Information Sciences 433–434 (2018) 204–220

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Relationship recommender system in a business and employment-oriented social network

Pablo Chamoso[a,*], Alberto Rivas[a], Sara Rodríguez[a], Javier Bajo[b]

[a] BISITE Research Group, University of Salamanca, Edificio I+D+i, Calle Espejo 2, Salamanca 37007, Spain
[b] Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Campus Montegancedo, Boadilla del Monte, Madrid 28660, Spain

ARTICLE INFO

Article history:
Received 13 June 2017
Revised 29 November 2017
Accepted 25 December 2017
Available online 26 December 2017

Keywords:
Case-based reasoning
Recommender system
Social networks
Virtual organizations

ABSTRACT

In the last ten years, social networks have had a great influence on people's lifestyles and have changed, above all, the way users communicate and relate. This is why, one of the main lines of research in the field of social networks focuses on finding and analyzing possible connections between users. These developments allow users to expand on their network of contacts without having to search among the total set of users. However, there are many types of social networks which attract users with specific needs, these needs influence on the type of contacts users are looking for. Our article proposes a relationship recommender system for a business and employment-oriented social network. The presented system functions by extracting relevant information from the social network which it then uses to adequately recommend new contacts and job offers to users. The recommender system uses information gathered from job offer descriptions, user profiles and users' actions. Then, different metrics are applied in order to discover new ties that are likely to convert into relationships.

1. Introduction

Two decades ago, the geographical distance between people was the parameter that had the most influence on our lives; it determined who we related with. We established contacts with people that lived close by, such as neighbors and colleagues. However, this traditional way of establishing contacts changed with the appearance of new technologies and more specifically with rapid advances in social networks. Twitter, Instagram, Google+ and especially Facebook, have changed the way contacts are established and how friends are made.

Among different approaches, the one presented in [46] determines which contacts should be suggested to social network users, by evaluating the level of affinity between their profiles.

Social networks have become more than just a means of communicating and sharing with friends. They have evolved to include the professional environment, expanding the users' work opportunities and possibilities. Social networks such as LinkedIn, Monster or XING, specialize in establishing professional links between users.

The research presented in this paper, focuses on a relationship recommendation system for a business and employment-related social network called beBee [4]. On beBee, companies and users share content, and users search for and apply to

* Corresponding author.
E-mail addresses: chamoso@usal.es (P. Chamoso), rivis@usal.es (A. Rivas), srg@usal.es (S. Rodríguez), jbajo@fi.upm.es (J. Bajo).

https://doi.org/10.1016/j.ins.2017.12.050
0020-0255/

the published job offers. Therefore, in the case of this social network, our recommender system will not only serve for connecting similar user profiles but also for providing work related suggestions to users and companies.

For this purpose, we identified different factors that could be extracted from the information provided by users, and we analyzed the information found in the published job offers. One of the challenges that we had to address was the extraction of information, since the required information is often not available or it is not properly structured. Thus, it was necessary to apply text mining and information extraction techniques in order to evaluate possible ties between users as well as users and job offers.

These techniques ensure that the job offers recommended to users, correspond to the information provided in their profile; they possess the skills required by the offer, they have been previously interested in similar offers, etc. However, the use of these techniques does not guarantee that the offer suggested by the system, satisfies all user expectations.

To provide accurate suggestions, it is important to analyze the reasons for which a particular user could reject or accept a suggestion. To do such an analysis, it is necessary to employ techniques such as case-based reasoning (CBR), which allows to feedback the system with information on previous cases. CBR is a problem solving methodology and its use in this project is essential since it will allow the recommender system to learn and improve over time by re-using similar past experiences which are stored in cases [30].

To evaluate the proposed system it has been divided into two parts: user-user relationships and user-job offer relationships. Different subsets of real and active users, as well as recently published job offers, have been selected to gather information for this evaluation. The results obtained in this evaluation are satisfactory.

The rest of the article is structured as follows: in the background Section 2, we make an analysis of the previously proposed relationship recommendation techniques for users in other environments. We also examined different information extraction and classification techniques for unstructured texts. In Section 3, the proposed system is described and in Section 4 the results of its evaluation are presented. Finally, conclusions are drawn on the performance of the proposed system and future work is discussed in Section 5.

2. Background

Three main technologies have been used in this work: Recommender Systems (RSs); their mechanism determines possible relationships between users and provides the system with suggestions. The use of text mining is crucial for the extraction of the different factors that can be contained in an untagged or tagged text, these factors are necessary for detecting affinity. The third essential technology are Virtual Organizations (VOs) of agents which allow the system to obtain the best solution; their modular basis provides the system with great independence when applying all these methodologies.

The rest of this section describes the role of these three technologies in similar studies found in the literature.

2.1. Recommender systems

The overwhelming amount of information on the Internet has made RSs a very important tool for those who provide information, as well as those who receive it; the users. RSs consider users' personal preferences, guiding them in a large space of possible options, to those items that they may find interesting or useful [10]. The implementation of RSs on the Internet has increased and with it, their application in different areas [35]. Among others, RSs have been deployed in contexts such as e-Commerce sites [44,45], online and mobile advertising [9], real time broadcasting [36], general and specific rating sites [2] and social networks [32,34]. Due to the large number of contexts, many different methodologies have been proposed.

In the context of the present work, we will focus specifically on social RSs. The popularity of social media sites causes an immense production of content, and users have to be helped to view or access the information they are interested in. Thus, social RSs aim to reduce the information overload for social site users. In addition, the ability to find the content we are looking for easily can encourage users to keep using the network, which means that recommendation technologies can play an important role in the success of social networks and the social web as a whole, assuring that each user is provided with content that they personally find interesting and attractive.

Therefore, this type of RSs use techniques that adapt to the needs and interests of individual users or groups [26] recommending content to [43] other users [46] or other groups [11].

Collaborative filtering is an approach used in social RSs [22]. This technique keeps track of users' likes and dislikes on the basis of their Internet history and it is usually used with other filtering techniques like content or social knowledge-based techniques [6].

Knowledge-based techniques are able to feedback the system and provide each user with better solutions. An example of knowledge-based technique is case based reasoning, which can be used in recommendation systems to exploit user histories, [8]. CBR systems rely on the knowledge they obtained from previous cases, when analyzing and finding solutions to a new problem. They do this by using algorithms to search, retrieve, compare and adapt a problem to a similar situation from the past. A case is defined as a past experience and is composed of: the description of the problem, the solution and the result [18].

Moreover, the evolution of RSs has shown the importance of using hybrid techniques [10]. In addition to knowledge-based techniques, using content-based techniques is really useful. Determining context information is an important factor

in RSs, if potentially useful information is not considered, the prediction may lose its precision. In [2], authors present a classification of the RS depending on how the contextual factors and the RSs' knowledge of them, changes over time.

What a RS knows about contextual factors can be: (i) Fully observable, when the factors are relevant to the application, and their structure and values are known explicitly. (ii) Partially observable, when only some of the information about contextual factors is known explicitly. In addition, their structure can also be unknown. (iii) Unobservable, if there is no explicit information about contextual factors. Depending on how contextual factors change, RSs can also be structured into two types: (i) Static, when the relevant contextual factors and their structure remain the same over time. (ii) Dynamic, when those relevant contextual factors or their structure can be modified.

In the present work partial and dynamic context knowledge is used. Partial because some factors can be explicitly unknown and dynamic since the structure of contextual factors can vary depending on how users and companies provide those factors in their profiles and job offers.

Thus, determining factors that possibly influence different relationships is very important. The factors that are found in unstructured texts also have to be extracted and analyzed. The following sections outline a variety of methodologies used for the extraction of information.

2.2. Text mining

Nowadays, we can find numerous sources of information such as websites, blogs or social networks, where users constantly share opinions, news or other kind of up-to-date information. Clearly, this massive amount of data coming from so many sources, should be processed in order to obtain from it as much knowledge as possible. Unlike data mining techniques, where information is extracted from structured databases, the goal of text mining techniques is to extract information from unstructured textual data. Usually, algorithms based on natural language processing are used together with text mining tools to improve final results [39].

The knowledge extraction process in collections of unstructured text, always works following similar steps, which can vary slightly depending on the aim. The first step of almost all techniques is preprocessing the document. Some of these preprocessing tasks are noise reduction and stemming. Noise reduction involves the elimination of stop words, white spaces, headers, footers, etc. which are not required for text analysis, meanwhile stemming is a process in which prefixes and suffixes are removed, leaving us just with the root of the word.

After this preprocessing step, text mining takes place, it can be done using various techniques, such as the ones outlined below:

A commonly used technique is the cleaning and parsing of input data in order to identify trends in the text document and label the dominant words and phrases [33]. Similarly, in [17] a term extraction is performed on the words that had been extracted previously and labeled as common. This technique gives us the option of eliminating meaningless words.

However, more recent techniques have also been used. They try to find links between words by performing association mining tasks after the term extraction process [21]. Moreover, frequent terms extraction is not the only main approach, clustering techniques are also used to label documents according to their dominant words; ordering the documents into hierarchical structures for browsing [3].

In the context of this work, text mining techniques will be used for the extraction of relevant information from job offers and from the text content and links shared by users. Carrying out the categorization process is the final step, it is done to sort all the similar documents, this will enable us to determine affinity among users. Based on the content and profiles of users, new relationships will be recommended to those with similar interests.

2.3. Virtual organizations

VOs of agents offer the system the possibility of developing flexible core software, with great independence and modularity in the application of recommendation methodologies to provide the best solution. By applying the concept of organization to computer systems, and in particular, to the Multi-agent System (MAS), a MAS can be described as a group of agents that are coordinated through (i) A series of standards and (ii) the establishment of several roles to achieve system objectives. This is very suitable in systems such as the one presented in this article, where there is a direct relationship between the users and the organization of entities that make up the social network, for this reason they can be identified directly in a similar software structure.

Structurally a VO is considered to be a set of entities whose association is established through relationships of inheritance and aggregation, and it is on this structure where a series of social relationships and workflows are defined [37]. The organization structure is applied to agents in groups, which may be formed by agents, roles, resources and applications, whose allocation results in an organizational purpose that facilitates their coordination.

Virtual organizations are structured to resemble human organizations. In a previous work, [41] open MAS, specifically those including organizational aspects such as VO of agents have been studied and analyzed. There are a lot of different platforms [1,5,7,23,38,40,42], that allow to create MAS. These platforms greatly facilitate the task of working with agents. However, in terms of platforms that allow for the creation of a VO, the number is drastically reduced; it is in fact difficult to find a single platform that covers all the requirements that a VO requires, such as reorganization and adaptation facilities, norm compliance, different organizational topologies, services and role management. For example, S-MOISE+ is an

organizational middleware that follows the MOISE+ model [28]. J-Moise+ [25] is very similar to S-Moise+ regarding the overall system concepts. AMELI is another middleware that can work with electronic institutions (similar to VO) [16]. One of the most complete and recent platforms that have been found in the literature review is Janus [19]. This platform was specifically designed to deal with the implementation and deployment of holonic and multi-agent systems (HMAS) [20]. To summarize, when dealing with all aspects of complex systems, such as MAS and VO, it is necessary to manage multiple levels of abstractions and openness, which is not the case of most solutions [13]. Moreover, although agent frameworks and platforms have similarities, there are subtle differences too. Each platform uses different models and syntax and provides different libraries. In our case [41], we have tried to use standards whose robustness has already been demonstrated and which are known within the research community, making the implementation of the system easier and highly reliable. A service oriented platform that allows for the implementation of an open MAS is used [41] to take maximum advantage of the distribution of resources. To this end, all services are implemented as Web Services. Due to its service orientation, different tools modeled with agents that consume Web services can be integrated and operated from the platform, regardless of their physical location or implementation.

Distributed MAS have become very sophisticated in the last years [14,48], with a rising potential to handle large volumes of data and coordinate the operations of many organizations. MAS offer a general computing paradigm for solving difficult computational problems as well as for characterizing complex systems based on concepts such as autonomy, self-aggregation, self-organization and emergent behavior. Due to its characteristics, MAS are especially suitable for solving large-scale and distributed problems such as social network presented in this research. Looking into the current researches background, it is possible to observe that MAS are been widely used to model and solve complex real-world problems in text mining and recommender systems in social networks. New methods have been proposed and applied to different areas including social network analysis, gene network analysis or web clustering. Most of the existing methods for mining communities are centralized, although in studies such as [24,47]. In this case, a group of autonomous agents work together to mine a network through a proposed self-aggregation and self-organization mechanism. In [27] a new model is presented for finding influential agent groups based on group centrality analyses in citation networks. The authors present a model for finding influential agent groups in multi-agent software systems considering that the impact of an agent is mainly characterized by its citation relations (a group's impact is determined by both direct and indirect citations). Another example is presented in [15], a novel distributed model for HMAS. The proposed model assumes an interaction network among the agents of a MAS. This decentralized and local-info-based nature of the suggested model makes it an appropriate approach for large-scale open MAS. The outcomes of this work can be used in a wide range of distributed applications.

There is a notable trend in filling the gap between social network analysis and control. This trend was triggered by the advancements in complex networks theory and MAS, the inclusion of new mathematical models describing dynamics of social groups, and the development of new computational tools for relevant information analysis.

Control of collective behavior is one of the most desirable goals in many applications related to social network analysis and mining. This work also intends to go a step further in this regard, by integrating a comprehensive recommender system into a social based agent architecture for the extraction and analysis of relevant information. The system was implemented in an agent framework that was subsequently used to extract the behavior of users in a social network.

3. Proposed system

This section describes the process that is carried out before a relationship is recommended (or not) to a user. To provide a recommendation, the system must detect a tie between two users or between a user and a job offer. According to the social tie proposals [31], which are based on mathematical sociology, ties can be classified into three types:

- *Strong ties*: these ties represent a formal and direct link. On the social network, a strong tie represents an existing and direct relationship between users or a user and a job offer.
- *Weak ties*: these are the most common ties on social networks. They are created when there is a link between two entities (user-user or user-job offer) that is not strong, but there are some elements connect users or a user and a job offer within the system.
- *Absent ties*: ties between objects that do not substantiate any true link, these are not considered by the system.

In such way, the system searches for weak ties and evaluates them, recommending suitable relationships to users. Once a user accepts a recommendation, a weak tie converts into a strong tie.

In addition, we should specify that these links are not bidirectional, they are unidirectional. Therefore, the system can suggest a relationship to only one of the users, since the affinity is calculated in such a way that it adapts to the profile of each user individually and automatically. In the job offer RS, particular job offers are recommended to compatible users.

The rest of this section addresses the findings and evaluations made in the four categories: (i) VOs of agents; (ii) extraction of relevant information from unstructured texts; (iii) user-user relationships; (iv) user-job offer relationships.

3.1. Virtual organizations

By using VO, the system can be structured as a group of agents which are coordinated through the integrated platform [41]. VO structures are very suitable for social networking systems, such as the one presented in this work. Since there is

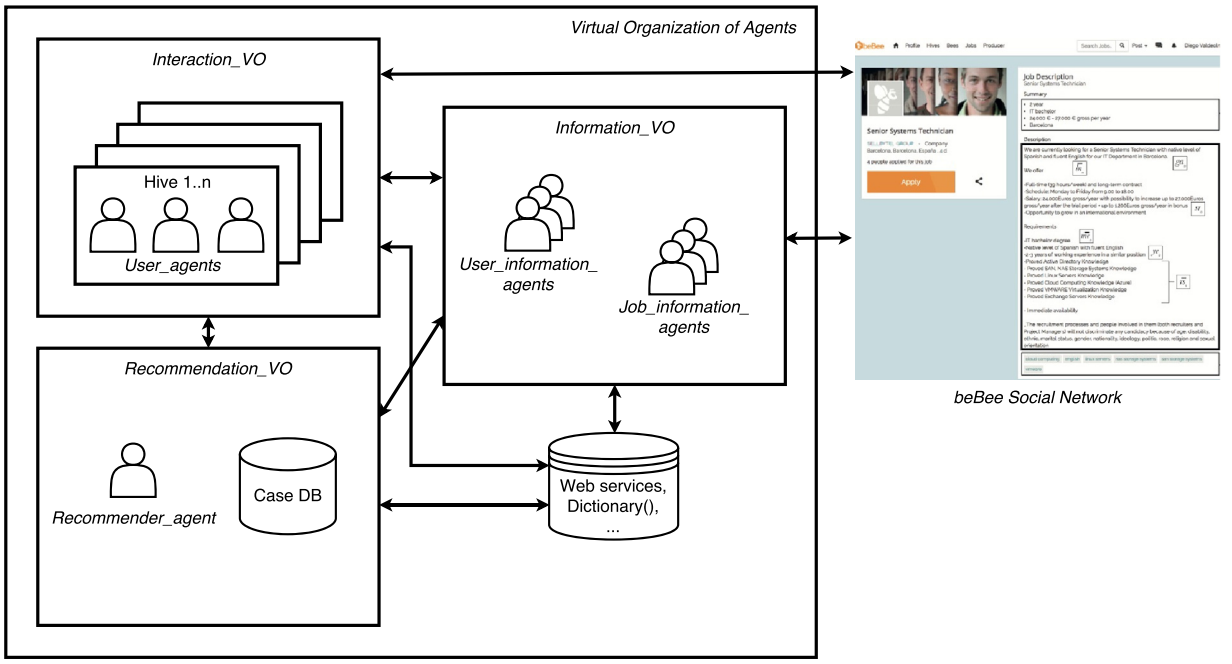

Fig. 1. VO architecture diagram.

a direct relationship between users and the organization of entities, the structure of the agent system can be determined directly on the basis of the social network structure.

The agents that form part of the organization, are deployed in several specialized sub-organizations. Each of the agents offers services (an interaction service) modeled as Web Services. The platform agents are implemented with Java to enable their interaction with the social network. Fig. 1 illustrates the VO architecture. The sub-organizations and agents are:

- *Interaction_VO*: beBee offers its users the possibility of joining hives, which are communities of users who share same interests or the same professional sector. To control this particular feature, the system has been implemented on the same terms; it has different sub-organizations in which several *User_agent* type agents interact with each other. A *User_agent* is considered to be the interface between the user, the social network and the recommendation functionalities. This agent is in charge of generating and updating the user profile.
- *Information_VO*: Is in charge of doing the search and treatment of content in the network. It consists of two main entities: *User_information_agent* and *Job_information_agent*. Both agents are responsible for managing all types of content on the network (offers, jobs, interests, etc.). The *Job_information_agent* will be responsible for collecting information about a job offer (required education, required skills, desirable skills, salary, geographic area). It will also make use of the appropriate web services in order to user a specific language dictionary. For its part, the *User_information_agent* will obtain information about users (colleagues, profile, job applications, contents published, ties, interests, geographic area). These agents make use of the information stored on the beBee network. Moreover, these agents are in charge of launching the RS process in order to recalculate all the affinities and thresholds when a profile has been included or updated. The system is fully scalable due to the fact that agents are independent of the network. That is, if the characteristics of the analyzed network changed, or if the use of the recommender system was to be used on another type of network, the implemented software would be very easy to adapt and transfer.
- *Recommendation_VO*: The core of the system will be the sub-organization responsible for carrying out the different recommendations. It will contain a *Recommender_agent* responsible for performing all factor and weight calculations as explained in the following sections. This agent will be in charge of supporting the CBR system. It will be a CBR-BDI type agent [12] where cases with information on past user actions and suggestions made by the system, will be stored. This means that it manages the CBR proposed in this study, where the cases are all the suggestions made to the user along with the user's actions before this suggestion was received. It communicates with the *User_agent* and *Information_agent* types, in order to obtain information from the user and the network.

The VOs work together with the beBee social network and also as an entity as shown in Fig. 1. All agents can make use of external web services to implement new capabilities such as new dictionaries, word frequency algorithms or search tools. The *Interaction_VO* makes it possible to interact with a specific user through the social network. This allows the system to define and manage the different profiles of people and device systems. Users interact with the system by creating ties

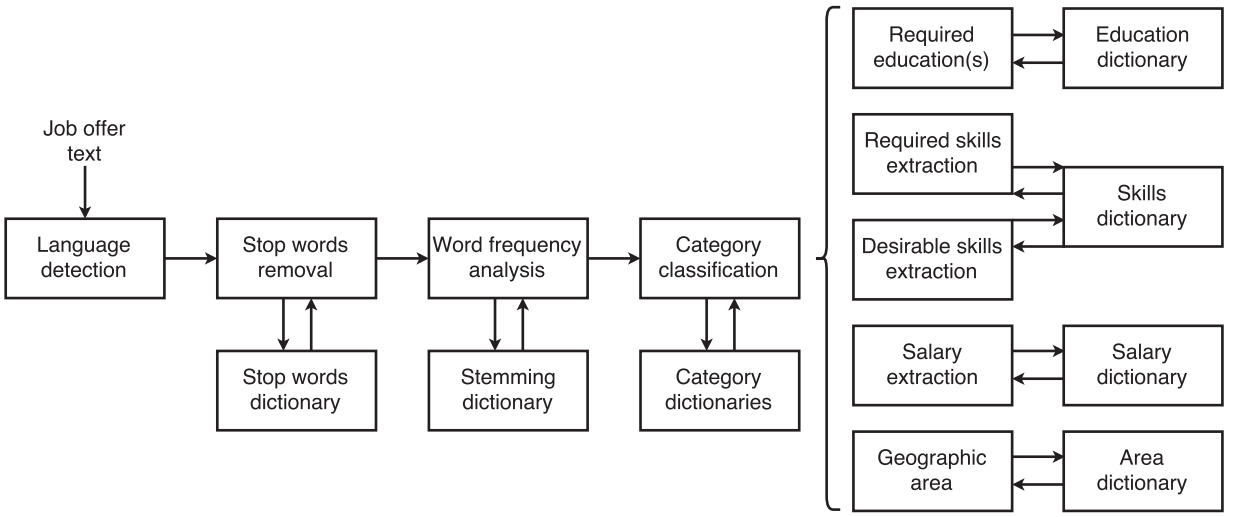

Fig. 2. Job offer factors extraction.

between other users and searching job offers, or accepting or rejecting new suggestions. This whole process constitutes the bulk of the research presented in this article and it will be described in the following sections.

3.2. Information extraction

It is desirable for a recommender system to have all the appropriate information in order to precisely determine whether a relationship should be recommended or not. Our work, in concrete, requires the extraction of information that provide us with user and job offer factors, necessary for the evaluation of possible relationships.

The first part of information extraction acts as a preprocessing stage in the obtaining of values, which are later used by the RS. Although the information on user profiles is structured, a semantic analysis is required in order to extract the contained factors, such as the user's level of education and category, their skills and the area they work in. It is less common to find structured information in job offers and therefore we need to take more steps to extract the information. This is because companies tend to copy and paste the content of job announcements in a single description field of the input form, making the extraction process more complex. This extraction workflow can be seen in Fig. 2.

First, the system detects the language of the text. beBee is an international social network, however, this preliminary study focuses exclusively on Spain. Once the language is detected, stop words are eliminated by using a specific dictionary for the detected language. Then, a stemming dictionary is used, words are stemmed and the frequency of each of the stemmed words is obtained.

Then, the offer is categorized in the Category classification step. To carry out this process, specific dictionaries have been written for each work category and the texts of existing offers were analyzed for each of the categories, (between 1000 and 5000 offers per category). Then, in each of the categories, an expert user reviewed 30% of the most repetitive words, keeping only the most relevant ones and discarding the rest.

In this way, when a new text is obtained, the classifier module makes use of these keyword dictionaries to find matches in the text and classify the area of work to which the offer belongs.

Once the offer is classified, required education, skills, salary and location information is obtained by using specific dictionaries for each factor.

In the extraction of information from the user profile, the dictionaries used to obtain the user's category, education, skills, salary and location are the same as those used in job offer analysis.

3.3. User-user relationships

An important aspect of context-based RSs is the order of priority that is given to certain factors. Not all factors have equal importance, thus, it is necessary to establish the key factors that determine if a recommendation should be made or not, and those that have a lesser influence on the recommendation. Therefore, it is not sufficient to identify the factors; the weight of their impact on the recommendation also has to be considered. For this reason, a metric is proposed for the evaluation of the weight of each factor and the assignment of its maximum weight ($max_{factorName}$). Then, the affinity (that represents the strength of the tie) can be obtained depending on these values.

Hence, the following factors determine the possibility of a weak tie between two users becoming a strong tie:

- *Colleagues (c)*: If two users obtained the same degree, at the same time and in the same University, it is quite possible that they wish to follow each other on a social network. The same goes for users who worked together in the same company.
- *User profile (p)*: The characteristics of users are a key factor in the establishment of relationships, especially on a job finding network. As a general rule, the more features two users have in common, the greater the affinity between them.
- *Shared strong ties (s)*: The number of contacts that two users share is another key factor in calculating affinity. The more contacts users have in common, the stronger the linkand affinity between them. To obtain the maximum weight of this factor, users must have 20% of shared users.
- *Strong ties profile (sp)*: It is possible to analyze strong tie profiles and compare them to the profiles of those users who are compatible and therefore are likely to be accepted if the system recommends them. To reach this factor's maximum value in the system, the user should have 50% of similarity with existing strong tie profiles.
- *Shared groups (g)*: beBee offers its users the possibility of joining "hives", which are groups or communities of users who share the same interests or professional sector. This means that users who are in the same group have certain affinity. The greater the number of groups in common, the greater the affinity between users.
- *Published offers (o)*: Job offers can also be posted by users. Users who publish job offers oriented at a certain work profile or category, may wish to establish new relationships and create new job opportunities. In this way, this factor provides some affinity to weak ties among users who post job offers.
- *Job applications (a)*: Two users that apply to the same job offers, could be interested in establishing professional links that will help them find new professional opportunities. The greater the number of job applications in common, the greater the affinity between these users.
- *Publication of contents (co)*: beBee is a social network where users can share content with others. This content can comprise of a text, an image or a link that provides access to the content of another webpage. In these situations, a mechanism in the system searches for similar content using text mining and image matching techniques. The system finds texts on the same topic regardless of their source or the content being slightly varied. Image matching works in the same way, alterations in the quality, color and rotation of the image or the presence of a watermark are changes that can be undone by the system and therefore they do not have any influence on the affinity level. In this way, when users publish similar content, the level of affinity increases between them with every match. The system is limited and therefore only a maximum of common *nc* contents is taken into account. Due to the activity of users on this social network, the value of *nc* has been fixed as 10, which represents the number of similar published content that users must have in common in order to obtain the maximum score.
- *Common interests (i)*: Users can also respond to the content published by other users. In this regard, when a user recommends (or likes) the content published by another user and they do not share a strong tie, the level of affinity increases. Similarly, if two users like the same content and do not have a strong tie, the affinity value also increases. The system is limited so that only a maximum of *ni* common interest is taken into account, each match counts the same percentage. Due to the activity of users on this social network, the value $ni = 10$ has been defined as the number of common interests that two users should obtain in order to reach the maximum score.
- *Following (fw)*: On beBee contacts between users are unidirectional. There may be a strong tie on the side of one user but it is not necessary for that tie to exist in the opposite direction. When evaluating affinity, an existing unidirectional strong tie is considered to be positive when evaluating affinity in the opposite direction.
- *Geographic area (ga)*: The geographical location is an important factor when assessing affinity between two users. This factor is considered to be positive if two users reside in the same location.
- *Previous cases*: The system also basis its recommendations on each user's previous actions. If a user tends to accept recommendations of profiles with specific characteristics, the system is able to adapt the weights of every factor so that the kind of profiles that is frequently accepted by the user is the most recommended. Weights are also modified when the user rejects a certain type of users, or even all types. This feedback system is created using CBR, where the cases of each of the suggestions sent to the user are stored along with the actions of that user before a suggestion was made.

A metric has been assigned to each of these factors to determine their weight on the affinity value. The Table 1 lists all the identified factors for the evaluation of user-user relationship recommendations with the equation that determines their weight in terms of the level of affinity.

To calculate the affinity value of user j to user i (in a unidirectional way, as already mentioned), the system evaluates the equations shown in Table 1 and the vector $\overline{f^{ij}}$ is obtained. This vector is defined as

$$\overline{f^{ij}} = [c, p, s, sp, g, o, a, co, i, fw, ga] \tag{1}$$

However, to determine the affinity between two users, it is necessary to define a number of additional concepts. Using these factors, with the exception of "Previous cases" one, a vector $\overline{v}$ is built for each user i so $\overline{v^i}$ is obtained as defined in (2).

$$\overline{v^i} = [c', p', s', sp', g', o', a', co', i', fw', ga'] \tag{2}$$

Value 1 is assigned to all the factors that are extracted from user information and value 0 is assigned otherwise. For example, the vector shown in (3) belongs to a user which has not joined any group and has published no job offers.

$$\overline{v^i} = [1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1] \tag{3}$$

Table 1
User-user recommendation factors and weight.

Factor	Weight
Colleagues	$c = \begin{cases} max_c & \text{true} \\ 0 & \text{false} \end{cases}$
User profile	$p = \frac{nCommonFeatures}{totalFeatures} * max_p$
Shared strong ties	$s = \min((\frac{nSharedStrongTies*5}{totalStrongTies} * max_s), max_s)$
Strong ties profile	$sp = \min((\frac{nStrongTiesSameArea*2}{totalStrongTies} * max_{sp}), max_{sp})$
Shared groups	$g = \frac{nCommonGroups}{totalGroups} * max_g$
Published offers	$o = \frac{nSimilarOffers}{totalOffers} * max_o$
Job applications	$a = \frac{nSameApplications}{totalApplications} * max_a$
Publication of contents	$co = \min((\frac{(nImg+nTxt+nLink)}{nc} * max_{co}), max_{co})$
Common interests	$i = \min((\frac{nInterests}{ni} * max_i), max_i)$
Following	$fw = \begin{cases} max_{fw} & \text{following} \\ 0 & \text{not following} \end{cases}$
Geographic area	$ga = \begin{cases} max_{ga} & \text{same areas} \\ 0 & \text{otherwise} \end{cases}$

The factors "User profile" (p), "Shared strong ties" (s), "Strong tie profile" (sp), "Following" (fw) and "Geographic area" (ga) are always required, thus their value in the factor vector is always 1.

Next, the vector $\overline{mw^i}$ (maximum weight) is defined as the vector of maximum value for the weight of each factor, depending on each user's criteria:

$$\overline{mw^i} = [max_c, max_p, max_s, max_{sp}, max_g, max_o, max_a, max_{co}, max_i, max_{fw}, max_{ga}] \tag{4}$$

To calculate the affinity value, a^{ij}, the following equation is used:

$$a^{ij} = \frac{\sum_k f^{ij}{}_k \cdot v^i{}_k \cdot v^j{}_k}{\sum_k mw^i_k \cdot v^i{}_k \cdot v^j{}_k} \tag{5}$$

where $a^{ij} \in [0, 1] \in \mathbb{R}$ and k represents every element contained in the vectors used. All of these vectors have the same dimension, and their value is increased by one from the first element to the 11th element (the last one in all cases) of the vectors $\overline{f^{ij}}$, $\overline{v^i}$, $\overline{v^j}$ and $\overline{mw^i}$.

The CBR system is responsible for determining if a weak tie between user i and user j with the affinity level a^{ij} should be suggested to user i in order to become a strong tie. The CBR system does this by: (i) determining the threshold value (trh) of affinity which must be surpassed, for a user to be suggested to another i; (ii) establishing the maximum weight of each of the identified factors, $\overline{mw^i}$.

Both tasks are carried out on the basis of the user's historical cases which display their reaction to past suggestions, in this way, when suggesting a new contact the recommendations adapt to each user's criteria. The system does this by establishing an initial situation, where threshold $thr^i = 0.3$, and whose interval can vary between [0.15, 0.7]. The values of the elements of the vector $\overline{mw^i}$ are defined as follows:

$$\overline{mw^i} = [35, 20, 20, 15, 15, 10, 10, 10, 10, 35, 20] \tag{6}$$

These input data have been determined in a supervised manner based on the average historical data of the social network. More specifically, a sample of 50 users has been selected manually on the basis of the users' activity. The range [0.15, 0.7] was established according to the affinity value of the accepted recommendations. The default value $thr^i = 0.3$ is the mode of the accepted recommendations. However, each profile will have its own maximum values, which will be recalculated after: (i) each interaction with the recommender system, and the cases will be added to the CBR case memory; (ii) each new strong tie added directly.

Each case that is part of the CBR memory is composed as defined in (7).

$$C_{i(n+1)} = (thr^i, \overline{f^{ij}}, \overline{v^i}, \overline{v^j}, \overline{mw}, a^{ij}, userResponse) \tag{7}$$

Where $C_{i(n+1)}$ is the case $(n+1)$ of user i and is calculated using the information of the previous n cases of that user. If the user has no previous cases, cases from users with similar profiles are used. Each user's information is saved by the system, in this way, it is able to learn from the user's past cases of accepted and rejected recommendations, by applying this criteria the system will be able to improve acceptance rates.

3.4. User-job offer relationships

One of the main goals of employment-oriented social networks is that users apply to job offers. Therefore, the proposal of suitable job offers to users is an essential role of the social network.

User profile analysis is carried out with the aim of evaluating if it is suitable for a particular job offer, a number of attributes is always considered and measured. However, for the job offers that specify the work area, the profiles that are taken into account are the ones that are categorized in this work area. The attributes that are identified on the user profile are described below. The metrics used to calculate their value are detailed later on in this section:

- *Main education ($\overline{me}$)*: This factor has two measures. The first one measures the quality of education received and the second factor measures its quantity. Quality is evaluated according to the Bologna Process [29], which proposes an ECTS (European Credit Transfer and Accumulation System) grading scale to evaluate the level. A coefficient value of 1 is assigned to three or four year studies, they are considered the lowest higher education level (usually awarding a bachelor's degree). A coefficient value of 2 is assigned to master degree studies that last one or two years. Lastly, a coefficient of 3 is assigned to the doctoral level. Regarding secondary education, the system considers two levels: junior secondary education (coefficient 0) and senior secondary education (coefficient 0.5). For all types of degrees, quality is evaluated according to the scale defined for factor *me*. The more qualified the user is, the greater their value. However, if the offer requires a high qualification, only these qualifications are taken into consideration.
- *Skills ($\overline{sk}$)*: Skills and keywords are also collected directly from users.
- *Unwanted skills ($\overline{us}$)*: The user's skills are a key factor that determines whether they are suited for a particular job or not. If a user rejects job offers that require certain skills, these skills will be stored in a vector as unwanted skills, along with the frequency with which they have been rejected. In this way, the system is able to learn that offers with certain features should not be suggested to the user. However, the offer can require the skills that the user has, mixed with other unwanted skills, so a threshold is determined dynamically depending on the actions of the user.
- *Years of experience (ye)*: One of the parameters that has the greatest influence on determining the knowledge of a user in an area are the years they have worked in it and the type of contract, considering a factor of 1 for each contract year and 0.5 for part time contracts. Thus, the years of experience are favorable on the user profile.
- *Languages ($\overline{ln}$)*: When languages are required by an offer, the level of their knowledge is an important factor. However, users do not always specify their level of knowledge of a foreign language and therefore it is not considered when suggesting offers to a user. What is taken into account is the knowledge of the language itself but not the level. It is left up to the companies to evaluate the applicants level at an interview.
- *Salary range (sr)*: If the user indicated the desired salary range, this category is also an important factor that must be taken into account when making a recommendation.
- *Geographic area (ga)*: The geographical area is often the most important factor in determining if a job offer may be of interest to a user. Initially, the system gives preference to the places that are the closest to the user's location, however, the weight of this value modifies if the user applies to offers that are at a distance form where they are situated.
- *Similar applications*: When a user applies to similar offers, the system takes this factor into account when making recommendations. A user is more likely to apply to an offer if the recommended job offer is similar to their previous applications. The cases of each of the job offer applications made by each user are stored on a CBR system which is used to obtain this feedback.
- *Previous cases*: As in the above case, the user's previous interactions with the RS serve as feedback which guides the system when making new recommendations to the user. For this factor the same CBR system is used, but the user's response can take positive values (if the user accepts the suggestion) or negative (if the suggestion is rejected), while for the Similar applications factor, only actions of acceptance are considered (applications).

In the same way that user profile factors are analyzed, job offer factors also have to be analyzed and evaluated so that the system is able to provide suitable recommendations. The following job offer factors are identified:

- *Required main education ($\overline{me_o}$)*: The level of education required by a job offer is a vital factor when determining whether a user is likely to be hired or not. In addition, it is possible that more than one qualification may be required.
- *Required skills ($\overline{rs_o}$)*: Most of the job offers on the website, in addition to a degree, usually require the applicant to have various abilities or skills that make them suitable for the position.
- *Desirable skills ($\overline{ds_o}$)*: This factor does not have the weight of the previous factor, but it does have the same reasoning. Although not essential, offers that list desirable skills look for profiles that will be the most eligible for the position offered.
- *Previous experience (years) (ye_o)*: Usually, the offer asks the applicant to have a certain level of experience that guarantees the user's ability to work in the required field.
- *Required languages ($\overline{ln_o}$)*: The number of languages a user knows is also a key factor for offers that involve the knowledge of languages. When a user applies for a position that requires the knowledge of various languages, the more languages they know from the languages desired in the offer, the higher their affinity value.
- *Salary range (sr_o)*: Although this factor is not taken into account in every case, since it can be affected by factors such as geographic location, it can be inferred that the higher the salary offered, the greater the salary of the user should be.
- *Geographic area (ga_o)*: An important factor to be extracted from a job offer is their geographical location and whether the candidates are required to reside in that location in order to determine which users should be recommended.

With the attributes extracted from both user profiles and job offers, two dataframes are built. Each dataframe is just a list where items of different type and dimension are stored.

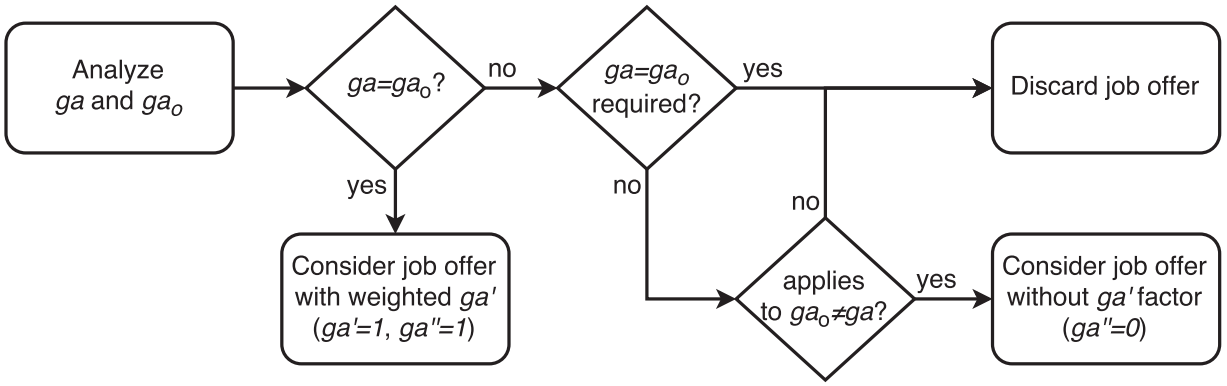

Fig. 3. Geographic area behavior analysis.

On the one hand, dataframe p^i is constructed, it represents the profile of user i and includes the previously described factors. The dataframe is defined as shown in

$$p^i = [\overline{me}, \overline{sk}, \overline{us}, ye, \overline{ln}, sr, ga] \tag{8}$$

On the other hand, o^j dataframe is constructed, it represents the characteristics of job offer j, and is defined as shown in

$$o^j = [\overline{me_o}, \overline{rs_o}, \overline{ds_o}, ye_o, \overline{ln_o}, sr_o, ga_o] \tag{9}$$

Once the characteristics of the job offer and the user profile have been obtained, the affinity between both is evaluated, indicating whether the offer should be recommended to the user or not. To this end, we established two offer and user affinity filters prior to the analysis.

In the first filtering, the unwanted skills of the user must be evaluated against the required skills of the offer, in order to determine the user's possible interest. To carry out this process the equation defined in (10) is established and the recommendation is only considered if the comparison is satisfied.

$$|\overline{rs_o} \cap \overline{us}| < |\overline{rs_o}| * thr_{us} \tag{10}$$

If the job offer ($\overline{rs_o}$) includes so many unwanted skills ($\overline{us}$) that the value exceeds the threshold (thr_{us}), the offer is automatically discarded for that user due to the number of skills that they are not interested in. The values of both, the threshold and the weighting (based on the frequency of occurrence in the rejected offers) of the unwanted skills is updated by the CBR system, as described below:

The second filtering prior to the affinity analysis, analyzes the geographic information of the offer and the user profile, taking into account the historical information of the user's behavior. This geographic analysis is denoted by the ga' parameter and as in the previous case, if conditions are not met the suggestion may be discarded immediately. This is because no matter how relevant the offer, if it does not fit the user's geographic area, it will not be of their interest.

Fig. 3 shows a flowchart that explains the behavior of the system in the different possible cases. Initially, the system detects if the location of the job offer and the user profile are the same ($ga = ga_o$). If so, the system will consider the ga' parameter in the subsequent affinity analysis by applying the weight max_{ga} to the factor. In the opposite case, in which the location of the job offer and the user profile are different, it is evaluated if the offer requires the candidate's location to match with the location of the offer ($ga = ga_o$ required?). If so, the suggestion is ruled out. However, if it is not a requirement that $ga = ga_o$, it is necessary to evaluate if the user has previously applied to offers in different geographic areas ($ga_o \neq ga$). If not, the possible recommendation is discarded, whereas if the user has applied to offers where $ga_o \neq ga$, the offer is not directly discarded, but the factor ga' is not considered in the subsequent final affinity analysis.

If the offer has not been discarded after passing through the two filters, the affinity between the job offer and the user profile is analyzed. This analysis evaluates the adaptation of characteristics of the profile of user i to the different requirements of offer j. To do this, the vector with the values of the factors $\overline{fo^{ij}}$, defined in (11), is calculated according to metrics indicated in Table 2.

$$\overline{fo^{ij}} = [me', rs', ds', ye', ln', sr', ga'] \tag{11}$$

Therefore, it is necessary to define a vector $\overline{vo}$ (12), which acts as a mask and determines which factors should be considered for the affinity calculation, just as it was done in vector $\overline{v}$ for user-user relationships. The same with vector $\overline{mwo^i}$ (13) with a similar behavior, but in this case it is applied to vector $\overline{mw}$.

$$\overline{vo^{ij}} = [me'', rs'', ds'', ye'', ln'', sr'', ga''] \tag{12}$$

$$\overline{mwo^i} = [max_{me'}, max_{rs'}, max_{ds'}, max_{ye'}, max_{ln'}, max_{sr'}, max_{ga'}] \tag{13}$$

Table 2
User-job offer recommendation factors and weight.

Factor	Weight
Main education (me')	$me' = \begin{cases} max_{me'} & \forall e \in \overline{me}/e \in \overline{me_o} \\ 0 & \text{otherwise} \end{cases}$
Required skills (rs')	$rs' = \begin{cases} max_{rs'} & \lvert\overline{sk} \cap \overline{rs_o}\rvert > \lvert\overline{rs_o}\rvert * thru_{rs'} \\ 0 & \lvert\overline{sk} \cap \overline{rs_o}\rvert < \lvert\overline{rs_o}\rvert * thrl_{rs'} \\ \frac{max_{rs'}}{2} & \text{otherwise} \end{cases}$
Desirable skills (ds')	$ds' = \begin{cases} max_{ds'} & \lvert\overline{sk} \cap \overline{ds_o}\rvert > \lvert\overline{ds_o}\rvert * thru_{ds'} \\ 0 & \lvert\overline{sk} \cap \overline{ds_o}\rvert < \lvert\overline{ds_o}\rvert * thrl_{ds'} \\ \frac{max_{ds'}}{2} & \text{otherwise} \end{cases}$
Previous experience (ye')	$ye' = \begin{cases} max_{ye'} & ye \geq ye_o * thr_{ye'} \\ 0 & \text{otherwise} \end{cases}$
Languages (ln')	$ln' = \begin{cases} max_{ln'} & \forall l \in \overline{ln}/l \in \overline{ln_o} \\ 0 & \text{otherwise} \end{cases}$
Salary range (sr')	$sr' = \begin{cases} max_{sr'} & sr \leq sr_o * thr_{sr'} \\ 0 & \text{otherwise} \end{cases}$

When a user does not have historical information gathered from previous cases of interaction with the recommender system, nor has previously applied to any offer, then the maximum values for each of the factors, that is, each of the elements of the vector $\overline{mwo^i}$, are initialized as defined in (14). These values have been established in a generic way based on the analysis of user preferences on the social network.

$$\overline{mwo^i} = [20, 10, 5, 10, 5, 25, 25] \tag{14}$$

Finally, the affinity values ao^{ij} linked to the user and the job offer are extracted, as defined in

$$ao^{ij} = \frac{\sum_k {fo^{ij}}_k}{\sum_k mwo^i_k \cdot {vo^i}_k} \tag{15}$$

If the threshold, which is established by the system on the basis of previous cases, is exceeded; the value ao^{ij} exceeds the threshold $thro_i$ the system recommends the job offer to the user. The system uses the default value $thro_i = 0.7$, but the value can vary in the interval [0.4, 0.8]. More specifically, a sample of 50 users has been selected manually on the basis of their job searching activity. The range [0.4, 0.8] was established according to the affinity value of the accepted recommendations. The default value $thro_i = 0.7$ is the mode of the accepted job offer recommendations.

$$\overline{thrs^i} = [thro_i, thr_{us}, thru_{rs'}, thrl_{rs'}, thru_{ds'}, thrl_{ds'}, thr_{ye'}, thr_{sr'}] \tag{16}$$

The maximum factor values ($\overline{mwo^i}$), and all the thresholds used ($\overline{thrs^i}$), are established automatically by the system using CBR, where cases follow the structure defined in

$$C_{i(n+1)} = (\overline{thrs}, \overline{fo^{ij}}, p^i, o^j, \overline{vo^{ij}}, \overline{mwo^i}, ao^{ij}, userResponse) \tag{17}$$

Where $C_{i(n+1)}$ is the case $(n+1)$ of the user i calculated from the user's information of the previous n cases. Within each case, all the necessary elements are stored so that the system can learn from the habits of individual users and recalculate each of the thresholds used for different factors. In addition, it updates the list of unwanted skills and information on geographic areas that are of interest to the user. In Fig. 4 we can see the flowchart that the system follows when evaluating if a new offer should be recommended to a user, along with the different stages of the CBR system (retrieve, reuse, check and retain) highlighted in gray.

If the system determines that a new job offer j should be recommended to user i, this offer is included in an ordered list of highest to lowest affinity (ao^{ij}). In case the user interacts with the RS and the thresholds are updated due to this interaction, the list of suggested offers is re-evaluated.

When a new user accesses the system and provides the required information in their profile but still has not interacted with the RS system and has not yet applied to any offers. Recommendations are given to that user on the basis of case DB cases of profiles with similar characteristics, until an interaction is made.

4. Experiment and results

The recommendation acceptance rate is used in this research as a measure in evaluating the effects of the presented recommendation systems.

To begin the evaluation, the initial value of different parameters and thresholds is defined and calculations are made. These values have been established on the basis of average historical behavior. There are two parts in the evaluation of the

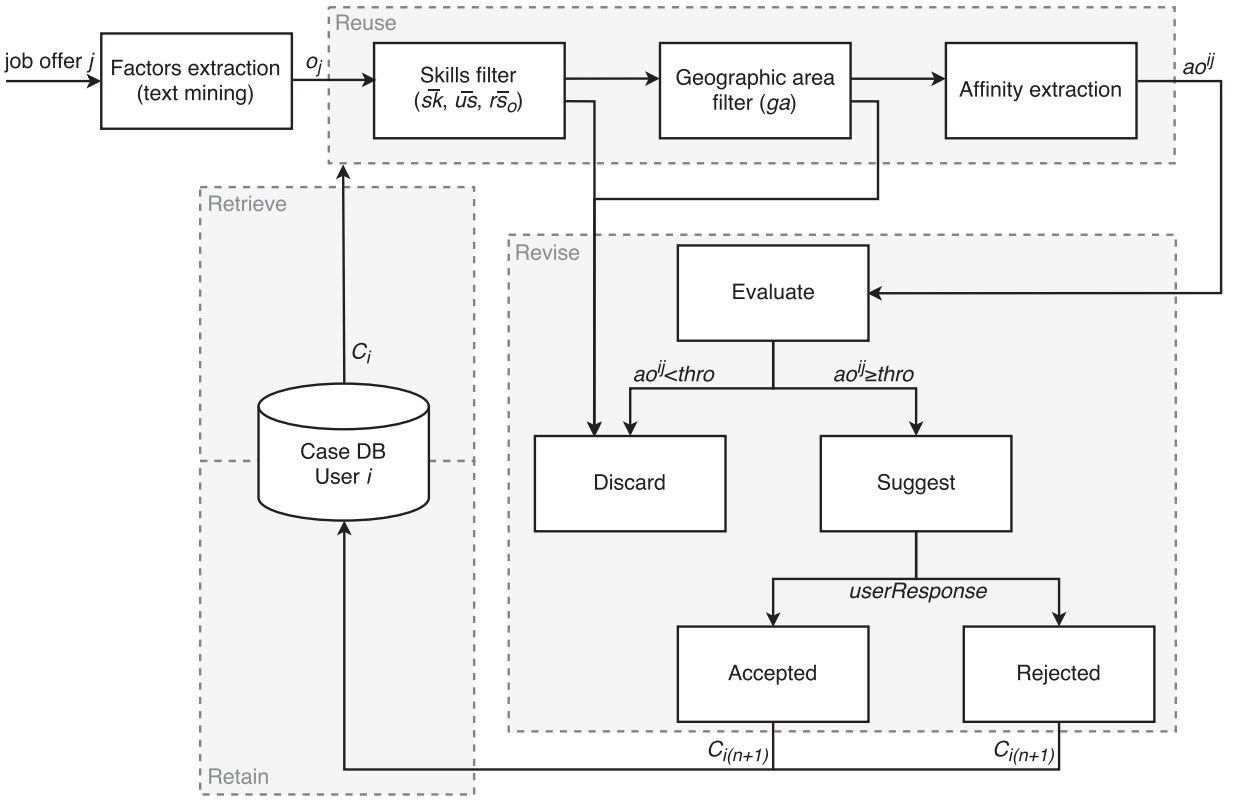

Fig. 4. Job offer recommendation system flowchart and CBR parts.

recommendation systems: (i) user-user relationships; and (ii) user-job offer relationships. However, beBee has no previous recommendation systems for these relationships so results cannot be compared.

In order to evaluate user-user relationships, a subset of 2000 random users has been selected. In order to evaluate the user-employment relationships, a study has been carried out with a subset of 250 users from each of the 6 areas with the most job offers published during the evaluation period (15 days). A total of 14,066 job offers were published in the 6 areas that have been evaluated.

The activity of users has been considered to initialize the different thresholds. To calculate the factors, each threshold has a mean value which is calculated on the basis of the available information on the offers that the user applied to.

Next, we describe the experiment data set and then we present the results of both the user-user relationship recommender system and the user-job offer recommender system.

4.1. Experiment data set

In order to evaluate the user-user relationship RS, a sample of 2000 users have been selected randomly and the 1500 most active users have been selected for the evaluation. Users were considered as active if they had expanded their network of contacts in the 15 days prior to the evaluation.

When evaluating the recommendation of a relatively new user in the system, j, to one of the selected users, i. Affinity a^{ij} has to be calculated and then it is determined whether this value exceeds the threshold thr^i or not.

The information is extracted from two profiles; vector $\overline{f^{ij}}$ is obtained for user i as shown in (18) and also vector $\overline{f^{ji}}$ for user j, as shown in (19). The values of the factors that compose these vectors are extracted by applying the metrics given in the Table 1. The default maximum values used are those defined in (4).

$$\overline{f^{ij}} = [35, 5, 4.44, 15, 15, 0, 0, 0, 0, 0, 20] \tag{18}$$

$$\overline{f^{ji}} = [35, 2.22, 20, 15, 7.5, 0, 0, 0, 0, 0, 20] \tag{19}$$

Vectors that determine the factors that will be used for calculating the affinity, are calculated for each user as defined in (2):

$$\overline{v^i} = [1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1] \tag{20}$$

$$\overline{v^j} = [1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1] \tag{21}$$

Therefore, using this information we can determine the affinity a^{ij}, whose value is $\frac{94.44}{180} = 0.52$. In this case, considering the default value of the threshold thr^i, 0.3, the result is that user j is recommended to user i.

In the opposite situation, user i is also recommended to user j, since a^{ji} is $\frac{99.72}{170} = 0.58$.

It should be noted that in this case, the values used are those established by default, however, they will be adjusted by the CBR system once users begin to interact with the RS.

The user-job offer RS has also been evaluated, different subsets of 250 users have been selected from those who had been active in the last 15 days and they were selected from each of the 6 areas with the highest numbers of published job offers (a total of 1500 users).

In all cases, all thresholds ($\overline{thrs^i}$: $thro_i$, thr_{us}, $thru_{rs'}$, $thrl_{rs'}$, $thru_{ds'}$, $thrl_{ds'}$, $thr_{ye'}$, $thr_{sr'}$) have been initialized with the default values as shown in

$$\overline{thrs^i} = [0.7, 0.85, 0.75, 0.25, 0.75, 0.25, 0.85, 0.9] \tag{22}$$

It is also necessary to extract information from the user profile, obtaining a value for each of the factors present in dataframe p_i. The salary range (sr) is private information so it cannot be viewed on the user's profile. So, if the user gives their permission, the system takes it into account to calculate the affinity. Information on the unwanted skills ($\overline{us}$) is also hidden on the public profile, but users can modify the skills that the CBR includes automatically.

When evaluating user-job offer suggestions, it is also necessary to describe the dataset related to job offers. In this case, the new offers published in each of the 6 categories that the 1500 selected users belong to, have been considered. These areas are: sales (4,868 job offers), trades and professions (3,359 job offers), engineers and technicians (2,291), IT (2,087 job offers), administrative (893), and teachers (568). In total, 14,066 offers have been evaluated.

Once job offers have been classified in their areas, the information necessary to build the dataframe o_j is extracted. This information is extracted from an input text provided by the user who published the offer on the social network.

In order to show the operation of the system with a practical case, we describe the process followed to evaluate if the offer j should or should not be recommended to the user with the profile p_i.

When the job offer is inserted into the system, the diagram shown in Fig. 2 is executed. Text mining techniques are applied to extract the required information and it is first classified in a professional area. Then, the values of the dataframe o_j are obtained. In this example, o_j is defined as follows:

o_j = [{"IT bachelor":2}, {"cloud computing", "linux servers", "nas storage systems", "san storage systems", "vmware"}, {}, 2, {"Spanish","English"}, "24.000 € - 27.000 €", "Barcelona"]

To evaluate the affinity of the job offer with user i, it is necessary to retrieve the current user profile to compose the dataframe p_i. In this case, the value of p_i is:

p_i = [{" IT bachelor" :2, " IT master" :3}, {" linux" , " java" , " cloud computing", " azure", " system administration", " scala", " apache cassandra", " apache spark", " open stack"}, {" frontend", " customer service", " cobol"}, 2, {" Spanish", " English"}, " 17.000 €", " Salamanca"]

In addition, all previous interactions of user i, e.g. C_i (CBR recovery stage), are recovered. Then, the system moves to the reuse CBR stage. At this stage, the first thing to check is whether the offer includes unwanted skills as defined in Eq. (10). In this example, the intersection of required skills ($\overline{rs_o}$) and unwanted skills ($\overline{us}$) is the empty set, so its cardinality is 0 and the equality is satisfied. The system therefore goes on to check the geographic information.

In this example, user i has previous job offer applications stored in the history where the geographic area (ga_o) is different from their place of residence (ga="Salamanca"), so offers are not discarded when $ga \neq ga_o$, however, this factor has no influence on the calculation of affinity, as described in Fig. 3.

With this information, the vector $\overline{vo^{ij}}$ is initialized with the values shown in

$$\overline{vo^{ij}} = [1, 1, 0, 1, 1, 1, 0] \tag{23}$$

The factor values of the vector $\overline{fo^{ij}}$ can be calculated by applying the metrics defined in Table 2. The values used in vector $\overline{mwo^i}$, are those defined in (14). The result is shown in

$$\overline{fo^{ij}} = [20, 5, 0, 10, 5, 25, 0] \tag{24}$$

According to Eq. (15), affinity ao^{ij} has the value of 0.928. This affinity is evaluated in the review stage of the CBR system. In this case, it exceeds the threshold $thro_i$ of the user, so the offer is recommended and shown in the user's offer recommendation list.

At the moment in which the user accepts or rejects the recommendation, the CBR system goes from the review stage to the reuse stage, and the case $C_{i(n+1)}$ is included as part of the case memory of the system. In the case of the recommendation being rejected, the system evaluates if any of the characteristics have to be added to the user's unwanted skills vector ($\overline{us}$).

Table 3
User-user recommendation results.

	Recommended	Accepted	Rejected	Ignored
Total	17,073	11,765	2223	3085
Average	11.38	7.84	1.48	11.38
Rate (%)	100	68.91	13.02	18.07

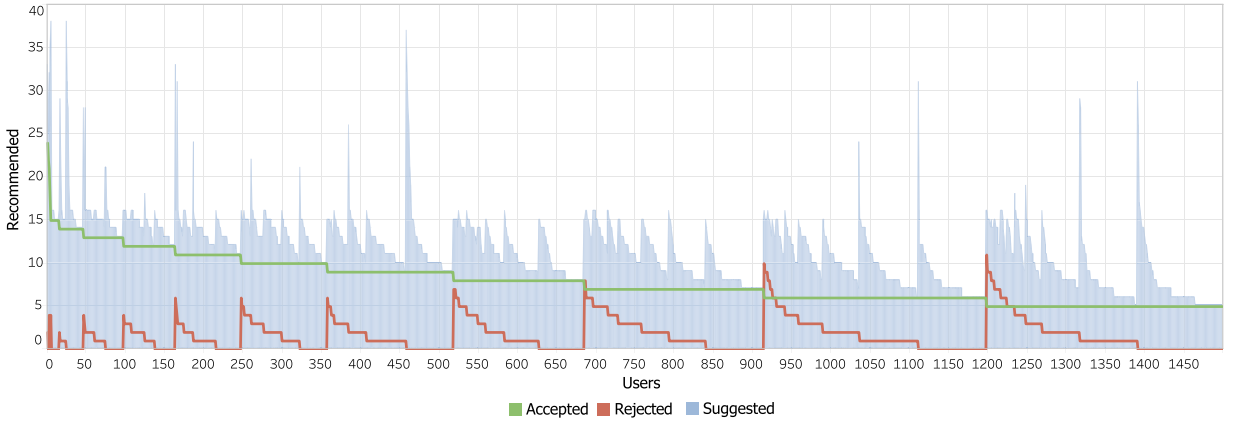

Fig. 5. User-user recommendations response graph.

4.2. Results

In this section the results obtained from this exemplary dataset are presented and the two proposed RSs are evaluated separately. First, the user-user and then the user-job offer relationship recommender systems.

4.2.1. User-User recommendations

A randomly selected sample of 2000 users has been given exclusive access to the RS functionality. A subset of 1500 users, who had the highest interaction rates during the 15 selection days, has been chosen. This was done because some of the selected users did not interact with the system during the evaluation period. The values collected for the 1500 selected users are shown in Table 3.

It can be seen that, in total, the system recommended 17,073 relationships to 1500 users. The number of accepted recommendations is 11,765, 68.91% of the total. However, only 13.02% of recommendations were rejected, since the remaining 18.07% of recommendations were ignored. However, if we only analyze the interactions with the recommender system, 84.11% are accepted and only 15.89% are rejected.

Similarly, Fig. 5 shows a graphical representation of the interaction of every user with the RS. It shows the number of accepted and rejected recommendations, along with the total number of recommendations. Users are ordered depending on the number of accepted recommendations first, then depending on the number of rejected recommendations and finally depending on the number of offers recommended. It can be seen that only 38 users have more rejected recommendations than accepted ones. In addition, 19.13% of users have an acceptance rate equal to or higher than 90% (including the ignored ones), which shows that the RS is well adapted to individual user profiles.

The recommendation of relationships with other users is performed in an ordered way, where the ones with the highest affinity are the first to be recommended and therefore, the study of the CBR system's evolution is not relevant. The acceptance rate is constant and it stays constant when the user interacts with the system. However, this fact can be evaluated favorably because the rate does not fall when the user increases their number of contacts.

4.2.2. User-Job offer recommendations

Regarding job offer recommendations to users, the results obtained from the subsets of described users are shown in Table 4.

More specifically, it is displaying the number of recommendations that the RS has made to the 250 users from each of the categories, when a new offer has been added. Each user that receives job offer recommendations can choose to interact or not to interact with the system. If the user interacts with the system, it has to accept or reject the recommendation.

As shown in Table 4, there are certain differences in the behavior of users depending on the area they belong to. The main reason for this is the current situation in that particular field of work in Spain. Thus, in addition to the affinity value, the number of recommendations depends on both the number of users inscribed in each of the professional areas and the number of job offers published in each area.

Table 4
User-Job offer recommendation results.

Area	Recomm.	Accepted	Rejected	New offers
Teachers	2176	1127	357	568
Administrative	4192	1922	529	893
IT	5246	1545	517	2087
Engineers & technicians	4975	1930	756	2291
Trades & professions	5873	2289	925	3359
Sales	7403	2613	822	4868

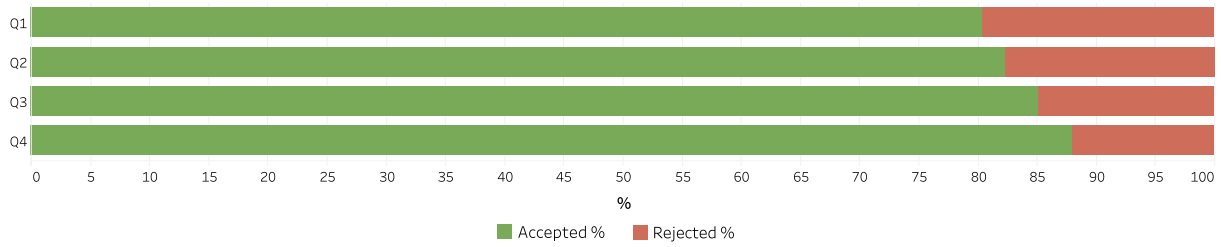

Fig. 6. 25% most active users evolution quartiles.

Table 5
25% most active users data grouped into quartiles.

	Q1	Q2	Q3	Q4	Avg
Accepted (Avg)	7.22	7.41	7.96	8.77	7.83
Rejected (Avg)	1.75	1.59	1.39	1.19	1.57
Accepted (%)	80.49	82.33	85.13	88.05	84.11
Rejected (%)	19.51	17.67	14.87	11.95	15.89

It can be observed that the lower the number of offers in an area, the higher the percentage of acceptance rates. On the other hand, the more offers in an area, the more selective the users are.

However, when viewing user interaction with job offer recommendations, the RS shows that the acceptance percentage is two times greater than the rejection percentage for all areas.

In addition, because of the use of the CBR system, the difference between the percentage of acceptance and the percentage of rejection increases as users interact with the RS, as shown in Fig. 6. We can therefore affirm that the RS improves its success rate when used over time.

For the evaluation of the CBR system, the data generated by the 1500 users have been considered. These data are displayed in Table 5. The percentage of applications for the recommended offers varies from 80.49% in the first quartile of interactions, to 88.05% in the last quartile, so efficiency improves after the users interact with an average of 10.98 offers (including the ignored ones).

5. Conclusion and future work

In this work, a context-aware RS based on a CBR system has been proposed in order to recommend user-user and user-job offer relationships. Both types of relationships follow the same procedure, however, the different factors that influence the recommendation and their weight have to be identified for each. Additionally, these weights adapt themselves to the profile that each user defines when interacting with the system.

The use of a CBR system allows to improve acceptance rates and reduce rejection rates with time, as the user continues to interact with the RS. For new user profiles, the system uses knowledge acquired from similar cases in order to determine different thresholds and calculate affinities.

As for future work, we are already working on making the system available to all users of this social network, regardless of their country (it has been tested only in Spain). To this end, independent evaluations will be carried out to check whether the system is able to obtain reasonably similar results to those presented in this article, despite markets having different behaviors.

In addition, from the continuous use of the CBR system, we are starting to determine situations in which alerts should be sent; with their help expert users will be able to act manually in cases that are identified as doubtful (those which are close to crossing the threshold), deciding whether or not the offer should be recommended. In this way, the system can be manually arbitrated by an expert user and the efficiency of the recommendations should increase.

Finally, as part of our future work, we believe it will be interesting to compare the performance of the proposed system with other existing recommender systems. In order to make a valid comparison other RSs will have to be specifically adapted to the context of the present case.

Acknowledgments

This work has been supported by project "MOVIURBAN: Máquina social para la gestión sostenible de ciudades inteligentes: movilidad urbana, datos abiertos, sensores móviles". ID: SA070U 16. Project co-financed with Junta Castilla y León, Consejería de Educación and FEDER funds. The research of Pablo Chamoso has been financed by the Regional Ministry of Education in Castilla y León and the European Social Fund.

References

[1] Agentbuilder.com, 2017, AgentBuilder. [online]. [Accessed 15 Dec. 2017]. Available from: http://www.agentbuilder.com/.
[2] G. Adomavicius, A. Tuzhilin, Context-aware recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 191–226.
[3] D. Agnihotri, K. Verma, P. Tripathi, Pattern and cluster mining on text data, in: Proceedings of the Fourth International Conference on Communication Systems and Network Technologies (CSNT), 2014, IEEE, 2014, pp. 428–432.
[4] Bebee.com, 2017, beBee, Affinity Networking; Professional social network | beBee. [online]. [Accessed 15 Dec. 2017]. Available from: https://www.bebee.com/.
[5] F. Bellifemine, G. Caire, A. Poggi, G. Rimassa (2003), 'JADE: A white Paper'. exp, vol. 3, No. 3, September 2003, Available at http://exp.telecomitalialab.com.
[6] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl. Based Syst. 46 (2013) 109–132.
[7] R.H. Bordini, J.F. Hübner, M. Wooldridge, Programming Multi-Agent Systems in Agentspeak Using Jason, 8, John Wiley & Sons, 2007.
[8] D. Bridge, M.H. Göker, L. McGinty, B. Smyth, Case-based recommender systems, Knowl. Eng. Rev. 20 (03) (2005) 315–320.
[9] R. Bulander, M. Decker, G. Schiefer, B. Kolmel, Comparison of different approaches for mobile advertising, in: Proceedings of the Second IEEE International Workshop on Mobile Commerce and Services, 2005. WMCS'05, IEEE, 2005, pp. 174–182.
[10] R. Burke, Hybrid recommender systems: survey and experiments, User Model User-adapt. Interact. 12 (4) (2002) 331–370.
[11] W.-Y. Chen, J.-C. Chu, J. Luan, H. Bai, Y. Wang, E.Y. Chang, Collaborative filtering for Orkut communities: discovery of user latent behavior, in: Proceedings of the 18th International Conference on World wide web, ACM, 2009, pp. 681–690.
[12] J.M. Corchado, R. Laza, Constructing deliberative agents with case-based reasoning technology, Int. J. Intell. Syst. 18 (12) (2003) 1227–1241.
[13] M. Cossentino, N. Gaud, V. Hilaire, S. Galland, A. Koukam, ASPECS: an agent-oriented software process for engineering complex systems, Auton. Agent Multi Agent Syst. 20 (2) (2010) 260–304.
[14] J.F. De Paz, J. Bajo, S. Rodríguez, G. Villarrubia, J.M. Corchado, Intelligent system for lighting control in smart cities, Inf. Sci. (NY) 372 (2016) 241–255.
[15] A. Esmaeili, N. Mozayani, M.R.J. Motlagh, E.T. Matson, A socially-based distributed self-organizing algorithm for holonic multi-agent systems: case study in a task environment, Cogn. Syst. Res. 43 (2017) 21–44.
[16] M. Esteva, B. Rosell, J.A. Rodriguez-Aguilar, J.L. Arcos, Ameli: an agent-based middleware for electronic institutions, in: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1, IEEE Computer Society, 2004, pp. 236–243.
[17] R. Feldman, M. Fresko, H. Hirsh, Y. Aumann, O. Liphstat, Y. Schler, M. Rajman, Knowledge management: a text mining approach, in: Proceedings of Practical Aspects of Knowledge Management, PAKM, 98, 1998, p. 9.
[18] R.L. Fidalgo, J. Rodriguez, Creation of deliberative agents using a CBR model, Comput. Inf. Syst. 8 (2) (2001) 33–39.
[19] S. Galland, N. Gaud, S. Rodriguez, V. Hilaire, Janus: another yet general-purpose multiagent platform, in: Proceedings of the Seventh AOSE Technical Forum, Paris, 2010.
[20] N. Gaud, S. Galland, V. Hilaire, A. Koukam, An organisational platform for holonic and multiagent systems, in: Proceedings of the International Workshop on Programming Multi-Agent Systems, Springer, 2008, pp. 104–119.
[21] V. Gupta, G.S. Lehal, et al., A survey of text mining techniques and applications, J. Emerg. Technol. Web intell. 1 (1) (2009) 60–76.
[22] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. (TOIS) 22 (1) (2004) 5–53.
[23] B. Hirsch, T. Konnerth, A. Heßler, Merging Agents and Services-the JIAC Agent Platform, in: Multi-Agent Programming:, Springer, 2009, pp. 159–185.
[24] J. Huang, B. Yang, D. Jin, Y. Yang, Decentralized mining social network communities with agents, Math. Comput. Model. 57 (11) (2013) 2998–3008.
[25] J. Hübner, J-Moise+ programming organisational agents with moise+ & jason, Techn. Fora Group at EUMAS 7 (2007).
[26] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, Recommender Systems: An Introduction, Cambridge University Press, 2010.
[27] J. Jiang, J. Yu, J. Lei, Finding influential agent groups in complex multiagent software systems based on citation network analyses, Adv. Eng. Softw. 79 (2015) 57–69.
[28] H.F. Jomi, S.S. Jaime, B. Olivier, S-moise+: a middleware for developing organised multi-agent systems, in: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, 2005, pp. 64–77.
[29] R. Keeling, The bologna process and the lisbon research agenda: the european commission's expanding role in higher education discourse, Eur. J. Educ. 41 (2) (2006) 203–223.
[30] J. Kolodner, C.-B. Reasoning, Morgan Kaufmann Publishers Inc, San Mateo, California, 1993.
[31] D. Krackhardt, N. Nohria, B. Eccles, The strength of strong ties, Networks in the knowledge economy, New York: Oxford University Press, 2003, pp. 82–105.
[32] D.H. Lee, P. Brusilovsky, How to measure information similarity in online social networks: a case study of citeulike, Inf. Sci. (NY) 418 (2017) 46–60.
[33] B. Lent, R. Agrawal, R. Srikant, Discovering trends in text databases, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining, KDD, 97, 1997, pp. 227–230.
[34] X. Liu, K. Aberer, Soco: a social network aided context-aware recommender system, in: Proceedings of the 22nd International Conference on World Wide Web, ACM, 2013, pp. 781–802.
[35] D.H. Park, H.K. Kim, I.Y. Choi, J.K. Kim, A literature review and classification of recommender systems research, Expert Syst. Appl. 39 (11) (2012) 10059–10072.
[36] Y. Park, J. Oh, H. Yu, Rectime: real-time recommender system for online broadcasting, Inf Sci (NY) 409 (2017) 1–16.
[37] J. Pavón, J.J. Gómez-Sanz, R. Fuentes, The Ingenias Methodology and Tools, in: Agent-oriented methodologies, IGI Global, 2005, pp. 236–276.
[38] S. Poslad, P. Buckle, R. Hadingham, The FIPA-OS agent platform: open source for open standards, in: Proceedings of the 5th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents, 355, 2000, p. 368.
[39] M. Rajman, R. Besançon, Text mining: natural language techniques and text mining applications, in: Data Mining and Reverse Engineering, Springer, 1998, pp. 50–64.
[40] S. Russell, H. Jordan, G.M. O'Hare, R.W. Collier, Agent factory: a framework for prototyping logic-based AOP languages, in: Proceedings of German Conference on Multiagent System Technologies, Springer, 2011, pp. 125–136.
[41] A. Sánchez, G. Villarrubia, C. Zato, S. Rodríguez, P. Chamoso, A gateway protocol based on FIPA-ACL for the new agent platform Pangea, in: Trends in Practical Applications of Agents and Multiagent Systems, Springer, 2013, pp. 41–51.

[42] J.J. Gomez-Sanz, R. Fuentes, J. Pavón, I. García-Magariño, INGENIAS development kit: a visual multi-agent system development environment, in: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: demo papers, International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 1675–1676.
[43] B. Sigurbjörnsson, R. Van Zwol, Flickr tag recommendation based on collective knowledge, in: Proceedings of the 17th International Conference on World Wide Web, ACM, 2008, pp. 327–336.
[44] S. Sivapalan, A. Sadeghian, H. Rahnama, A.M. Madni, Recommender systems in e-commerce, in: Proceedings of World Automation Congress (WAC), 2014, IEEE, 2014, pp. 179–184.
[45] J. Wang, Y. Zhang, Opportunity model for e-commerce recommendation: right product; right time, in: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2013, pp. 303–312.
[46] Z. Wang, J. Liao, Q. Cao, H. Qi, Z. Wang, Friendbook: a semantic-based friend recommendation system for social networks, IEEE Trans. Mob. Comput. 14 (3) (2015) 538–551.
[47] B. Yang, J. Huang, D. Liu, J. Liu, A multi-agent based decentralized algorithm for social network community mining, in: Proceedings of International Conference on Advances in Social Network Analysis and Mining, 2009. ASONAM'09, IEEE, 2009, pp. 78–82.
[48] Y. Yang, D. Yue, C. Dou, Distributed adaptive output consensus control of a class of heterogeneous multi-agent systems under switching directed topologies, Inf. Sci. (NY) 345 (2016) 294–312.