* ***RatingAverage*** *is low and* ***UserParticipation*** *is sufficient then* ***recommendation*** *is high.*

3. *If* ***RatingAverage*** *is high and* ***UserParticipation*** *is insufficient then* ***recommendation*** *is low.*

4. *If* ***RatingAverage*** *is high and* ***UserParticipation*** *is sufficient then* ***recommendation*** *is high.*

Content-based uses cosine similarity to compare the binary vectors representing the profile of each item, thereby obtaining a numerical value that determines similarity, based on a threshold.

In other words, it makes a comparison of profiles of each item to determine the most similar to items the user has rated with highest score, context-aware recommender system proposed has a scale from 1 to 5. In the next step the outputs of every recommender algorithm is represented by a list of recommended items. Subsequently applies the context filter and context-aware recommender system gets the final con-

Table 5.5: Example of contextual ratings in the user profile.

User profile		
Item1	Rating1	Context1
Item2	Rating2	Context2
Item3	Rating3	Context3

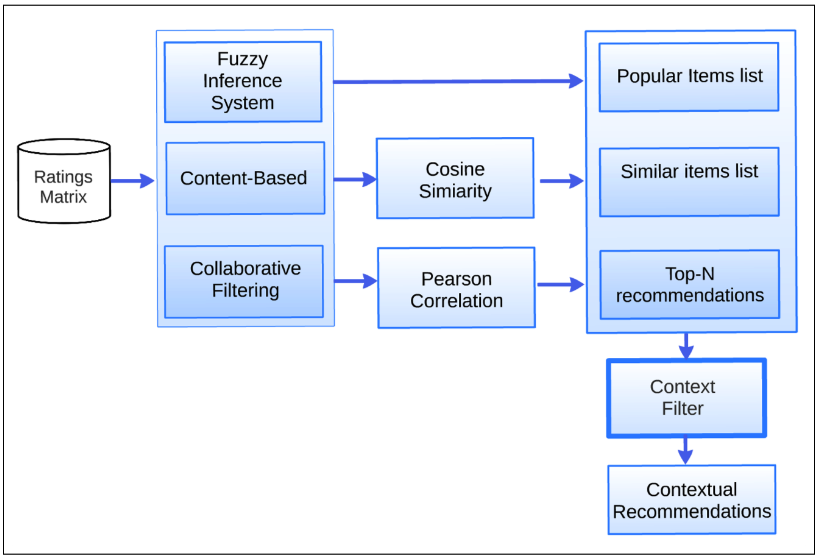


Figure 5.8: Recommender system architecture

textual recommendations.

Context-aware recommender system identifies contextual data of the user profile (see table 5.5), and compares recommended items to filter those items that are adjusted to the user context.

The context filtering is the last step before to get the recommended items. The schema of architecture for context-aware recommender system is depicted in figure 5.8. Two experiments were performed using TripAdvisor dataset, table 5.6 describes the data sets and the scarcity percentage of the specified data. Scarcity of 99% mean

Table 5.6: Datasets description.

Dataset	Users	Items	Ratings	Scarcity (percent)
TripAdvisor v1	1202	1890	4669	99.79
TripAdvisor v2	2731	2269	14175	99.77

Table 5.7: Comparison of RMSE.

Dataset	Algorithm	RMSE
TripAdvisor v2	FC + Post-filtering	0.504
	FC	0.994
	Pre-filtering + Relaxation	0.985

that there are problems to recommend items because the information is not enought to get good recommendations.

By other side, in table 5.7 the comparison shows that the algorithm has a acceptable performance, i.e., the error falls into the range of results obtained with others algorithms. Then, contextual recommendations were evaluated with the Root Mean Square Error in order to compare the results with context relaxation algorithm[48] that is evaluated with the same dataset.

The fundament of content-based is the cosine similarity; this means that if similarity value among items is high, the recommendations will improve the degree of user satisfaction. This is observed when calculating the similarity average in each dataset as shown in table 5.8.

FIS can provides a list of popular items for each dataset, recommendations through averages are obtained, and recommendations are conditioned to show it

Table 5.8: Level of similarity among items in datasets.

Dataset	Similarity	Avg.votes per user.
TripAdvisor v1	0.448	5
TripAdvisor v2	0.508	8

when the collaborative filtering and content- based are not delivering recommendations because of data scarcity. However, the majority of popular items of dataset were rated in contexts: romantic, family and bussines, that means that the dataset has biases.

In this experiment the context-aware recommender system proposed involves the paradigm of post-filtering for contextual recommendations. The structure of the datasets facilitated the evaluation of recommendations although the rating matrix has been scarce in both cases. Anyway, information of items and users was used to test the system and a good performance of the system was done.

With respect the performance, post-filtering allows select relevant items that are adjusted into the context, indeed, post-filtering and implementation of different recommendation techniques the system has suitable performance and the datasets help the processes performed.

Table 5.9: Contexts in InCarMusic dataset.

Context	Values
Driving style	elaxed, driving, sport driving.
Road type	city, highway, serpentine.
Landscape	coast line, country side, mountains/hills, urban.
Sleepiness	awake, sleepy.
Traffic conditions	free road, many cars, traffic jam.
Mood	active, happy, lazy, sad.
Weather	cloudy, snowing, sunny, rainy.
Natural phenomena	day time, morning, night, afternoon.

5.4 Datasets in matrix factorization

Filmtrust dataset

FilmTrust is a small dataset crawled from the entire FilmTrust website in June, 2011. Filmtrust contains a ratings matrix of 35498 ratings, 1504 users and 2071 movies. The dataset has a density of 1.14% and was used in [24] using the trust level such as context. The web page is *http://www.librec.net/datasets.html*.

InCarMusic dataset

InCarMusic dataset[10] has 8 contextual factors and the possible values for contextual conditions are explained in table 5.9. Music tracks were ten different genres. There is not unified music genre taxonomy, for this reason the recommender system uses the genres defined in [42]: classical, country, disco, hip hop, jazz, rock, blues, reggae, pop and metal, 50 music tracks and 42 users in dataset.

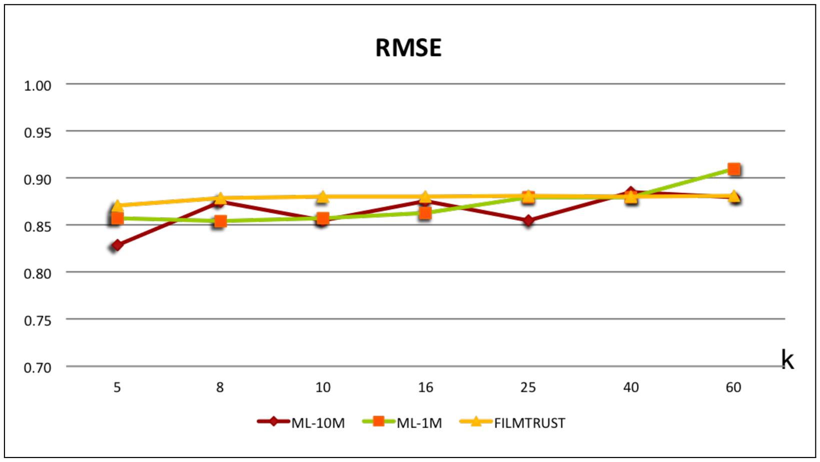


Figure 5.9: RMSE results of matrix factorization test.

5.4.1 Results

For experiments with matrix factorization technique the Graphlab toolbox was used. Both mentioned datasets and Movielens (1 millon and 10 millions) were used to test the algoritmh. The test involves K factors that are increasing for 50 iterations. previously, was done a test to identify what number of iterations are enough to get a good result with no overload of process in the algorithm. Results are depicted in the chart 5.9 where the axis (x, y) represent the K value and the error value, respectively. The observations deal to small differences among the datasets, in a range of 0.80-0.90, and the high variability is in MovieLens dataset 10 millions. The big dataset implies more unstable behaviour, while in a small dataset (Filmtrust) the error is less variable. A comparison among MovieLens 1 million and 10 millions shows that there's not a significant difference.

Table 5.10: RMSE of datasets using matrix factorization.

Dataset	Ratings	Cosine Sim.	RMSE
Tijuana Rest.	896	0.67	0.60
Mexico Rest.	1161	0.25	0.54
InCarMusic	4012	0.45	0.93
TripAdvisor	4669	0.17	0.85
MovieLens	10000	0.46	0.51
Movielens	100000	0.94	0.42

By other side, other datasets were used to test matrix factorization under the same parameters to calculate the RMSE for each one. Table 5.10 presents the total of ratings of each dataset, the cosine similarity, it means how similar are the items into the dataset, and the RMSE error obtained in the test with matrix factorization technique. The datasets contain less ratings than the presented in the chart 5.10, according the table 5.10 is not possible to assum that matrix factorization has a better performance with small datasets, because TripAdvisor and InCarMusic datasets obtain an error in the same range that the large datasets of the previuos chart.

Chapter 6

System evaluation

6.1 Metrics

To evaluate context-aware recommender system was used the **task success** and **time-on-task** metrics.

The **task success metric** is perhaps the most widely used performance metric. It measures how effectively users are able to complete a given set of tasks. The **time-on-task metric** is a common performance metric that measures how much time is required to complete a task[4].

Task success is something that almost anyone can do. If the users cant complete their tasks, then something is wrong. When the users fail to complete a simple task can be an evidence that something needs to be fixed in the recommender system.

The usability test consist of a list of simple tasks for users that they shall perform in the system to complete the test. Before to start, a minimal description about the system for user was explained. The tasks list are the following:

1. *Rated a restaurant without context.*
2. *Add context to the user profile.*
3. *Filter restaurants by favorite context.*
4. *Find information of a specific restaurant.*
5. *Find all the reviews of a specific restaurant.*
6. *Find section of my favorite restaurants.*
7. *Add a review of a restaurant.*
8. *Find the most popular restaurants.*
9. *Add a restaurant to your wishlist.*
10. *Get recommendations based on expert opinion.*
11. *Get the recommendations content-based.*
12. *Get the collaborative recommendations.*
13. *Get recommendations of the nearby restaurants.*

6.2 Enviromental set up

Each user did the task list, one by one, with previous instructions. It gives a brief explanation about the general features of system before to start. The time average for each user was around 10 minutes to finished all activities without disruptions. After, the results was depicted in a chart to observe the user behaviour for each task, in the figure 6.1 the axis (x, y) represent the task number and percent of success, respectively. The chart shows that only 3 tasks werent accomplished successfuly, the task 5, 6 and 7.

The issue with task 5 was that users can not found easily the reviews section in the interface, the issue in task 7 is derived of task 5 because the user couldnt find the manner to add a review. The task 6 correspond to the favorite restaurants, but the issue is that it was confused to chose favorite restaurants in place of wishlist section. In general, these results mean a possible redisign in the interfacte to facilitate the performance of these tasks. The time it takes a participant to perform a task says a lot about the usability of the application. In almost every situation, the faster a participant can complete a task, the better the experience. In fact, it would be pretty unusual for a user to complain that a task took less time than expected [4]. Then, task-on-time was applied to measure time that an user did the task. A resume of the time tasks for each user it is in table 6.1, *null* values mean that the user didn't

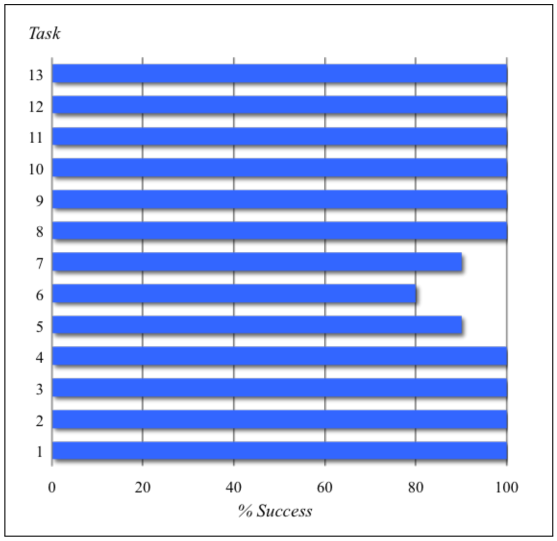


Figure 6.1: Representation of the percent of success for each task.

the task.

6.3 Results

To measure the efficiency of the metric it was chose an confidence interval. In this way, it is observed the time variability within the same task and also helps visualize the difference across tasks to determine whether there is a statistically significant difference between tasks. The obtained information is in table 6.2, the median was used to calculate the confidence interval. In the next step the USE *(Usefulnes, Satisfaction, and Ease of Use)* questionnaire [32] was applied in order to get the user's feedback and comments for to know about the difficults in the test. The USE questionnaire consists of 30 rating scales divided into 4 categories:

Table 6.1: Time on task data for 10 users and 13 tasks.

Task	Us1	Us2	Us3	Us4	Us5	Us6	Us7	Us8	Us9	Us10
1	12	28	24	30	19	33	23	16	5	7
2	3	4	17	5	17	134	9	16	12	11
3	123	69	159	53	69	113	44	41	70	98
4	20	4	86	40	13	4	17	3	20	3
5	50	10	63	50	7	11	10	5	20	Null
6	10	30	28	27	5	46	Null	7	Null	34
7	10	20	16	8	15	Null	9	24	16	28
8	18	24	10	10	5	3	27	4	5	6
9	5	6	31	4	45	9	12	5	3	8
10	15	17	15	11	10	19	13	10	20	20
11	30	15	20	16	20	22	15	13	18	20
12	12	14	19	14	40	10	17	17	15	15
13	25	15	15	14	10	10	11	10	10	25

Table 6.2: Confidence interval per task with a confidence level of 95%.

Task	Median	CI 95%	Upper bound	Lower bound
1	20	5.96	25.96	14.04
2	11.5	0.81	12.31	10.69
3	69.5	25.57	95.07	43.93
4	15	16.34	31.34	-1.34
5	15.5	14.84	30.34	0.66
6	27.5	11.57	39.07	15.93
7	16	5.19	21.19	10.81
8	8	5.80	13.80	2.20
9	7	9.43	16.43	-2.43
10	15	2.44	17.44	12.56
11	19	3.00	22.00	16.00
12	14.5	5.51	20.01	8.99
13	12.5	3.89	16.39	8.61

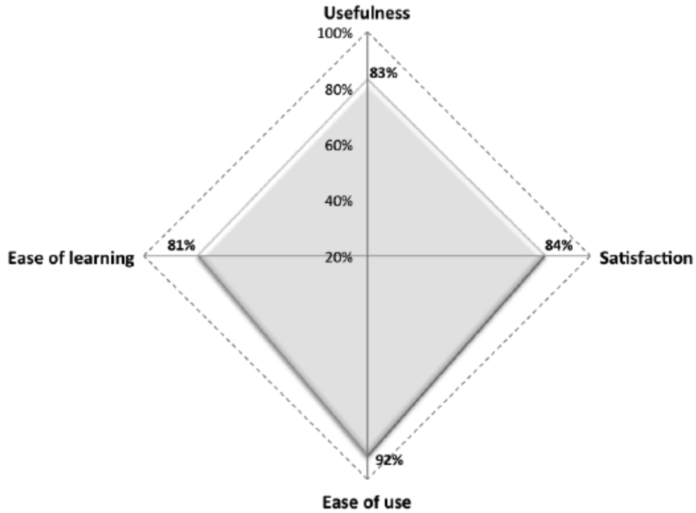


Figure 6.2: The radar chart that depicts the four axis evaluated in the questionnaire.

Usefulness, Satisfaction, Ease of Use, and Ease of Learning. Each is a positive statement to which the user rates level of agreement on a 7-point Likert scale. The USE questionnaire(see appendix B) allows to get values for Usefulness, Satisfaction, Ease of Use, and Ease of Learning, the visualizing the results is in the Fig.6.2 , where the four axis of the radar chart represent the values of percent which users rated positively this factors with respect to their interaction with the context-aware recommender system. The accurate values are *Usability 83%, Satisfaction 84%, Easy of use 92%, and Easy of Learning 81%.*

Chapter 7

Conclusions and future work

We observed the users behaviour to identify the most frecuently difficults and doubts about tasks. We did a brief interview with users after the test in order to understand their feelings or mood, their ideas about the experience, and overall, their opinion about the context-aware recommender system. The conclusions are based in user's comments, then the main errors in the system interface are summarized in three points:

1. Incomplete information for user, i.e., the system doesn't had enough and clear information to be a friendly interface, and therefore the user couldn't do easily a task.

2. Fails in design, because of unordered elements in the screen, in other words,

the elements are not in the correct site into the screen to be easily identified per users.

3. Fails in the language and confusion, because of the english language is not the native language of the users.

The three points mentioned are related to the null values in data table (see Table 6.1), some users didn't the task because they were confused, so they decided to omit the task. The null values weren't took in account when the median was calculated (see Table 6.2).

The USE questionnaire was useful to identify the weaknesses in the context-aware recommender system. The percent is upper of the acceptable (80%), the results allow to say that the system has a good performance.

For the future work we proposed to improve the problems found in the user interface, so the proposals are the following:

1. Redesign the user interface could helps to be more friendly for users. Due to the issues, the redesign involves:
   (a) Analyze the amount of information enough for a easy understanding, i.e., how much information the user needs seeing without overload it.
   (b) Modify the tasks descriptions in the most simple way to avoid confusion.

   (c) Add more language functionalities for to facilitate the tasks for users.

2. To apply the usability test again with the changes in the interface in order to observe the level of improves and to compare the results.

3. Apply an statistical test to analize the results.

4. Add collaborative filtering based on model (matrix factorization technique) within the context-aware recommender system in order to improve the level of user satisfaction in the context.

5. Add any contextual factors (such as companion, time of day, budget, etc.) in order to include more context information that could be relevant in the recommendations.

Publications

1. *Restaurant Recommendations based on a Domain Model and Fuzzy Rules. Xochilt Ramírez-García, Mario García-Valdéz. International Seminar on Computational Intelligence. Tijuana Institute of Technology. Tijuana Mexico. (2012).*

2. *Post-filtering for a Context-Aware Recommender System. Xochilt Ramírez-García, Mario García-Valdéz. Recent Advances on Hybrid Approaches for Designing Intelligent Systems . Springer International Publishing Switzerland. (2013).*

3. *Recomendaciones contextuales basadas en el enfoque de post-filtrado. Xochilt Ramírez-García, Mario García-Valdéz. Modelado computacional de Habilidades Linguisticas y Visuales. Vol.74. Research in Computer Sciences, IPN. 2014.*

4. *Context-aware Recommender System Based in Pre-filtering Approach and Fuzzy*

Rules. Xochilt Ramírez-García, Mario García-Valdéz. Recent Advances on Hybrid Approaches for Designing Intelligent Systems . Springer International Publishing Switzerland. (2014).

5. *Context-Aware Recommender System Using Collaborative Filtering, Content-Based Algorithm and Fuzzy Rules. Xochilt Ramírez-García, Mario García-Valdéz, 2016.*

6. *A Hybrid Context-aware Recommender System for Restaurants. Xochilt Ramírez-García, Mario García-Valdéz, 2016.*

Appendix A

Pseudocode

Algorithm 1 Get Cosine similarity values

Require: The list of itemProfilesUser and itemProfilesAll in binary format.
Ensure: The list of cosine similairty value for each item of the itemProfilesUser with each element of itemProfilesAll.

```
allProfiles ←[ ]
for itemu to size of itemProfilesUser do
  for itema to size of itemProfilesAll do
    if itemu = itema then
      jump next item
    else
      cosineSimilarityValue ← among itemu and itema
      itemProfiles ← itemu, itema, cosineSimilarityValue
    end if
  end for
end for
return allProfiles
```

Algorithm 2 Collaborative filtering algorithm

Require: The userId.
Ensure: The Top-N list of recommendations for the current user.
  $ratingMatrix \leftarrow allRatings$
  Call $Recommendations \leftarrow getRecommendations()$ module
  return $Recommendations$

Algorithm 3 Content-Based Algorithm

Require: The user id.
Ensure: The Top-N list of recommendations.
  $RV \leftarrow$ All items that user rated with 5
  for $item$ to size of RV **do**
    if $item$ is not in RV **then**
      $UV \leftarrow itemid$
    end if
  end for
  $allItems \leftarrow [\,]$
  $getItemsProfilesUser \leftarrow$ Binary vectors of RV
  $allRatings \leftarrow$ Rating matrix
  for $item$ to size of $allRatings$ **do**
    if $itemid$ is not in $allItems$ **then**
      $allItems \leftarrow item$
    end if
  end for
  $getAllItemsProfiles \leftarrow$ Binary vectors of allItems
  $getCosineSim \leftarrow$ getItemsProfilesUser,getAllItemsProfiles
  for $item$ to size of $highCosineSim$ **do**
    if $itemsimilarity \geq 0.8$ **then**
      $highCosineSim \leftarrow item$
    end if
  end for
  Sort $highCosineSim$ list
  return $itemProfiles$

Algorithm 4 Get item profiles

Require: The UV vector, allItems vector and boolean value of userProfile.
Ensure: The list of temProfiles in binary vectors.

```
if userProfile true then
    getItemsProfilesUser ← UV
    for itemp to size of UV do
        get binary vector of itemp
        itemProfiles ← itemp
    end for
else
    allItemProfiles ← allItems
    for itemp to size of allItems do
        get binary vector of itemp
        itemProfiles ← itemp
    end for
end if
return itemProfiles
```

Algorithm 5 Calculate Cosine similarity

Require: The itemProfileUser and itemProfileAll, both vectors in binary format.
Ensure: The cosine similarity value.

```
sum ← 0
normaItemUser ← 0
normaItemAll ← 0
for position to size of itemProfileUser do
    sumProduct ← sumProduct + (itemProfileUser[position] * itemProfileAll[position])
end for
for item to size of itemProfileUser do
    normaItemUser ← normaItemUser + itemProfileUser[item]^2
end for
for item to size of itemProfileAll do
    normaItemAll ← normaItemAll + itemProfileAll[item]^2
end for
squareRootUser ← squareroot(normaItemUser)
squareRootAll ← squareroot(normaItemAll)
cosineSimilarity ← sumProduct/(squareRootUser * squareRootAll)
return cosineSimilarity
```

Algorithm 6 Create a binary vector of item profile

Require: The tem profile content in r.
Ensure: The temProfile of r in a binary vector.

$price \leftarrow [4]$
$payment \leftarrow [2]$
$alcohol \leftarrow [2]$
$smokingarea \leftarrow [2]$
$dresscode \leftarrow [3]$
$parking \leftarrow [3]$
$installation \leftarrow [4]$
$atmosphere \leftarrow [5]$
$cuisine \leftarrow [30]$
$price[positionPriceId - 1] \leftarrow 1$
$payment[positionPriceId - 1] \leftarrow 1$
$alcohol[positionPriceId - 1] \leftarrow 1$
$smokingarea[positionPriceId - 1] \leftarrow 1$
$dresscode[positionPriceId - 1] \leftarrow 1$
$parking[positionPriceId - 1] \leftarrow 1$
$installation[positionPriceId - 1] \leftarrow 1$
$atmosphere[positionPriceId - 1] \leftarrow 1$
$cuisine[positionPriceId - 1] \leftarrow 1$
$itemProfile \leftarrow price + payment + alcohol + smookingarea + dresscode + parking + installation + atmosphere + cuisine$
return $itemProfile$

Algorithm 7 Get recommendations

Require: The currentUser and ratingMatrix.
Ensure: The Top-N list of recommendations for the current user.

Dictionaries $totals \leftarrow \{\}, sumSimilarity \leftarrow \{\}$
$predictions \leftarrow [\ ]$
for $otherUser$ to size of $ratingMatrix$ **do**
  if $otherUser = currentUser$ **then**
    jump next $otherUser$
  end if
  $similarityValue \leftarrow$ get pearsonSimilarity
  if $similarityValue \leq 0$ **then**
    jump next $otherUser$
  end if
  for $item$ to size of $profileOther$ **do**
    if $item$ is not in $profileUser$ **then**
      if $profileUser[item] = 0$ **then**
        Set in $totals \leftarrow item$
        $totals[item]$ Add $ratingMatrix[otherUser][item] * similarityValue$
        Set in $sunSimilarity \leftarrow item$
        $sumSimilarity$ Add $similarityValue$
      end if
    end if
  end for
end for
for each $(item, total)$ in $totals$ **do**
  $predictions \leftarrow [(total/sumSimilarity[item], item)]$
end for
Ranking of $predictions$
return $predictions$

Algorithm 8 Get Pearson correlation

Require: The currentUser, otherUser and preferences.
Ensure: The pearsonCorrelation score.

Dictionaries $itemsRatedMutually \leftarrow \{\}$
for each $item$ in preferences of $currentUser$ **do**
  if $item$ is in preferences of $currentUser$ **then**
    jump next $itemsRatedMutually[item] \leftarrow 1$
  end if
end for
$numberElements \leftarrow$ size of $itemsRatedMutually$
if $itemsRatedMutually = 0$ **then**
  return 0
end if
for $item$ to size of $itemsRatedManually$ to get all preferences **do**
  $sumCurrentUser \leftarrow preferences[currentUser][item]$
  $sumOtherUser \leftarrow preferences[otherUser][item]$
end for
for $item$ to size of $itemsRatedManually$ to get squares **do**
  $squareCurrentUser \leftarrow square(preferences[currentUser][item])^2$
  $squareOtherUser \leftarrow square(preferences[otherUser][item])^2$
end for
for $item$ to size of $itemsRatedManually$ to get sum of products **do**
  $sumProduct \leftarrow preferences[currentUser][item] * preferences[otherUser][item]$
end for
$pearsonNumerator \leftarrow sumProduct - ((sumCurrentUser * sumOtherUser)/numberElements)$
$pearsonDenominator \leftarrow square(squareCurrentUser - ((sumCurrentUser)^2/numberElements) * squareOtherUser - ((sumOtherUser)^2/numberElements))$
$pearsonCorrelation \leftarrow pearsonNumerator/pearsonDenominator$
return $pearsonCorrelation$ among two users

Algorithm 9 Matrix factorization

Require: R is a matrix to be factorized, dimension N * M, P an initial matrix of dimension N * K, Q an initial matrix of dimension M * K, K is the number of latent features, steps for the maximum number of steps to perform the optimization, alpha is the learning rate and beta is the regularization parameter.
Ensure: The factorized matrix P and Q.

```
alpha ← 0.0001, beta ← 0.001
QMatrix ← QMatrix ∗ T
for step to rangeSteps do
    for i to size of RMatrix do
        for j to size of RMatrix[i] do
            if RMatrix[i][j] > 0 then
                e_{i,j} ← RMatrix[i][j] − dotProduct(PMatrix[itoend], QMatrix[inittoj])
            end if
            for k to range of KFactors do
                PMatrix[i][k] ← PMatrix[i][k] + alpha ∗ (2 ∗ e_{i,j} ∗ QMatrix[k][j] − beta ∗
                PMatrix[i][k])
                QMatrix[k][j] ← QMatrix[k][j] + alpha ∗ (2 ∗ e_{i,j} ∗ PMatrix[i][k] − beta ∗
                QMatrix[k][j])
            end for
        end for
    end for
    eR ← dotProduct(PMatrix ∗ QMatrix)
    for i to range of RMatrix do
        for j to size of RMatrix[i] do
            if RMatrix[i][j] > 0 then
                e ← e + (beta/2) ∗ PMatrix[i][k]^2 + QMatrix[i][j]^2
            end if
        end for
    end for
    if e < 0 then
        break
    end if
end for
return PMatrix, QMatrix ∗ T
```

Appendix B

USE Questionnaire

Usefulness

- It helps me be more effective.
- It helps me be more productive.
- It is useful.
- It gives me more control over the activities in my life.
- It makes the things I want to accomplish easier to get done.
- It saves me time when I use it.
- It meets my needs.
- It does everything I would expect it to do.

Ease of Use

- It is easy to use.
- It is simple to use.
- It is user friendly.
- It requires the fewest steps possible to accomplish what I want to do with it.
- It is flexible.
- Using it is effortless.
- I can use it without written instructions.
- I don't notice any inconsistencies as I use it.
- Both occasional and regular users would like it.
- I can recover from mistakes quickly and easily.
- I can use it successfully every time.

Ease of Learning

- I learned to use it quickly.
- I easily remember how to use it. It is easy to learn to use it.
- I quickly became skillful with it.

Satisfaction

- I am satisfied with it.
- I would recommend it to a friend.
- It is fun to use.
- It works the way I want it to work.
- It is wonderful.
- I feel I need to have it.
- It is pleasant to use.

Source: From the work of Lund (2001). Note: Users rate agreement with these statements on a 7-point Likert scale, ranging from strongly disagree to strongly agree. Statements in italics were found to weight less heavily than the others.

Bibliography

[1] Gregory D Abowd, Christopher G Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: A mobile context-aware tour guide. *Wireless networks*, 3(5):421–433, 1997.

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

[3] Wasfi Al-Khatib, Y Francis Day, Arif Ghafoor, and P Bruce Berra. Semantic modeling and knowledge representation in multimedia databases. *Knowledge and Data Engineering, IEEE Transactions on*, 11(1):64–80, 1999.

[4] William Albert and Thomas Tullis. *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Newnes, 2013.

[5] Susan Auty. Consumer choice and segmentation in the restaurant industry. *Service Industries Journal*, 12(3):324–339, 1992.

[6] Robert Babuška. Fuzzy systems, modeling and identification. *Department of Electrical Engineering, Delft University of Technology*, 1996.

[7] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[8] Linas Baltrunas and Xavier Amatriain. Towards time-dependant recommendation based on implicit feedback. In *Workshop on context-aware recommender systems (CARS09)*, 2009.

[9] Linas Baltrunas and Francesco Ricci. Context-based splitting of item ratings in collaborative filtering. In *Proceedings of the third ACM conference on Recommender systems*, pages 245–248. ACM, 2009.

[10] Linas Baltrunas, Marius Kaminskas, Bernd Ludwig, Omar Moling, Francesco Ricci, Aykan Aydin, Karl-Heinz Lüke, and Roland Schwaiger. Incarmusic: Context-aware music recommendations in a car. In *EC-Web*, volume 11, pages 89–100. Springer, 2011.

[11] Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci. Context-aware places of interest recommendations and explanations. In *Joint Proceedings of the Workshop on Decision Making and Recommendation Acceptance Issues in Recommender Systems (DEMRA 2011) and the 2nd Workshop on User Models for Motivational Systems: The Affective and the Rational Routes to Persuasion (UMMS 2011). CEUR Workshop Proceedings*, volume 740, pages 19–26, 2011.

[12] Ronald J Brachman, Hector J Levesque, and Raymond Reiter. *Knowledge representation*. MIT press, 1992.

[13] Rebecca Bulander, Michael Decker, Gunther Schiefer, and Bernhard Kölmel. Comparison of different approaches for mobile advertising. In *Mobile Commerce and Services, 2005. WMCS'05. The Second IEEE International Workshop on*, pages 174–182. IEEE, 2005.

[14] Robin Burke. Integrating knowledge-based and collaborative-filtering recommender systems. In *Proceedings of the Workshop on AI and Electronic Commerce*, pages 69–72, 1999.

[15] Landro Castro and Silvana Aciar. Prototype of a tourism recommender system. In *Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En*, pages 1–7. IEEE, 2012.

[16] Federica Cena, Luca Console, Cristina Gena, Anna Goy, Guido Levi, Sonia Modeo, and Ilaria Torre. Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Communications*, 19(4):369–384, 2006.

[17] Chung-Hua Chu and Se-Hsien Wu. A chinese restaurant recommendation system based on mobile context-aware services. In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, volume 2, pages 116–118. IEEE, 2013.

[18] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and

Matthew Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR workshop on recommender systems*, volume 60. Citeseer, 1999.

[19] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[20] Didier J Dubois. *Fuzzy sets and systems: theory and applications*, volume 144. Academic press, 1980.

[21] Brian Fling. *Mobile Design and Development: Practical concepts and techniques for creating mobile sites and web apps.* " O'Reilly Media, Inc.", 2009.

[22] Mario García-Valdez and Brunett Parra. A hybrid recommender system architecture for learning objects. In *Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control*, pages 205–211. Springer, 2009.

[23] Jennifer Golbeck, James Hendler, et al. Filmtrust: Movie recommendations using trust in web-based social networks. In *Proceedings of the IEEE Consumer communications and networking conference*, volume 96, pages 282–286. Citeseer, 2006.

[24] G. Guo, J. Zhang, and N. Yorke-Smith. A novel bayesian similarity measure for recommender systems. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2619–2625, 2013.

[25] Liang He and Faqing Wu. A time-context-based collaborative filtering algorithm. In *Granular Computing, 2009, GRC'09. IEEE International Conference on*, pages 209–213. IEEE, 2009.

[26] Jaksa Jack Kivela. Restaurant marketing: selection and segmentation in hong kong. *International Journal of Contemporary Hospitality Management*, 9(3):116–123, 1997.

[27] Amit Konar. *Computational intelligence: principles, techniques and applications*. Springer Science & Business Media, 2006.

[28] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[29] Brian Y Lim and Anind K Dey. Assessing demand for intelligibility in context-aware applications. In *Proceedings of the 11th international conference on Ubiquitous computing*, pages 195–204. ACM, 2009.

[30] Matteo Manca, Ludovico Boratto, and Salvatore Carta. Mining user behavior in a social bookmarking system-a delicious friend recommender system. In *DATA*, pages 331–338, 2014.

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

[32] Margaret Morris and Arnie Lund. Experience modeling: How are they made and what do they offer. *LOOP: AIGA Journal of Interaction Design Education*, (7), 2001.

[33] José M Noguera, Manuel J Barranco, Rafael J Segura, and Luis Martínez. A mobile 3d-gis hybrid recommender system for tourism. *Information Sciences*, 215:37–52, 2012.

[34] Bing Pan and Daniel R Fesenmaier. Online information search: vacation planning process. *Annals of Tourism Research*, 33(3):809–832, 2006.

[35] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6):393–408, 1999.

[36] Sasank Reddy and Jeff Mascia. Lifetrak: music in tune with your life. In *Proceedings of the 1st ACM international workshop on Human-centered multimedia*, pages 25–34. ACM, 2006.

[37] Norma Saiph Savage, Maciej Baranski, Norma Elva Chavez, and Tobias Höllerer. *Im feeling loco: A location based context aware recommendation system*. Springer, 2012.

[38] Rossano Schifanella, André Panisson, Cristina Gena, and Giancarlo Ruffo. Mobhinter: epidemic collaborative filtering and self-organization in mobile ad-hoc networks. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 27–34. ACM, 2008.

[39] William Siler and James J Buckley. *Fuzzy expert systems and fuzzy reasoning.* John Wiley & Sons, 2005.

[40] Thomas Tran and Robin Cohen. Hybrid recommender systems for electronic commerce. In *Proc. Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, Technical Report WS-00-04, AAAI Press*, 2000.

[41] Gytis Tumas and Francesco Ricci. Personalized mobile city transport advisory system. *Information and Communication Technologies in Tourism 2009*, pages 173–183, 2009.

[42] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE transactions on*, 10(5):293–302, 2002.

[43] Ramírez-García Xochilt and Mario García-Valdez. Restaurant recommendations based on a domain model and fuzzy rules. In *Recent Advances on Hybrid Intelligent Systems*, pages 533–546. Springer, 2013.

[44] Jiawei Yao, Jiajun Yao, Rui Yang, and Zhenyu Chen. Product recommendation based on search keywords. In *Web Information Systems and Applications Conference (WISA), 2012 Ninth*, pages 67–70. IEEE, 2012.

[45] Chien-Chih Yu and Hsiao-Ping Chang. *Personalized location-based recommendation services for tour planning in mobile tourism applications.* Springer, 2009.

[46] LA Zedeh. Knowledge representation in fuzzy logic. *Knowledge and Data Engineering, IEEE Transactions on*, 1(1):89–100, 1989.

[47] Yong Zheng. Context-aware datasets. url: http://students.depaul.edu/ yzheng8/datasets.html. 2015. URL http://students.depaul.edu/~yzheng8/DataSets.html.

[48] Yong Zheng, Robin Burke, and Bamshad Mobasher. *Differential context relaxation for context-aware travel recommendation.* Springer, 2012.

[49] Yong Zheng, Bamshad Mobasher, and Robin Burke. Context recommendation using multi-label classification. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*, volume 2, pages 288–295. IEEE, 2014.