Semantic Search Itinerary Recommender System

LIVIU ADRIAN COTFAS, ANDREEA DIOSTEANU, STEFAN DANIEL DUMITRESCU, ALEXANDRU SMEUREANU

***Abstract*—** In this paper we present a novel approach based on Natural Language Processing and hybrid multi-objective genetic algorithms for developing mobile tourism itinerary recommender systems. The proposed semantic matching technique allows users to find Points of Interest – POIs that match highly specific preferences. Moreover, it can also be used to further filter results from traditional recommender techniques, such as collaborative filtering, and only requires a minimal initial input from the user to generate relevant recommendations. A hybrid multi-objective genetic algorithm has been developed in order to allow the tourists to easily choose between several Pareto optimal itineraries computed in near real-time. Furthermore, the proposed system is easy to use, thus it can be stated that our solution is both complex and at the same time user-oriented.

***Keywords*—** semantic similarity, free text document indexing, multi-objective genetic algorithm, itinerary recommender system.

INTRODUCTION

Nowadays, designing flexible, efficient and user-friendly mobile tour guide applications is of a great interest both from a commercial and research point of view. Such systems are useful for tourists visiting a location in a limited period of time. Without any support from a system, a user manually building an itinerary, should spend a lot of time prior to the trip, searching for information regarding the tourist attractions, reading facts about them and designing possible itineraries by taking into account factors such as opening hours, visiting durations, distances and available means of public transport. Conversely, itinerary recommender systems not only assist users to schedule the initial itinerary, but can also easily revise it during the trip.

Manuscript received April 12, 2011. Parts of this article are the result of the project „Doctoral Program and PhD Students in the education research and innovation triangle”. This project is co funded by European Social Fund through The Sectorial Operational Program for Human Resources Development 2007-2013, coordinated by The Bucharest Academy of Economic Studies (Project no. 7832, “Doctoral Program and PhD Students in the education research and innovation triangle, DOC-ECI”). Parts of this article are the result of POSDRU, financing contract POSDRU/6/1.5/S/19.

L.A. Cotfas is with the Economic Informatics Department, Bucharest Academy of Economic Studies, Bucharest, Romania (phone: 40-772-402-305 e-mail: liviu.cotfas@gmail.com).

A. Dioșteanu is with the Economic Informatics Department, Bucharest Academy of Economic Studies, Bucharest, Romania (e-mail: andreea.diosteanu@gmail.com).

S.D. Dumitrescu is with the Computers and Information Technology Department, Politehnica University of Bucharest, Bucharest, Romania (e-mail: dumitrescu.stefan@gmail.com).

A. Smeureanu is with the Economic Informatics Department, Bucharest Academy of Economic Studies, Bucharest, Romania (e-mail: alexandru.smeureanu@gmail.com).

Furthermore, another important research trend in the integration of semantic technologies in many fields as they enhance the process of knowledge retrieval, processing and structuring. Paper [1] presents for example a Fuzzy-Neural Network (FNN) model to construct Q&A knowledge base automatically. In this paper we use an approach based on document indexing and semantic search in order to allow users to easily find interesting tourist attractions.

Recommender systems also represent a trend of great interest for the current research. They consist of software tools and techniques providing suggestions for items to be of use to a user [2]. In order to determine the most suitable suggestions, most implementations rely on collaborative filtering and content-based methods. However, these solutions individually fail to provide good solutions in many situations. In order to overcome these failures, [3] proposes an alternative method to content-based filtering based on neural networks. The article presents a framework used for designing recommender systems for combining neural networks and collaborative filtering. Recommender Systems also have a lot of applications in a wide range of domains from education and training to economics [4], or tourism as it is the case of the application presented in this paper.

One of the first attempts of applying recommender systems in tourism is represented by the Cyberguide [5] project, in which the authors develop a mobile application targeted to model the tour guide activity. The paper also acknowledges the importance of context, but limited only to location awareness. The authors present the overall architecture and the development procedures for multiple different hand-held platforms. Moreover, the general research issues that have emerged in context-aware applications development in a mobile environment are discussed.

The Guide system presented in [6] provides POI recommendations based on the user’s profile and context. The authors also present an evaluation of the visitor’s experience with the system. A drawback of the approach is represented by the amount of information required from the user in order to customize the tour. Guide offers users the possibility of creating a tour by selecting objectives, presenting their domains of interest and preferences. The restrictions that the system takes into account refer to the schedule of objectives, the best visiting intervals. The tour can be changed dynamically, based on the actual time the tourist already spent at some attractions or based on the weather conditions. For example, if the weather is inappropriate for visiting open-air points of interest, than the system automatically suggests indoors activities). The user profile is also continuously updated, using information from previous trips. [7] presents an artificial intelligence and metaheuristic approach for dealing with the tourist planning issue on mobile devices. The solution presented offers a decision support and a modeling mechanism starting from the orienteering problem. The problem involves a set of candidate objectives for which a score is being associated. The problem is thus reduced to maximizing the total score of the visited places, while complying with the constraints related to the total amount of time or the total distance. The score of a Point of Interest - POI is defined as the interest of a tourist in that place. Scores are calculated using the vector space model. The trip planning problem is solved using a guided local search metaheuristic. In order to compare the performance of this approach with an algorithm that appeared in the literature, both are applied to a real data set from the city of Ghent.

Our approach is also based on semantic search as [7] but uses neuronal network based document indexing mechanisms. Furthermore, we present the user a list of possible points of interest as [6], but we dynamically create the tour and we don't require data introduction effort from the user. Moreover, our approach also generates public transport routes to link objectives and also offers information regarding the pollution level from the areas in which pollution sensors are available. The trip planner presented in this paper is tested in Bucharest city.

This paper is organized as follows: in the second section we make a short overall presentation of the main characteristics of the proposed solution and we highlight the general features of the recommender systems, in the third part we present the semantic technologies and algorithm applied for semantic search and document indexing by using keywords, in the fourth section we formalize the model used for generating itineraries, while the fifth section enlarges upon the overall distributed architecture of our web service based platform, and in the sixth section we present the entire itinerary generation flow and the scenario, in addition we also introduce the mobile itinerary recommender application. The last section will conclude the article and present the future research guidelines.

ITINERARY RECOMMENDER SYSTEMS

In order to offer users the possibility of flexibly selecting the itinerary based on their preferences, we propose an algorithm for indexing documents and a flexible semantic search mechanism.

Several reviews of existing itinerary recommender solutions can be found in [8] and [9]. In most approaches, the process of generating itineraries and guiding users can be divided in three distinct steps, which also represent the main functionalities of itinerary recommender systems (Fig 1.):

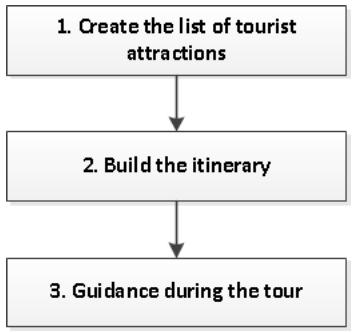
1. Create the list of tourist attractions
2. Build the itinerary
3. Guidance during the tour

Fig. 1 The main functionalities of itinerary recommender systems

- **Creating the list of POI:** In the simpler approaches like the one presented in P-Tour [10], the user is requested to select the desired POI from a list without receiving any recommendations from the system. On the opposite side, other systems [11] opt for a completely automatic approach in which the user has no control over the selected Points of Interest – POI. mTripAssistent combines the manual and automatic approaches for building the list of POI, by both allowing the user to manually select locations and using an algorithm to choose the best locations for the remaining time.
- **Building the itinerary:** Building tourist itineraries is an extension of the Traveling Salesmen Problem which is known to be already NP-hard. Existing approaches either only try to maximize the sum of scores associated to the visited POIs [11], or if multiple criteria are taken into consideration, they are combined using a simple weighted sum. [4] presents an approach based on using weighted sum that takes into consideration the preference expressed by the user and the distance between the Points of Interest. mTripAssistent uses a novel multi-objective algorithm that allows finding several Pareto-optimal solutions in near real-time.
- **Guidance during the tour:** As it was shown in the user studies performed by [12], tourists typically modify the list of POI during the itinerary. Therefore mTripAssistent allows the user to easily change the itinerary at any moment during the tour.

As it can be seen from the features presented above the novelty of our approach resides in the fact that it requires less input and effort from the user, is flexible and dynamic- by applying the multi-objective algorithm a wider set of constraints and rules can be applied. Furthermore the proposed solution also recommends public transport routes.

DOCUMENT INDEXING FOR EXTRACTING CANDIDATE POIS

In order to determine the possible candidate points of interest, we developed a semantic search algorithm that has two main steps: the semantic classification of the description files of POI and also a document indexing step. Afterwards we perform the semantic matching between the indexed words and the search phrases introduced by the user. An approach for key words extraction based on WordNet is also presented in

[13]. The article uses key sentences for automatically summarizing documents by applying statistical methods. Our paper uses a neuronal network approach for identifying keywords.

Document indexing is a useful in a lot of Natural Language Processing application such as text mining, knowledge retrieval, keywords extraction, etc [14]. Indexing defines the process of converting a document into a list of words included in it. The result of indexing is represented by a list of words called index language [15].

The selection process of document indexing has the following steps:

Step 1: Eliminate stop words. Stop words are words which have grammatical functionality, but are not related to the content of the document. Some possible stop words are: conjunction, prepositions, auxiliary verbs, articles, pronouns, etc.

Step 2: For the remaining of the document words in the list perform word stemming. This means that we extract the root of the words. In order to achieve this, we use Porter stemmer for English language.

Step 3: This step consists in performing a neuronal network based filtering so that to extract only those words that are relevant to the context.

The input nodes for the neural network are represented by the following features that can be grouped into statistical features (TF, IDF, ITF) [15], position features (T, F_{pos}) [16], grammatical syntax (subject, object, predicate (SPO)):

- term frequency(TF)- the frequency of each term in the document;
- inverted document frequency(IDF)-the number of documents that contain the word from the initial document sample;

IDF= $-log_2 \frac{TF}{Tot}$, where Tot - total occurrence [17]

- inverted total frequency(ITF)-total frequency of the word in the initial document sample;
- T-Boolean value indicating whether the word appears in the title or not;
- $F_{pos}=\frac{1}{\sqrt{i}}$ where i-the number of the sentence in which the word appears for the first time
- SPO-Boolean value indicating whether the word is a subject, predicate or an object in the sentences from the first paragraph. For determining the grammatical category of a word we used Gate and we integrated the Stanford Parser [18][19].

Therefore, the neuronal network will have six input nodes. Other characteristics of the ANN are: one hidden layer, usually it is not indicated to use more than one hidden layer; the number of hidden nodes is given by the formula:

$$N_h = \frac{(no.attributes + no.classes)}{2}$$

where:
$no.attributes$ - represents the number of features (in our case equals six)
$no.classes$ - represents the number of classes in which words can be classified (in our case 2: keyword and non-keyword).

The artificial network model can be seen in Fig. 2.

In order to train the neuronal network we used the back propagation algorithm implemented in Weka [20] on a set of manually classified instances. For the training set we have chosen documents from different sources, with a focus on newspaper articles. We created a java application that automatically extracts words in the document by following the steps presented above and computes the values for them. We then manually classify the extracted potential keywords into two groups: keywords and non-keywords. We use the resulting set of classified instances to train the ANN.

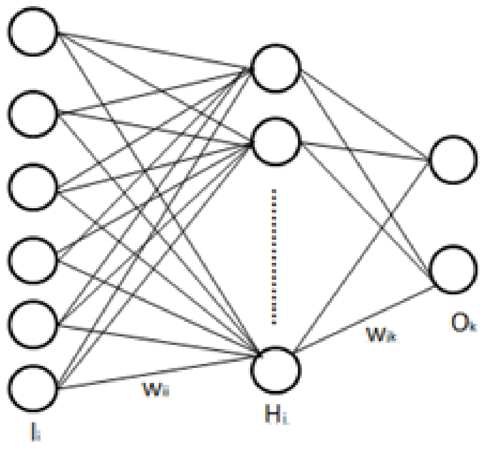

Fig. 2 Artificial neuronal network for extracting relevant words

We tested the artificial neuronal network on a set of 50 English documents. From the set of 3850 identified possible keywords, only 10% were selected in the manual classification phase. In order to choose the best configuration for the ANN, several choices for the number of neurons in the hidden layer have been tested.

Below, we present the comparison results for 3 ANN configurations:

- $N_h = \frac{(no.attributes+no.classes)}{2}$ with the results:

Confusion Matrix =$\begin{array}{c} \begin{pmatrix} 43 & 342 \\ 29 & 3436 \end{pmatrix} \end{array} \begin{array}{c} a \\ b \end{array}$

where:
first column(a) - is keyword
second(b) - non-keyword

Correctly Classified Instance	3479	90.3636 %
Incorrectly Classified Instances	371	9.6364 %

- $N_h = no.attributes + no.classes$, with the results:

Confusion Matrix =$\begin{pmatrix} 44 & 341 \\ 45 & 3420 \end{pmatrix} \begin{array}{c} a \\ b \end{array}$

Correctly Classified Instance	3464	89.974%

Incorrectly Classified Instances	386	10.026%

- $N_h = 10$, with the results:

Confusion Matrix $= \begin{pmatrix} 45 & 340 \\ 49 & 3416 \end{pmatrix} \begin{matrix} a \\ b \end{matrix}$

Correctly Classified Instances	3461	89.8961 %
Incorrectly Classified Instances	389	10.1039 %

- We also compared the results with the *Naive Bayesian* Classifier:

Confusion Matrix $= \begin{pmatrix} 35 & 350 \\ 28 & 3437 \end{pmatrix} \begin{matrix} a \\ b \end{matrix}$

Correctly Classified Instances	3472	90.1818 %
Incorrectly Classified Instances	378	9.8182 %

We conclude that the ANN configuration initially presented gives us better results.

After the document indexing phase is completed, we can perform the semantic matching.

First in order to understand the sense of the words as it is used in the documents we perform word disambiguation. For this, we query an external corpus, YAGO [21], based on Wikipedia and WordNet that contains both word definitions and the semantic relations with other words in the same synonymic class. For word sense disambiguation we identify all possible senses of the words that were previously POS – Parts of Speech tagged. For each meaning of the word we identify all possible solutions:

- Its own definition that includes example texts that WordNet or other corpus provides to the glosses.
- The dictionary definition of the synonymic groups that are connected to it through the “has a” relations. If there is more than one relation for a word sense, then the glosses for each relation are concatenated into a single gloss string.
- The dictionary definition of the synonymic group that are connected to it through the “is a” relations.
- The dictionary definition of the synonymic group that are connected to it through the “part of” relations.

After these steps are taken, the semantic similarity of the synonymic group the words belong to is measured. If a word has more than one sense, it will appear in multiple synonymic groups at various locations in the classification. WordNet defines relations between synonymic groups and relations between word senses. To measure the semantic similarity between two synonymic groups, the most appropriate relations that are to be used are the semantic hierarchic ones. Several types of semantic similarity formulas exist: Conrath, Lin, Resnik. In our approach, Lin formula was chosen as it fully takes into account the information content:

$$Sim_{Lin}(X_1, X_2) = \frac{2 * S(X_1, X_2)}{(IC(X_1) + IC(X_2))} \qquad (1)$$

where:
$Sim_{Lin}(X_1, X_2)$ – the level of similarity between concepts 1 and 2 computed by Lin formula
$S(X_1, X_2)$- the degree of shared information for X_1 and X_2 concepts
X_1, X_2 - concepts 1 and 2
IC(X) - information content (IC) of X concept.

The semantic matching evaluation will be performed against the user key search phrases and the keywords extracted from the documents.

For instance, if a tourist introduces “modern art” as search phrase, and in a document about a military museum it is specified that there is a painting created by a modern art painter “Paul Gauguin”, then the POI will be suggested to the user.

Multi-objective Evolutionary Algorithm for Generating Itineraries

Besides the semantic search approach described in the previous section, users can also choose POI from a suggestion list built using collaborative filtering. All selected POIs are displayed in a common list, allowing the user to change their importance, by changing the order.

We frame the problem of building tourist itineraries as an extension of the classic Orienteering Problem - OP, also known as the Time Dependent Orienteering Problem with Time Windows - TDOPTW. As the Orienteering Problem is known to be NP-hard, finding an exact solution for the TDOPTW in near real-time is considered to require a high computational power [22]. The proposed system uses a hybrid multi-objective genetic algorithm in order to generate itineraries in a short amount of time. Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover. In order to decrease the number of required generations, we use the following heuristic: each time we add a new POI to an itinerary we choose the one with the highest ratio between the associated score, S_i and the time required to visit the POI.

The itinerary building algorithm has as inputs:

- T_S - the start time;
- T_F - the finish time;
- L_S - the start location;
- L_F - the finish location;
- $POI_{Selected}$ - the set of manually selected tourist attractions that will be included in all generated itineraries;
- $POI_{Favorites}$ - the set of attractions that were marked as favorites by the user;
- POI - the global set of tourist attractions;
- N_{Sol}^* - the number of requested solutions.

The output of the algorithm consists in a number of $N_{Sol} \leq N_{Sol}^*$ Pareto-optimal itineraries including both the tourist attractions and the traveling directions between them.

The considered optimization criteria are:

- **C1** - represents the useful visiting time calculated as shown in (2);

- **C2** - represents the average score of the visited locations calculated as shown in (3);
- **C3** - represents the diversity of the itinerary calculated as shown in (4).

Other criteria can also easily be taken into consideration. The criteria **C3** can also take into consideration the structure of the itinerary.

$$VisitT_j = \sum_{k=1}^{n_j} VisitD_k \tag{2}$$

$$MR_j = (\sum_{k=1}^{n_j} S_k)/n_j \tag{3}$$

$$DV_j = n_j \tag{4}$$

The start and finish locations, L_S and L_F, represent the GPS coordinates associated to the locations selected by the user either from the map or from the list of tourist attractions.

Each Point of Interest $i = 1 \dots N_{POI}$ is characterized by:

- O_i - opening time;
- C_i - closing time;
- S_i - associated score;
- Fee_i - entrance fee;
- $VisitD_i$ - medium visit duration;
- Loc_i - GPS coordinates.

For each user u and tourist attraction i, $VisitD_i^u$ represents the real visiting duration and $VisitD_i^{*u}$ the predicted one. For a tourist attraction that has not yet been visited, the estimated visiting duration is computed as shown in (5), where $N_{POI}^{u,k}$ represents the number of tourist attractions in the category k to which the tourist attraction i belongs that have already been visited by the user u

$$VisitD_i^{*u} = VisitD_i * \sqrt[N_{POI}^{u,k}]{\prod_{j=1}^{N_{POI}^{u,k}} \frac{VisitD_j^u}{VisitD_j}} \tag{5}$$

The algorithm is run until a maximum number of generations, G_{MG} is reached or until the best found solution doesn't change for G_{CG} consecutive generations. Depending on the purpose G_{MG} can be adjusted either for accuracy or speed. In order to increase the performance with compute in advance the minimum - $TravelD_{ij}^{min}$ and maximum time - $TravelD_{ij}^{max}$ required to travel between locations i and j. The values can be calculated for different hours or week days.

We have chosen an Elitist approach [13] in which we use two populations, the normal one P_g and the population of non-dominated solutions in generation $g < G_{MG}$, E_g.

Initialization: For $g = 0$, we create the initial population P_0, with a number of N_P itineraries called chromosomes. The population of non-dominated solutions is also initialized $E_0 = \emptyset$. In order to limit the number of required algorithm iterations, all itineraries in the initial population are generated valid. First, the user selected tourist attractions, $POI_{Selected}$ are added to the new itinerary in a random manner.

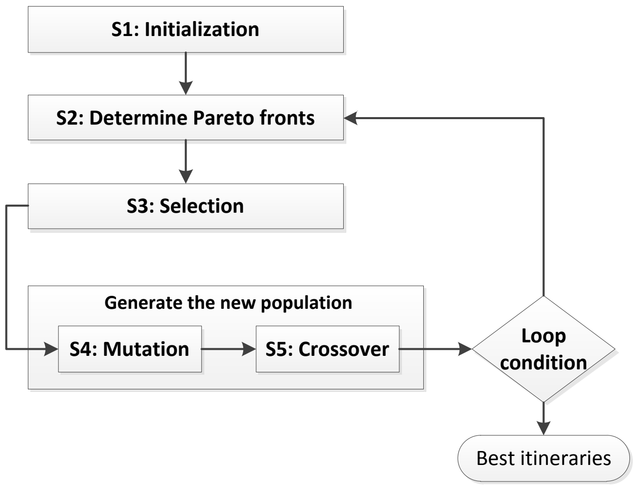

Fig. 3 Itinerary building algorithm

Afterwards, the remaining time is completed with tourist attractions selected from the set of favorites, $POI_{Favorites}$ and from the normal set, POI. A tourist attraction i can only be inserted if the arrival time, A_i, satisfies the conditions in (6) and (7).

$$A_i < C_i - VisitD_i \tag{6}$$

$$A_i + VisitD_i + TravelD_{iL_F} < T_F \tag{7}$$

The time needed to visit a tourist attraction can be computed as shown in (8) by summing the travel time from the previous location, the waiting time and the visit duration.

$$VisitTD_i = TravelD_{i-1,i} + WaitD_i + VisitD_i \tag{8}$$

The waiting duration, $WaitD_i$ can be computed as shown in (9), where A_i represents the arrival time.

$$WaitD_i = \max(0, O_i - A_i) \tag{9}$$

Evaluation: All N_P itineraries are evaluated based on the three selected criteria: **C1**, **C2**, **C3**. In order to determine the Pareto-optimal fronts we use an approach similar to the one in the NSGA-II algorithm [23].

Selection: As we have chosen an elitist approach in which we copy all non-dominated solutions from P_g in E_g. We remove all other solutions from P_g and we automatically include all solutions from $E_g$in P_{g+1}. Using mutation and crossover we generate $N_P - |E_g|$solutions that are added to P_g in order to have $|P_{g+1}| = N_P$. If $|E_g| > N_P/2$ we use a clustering approach to select $N_P/2$ solutions as different as possible.

Mutation: Adds random variation to the evolving population. Two links are randomly chosen from the itinerary and all the locations between them that do not belong to the manually selected set of tourist attractions, $POI_{Selected}$, are removed. The itinerary is then completed by randomly inserting new POIs.

Crossover: Combines the features of two parent chromosomes to form new children by swapping corresponding itinerary segments of the parents. At first, one link offset is selected randomly. The resulting segments are swapped and a repair step is performed in order to preserve the manually selected tourist attractions from $POI_{Selected}$, in both itineraries. If necessary the POIs with the lowest S_i are removed in order to keep the itineraries valid.

The walking distances can be determined using either Dijkstra or A* for large graphs. The functionality is also provided by several libraries such as pgRouting and API's like Google Maps Directions API.

SYSTEM ARCHITECTURE

The proposed solution presented in Fig. 4 relies on a multi-tier paradigm for both the server and the client implementations. Even though a client only architecture would offer several benefits, such as the possibility to work entirely offline, it is not a feasible option for our approach due to the big amounts of data used both by the semantic search and by the collaborative filtering algorithms. Therefore, we have chosen a mixed approach in which the resource intensive computations are performed on the server, while the simple ones are performed directly on the client. Client implementations rely on a local data persistence management solution in order to avoid unnecessary server requests. The server code is written in Java and uses Apache Tomcat as a web application server.

The **Interface Layer** has the role of facilitating the communication with the client implementations using HTTP requests, Web Service Calls and Socket Connections. Thanks to the multiple communication methods supported by this layer, client side applications can be implemented using a wide array of technologies.

The **Business Layer** includes the main functionalities of the system, implemented as semantically annotated web services [24]. The itinerary web service implements the multi-objective genetic algorithm described in the previous section, the recommender web service implements a context-aware collaborative filtering algorithm and the route finding web service implements the public transport route finding algorithm presented by the authors in [25].

The **Persistence Layer** stores information regarding the available tourist attractions, the public transport network, as well as user information. The POI Data database stores for all the attractions: the name, a short and a long description, photos, the location and the opening hours. The User Data database stores the user information as well as the tourist attractions ratings used by the recommender algorithm. A separate database is used to store the data required by the public transport route finding algorithm.

Using **External Data Providers**, such as external web services, the system collects weather forecasts, retrieves the changes in the list of available tourist attractions and updates the data for the public transport network.

An **Intelligent Agents Module** has the purpose of monitoring the users' behavior as well as of notifying them of any context changes, like weather changes or changes in the opening hours of the attractions included in the itinerary.

MOBILE USER INTERFACE

The reference client implementation for mobile devices shown in Fig. 5 relies on the latest standard web technologies such as WebSQL and WebStorage to offer portability across different platforms as well as a rich user experience. W3C GeoLocation API was used to determine and monitor the position of the user.

The application can either be used directly from the browser, or can be installed using a thin native wrapper that provides the required translation from JavaScript method calls to native method calls. The application works on all devices that comply with the HTML5 standard specifications

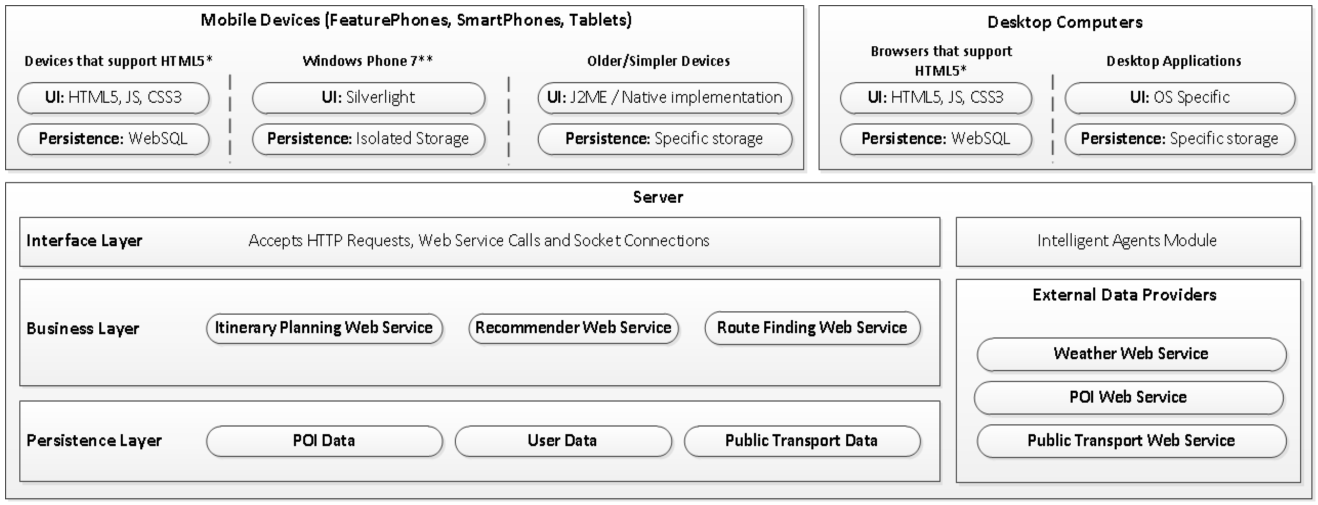
Mobile Devices (FeaturePhones, SmartPhones, Tablets)
Devices that support HTML5*
UI: HTML5, JS, CSS3
Persistence: WebSQL
Windows Phone 7**
UI: Silverlight
Persistence: Isolated Storage
Older/Simpler Devices
UI: J2ME / Native implementation
Persistence: Specific storage
Desktop Computers
Browsers that support HTML5*
UI: HTML5, JS, CSS3
Persistence: WebSQL
Desktop Applications
UI: OS Specific
Persistence: Specific storage
Server
Interface Layer
Accepts HTTP Requests, Web Service Calls and Socket Connections
Intelligent Agents Module
Business Layer
Itinerary Planning Web Service
Recommender Web Service
Route Finding Web Service
External Data Providers
Weather Web Service
POI Web Service
Public Transport Web Service
Persistence Layer
POI Data
User Data
Public Transport Data

Fig. 4 Distributed architecture

including both smartphones and feature phones.

The steps that a user should perform in order to build an itinerary are presented below.

Step 1 - Create the list of tourist attractions

In order to build an itinerary, the user can both semantically search for POI as described in this paper or can select from a list of suggested tourist attractions. The list of suggestions is generated using a collaborative filtering algorithm that also takes into consideration the current context including weather, time, week day and season. The tourist attractions can either be added manually to the itinerary, corresponding to the $POI_{Selected}$ set, or can be added to the favorites list, corresponding to the $POI_{Favorites}$ set that are used as input parameters for the multi-objective itinerary building algorithm. As shown in Fig. 6, users can change the start time - T_S, the finish time - T_F, the start and finish locations - L_S, L_F.

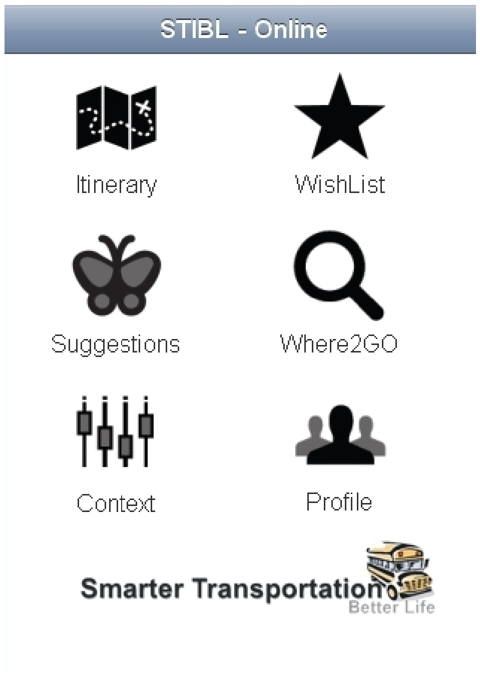

Fig. 5 Mobile HTML5 interface

Step 2 – Choosing the itinerary

The itinerary configuration screen also allows choosing if one or several itineraries will be generated. If the user chooses to generate multiple itineraries, a number of N_{Sol} possible itineraries are generated on the server using the multi-objective itinerary algorithm. The itineraries are displayed on the mobile phone including the score for the 3 optimization criteria **C1**, **C2**, **C3**. As shown in the fourth section, the system used previously recorded data to predict the visit duration for the current user.

Step 3 – Guidance during the tour

The selected itinerary can be viewed either as a list or using the map. In case the tourist exceeds the average duration for some of the points of interest, the itinerary is dynamically updated. Moreover, as shown in the fourth section, a crowd sourcing approach was implemented in order to determine the real visiting duration for the users of the application. For each user, the system constantly updates the difference for each category of POI between the current user's visiting duration and the average duration. This information is used to better predict the visiting durations in future itineraries.

Public transport route finding is also included in the application in order to allow the user to easily travel between the locations selected in the itinerary.

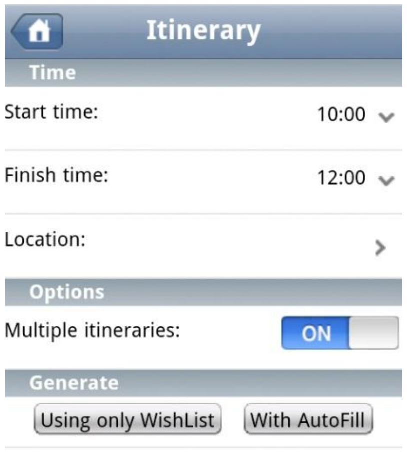
Itinerary
Time
Start time: 10:00
Finish time: 12:00
Location:
Options
Multiple itineraries: ON
Generate

Fig. 6 Itinerary settings screen

A comparison between native and mobile web applications built using the latest standard web technologies is provided in Table I. As shown also in the table, developing native versions of an application for several mobile platforms requires more time and increases the costs due to the different development approaches characteristic for each platform

Table I Comparison between Native and Mobile Web Applications

	Native Applications	**Applications based on web technologies**
General		
Portable across many platforms	no	yes
Can run in the browser	no	yes
Can be installed	yes	yes
Development		
Technologies	different for each platform	the same for all platforms (HTML5, CSS3, JavaScript)
Local persistence	yes	yes
Possibility to work offline	yes	yes
Advanced hardware features	yes	yes
Resource intensive computations	better performance	worse performance

Others		
Accepted in Application Stores	yes	yes*
Easy to update	no	yes

Mobile web applications can be installed on mobile devices using a thin native wrapper. When running directly from the browser the local storage is limited to approximately 10Mb. This limitation can easily be avoided by using a native wrapper around the web application.

CONCLUSIONS

In this paper we presented an itinerary recommender system that uses both semantic search and collaborative filtering to allow users to find interesting Points of Interest. A multi-objective itinerary hybrid genetic algorithm that allows finding routes in near real-time was also presented. The novelty of our approach consists of the fact that we use neuronal network document indexing based on key word extraction and also a modified version of the multi objective genetic algorithm. This enables users to easily generate itineraries in a very flexible and dynamic manner with less effort and input. The application takes into considerations user's preferences and profile. A crowd sourcing approach was implemented in order to compare the differences between the estimated and real times need to travel between the tourist attractions. The semantic search approach allows users to find POIs related to highly specific information.

ACKNOWLEDGMENT

L.A. Cotfas is grateful to Professor Ricci for the very useful suggestions and to the Free University of Bozen-Bolzano for kind hospitality during the time this work was carried out.

REFERENCES

[1] Y.H Kuo, C.S Lee, S-M Guo, and F.T Tu, "Apply FNN Model to Construct Ontology-based Q&A System," *WSEAS Transactions on Communications*, vol. 3, Issue 1, pp. 328-335, Jan. 2004
[2] F. Ricci, L. Rokach, L., and B. Shapira, B. *Recommender Systems Handbook*, Springer-Verlag, 2011, ch1.
[3] C. Vassiliou, D. Stamoulis, D. Martakos "A Recommender System Framework combining Neural Networks & Collaborative Filtering", *in Proc. of the 5th WSEAS Int. Conf. on Instrumentation, Measurement, Circuits and Systems*, Hangzhou, China, 2006, pp.285-290
[4] H.F Wang, C.T. Wu, "A Strategy-Oriented Operation Module for Recommender Systems in E Commerce",in *Proc of the 9th WSEAS International Conference on Applied Informatics and Communications (AIC '09)*,Moscow, Russia, 2009, pp. 78-83
[5] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, M. Pikerton, "*Cyberguide: A mobile context-aware tour guide,*" Wireless Networks, vol. 3, pp. 421-433, 1997.
[6] K. Cheverst, N. Davies, and K. Mitchell, "*The role of adaptive hypermedia in a context-aware tourist guide*," Communications of the ACM: Special Issue on Adaptive Web-Based Systems and Adaptive Hypermedia, No. 45, 2002, pp. 47-51.
[7] W. Souffriau, P. Vansteenwegen, J. Vertommen, G.V. Berghe, D.V. Oudheusden, "*A personalized tourist trip design algorithm for mobile tourist guides,*" Applied Artificial Intelligence No. 22, pp. 964-985, 2008.
[8] A. Garcia, M.T. Linaza, O. Arbelaitz, and P. Vansteenwegen, *"Intelligent routing system for a personalised electronic tourist guide,"* Information and Communication Technologies in Tourism, 2009, pp. 185–197.
[9] R. Kramer, M. Modsching, K. Hagen, and U. Gretzel, "*Behavioural impacts of mobile tour guides,*" Information and Communication Technologies in Tourism, 2007, pp. 109–118.
[10] A. Maruyama, N. Shibata, Y. Murata, K. Yasumoto, and M. Ito, "*P-tour: A personal navigation system for tourism,*" Proc. of 11th World Congress on ITS, 2004, pp. 18–21.
[11] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden, "*Iterated local search for the team orienteering problem with time windows*," Computers & Operations Research, vol. 36, 2009, pp. 3281-3290.
[12] R. Kramer, M. Modsching, K. Hagen, and U. Gretzel, "*Behavioural impacts of mobile tour guides,*" Information and Communication Technologies in Tourism, 2007, p. 109–118.
[13] C. Dang, X. Luo,"WordNet-based Document Summarization", in *Proc. 7th WSEAS International Conference on Applied Computer & Applied Computational Science (ACACOS '08)*, Hangzhou, China, 2008, pp. 383-387
[14] I. Smeureanu, A. Diosteanu, C. Delcea, and L.A. Cotfas, "Busines Ontology for Evaluating Corporate Social Responsibility," *Amfiteatru Economic*, vol. 29, pp. 28-42, 2011.
[15] Taeho, J., "Neural Based Approach to Keyword Extraction from Documents," *Computational Science and Its Applications*, Vol. 2667, Springer Berlin, pp 263-270, 2003.
[16] K. Sarkar., M. Nasipuri, S. Ghose, "A New Approach to Keyphrase Extraction Using Neural Networks," *International Journal of Computer Science Issues*, Vol. 7, Issue 2, No 3, pp 16-25, 2010.
[17] O. Medelyan, Human-competitive automatic topic indexing, PhD Thesis, University of Waikato, New Zeeland, 2009.
[18] The Stanford Natural Language Processing Group, "The Stanford Parser," 2011. [Online]. Available: http://nlp.stanford.edu/software/lex-parser.shtml . [Accessed: January. 15, 2010].
[19] GATE, "General Architecture for Text Engineering," 2011. [Online]. Available: http://gate.ac.uk/.[Accessed: January. 15, 2010].
[20] University of Waikato, "Weka," 2011. [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/. [Accessed: January. 15, 2010].
[21] F.M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," Proceedings *of the 16th international conference on World Wide Web*, New York, 2007, pp. 697-706.
[22] M. Kenteris, D. Gavalas, G. Pantziou, and C. Konstantopoulos, "Near-optimal personalized daily itineraries for a mobile tourist guide," IEEE Symposium on Computers and Communications, pp. 862–864), 2010.
[23] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182-197, 2002.
[24] L.A. Cotfas, A. Dioșteanu, and A. Smeureanu, I. "Fractal web service composition framework". 8th International Conference on Communications, 2010, pp. 405-408).
[25] L.A. Cotfas, M.C. Croicu, and D. Cotfas "A Collaborative GIS Solution for Public Transport". Informatica Economică, Vol. 13(2), pp. 50-58, 2009