Integrating DBpedia and SentiWordNet for a Tourism Recommender System

Bernadette Varga
Computer Science Department
Technical University of Cluj Napoca, Romania
Email: vargabernadette@utcluj.ro

Adrian Groza
Computer Science Department
Technical University of Cluj Napoca, Romania
Email: adrian.groza@cs.utcluj.ro

***Abstract*—The popularity of the social web introduces opportunities for the recommender systems, whilst new challenges arise when semantic knowledge is integrated in the landscape. The large amount of opinions available from Web 2.0 are exploited here to improve recommendation techniques in a semantic context. The developed recommendation system matches the crawled opinions against tourist objectives within the DBpedia ontology. Following a natural language processing step in Gate, several metrics are employed to build a recommendation plan, and formal justification is provided in case of need.**

I. Introduction

In the context of large amount of unstructured data on the Internet [1] and the need of decision support systems [2], the proposed system intends to give recommendations for trip planning. It gathers the objectives that should be visited, gets visitor's reviews, and based on sentiment analysis and evaluation functions, it is capable to give the user a plan for a few-day-trip. One advantage of the framework is that it provides explanations on the decisions that were taken.

The research conducted here can be integrated in the larger context of urbanisation within semantic web [3]. The challenges appear with the heterogeneity of semantics, data dependence, noise, inconsistency and representation. The LarKC application is designed for geo-spacial social communities, and uses information from ontologies like DBpedia and Sindice [4]. Similarly, we exploit DBpedia knowledge base for constructing a holiday trip, choosing a destination city. The DBpedia knowledge base has several advantages over existing knowledge bases: it covers many domains; it represents real community agreement; it automatically evolves as Wikipedia changes, and it is truly multilingual.

From the opinion mining perspective [5], an ideal solution for a recommendation system would be a framework that processes automatically a set of results gathered by the search process, generates a list of attributes for the products collected and aggregates the reviews about each product. The main issues regard sentiment polarity, subjectivity detection, opinion identification, extraction of significant attributes and classification based on relations [6]. The system proposed here has as a starting point the SentiWordNet lexical resource [7]. The system calculates scores for any of the reviews and review related data, on which a further classification can be done.

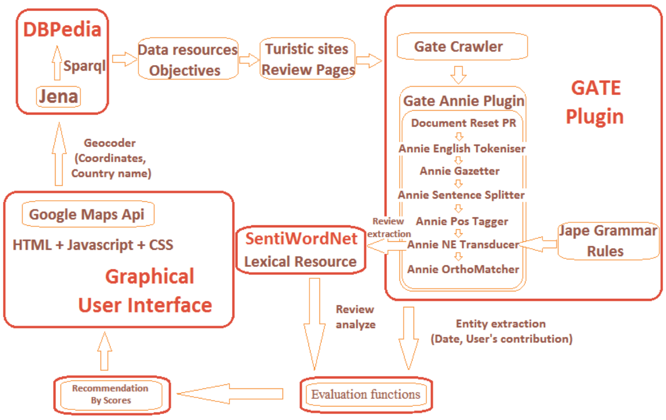


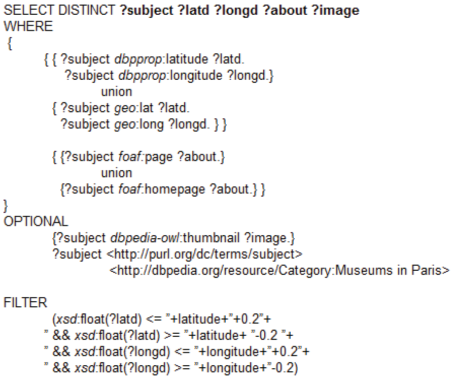
Fig. 1. System architecture

II. Architecture and components description

The system's architecture is presented in figure 1. There are two main data sources. DBpedia offers information about touristic objectives, while the travel agencies' web pages gather visitor reviews, on which opinion analysis can be done. According to the flow of data, the system first takes the tourism sights, from the DBpedia ontology by querying it in Sparql [8], with the Jena library. The tourism objectives represent the data in the recommendation system. These are the candidate elements of the final recommendation list. In the next part, the framework uses the GATE platform in collecting and analysing the associated reviews and opinion for each candidate element [9]. Reviews are taken from travel agencies' websites by GATE Web Crawler Plugin, that enables GATE to build a corpus starting from an URL address. The whole content of a web page related to a tourism attraction is further analysed by the Gate's Annie Plugin, aiming at information extraction [10].

In the analysis process, *Annie* starts working on the standard annotation list called *Original markups*, which contains types resulting from the documents format analysis. In case of an HTML document, it contains various types like body, div, form and others.Different modules in the *Annie Plugin* are used in a pipeline described below: Firstly, the *Document Reset*module resets the already generated

978-1-4577-1481-8/11/$26.00 ©2011 IEEE



```
SELECT DISTINCT ?subject ?latd ?longd ?about ?image
WHERE
 {
        { { ?subject dbpprop:latitude ?latd.
             ?subject dbpprop:longitude ?longd.}
                    union
          { ?subject geo:lat ?latd.
            ?subject geo:long ?longd. } }

          { {?subject foaf:page ?about.}
                    union
            {?subject foaf:homepage ?about.} }
}
OPTIONAL
          {?subject dbpedia-owl:thumbnail ?image.}
          ?subject <http://purl.org/dc/terms/subject>
                     <http://dbpedia.org/resource/Category:Museums in Paris>

FILTER
          (xsd:float(?latd) <= "+latitude+"+0.2"+
        " && xsd:float(?latd) >= "+latitude+ "-0.2 "+
        " && xsd:float(?longd) <= "+longitude+"+0.2"+
        " && xsd:float(?longd) >= "+longitude+"-0.2)
```

Fig. 2. Sparql query on DBpedia

annotation lists to their original, empty state. After that, the $English\ Tokeniser$ module splits the analysed content into very simple tokens, necessary to the $Gazetteer$ module, which consists of a set of lists with names of entities. Based on these lists, annotations of type Lookup are created for each matching string in the text. This type contains information like Date type, Person names, Places and others. The $Sentence\ Splitter$ module is used for segmenting the original text into sentences. This module is required for the $Part-Of-Speech$ (POS) Tagger module, which generates a POS tag as an annotation on each word or symbol. The resulting values are used by the SentiWordNet.$Annie\ NE\ Transducer$ module continues the processing by using rules to work upon the annotations from the $Gazetteer$ along with annotations from other processing resources to further identify patterns or entities from the text. This module uses Java Annotation Patterns Engine (JAPE) rules for extracting relevant data for the analysis process. With this module, from the collected pages, the system retrieves several information, like: the total number of reviews for a travel sight, and for each review retrieves it's text, the date, when the review was written, the stars granted by the author, and the total number of reviews of an author, measuring the reliability of the users. After the entity extraction phase, the reviews are analysed using SentiWordNet [11]. The last step of the analysis consists of the evaluation part. Various evaluation functions are applied to candidate data, resulting in a list of ranked travel sights. Finally, these touristic objectives are placed back on a Google map. Explanations of the reasoning process are also available for the user, as a PDF file.

III. Data collection

Collecting Dynamic Data. During the information gathering process the system generates the correspondent Sparql code (see figure 2). The query starts by prefix names definitions. These prefixes define datasets, that are interrogated. *Dbpprop*, for instance, represents the *Infobox Dataset*. Geo coordinates are used within the *geo* namespace. The Ontology Infobox Types used here contains the rdf:types of the infobox instances. The Ontology Infobox Properties dataset contains the actual data values by using properties based on the *http://dbpedia.org/ontology/propertyname* naming schema.

The example shows a *SELECT* clause, where *distinct* keyword specifies that the results are different, *?subject* represents the objective, *?latd* and *?longd* represents the latitude and longitude coordinate, where the travel sight situates, *?about* is the touristic objective's homepage, if it exists, otherwise the wikipedia page's address, and finally *?image* represents the address of the related image. All the selected data references begin with the "?" sign. In the *WHERE* clause it is mentioned how the *?latd* and *?longd* values are obtained from the union of two subclauses. The first subclause tries to retrieve the coordinates from infobox latitude and longitude properties. The second subclause contains the same information from the geo namespace's dataset. If one of the two subqueries is successful, the travel sight is returned. The second part of the *WHERE* clause carries out the union between the *page* property value within the *foaf* namespace (the correspondent wikipedia page) and the topic's homepage. In the *OPTIONAL* clause the image is obtained from the *dbpedia-owl*'s thumbnail property, and finally, the subject represents the travel sight being the term within a specified category. In our example about museums in Paris the category is *http://dbpedia.org/resource/Category:Museums_in_France*. A *FILTER* clause specifies that the *?latd* and *?longd* values must be between some predefined values. In the given query latitude and longitude are parameters, and 0.2 represent about 25 km. This means, that the touristic objectives are searched within a 25 km distance from a given point.

Getting reviews. After collecting the candidate objectives, the system starts to search and extract opinions from http://www.tripadvisor.com/, which has over 45 million reviews and opinions. In the next step with Gate Crawler Plugin the related page's content is captured as an HTML document, and it also makes standard annotation for them.

Information Extraction. Jape transducer uses rules based on regular expressions to extract some already annotated data, or to define more precise annotations. The target page is a semi-structured HTML document, which means, that the framework uses special tags too behind the default annotations.

A Jape grammar has been developed for extracting the date, when a review was written. Rules have been written for extracting the number of total reviews according to a travel sight, and for each review: for the reviews' text, number of stars granted and author's contribution.

IV. Opinion aggregation

The proposed recommendation system is based on analysis and reasoning functions. This section presents the basis of sentiment analysis, and the evaluation functions, which were used during the ordering phase.

a) Evaluation functions: The number of the opinions related to a topic can show how many people are interested in the corresponding domain. It could happen that a sight has a low ranking, or the opinions are not convincing, but if the number of related reviews is very significant, than the topic can be considered a "must see". Reviews are analysed by the system, assigning scores to each word or word structure.

Reviews' credibility depends on the author. But how does the system decide, whether a certain author or an older review should be trusted?

Firstly, the α function determines how the score changes when the total number of reviews associated to a topic increases: $\alpha(x) : \mathbf{N} \rightarrow [0, 100]$, and:

$$\alpha(x) = \begin{cases} 39.3log(1+x) & \text{if } x < 350, \\ 100 & \text{if } x \geq 350. \end{cases}$$

The score will be calculated for values between 0 and ∞. If there are no reviews the score assigned by α is 0. Otherwise, the score increases logarithmically, until it reaches the limit of 100, case in which the number of reviews is 350. For example, for a number of 10 reviews: $\alpha(10) = 39.3log(1+10) = 40.9267$ and for a number of 250 reviews: $\alpha(250) = 39.3log(1+250) = 94.2390$.

According to the Lightspeed Research's survey entitled Consumer Reviews and Research Online, published in March 2011, 62% of the people read a review about a product or a service in the last 6 months. Within these people, an amount of 66% was looking for hotel accommodation and tours online. Reevoo (www.reevoo.com/b2b) says in The Six Essentials of Social Commerce: there is a direct relationship between number of reviews per product and sales for that product, where 350 reviews is considered a significant number.

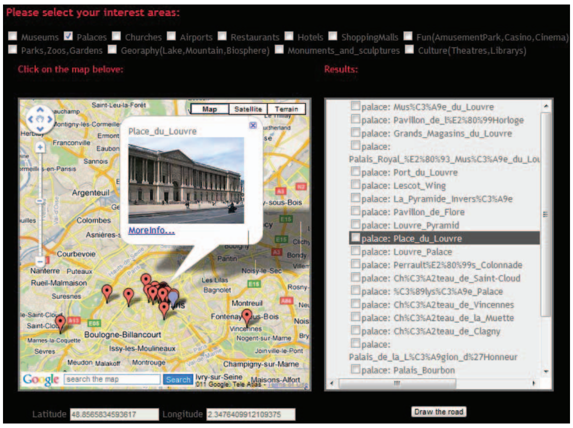
b) Review related functions: Besides the total number of reviews, the system needs to extract the following data: i) the date and time, when a review was written (τ); ii) the number of contributions of the user, who wrote the review, that the system currently analyses (ν); iii) the number of the stars, that user gave to the sight (σ); and iv) the plain text of the review (o).

i) Date and time analysis. Those reviews that have been written recently, have a higher weight, formalised as $\tau(x, a) : \mathbf{N} \rightarrow [0, 100], with : \tau(x, a) = 365a/(3.65a + x)$, where *x* represents the number of days passed after the review was written, and *a* the number of years, after which a review loses its importance. For example, if $\tau(x) = \frac{365}{3.65+x}$ (a=1):

$$\tau(2) = \tfrac{365}{3.65+2} = 64.6018 \text{ and } \tau(365) = \tfrac{365}{3.65+364} = 0.9928$$

ii) Number of user contributions behave similar to the total number of reviews associated to a sight. This means, that a user can have an unlimited number of reviews already written, but if this number is small, than the score tends to 0, else it increases to 100. Because of the similarity observed between the behaviour of this phenomenon and the behaviour of the α function, this function will be used here as well: $\nu(x) = \alpha(y)$, where *x* represents the number of reviews written by the same author, and *y* represents the total number of reviews related to the tourism sight.

iii) Number of stars. There could be a minimum of 0 and a maximum of 5 stars. This means that $\sigma(x) \in \{0, 1, 2, 3, 4, 5\}$, where *x* represents the review.



Please select your interest areas:

□ Museums ☑ Palaces □ Churches □ Airports □ Restaurants □ Hotels □ ShoppingMalls □ Fun(AmusementPark,Casino,Cinema)
□ Parks,Zoos,Gardens □ Georaphy(Lake,Mountain,Biosphere) □ Monuments_and_sculptures □ Culture(Theatres,Librarys)

Click on the map belove:

Results:

□ palace: Mus%C3%A9e_du_Louvre
□ palace: Pavillon_de_l%E2%80%99Horloge
□ palace: Grands_Magasins_du_Louvre
□ palace: Palais_Royal_%E2%80%93_Mus%C3%A9e_du_Lou
□ palace: Port_du_Louvre
□ palace: Lescot_Wing
□ palace: La_Pyramide_Invers%C3%A9e
□ palace: Pavillon_de_Flore
□ palace: Louvre_Pyramid
□ palace: Place_du_Louvre
□ palace: Louvre_Palace
□ palace: Perrault%E2%80%99s_Colonnade
□ palace: Ch%C3%A2teau_de_Saint-Cloud
□ palace: %C3%89lys%C3%A9e_Palace
□ palace: Ch%C3%A2teau_de_Vincennes
□ palace: Ch%C3%A2teau_de_la_Muette
□ palace: Ch%C3%A2teau_de_Clagny
□ palace: Palais_de_la_L%C3%A9gion_d%27Honneur
□ palace: Palais_Bourbon

Latitude 48.8565834593617 Longitude 2.3476409912109375

Fig. 3. Running Scenario

iv) Opinion. The system uses the SentiWordNet lexical resource for determining the "neutrality" score of a word within a review. Neutrality (or objectivity) has the constraint: $\eta(x) \in [0, 1]$, where *x* is the word, whose neutrality is calculated. More precisely: $\eta(x) = 1 - (PosScore + NegScore)$, where $PosScore$ represents the positivity score, while $NegScore$ the negativity one. For example, for the word *unacceptable* as an adjective, with wordnet identifier: 00018584 (meanings: not acceptable; not welcome; "a word unacceptable in polite society") the objectivity is $\eta(unacceptable\#1) = 1-(0.125+0.375) = 0.5$.

The identification of a meaning is a pretty hard operation, so the framework calculates the arithmetical mean of the neutrality scores for a word with a specified POS type. The next step is calculating the review's total score. This represents the sum of neutrality values of all the words or structures (structure of words with a single meaning within a review, for example *"well done"*). The score can be determined from $o(x) = (\sum_{k=1}^{n} \eta(k))/n$, where *n* represents the total number of words within a review and $\eta(x) \in [0, 1]$. The system is also able to detect negative forms. For example, if it encounters the word *"not"* in front of an adjective, it calculates the objectivity score using the equation: $\eta(\overline{x}) = 1 - \eta(x) = PosScore(x) + NegScore(x)$.

c) Recommendation generation: Based on the above functions, the final score of an objective is:

$$Score_{obj} = \frac{\alpha_{obj} + \omega \displaystyle\sum_{k=1}^{n} \frac{\tau_k * o(k) * \nu_k + 20\sigma(k)}{2n}}{\omega + 1},$$

According to the function, the final score of a sight is in [0,100]. In the function α_{obj} gives the score for total number of reviews related to the analysed objective. Given that sigma is in [0..5], the formula contains $20\sigma(k)$, resulting in a range of [0..100]. Similarly, τ_k $*o(k)*\nu_k$ has $o(k) \in [0..1]$, $\nu_k \in [0..100]$ and $\tau_k \in [0..100]$, where τ represents the score related to the date of a review and ν_k the user's contribution. The used weight is ω, that specifies how important are opinion related data as opposed to the total number of opinions.

Opinion	Date	Posts	τ	ν	σ	o	Score
This is the second time I've visited the louvre museum. It will take many more visits to see it all. The mona lisa is amazing and the sculptures fare beautiful but I especially love the glass pyramid, Psyche Revived by Cupid's Kiss and the venus de milo, they all there to give you the best emotions.	May 11, 2011	16	54.8	48.3	5	0.4	57.1
I absolutely love this museum. There is so much art from many different cultures, even the building itself has its own history. However, I once read a guidebook that said it would take 3 months to stand in front of every artwork in the Louvre and read the plaque! I highly recommend figuring out what you want to see ahead of time so you don't get swamped.	Apr 2, 2011	2	7.9	18.7	4	0.4	40.9

TABLE I
OPINION EVALUATION FOR THE LOUVRE MUSEUM IN MAY 14, 2011

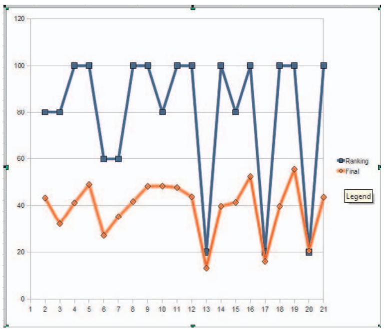


Fig. 4. Opinion score vs. stars granted

V. RUNNING SCENARIO

The system outputs a list of recommended sights which are worth visiting according to opinion mined form different tourist sites, where the objectives are categorised based on DBpedia ontology. The user is able to select his interests from an already specified list, which represent Wikipedia categories inside DBpedia. After clicking on the map, latitude and longitude values are obtained through the Google service, and finally the address is determined from the coordinates by "reverse geocoding" service (see figure 3). Based on the selected categories the system generates a query on DBpedia. The Sparql endpoint responds with the travel sight's name, an image, coordinates, and page address. On the current example all the palaces from Paris with coordinate values within a distance of 25 km from the destination point are listed. After the sights are collected, the system gathers the related reviews from www.tripadvisor.com. Table I shows two opinions about the Louvre museum. For the evaluation functions, a value of 3 years was established for a, meaning that after 3 years the related reviews lose their importance.

In the evaluation process, we compared the scores accorded by ratings (stars only) and the final scores including the opinion. In figure 4 there are 20 opinion related function. The red line shows how the final scores behave, the blue line shows the behaviour of the stars. The stars has a significant part in the final behaviour, but there are also differences between the two establish order. For example, at opinion number 10 the accorded star is 4 (the score is 20*4=80), but in the final list it becomes of the best reviews. Opinion 14 has 5 stars, but after the analysis it takes an average final value.

VI. CONCLUSION

Whilst traditional recommender systems adopts a static approach when computing and ranking recommendations [12], the system proposed here dynamically collects and analyses opinions. In the same time, it benefits from the continuous update of the Wikipedia, and it extracts important information from a large quantity of text to feed several evaluation functions responsible to generate recommendation plans.

ACKNOWLEDGMENT

We are grateful to the anonymous reviewers for their useful comments. This work was supported by the grant ID 160/672 from the National Research Council of Romanian Ministry of Education and Research and POSDRU/89/1.5/S/62557/EXCEL.

REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a web of open data," in *The Semantic Web*, ser. LNCS, K. Aberer and all, Eds. Springer Berlin/Heidelberg, 2007, vol. 4825, ch. 52, pp. 722–735.

[2] I. Garcia, L. Sebastia, and E. Onaindia, "On the design of individual and group recommender systems for tourism," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7683 – 7692, 2011.

[3] E. D. Valle, I. Celino, and D. Dell'Aglio, "The experience of realizing a semantic web urban computing application," *T. GIS*, vol. 14, no. 2, pp. 163–181, 2010.

[4] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello, "Sindice.com: a document-oriented lookup index for open linked data." *IJMSO*, vol. 3, no. 1, pp. 37–52, 2008.

[5] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: opinion extraction and semantic classification of product reviews." in *WWW*, 2003, pp. 519–528.

[6] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Info. Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2007.

[7] B. Ohana, B. Tierney, and S. J. Delany, "Domain independent sentiment classification with many lexicons," in *AINA Workshops*. IEEE Computer Society, 2011, pp. 632–637.

[8] B. Quilitz and U. Leser, "Querying distributed RDF data sources with SPARQL," in *ESWC*, ser. Lecture Notes in Computer Science, S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, Eds., vol. 5021. Springer, 2008, pp. 524–538.

[9] H. Cunningham, "Gate, a general architecture for text engineering," *Computers and the Humanities*, vol. 36, no. 2, pp. 223–254, 2002.

[10] K. Bontcheva, H. Cunningham, D. Maynard, V. Tablan, and H. Saggion, "Developing reusable and robust language processing components for information systems using gate," in *NLIS2002, Aix-en-Provence*. Society Press, 2002, pp. 223–227.

[11] A. Esuli and F. Sebastiani, "Sentiwordnet: A publicly available lexical resource for opinion mining," in *LREC06*, 2006, pp. 417–422.

[12] I.-C. Hsu, "Sxrs: An xlink-based recommender system using semantic web technologies," *Expert Systems with Applications*, vol. 36, no. 2, Part 2, pp. 3795 – 3804, 2009.