Collaborative information extraction for adaptive recommendations in a multiagent tourism recommender system

Víctor Sánchez-Anguix, Sergio Esparcia, Estefanía Argente, Ana García-Fornes and Vicente Julián

Abstract Recommender systems are capable of providing users with personalized information that is adjusted to their needs. One particular case of these systems is applied to the tourism industry. Tourism recommender systems offer services such as hotel reservation, restaurant reservation, and visit plans personalized for each tourist. The Social-Net Tourism Recommender System is a tourism recommender system that is based on agent technology and uses social networks as its basis for recommendations. Nevertheless, these systems face some problems. On the one hand information needs to be mantained up-to-date, which can result in a costly task if not performed automatically. On the other hand, it may be interesting to include third parties services in the recommendation since they improve the quality of the recommendations. In this paper, we present an add-on for the Social-Net Tourism Recommender System that uses information extraction and natural language processing techniques in order to automatically extract and classify information from the Web. Its goal is to maintain the system updated and obtain information about third parties services that are not offered by service providers inside the system.

1 Introduction

Over the last few years, the Web has become the greatest source of available information. A goal for researchers is to obtain optimal ways of recovering specific information from the Web. One of the most important information filtering techniques are recommender systems, whose goal is to present information items that could be interesting to the user. Recommender systems attempt to reduce information overload by selecting subsets of items based on user preferences. This kind of systems are able to provide users with personalized information that covers their

Universidad Politécnica de Valencia
Departamento de Sistemas Informáticos y Computación
Grupo de Tecnología Informática - Inteligencia Artificial
Camí de Vera s/n 46022, Valencia, Spain
{sanguix,sesparcia,eargente,agarcia,vinglada}@dsic.upv.es

needs. There are many domains in which recommender systems have been applied, such as the tourism industry. Tourism recommender systems are able to provide tourists with custom recommendation based on their preferences.

There are two main approaches on recommender systems: content-based and collaborative filtering. On the one hand, content-based algorithms use item content, like name and other features. Triplehop's Technologies TripMatcher[10], DIETORECS [3, 5] and [5], and Vacation Coach's Me-Print are three examples for e-commerce implementations that use content-based algorithms. In this kind of systems, recommendations depend on item description. On the other hand, collaborative recommender systems use algorithms that give recommendations based on the user's behavior (other users' experiences and opinions). Some examples can be found in [7, 11]. Collaborative filtering is currently the most used technique in recommender systems.

One example of collaborative recommender system is the Social-Net Tourism Recommender System (STRS)[8], a tourism application that helps tourists to make their visits to a city more profitable and adapted to their preferences. It uses a mobile device, such a phone or a PDA, allowing tourists to make reservations in restaurants or cinemas, and providing a visit plan for a day. STRS is based on multi-agent system technology and employs social networks as a mechanism to give recommendations.

However, some problems arise in STRS and most tourism recommender systems. First, each service provider is responsible for keeping up-to-date their business information in the recommender system. If it is not carried out by an automatic process, keeping information up-to-date is a costly task. Therefore, the use of non-automated update mechanisms may lead to negligent behaviours conducted by the users, whom rely on the performance of the recommender system despite its current information. This information may be available on the user's website. Second, there may be third parties (tourist service providers that are not part of the system) which offer types of services that are not offered in STRS. The inclusion of information of third party services may enhance recommendations given to tourists, and therefore improves the system. For instance, if we know that the tourist likes theater and our system users do not offer theater services, it may be wise to include information about a third party event related to theater in the recommendation. Even although third parties are not part of the system, it should be noted that the final goal of the system is to satisfy tourists and, consequently, increase the benefits of those parties that are part of the system. This information about third party services may be available on the Web. Again, an automated update mechanisms is convenient in order to be able to cope with the possible dynamic content of these third party services.

The main aim of this work is to create an add-on that can extract and keep up-to-date information from the websites of third parties and tourist service providers. Two are the advantages of the inclusion of the add-on. On the one hand, the tourism recommender system dynamically adapts its information. On the other hand, recommendations can be richer and thus more adaptable to the tourist profile. Wrapper agents and natural language processing based agents are used to extract and classify information respectively. The information classification process is performed by means of a voting process which is governed by a trusted mediator that can adjust the voting power of each classification agent.

The remainder of this paper is organized as follows. Section 2 provides an overview for the STRS architecture; section 3 gives a description of the add-on: the description for information extraction and information classification agents, and an explanation for the extended architecture; section 4 presents the experiments used to test the system, showing the classification accuracy of the system and an experiment in which a simple update rule for the voting power is used; finally, section 5 presents some conclusions and future work.

2 The Social-Net Tourism Recommender System architecture

The Social-Net Tourism Recommender System (STRS)[8] is a tourism application, which offers different services to tourists. Its goal is to improve their stay in a city, spending their time in the most efficient way. Tourists can find two kinds of information, according to their interest: personal interesting places (restaurants, cinemas, museums, theaters...) and general interesting places (monuments, churches, beaches, parks...). Users can set their preferences using a mobile phone or PDA. Tourists can make a reservation in a restaurant, buy tickets for a film or a concert, and so forth. Also, users can add a reservation into a plan to visit a city.

STRS integrates Multi-Agent technology and a recommender system based on social network analysis. It uses social networks to model communities of users, trying to identify the relations among them, to identify users similar to others so as to get a recommendation and recommend the items the users like the most.

STRS is formed by two subsystems that cooperate to provide comprehensive and accurate tourism recommendations: the Multi-Agent Tourism System (MATS) and the Multi-Agent Social Recommender System (MASR). Both systems are shown in Figure 1:

Fig. 1 Social-Net Tourism Recommender System

- MASR is formed by four types of agents: (i) *user agent*: an interface between the user and the MASR; (ii) *data agent*: an agent responsible for the management of a database with user's data; (iii) *recommender agent*: receives all the recommendation and users registration queries; and (iv) *social-net agent*: adds a node onto the social agent when a user joins the platform and determines the new user's similarity with regards to the other profiles.
- MATS is also composed by four agents: (i) *broker agent*: in charge of establishing a communication between the user and sight agents; (ii) *sight agent*: manages all the information regarding the characteristics and activities of a specific place of interest in the city; (iii) *user agent*: allows tourists to use the different services

by means of a GUI on their mobile devices; and (iv) *plan agent*: establishes and manages all the planning process offered by the system, taking into account preferences and searches.

As commented before, one of the main problems of this proposal is to maintain information up-to-date and to integrate information that could be interesting for the tourists and which is placed outside of our system. Therefore, next section presents an add-on for STRS that is capable of maintain the information updated and is able to introduce new information which is not supplied by any of the service providers of the system.

3 The collaborative information extraction for adaptive recommendations add-on

The proposed add-on is based on a collaborative strategy that extracts and classifies leisure services available on the Web. It is composed of two different types of agents: information extraction agents (IE agents) and information classification agents (IC agents). IE agents are needed to extract information that is available on the Web. However, the mere extraction of information may not provide reliable evidence about its contents. Therefore, additional techniques are needed to classify such information. In our case, we employ a set of collaborative agents that use natural language processing techniques (NLP) in order to classify information into a category of leisure service (concert, theater play, exhibition, etc...).

This section describes how the add-on was designed. IE agents are described in subsection 3.1. Then, IC agents are explained.

3.1 Information extraction agents

IE agents are the ones entrusted with the task of extracting information from the different websites. They are designed following a wrapper architecture. Thus, one wrapper agent is needed per each website that is analyzed. Our wrapper agents transform the information available in a website into fields that are interesting for the application. For instance, it may be interesting to convert all the information contained in a leisure service into specific fields such as *name, address, price, time*, and so forth. In order to extract these specific fields, our IE agents examine the HTML structure of the website looking for specific patterns that point to the place where such fields are.

Even though some information has been extracted, some other fields may not appear explicitely. Thus, additional mechanisms are needed to infer missing information. In our case, the event category needs to be inferred since many times it does not appear in the original website. Our IE agents send the event description they have extracted to an organization of IC agents, a team-based organization, and wait for their opinion. IC agents decide the category of such information and send their decision to the corresponding IE agent. Once the decision from IC agents has arrived, the service information along with its category is sent to a Sight agent that is responsible for storing and managing information related to that specific service category.

It must be noted that our wrapper agents were specifically created for some test websites. However, adapting wrappers to new websites does not pose a major problem since there are techniques that allow to generate wrappers automatically[4, 6].

3.2 Information classification agents

Each IC agent is specialized in classifying into one specific event category. As a matter of fact, each IC agent gives a score that represents its confidence. The higher the score, the more confident the agent is in the classification of the service description into the agent's specializing category. In order to analyze how relevant a service description is with respect to a specific service category, each IC agent uses a rule based system based on NLP knowledge. Rules are applied over a preprocessed service description that only contains filtered words (using a stop-word list) and their corresponding lemma and lexical category. The Freeling library [2] is used in order to infer words' lemmas and their lexical category. Once the description has been preprocessed, two different types of rule can be applied over inferred lemmas:

1. Term Strength rules: Term Strength (TS) [12, 13] is a measure of how relevant a word/lemma is with respect a specific category. More specifically, the TS of a word with respect a specific category can be calculated as follows:

$$TS(w_i) = \frac{\sum_{i=1}^{|D|}\sum_{j\neq i}^{|D|} occurs(w_i,d_i) * occurs(w_i,d_j)}{\sum_{i=1}^{|D|}\sum_{j\neq i}^{|D|} occurs(w_i,d_i)} \tag{1}$$

   where w_i is a word/lemma, $D = \{d_1, d_2, ..., d_{|D|}\}$ is a set of documents related to a specific category, and $occurs(w, d_i)$ is a function that returns 1 if the word/lemma w_i can be found in the document d_i and 0 otherwise. The TS of words with respect a specific category is precalculated using a corpus. We propose the following mechanism for TS rules: TS rules look for lemmas whose TS value has been precalculated during the training phase. If a match is found, the matched TS rule r_j produces a score SC equal to the precalculated TS.

$$SC_{TS}(w_i) = TS(w_i) \tag{2}$$

   where SC_{TS} is the score, and w_i is a word/lemma.
2. Hyperonym rules: These rules are based on hyperonym trees found in Wordnet [9]. Hyperonymy is the semantic relation between a more general word and a more specific word. In hyperonym trees, the root is the word/lemma that is analyzed whereas leaves are the most general words that are related to the root. It

must be noted that tree nodes have different branches and each branch represents a different word sense. Additionally, branches are ordered according to the frenquency of that specific sense. We propose the following mechanism for Hyperonym rules: specific patterns are searched in the hyperonym tree. These patterns should be indicated by an expert in the area. If the rule matches, it produces a score SC_H that is equal to:

$$SC_H(w_i) = \frac{|S(w_i)| - (Order(s_i) - 1)}{\sum_{k=1}^{|S(w_i)|} k} \tag{3}$$

where w_i is the word/lemma analyzed, $S(w_i)$ is the ordered set of senses of w_i, s_i is the sense where the pattern was found, and $Order(s_i)$ calculates the position of s_i in the ordered set $S(w_i)$. Using this score function, those senses that are less frequent give lower score than those that are the most frequent ones.

Each IC agent tries to apply each of its own rules to the filtered service description. Then, each word/lemma has an associated score that is equal to the rule that matched with that specific word/lemma, and produced the highest score. The score of the event description with respect to a specific category is the sum of the associated score of every word/lemma that is part of the event description. These statements can be formalized as follows:

$$SC(W) = \sum_{i=1}^{|W|} \max_{r_j \in R} \; SC(w_i, r_j) \tag{4}$$

$$SC(w_i, r_j) = \begin{cases} SC_{TS}(w_i) & \text{if } r_j \in \text{TS rule} \\ SC_H(w_i) & \text{if } r_j \in \text{Hyperonym rule} \end{cases} \tag{5}$$

where W is the set of filtered words/lemmas, R is the set of rules of the IC agent, w_i is a word/lemma, r_j is a rule, and $SC(w_i, r_j)$ is the score produced by rule r_j when applied to w_i.

Fig. 2 The architecture of the STRS system and its add-on. 1.An IE agent extracts a service description from the Web; 2.This agent requests a classifiction service to the IC organization and includes the service description; 3.The contact agent of the organization broadcasts the service call and its associated service description to every IC agent; 4. Each IC agent emits a vote/score and its specializing category that is sent to the trusted mediator; 5.The mediator agent decides which category to assign to the service description based on the highest score and each agent voting power; 6.The classification results are sent back to the invoking IE agent; 7.This IE agent passes the service information to the corresponding Sight agent in the STRS

All of the IC agents form a team-based organization. The fact that IC agents form an organization has advantages over non-organized agents. First, IE agents do not need to know every single IC agent. They only need to know the organization instead. The organization offers a classification service that takes an event description as an argument and returns the appropiate event category. One agent, that acts as contact agent, broadcasts the service call to every IC agent. Then, the information classification process is performed by means of a voting process which is governed by a trusted mediator that can adjust the voting power of each classification agent. Each agent calculates and sends a score and its associated category to the trusted mediator. The trusted mediator classifies the service description based on which agents are more confident with their decisions (higher score). The final classification is sent to the service invoker (a IE agent).

As it was mentioned above, the trusted mediator can adjust the voting power (vp) of each IC agent according to some past experiences. More specifically, the mediator can penalize agents whose decisions have shown to be wrong in the past. This can be formalized as follows:

$$Category(W) = \underset{a_i \in IC}{\operatorname{argmax}} \; vp_{a_i} * SC_{a_i}(W) \tag{6}$$

where IC is the set of IC agents, and vp_{a_i} is the voting power that the mediator grants to the agent a_i.

The complete architecture of the Social-Net Tourism Recommender System and the designed add-on can be found in Figure 2. It shows the whole process of information extraction, information classification and its integration in the STRS system.

4 Experiments

Two experiments were carried out in the implemented version of the STRS under Magentix[1]. The goal of the first experiment was to test the classification accuracy of IC agents. Three service categories were selected as a testbed for this experiment: Concerts, exhibitions, and theater plays. A corpus of 600 service descriptions was built (200 event descriptions per category). The 70% of the corpus was used as training, whereas the other 30% was used for testing purposes. The voting power of each agent was fixed to w_{a_i} =1, and it remained static during the whole process. The results of the first experiment can be found in Table 1. The table shows the accuracy of the add-on according to its classification error. It can be observed that the designed system performs similarly with respect to the training corpus and the test corpus. Both values are around 11% of classification error, which can be considered as a good performance.

Corpus	Classification Error (%)
Training	11.79
Test	11.11

Table 1 This table shows the error of the designed IC agents according to the corpus obtained for training and testing purposes

The second experiment was performed in order to observe how the IC Agent Organization deals with agents that have a bad behaviour, i.e. malicious agents or agents that have been bad designed. The three agents that were used in the first experiment were also used in this second experiment (music agent, theater agent, exhibition agent). Additionally, three malicious(or bad designed) agents that represent *music, theater, exhibition* categories were also introduced in the system. These malicious agents generate high scores with a high probability. The mediator checks agents' behaviours at intervals of 10 service calls. Then, it applies a decay on the voting power vp_{a_i} based on the behaviour of the agent in the past 10 service calls. The decay formula can be formalized as follows:

$$vp_{a_i}^{t+1} = vp_{a_i}^{t} - \frac{FP_{a_i}}{|N_{other}|} + \frac{TP_{a_i}}{|N|} \tag{7}$$

where $vp_{a_i}^{t+1}$ is the new voting power, $vp_{a_i}^{t}$ is the voting power of agent a_i in the last check, FP_{a_i} is the number of times where the system decision was given by agent a_i and the correct service category was not the one a_i represents, TP_{a_i} is the number of times where the system decision was given by agent a_i and the correct service category was the one a_i represents, $|N|$ is the total number of service calls (10 in this case), and $|N_{other}|$ is the total number of service calls whose associated service category is not the one agent a_i represents.

It is acknowledged that there may be more sophisticated decay functions. Nevertheless, the aim of this experiment was to design a simple experiment where the benefits of the agent organization could be observed. The experiment was run for 100 random service calls. The results of this experiment can be observed in Figure 3. This Figure shows the evolution of agents' voting power as the number of service calls increases. It can be observed how agents that were bad designed (malicious agents) had their voting power reduced to values close to zero, whereas the other agents's voting power remained at the maximum. Therefore, the agent organization was capable of reducing the voting power of those agents that introduced error in the system.

Fig. 3 Evolution of agents' voting power

5 Conclusions

In this work, an add-on for the Social-Net Tourism Recommender (STRS) was presented. The recommender system is based on multi-agent system technology and it uses social networks to make its recommendations. The designed add-on solves two problems that recommender systems are affected by. At one hand, keeping information up-to-date is a crucial goal that recommender systems must achieve. At the other hand, it is a very interesting feature for our system to offer tourists with information provided by third parties that are not registered as service providers of the recommender system.

The Web is used as a source of information for third party and system user services. In order to extract the desired information, two types of agents were created: information extraction agents (IE agents) and information classification agents (IC agents). IE agents are based on wrapper technology, and their goal is to extract the information that is required in order to keep the system up-to-date or extract information from third parties. Nevertheless, sometimes the category of the service that has been extracted does not appear explicitely. Therefore, IE agents need to pass the information to IC agents.

An IC agent is capable of scoring/voting the information with respect to the leisure service category it is specialized in. It uses natural language processing techniques to accomplish such task. All IC agents form a team-based organization where a trusted mediator governs a voting process where the category of the service is decided. The trusted mediator is capable of adjusting the voting power of each agent.

Some experiments were carried out in order to test the performance of the add-on. First, the classification accuracy of IC agents was tested. Results show that the system classified incorrectly in the 11.79% of the cases in training and 11.11% of the cases in test phase. Additionally, an experiment was carried out where a simple update rule for the voting power of each agent was used. Results show that bad designed agents or malicious agents have their voting power almost nullified by the mediator.

Acknowledgements This work is supported by TIN2008-04446 of the Spanish government, CONSOLIDER-INGENIO 2010 under grant CSD2007-00022, and FPU grant AP2008-00600 awarded to V.Sánchez-Anguix.

References

1. J. M. Alberola, J. M. Such, A. Espinosa, V. Botti, and A. García-Fornes. Scalable and efficient multiagent platform closer to the operating system. In *Artificial Intelligence Research and Development*, volume 184, pages 7–15. IOS Press, 2008.
2. J. Atserias, B. Casas, E. Comelles, M. González, L. Padró, and M. Padró. FreeLing 1.3: Syntactic and semantic services in an open-source NLP library. In *Proc of the LREC'06*, pages 48–55, 2006.
3. D. Fesenmaier, F. Ricci, E. Schaumlechner, K. Wober, and C. Zanella. Dietorecs: Travel advisory for multiple decision styles. In *Proc of the ENTER'03*, pages 232–242, 2003.
4. S. Flesca, G. Manco, E. Masciari, E. Rende, and A. Tagarelli. Web wrapper induction: a brief survey. *AI Commun.*, 17(2):57–61, 2004.
5. J. Herlocker and K. J.A. Content-independent task-focused recommendation. *IEEE Internet Comput.*, 5:40–47, 2001.
6. N. Kushmerick and B. Thomas. Adaptive information extraction: Core technologies for information agents. In M. Klusch, S. Bergamaschi, P. Edwards, and P. Petta, editors, *AgentLink*, volume 2586 of *LNCS*, pages 79–103. Springer, 2003.
7. S. Loh, F. Lorenzi, R. Saldana, and D. Litchnow. A tourism recommender system based on collaboration and text analysis. *Information Technology and Tourism*, 6:157–165.
8. J. S. Lopez, F. A. Bustos, V. Julian, and M. Rebollo. Developing a multiagent recommender system: A case study in tourism industry. *International Transactions on Systems Science and Applications*, 4:206–212, 2008.
9. G. Miller. WordNet: a lexical database for English. *Commun. ACM*, 38(11):41, 1995.
10. F. Ricci and H. Werthner. Case-based querying for travel planning recommendation. *Information Technology and Tourism*, 4(3-4):215–226, 2002.

11. A. Rudstrom and P. Fagerberg. Socially enhaced travel booking: a case study. *Information Technology and Tourism*, 6(3)(special issue on Travel Recommender Systems).
12. W. J. Wilbur and K. Sirotkin. The automatic identification of stop words. *J. Inf. Sci.*, 18(1):45–55, 1992.
13. Y. Yang. Noise reduction in a statistical approach to text categorization. In *In Proc of the SIGIR'95*, pages 256–263, 1995.