Expert Systems With Applications 87 (2017) 199–208

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Designing the Content Analyzer of a Travel Recommender System

Carla Binucci, Felice De Luca, Emilio Di Giacomo*, Giuseppe Liotta, Fabrizio Montecchiani

Dipartimento di Ingegneria, Università degli Studi di Perugia, Italy

ARTICLE INFO

Article history:
Received 27 March 2017
Revised 16 June 2017
Accepted 17 June 2017
Available online 19 June 2017

Keywords:
Content Analysis
Travel Recommender Systems
Content-based Approach

ABSTRACT

Content-based travel recommender systems suggest touristic attractions based on a best match between users' preferences and a given set of points of interests, called POIs for short. When designing such systems, a critical aspect is to equip them with a rich enough knowledge base that, for each POI, indicates how much the POI is relevant for a set of possible topics of interests, also called TOIs for short. This paper focuses on the problem of designing the Content Analyzer of a content-based travel recommender system. The Content Analyzer is a module that receives as input a set of POIs and a set of TOIs and it computes the relevance of each POI with respect to each TOI. The proposed approach is unsupervised, fully automatic, and it relies on publicly available sources of information. We describe an implementation of the technique in a system called Cicero and present an experimental evaluation of its effectiveness against a ground truth generated by experts.

1. Introduction

Planning a trip by taking into account personal preferences often becomes a time-consuming and difficult task, given the overwhelming amount of information available on a large variety of digital sources (such as institutional web sites, travel blogs, travel guides, etc.). A *travel recommender system* guides a tourist through this large space of possible options by matching touristic and leisure attractions (technically called *Points of Interest*, or *POIs* for short) with traveler's interests. As pointed out in an early work of Staab et al. (2002) and in a more recent survey of Borrás, Moreno, and Valls (2014), the most common travel recommender systems use a content-based approach, in which the user expresses her needs by associating values to a set of attributes and the system matches these needs with a given set of available POIs. As depicted in Fig. 1, the architecture of a content-based travel recommender system consists of three main modules: a *Content Analyzer*, a *Profiler*, and a *Matching Module*. The Content Analyzer gathers information from various sources and it computes a suitable description of the POIs that is stored in a *Knowledge Base*. The Profiler constructs users' profiles by collecting and analyzing data representative of their interests. The Matching Module outputs a *travel recommendation*, i.e., a ranked list of those POIs that are most suitable for the users' profiles. A limited list of content-based travel recommender systems includes, for example, (Brilhante, Macedo, Nardini, Perego, & Renso, 2015; Cenamor, de la Rosa, Núñez, & Borrajo, 2017; Győrödi, Győrödi, & Dersidan, 2013; Huang & Bian, 2009; Lee, Chang, & Wang, 2009; Lim, Chan, Leckie, & Karunasekera, 2015; Martínez Santiago, Ariza López, Montejo-Ráez, & Ureña López, 2012; Montejo-Ráez, Perea-Ortega, García-Cumbreras, & Martínez-Santiago, 2011; Ruotsalo et al., 2013; Vansteenwegen, Souffriau, Berghe, & Oudheusden, 2011).

We focus on the widely adopted keyword-based vector space model to represent users' profiles and POIs' descriptions (de Gemmis, Lops, Musto, Narducci, & Semeraro, 2015). According to this model, the POIs' description and the users' profiles are represented as two vectors in an n-dimensional space, where each dimension corresponds to a *Topic of Interest*, or *TOI* for short. A TOI is a keyword, that describes a subject on which a traveler can find interesting attractions in the region; examples of TOIs could be *History, Religion, Art*, and so on. The travel recommendation is then computed by means of a suitable matching technique between these two vectors. For example, suppose that the profile of a tourist in Rome defined on the TOIs *History, Religion*, and *Art* has higher scores on *History*, then the Matching Module will return a list of POIs where the Colosseum and the Roman Forum are ranked higher than the National Gallery of Modern and Contemporary Art.

A critical aspect in the design of an effective content-based travel recommender system is to create a rich enough Knowledge Base with a suitable description of a large set of POIs. The description of a POI in the Knowledge Base should encode how much this POI is relevant for the set of TOIs presented to the user.

* Corresponding author.
E-mail addresses: carla.binucci@unipg.it (C. Binucci), felice.deluca@studenti.unipg.it (F. De Luca), emilio.digiacomo@unipg.it (E. Di Giacomo), giuseppe.liotta@unipg.it (G. Liotta), fabrizio.montecchiani@unipg.it (F. Montecchiani).

http://dx.doi.org/10.1016/j.eswa.2017.06.028
0957-4174/

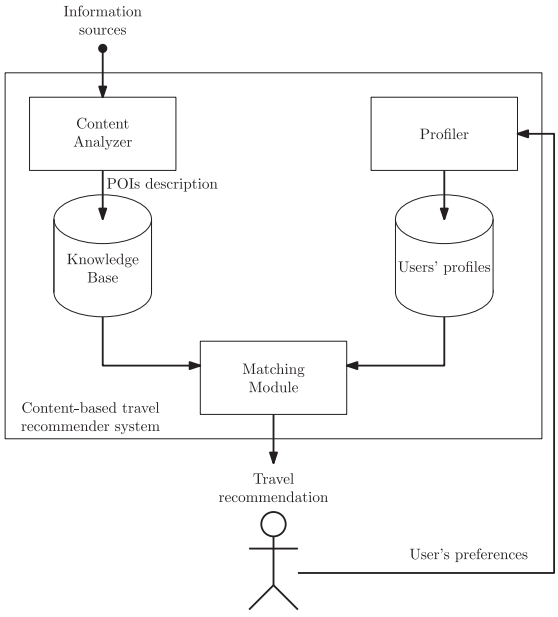


Fig. 1. High level architecture of a content-based travel recommender system.

Obtaining such a description for each POI often relies on a manual classification which is an expensive task (see, e.g., Batet, Moreno, Sánchez, Isern, and Valls (2012); Gavalas et al. (2015); Gavalas and Kenteris (2011); Lim et al. (2015); Lucas et al. (2013); Martínez Santiago et al. (2012); Meehan, Lunney, Curran, and McCaughey (2013); Savir, Brafman, and Shani (2013); Umanets, Ferreira, and Leite (2014); Vansteenwegen et al. (2011)). Also, if the list of TOIs changes, a different description for the POIs may be needed. Thus the maintenance of these systems is an expensive task and it may become particularly critical in those contexts where territorial policy makers want to offer their tourists TOIs that vary based on available seasonal attractions and events. For example, suppose that in a certain period there is a special event in a region, such as a music festival; a regional policy maker can decide to introduce the new TOI *Music* in the system so that tourists who visit the region because of the festival will find POIs that are particularly interesting for them.

This paper focuses on the design of the Content Analyzer for a travel recommender system. Motivated in part by a research grant of the Umbria Region of Italy[1] (see also Binucci, Didimo, Liotta, Montecchiani, & Sartore (2013)), we study the following problem: Given a geographic region $\mathcal{R}$ and a set of TOIs (defined by a policy maker who wants to promote the touristic attractions of a territory), compute the relevance of the POIs in $\mathcal{R}$ with respect to the given TOIs. We propose a graph-based algorithmic method specifically tailored for the context of a travel recommender system and we embed such method in a content analyzer called Cicero[2]. The main features of Cicero are as follows.

- Cicero receives as input a region $\mathcal{R}$ and a set of TOIs $T = \{t_1, t_2, \ldots, t_n\}$; it retrieves a set of POIs $P = \{p_1, p_2, \ldots, p_m\}$ in region $\mathcal{R}$ and it computes as output a *description* for each POI. The description of a POI $p_i \in P$ is an array whose j-th element is a numeric value in the range [0, 1] that represents the relevance of p_i with respect to TOI t_j.
- The computation of the descriptions of the POIs is unsupervised and automatic. Since it is unsupervised, it does not require a training set of reliable data, which can be difficult to obtain. The fact that it is automatic allows us to handle large geographic regions with thousands of POIs, for which a manual assignment would be impractical or too expensive.
- Cicero relies on publicly accessible Web resources and does not use an ontology. Ontologies are often used to represent (and reason about) the tourism domain knowledge (see, e.g., Castillo et al. (2008); Lee et al. (2009); Moreno, Valls, Isern, Marin, and Borrás (2013); Ruotsalo et al. (2013); Wang, Zeng, and Tang (2011)). However, ontologies for travel recommender systems are typically designed ad-hoc and built manually, which can be a time-consuming and labor-intensive task. Instead, the proposed technique computes a description of each POI by executing a shortest path algorithm on a suitable concept network that is extracted by automatically crawling Wikipedia[3] and OpenStreetMap[4].

The rest of this paper is organized as follows. A critical discussion about the main differences and similarities between the approach of Cicero and existing literature in the areas of travel recommender systems and of semantic technologies can be found in Section 2. In Section 3 we present a reference architecture upon which we designed and implemented our system. In Section 4 we discuss the main principles and methods behind Cicero. We describe how public encyclopedic sources can be used to construct a network of concepts that is then exploited to compute the descriptions of the POIs with respect to the given TOIs. To have an indication about the effectiveness of the proposed techniques, in Section 5 we:

1. Use Cicero to create a concept network for two Italian touristic cities, namely Rome and Perugia; the network is computed with respect to three popular TOIs that are particularly relevant for the two chosen cities, namely *Art, History*, and *Religion*.
2. Evaluate some structural properties of the created network that affect the effectiveness and the efficiency of our approach.
3. Use the network to create a Knowledge Base and compare the results against a ground truth defined by professional touristic guides and experts of the two cities. While for their sizes and number of available POIs the two chosen cities have different characteristics, the experimental analysis suggests that the algorithmic technique behind Cicero can be effective in both scenarios.
4. Realized a proof-of-concept implementation of a content-based recommender system that uses the computed Knowledge Base.

Finally, conclusions and future research directions are discussed in Section 6.

2. Related Work

The research in this paper naturally relates with both the literature about (travel) recommender systems and the literature about semantic relatedness analysis. In this section we briefly recall some of the most relevant references of these research areas and spend a few more words about those contributions that adopt an approach similar to ours.

Several travel recommender systems have been described in the literature; as already pointed out in Section 1, the majority of them (see, e.g., Brilhante et al. (2015); Cenamor et al. (2017); Győrödi et al. (2013); Huang and Bian (2009); Lee et al. (2009); Lim et al. (2015); Martínez Santiago et al. (2012); Montejo-Ráez et al. (2011); Ruotsalo et al. (2013); Vansteenwegen et al. (2011))

[1] POR-FESR Project "TRART: Telematic Representation-Augmented Reality-Territories".

[2] http://www.felicedeluca.com/cicero/.

[3] https://www.wikipedia.org/.

[4] https://www.openstreetmap.org/.

uses the content-based approach where suggestions are based on a matching between the user's preferences and the POIs' features. We will refer to this approach throughout the paper describing a novel technique to devise a Content Analyzer that is a crucial component of this approach. Beside the content-based approach, some travel recommender systems adopt a *collaborative filtering approach*, where travel recommendations are made based on groups of users with similar preferences; when the system identifies those people sharing similar interests with the current user, it recommends to the users the same POIs that those people liked. Travel recommender systems that adopt a collaborative filtering approach typically suffer of the so called "cold start problem", that is they become effective only after a sufficient amount of data about past travel recommendations for a large enough set of people is gathered. A limited list of papers that describe travel recommender systems based on collaborative filtering include the work of Savir et al. (2013); Umanets et al. (2014); Yang and Hwang (2013). A variant of the collaborative filtering approach that tries to partially overcome the cold start problem is the *demographic based approach* (see, e.g., Wang et al. (2011)), where a user is assigned to a stereotypical class based on its demographic data (age, nationality, level of studies, etc.) and travel recommendations are made based on this stereotypical classification. The previous approaches are often combined in what is generically called a *hybrid approach* (see, e.g., Batet et al. (2012); Braunhofer, Elahi, Ricci, and Schievenin (2013); Castillo et al. (2008); Gavalas et al. (2015); Gavalas and Kenteris (2011); Lucas et al. (2013); Meehan et al. (2013); Moreno et al. (2013); Niaraki and Kim (2009)).

Concerning the construction of the knowledge base in a travel recommender system, various solutions exist. Often the POIs are manually classified (see, e.g., Batet et al. (2012); Gavalas et al. (2015); Gavalas and Kenteris (2011); Lim et al. (2015); Lucas et al. (2013); Martínez Santiago et al. (2012); Meehan et al. (2013); Savir et al. (2013); Umanets et al. (2014); Vansteenwegen et al. (2011)), possibly with a following refinement based on the use of ontologies (see, e.g., Castillo et al. (2008); Lee et al. (2009); Moreno et al. (2013); Ruotsalo et al. (2013); Wang et al. (2011)) or feedback from the users (Győrödi et al., 2013). Clearly, manual classification, especially if done by experts, can be very precise but it is a hard and tedious task. As an alternative to manual classification, some systems retrieve information about the POIs from the Web or other on-line sources. For example, Huang and Bian (2009) integrate heterogeneous on-line information by means of an ad-hoc defined ontology. Ontologies are a valuable tool to represent the domain knowledge and to reason about it. However, designing and building an ontology is a very time-consuming and labor-intensive task. The Otiŭm system (Montejo-Ráez et al., 2011) retrieves information about tourist attractions and events to be suggested from the Web. The proposed approach aims at finding attractions that are similar to each other, but it does not produce a description of each attraction in terms of a set of TOIs that can be matched with an analogous description of the user's profile.

An approach similar to the one described in this paper can be found in a paper by Brilhante et al. (2015). By using data retrieved from Wikipedia, Brilhante et al. compute for each POI p_i a value that represents the relevance of p_i with respect to a set of keywords. The main difference with our paper is that the keywords used by Brilhante et al. (2015) on which p_i is ranked are tags automatically extracted from the Wikipedia pages of p_i (they are the Wikipedia categories at the bottom of the page); in our application scenario, we have a set of TOIs that are part of the input and these TOIs may not even appear in the Wikipedia page of p_i. This difference has a fundamental impact not only in the application scenario for which the content analyzer is designed, but also on the algorithmic technique that is used to compute a description of a POI with respect to the set of TOIs.

Table 1
Notation used in throughout the paper.

Symbol	Description
$\mathcal{R}$	A geographic region
$T = \{t_1, \ldots, t_n\}$	A set of TOIs
$P = \{p_1, \ldots, p_m\}$	A set of POIs
R_i	Description of POI p_i
$R_i[j]$	Relevance of POI p_i with respect to TOI t_j
$\Pi_i[j]$	Pertinence of POI p_i with respect to TOI t_j
l_c	List of tag keys from OpenStreetMap
l_v	List of tag values from OpenStreetMap
$\langle c_i, v_i \rangle$	A key-value pair with $c_i \in l_c$ and $v_i \in l_v$
$\tau(p_i)$	Set of tag values associated with POI p_i
$\mathcal{N}$	Concept network
$\pi_{i,j}$	Pertinence score of tag value v_i with respect to TOI t_j
pop_i	Popularity score of POI p_i

Finally, the technique in this paper can be related with the rich literature about computing semantic similarities between concepts by means of their graph theoretic distance in networks of concepts (see, e.g. Leacock and Chodorow (1998); Rada, Mili, Bicknell, and Blettner (1989); Wu and Palmer (1994)). Among the early papers about this approach, we recall the work of Jarmasz and Szpakowicz (2003), who use a distance function between terms on a thesaurus constructed from Roget's Thesaurus. The main difference between the approach in this paper and the one of Jarmasz and Szpakowicz are: (i) The work of Jarmasz and Szpakowicz is devoted to computing the semantic relatedness between two documents while we are interested in computing a vector-based representation of a set of POIs; (ii) Roget's Thesaurus provides structured access to semantic properties of terms, while the information of Wikipedia concepts is unstructured; (iii) Roget's Theseaurus (as well as other corpora like WordNet) are manually maintained up-to-date by experts. Roget's Thesaurus dates back to the nineteenth century and it is updated every few years; on the other hand, Wikipedia represents a rapidly evolving body of knowledge that can be adapted much faster to possible changes in the touristic world. It is worth remarking that, while computing the semantic relatedness based on encyclopedic knowledge, including Wikipedia, has received increasing attention in recent years (see, e.g., Banerjee, Ramanathan, and Gupta (2007); Egozi, Markovitch, and Gabrilovich (2011); Gabrilovich and Markovitch (2009); Ponzetto and Strube (2007); Taieb, Aouicha, and Hamadou (2013)), to the best of our knowledge the present paper is the first application of a path based measure of semantic relatedness in the contest of travel recommender systems.

3. A Reference Architecture

In this section we discuss the main ideas behind the design of our Content Analyzer. In order to help the reader, we report in Table 1 the main symbols and notations used, which will be defined and explained in the following. The reference architecture upon which we designed and implemented Cicero is given in Fig. 2. The architecture is composed of two main modules described in the following.

Graphic User Interface (GUI). The GUI enables the end-user to select a region and define a set of TOIs. An example of the GUI implemented in the Cicero system is given in Fig. 3(a); the figure shows the selection of a geographic region around the city of Rome. The selected region and the chosen set of TOIs are then given as input to the Algorithmic Engine, which is the second module of the architecture. As explained in the following, this second module contains the logic to extract the POIs from the region and to compute a description of each POI with respect to each TOI. This information is returned to the GUI, which shows the set of retrieved POIs on a map. For example, Fig. 3(b) shows the set of POIs retrieved for the region selected in Fig. 3(a). Note that the

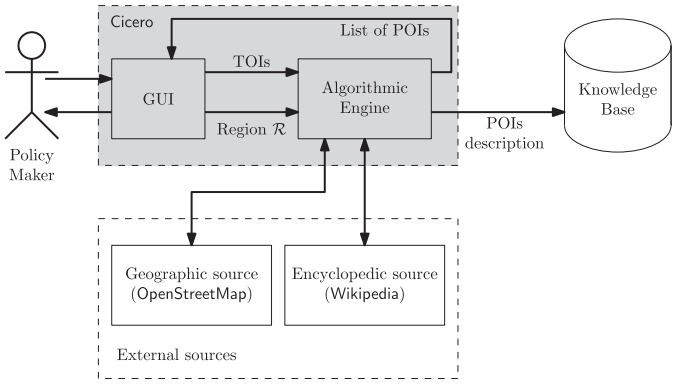


Fig. 2. A reference architecture for the content analyzer of a content-based travel recommender system.

POIs are automatically clustered based on the zoom level of the map (adjustable by the end-user), and the numbers inside the circles indicate the number of clustered POIs.

Algorithmic Engine. The Algorithmic Engine extracts the POIs from the selected region and creates a Knowledge Base that stores the retrieved POIs. Each POI is stored together with a *description* of its relevance with respect to the chosen TOIs. The description of p_i is a vector R_i, whose component $R_i[j]$ indicates the *relevance* of POI p_i with respect to TOI t_j, also called the *relevance score* of p_i with respect to t_j. At high level, the score $R_i[j]$ should encode both the pertinence and the popularity of p_i with respect to t_j. The pertinence has to do with the semantic relatedness between the POI and the given TOI; the popularity has to do with how famous is a POI as a tourist attraction. For example, any POI that is a church in the city of Rome should have a high pertinence score for the TOI *Religion* and a low pertinence score for the TOI *Nature* (for a park the situation should be the opposite one). Among all churches, St. Peter's cathedral in Rome is by far the most famous one, and thus it has the highest popularity; as a result, the relevance score of St. Peter's cathedral should be the highest in the city of Rome with respect to the TOI *Religion*. The Algorithmic Engine of Cicero computes the description of the POIs by exploiting publicly accessible sources of knowledge (such as OpenStreetMap and Wikipedia). This module uses the extracted data to construct a network of concepts and to estimate the relatedness of a POI to a set of TOIs based on the graph theoretic distance in the network (see also Section 4). As it will be discussed in Section 5.3, Cicero and its Knowledge Base can be embedded in the architecture of a generic content-based travel recommender system (see also Fig. 1). For this purpose, and in order to guarantee the portability of the output of Cicero, the descriptions of the POIs can be exported as a CSV file through the GUI.

4. The Algorithmic Pipeline of Cicero

The Algorithmic Engine of Cicero implements the pipeline shown in Fig. 4. This pipeline consists of the following five steps:

1. The POIs of the geographic region of interest are retrieved from OpenStreetMap together with the tags associated with them. OpenStreetMap is a collaborative project to create a free editable map of the world that allows users to tag geographical elements including touristic attractions.
2. A *concept network* is constructed by crawling the Wikipedia pages (starting from the pages of the retrieved tags). Each node of the network represents a concept (related to tourism) and the network describes the connection between these concepts.
3. For each POI p_i, a vector Π_i is computed based on the above concept network. The component $\Pi_i[j]$ of Π_i indicates the *pertinence* of POI p_i with respect to TOI t_j, also called *pertinence score* of p_i with respect to t_j. To this aim, for each tag v retrieved in the first step, we compute a shortest path from v to the nodes representing the TOIs $t_1, t_2, \ldots, t_n$ and use the length of such a path to assign a score to v for each such a TOI. The pertinence score $\Pi_i[j]$ of POI p_i with respect to TOI t_j is a suitable combination of the scores of the tags associated with the POI and of some other information obtained from its Wikipedia page.
4. A second score is computed to take into account the popularity of each POI.
5. The pertinence score and the popularity score are combined to compute the relevance score $R_i[j]$ for each POI p_i and for each TOI t_j.

In the remainder of this section we give more details about the above five steps.

POI retrieval. Given a geographic region $\mathcal{R}$, we identify the geographical coordinates of a bounding polygon B containing such a region and download from OpenStreetMap all the POIs whose coordinates fall in B. The set of POIs P' retrieved in this way must be filtered because it contains several POIs that are not touristic attractions. For example, it includes gas stations, supermarkets, etc. In order to select only the touristic POIs, we filter them based on the tags associated with the POIs. Namely, OpenStreetMap allows the users to tag the various element of a map, including POIs. Each tag is a key-value pair and describes a geographic attribute of a map element. The OpenStreetMap community agrees on certain key-value combinations for the most commonly used tags, although users can create new tags to enrich the description of the various elements.

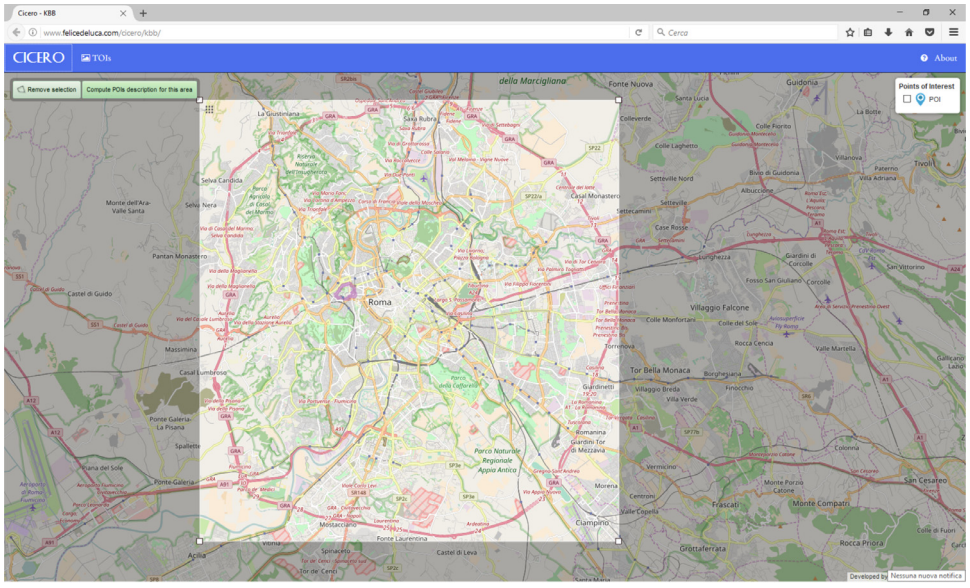
Based on the OpenStreetmap list of tags[5] we defined a list l_c of keys that are related to tourism. For example, the list contains keys like *historical, tourism, leisure, amenity*, etc. We defined a second list l_v that contains, for each key in l_c, the possible values of the key that are relevant for tourism. For example, several possible values exist for the key *amenity*. List l_v contains those values that are relevant to tourism (for example, *fountain, arts_center, theatre*, etc.), while it does not contain those that are not relevant (for example, *school, car_wash, post_office*, etc.).

The lists l_c and l_v are used to filter the set P' and to construct the set P that will be used in the next step. Let p_i be a POI of P' and let $\langle c_1, v_1\rangle, \langle c_2, v_2\rangle, \ldots, \langle c_h, v_h\rangle$ be the set of key-value pairs associated with p_i in OpenStreetMap. We put p_i in P only if there exists at least one pair $\langle c_j, v_j\rangle$ such that $c_j \in l_c$ and $v_j \in l_v$. If a POI p_i is selected to be in P, we associate with it the set $\tau(p_i)$ of all tag values v_j such that $c_j \in l_c$ and $v_j \in l_v$. The values in $\tau(p_i)$ will be used in the next steps to compute the relevance of p_i with respect to the TOIs in T.

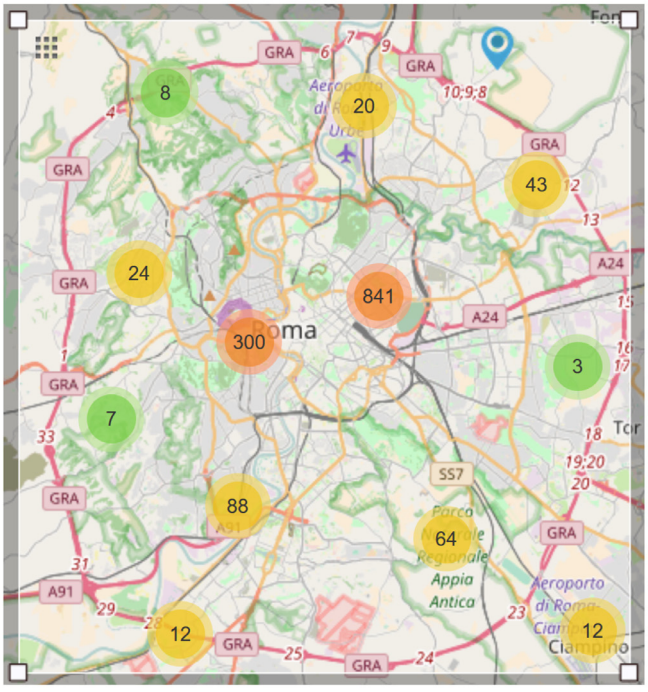
Network construction. The second step creates an oriented network $\mathcal{N}$ that will be used to compute the pertinence scores of the various POIs. The nodes of the network are "concepts" and two concepts are connected by an edge if they are semantically related. Network $\mathcal{N}$ is constructed as follows.

Each POI in P and each tag value in $W = \bigcup_{i=1}^{m} \tau(p_i)$ is a node of $\mathcal{N}$. New nodes are added to $\mathcal{N}$ by crawling Wikipedia starting from the pages associated with POIs in P and with the tag values in W; each new node of $\mathcal{N}$ corresponds to a Wikipedia page that is encountered in the crawling. The arcs correspond to wikilinks: an arc (u, v) is oriented from u to v if the Wikipedia page of u has a wikilink to the Wikipedia page of v. The crawling continues until the Wikipedia pages that correspond to the TOIs are included in the network or a maximum number of hops in the crawling has been

[5] http://wiki.openstreetmap.org/wiki/Map_Features.



(a)



(b)

Fig. 3. The interface of Cicero: (a) Selecting a Region $\mathcal{R}$ in the map of Italy. (b) The set of POIs exctracted from $\mathcal{R}$.

reached. The diameter of the Wikipedia graph is believed[6] to be about 70, which would be a natural upper bound to the number of hops performed by the crawler. As it will be discussed in Section 5, we have some experimental evidence that TOIs can be included in a concept network with a much smaller number of hops.

Fig. 5 shows a portion of the concept network for one POI *Colosseum* of the city of Rome and three TOIs, namely *Art, History*, and *Religion* (green nodes). The light blue nodes represent (some of) the tag values associated with the POI *Colosseum*, namely *Amphitheatre, Archaeological Site, Attraction*, and *Principate*[7]. A full view of the concept network for all POIs of Rome and the three TOIs *Art, History*, and *Religion* is given in Fig. 6. After just three hops all TOIs were included in the network and were all reachable by every POI; the network has 24,912 nodes, 60,292 arcs, and diameter 33.

Pertinence computation. In this step we use the concept network to compute the pertinence values of the various POIs. For each tag value $v_i \in W$ and for each TOI $t_j \in T$, we compute the shortest path in $\mathcal{N}$ from the node representing v_i to the node representing t_j. The idea is that the length of the shortest path is a

[6] http://mu.netsoc.ie/wiki/.

[7] Principate is the name of the first period of the Roman Empire, when the Colosseum was built.

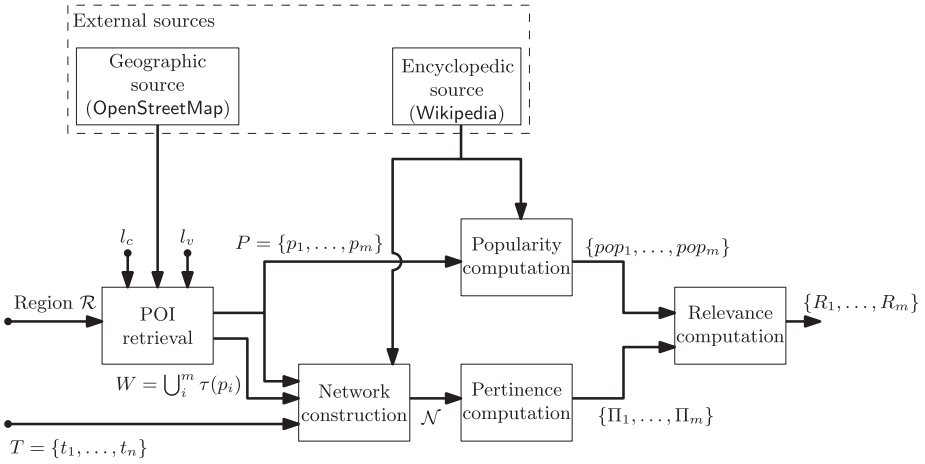


Fig. 4. The algorithmic pipeline.

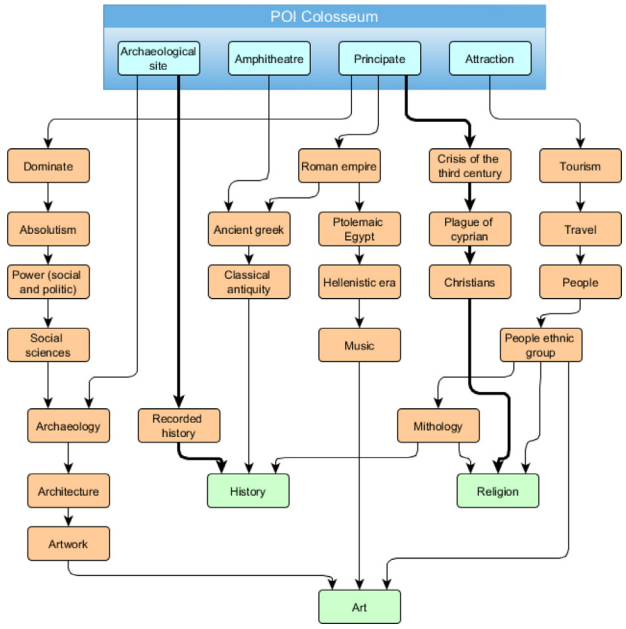


Fig. 5. Portion of the concept network of Rome for TOIs *Art, History*, and *Religion* (the green nodes) and the POI *Colosseum* (the light blue node).

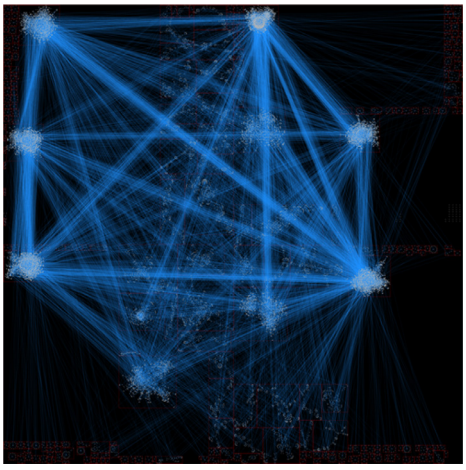


Fig. 6. The concept network of Rome for TOIs *Art, History*, and *Religion*. It consists of 24,912 nodes, 60,292 arcs, and it has diameter 33. The visualization is computed with the technique described in (Didimo & Montecchiani, 2014).

measure of how semantically related v_i and t_j are. For example, in Fig. 5 the bold edges highlight the two shortest paths that connect one of the tag values associated with the POI *Colosseum* to the TOIs *History* and *Religion*. In particular, the one connecting *Archaeological Site* to *History* is shorter than the one connecting *Principate* to *Religion*, which implies a higher level of relatedness between the Colosseum and *History* than between the Colosseum and *Religion*[8].

Once the shortest path has been computed for each tag value and for each TOI, we obtain a matrix Π' with a row for each tag value and a column for each TOI. The element $\pi'_{i,j}$ of the matrix is the length of the shortest path from the tag value v_i to the TOI t_j. We normalize the values in the matrix so that they become integer values in the range [0, 9], and such that the longer the shortest path the lower the corresponding value. The value 0 is assigned when a path does not exist. The element $\pi_{i,j}$ of the normalized matrix Π is computed as follows.

$$\pi_{i,j} = \begin{cases} 0 & \text{if there is no path from } v_i \text{ to } t_j \\ 9 - 8\frac{\pi'_{i,j} - m_j}{M_j - m_j} & \text{otherwise} \end{cases}$$

where m_j is the minimum value in the j-th column of Π' and M_j is the maximum value in the same column[9]. In other words, we make a linear mapping from the interval $[m_j, M_j]$ to the interval [9, 1]. The value $\pi_{i,j}$ of the normalized matrix Π is the *pertinence score* of tag v_i with respect to the TOI t_j.

The idea now is to compute the pertinence score of a POI p_i with respect to a TOI t_j, by combining the pertinence scores $\pi_{i_1,j}, \pi_{i_2,j}, \ldots, \pi_{i_t,j}$ of the tags $\{v_{i_1}, v_{i_2}, \ldots, v_{i_t}\}$ associated with p_i. Denote by $\pi_1, \pi_2, \ldots, \pi_t$ the values $\pi_{i_1,j}, \pi_{i_2,j}, \ldots, \pi_{i_t,j}, \pi_{i_t,j}$ sorted in non increasing order. The pertinence score of p_i with respect to t_j is $\Pi_i[j] = \sum_{h=1}^{t} \pi_h \cdot 10^{-h}$, which is clearly in the range [0, 1). With this definition of the pertinence score, if we compare two POIs, they are first compared based on the highest pertinence score of their tag values; if this value is the same, they are compared based on the second highest score, and so on.

To introduce additional information in the computation of the pertinence scores, we exploit the Wikipedia portals. Wikipedia portals are pages intended to serve as "main pages" for specific topics or areas. Each portal lists all the Wikipedia pages that are related to its topic/area. For example, there exists a portal of Mathe-

[8] It may be worth remarking that the Roman Colosseum is somewhat related to religion since it was declared a holy place by the Church in the 18th Century.

[9] Clearly, when computing the maximum and the minimum value in the j-th column we only consider the entries for which a path exists.

Table 2
An excerpt of the output of Cicero. Description of 20 POIs of the city of Rome for TOIs *Art, History*, and *Religion*; the entries are ranked for decreasing relevance with respect to *History*.

POI	Art	History	Religion
Colosseum	0.665582609	0.828402609	0.679582609
Pantheon	0.623930017	0.763930297	0.778786117
Vatican Museums	0.759204348	0.759204348	0.523304348
Trajan's Column	0.6018	0.758432	0.6032
Palatine Hill	0.666391304	0.748291304	0.596391304
Forum Romanum	0.574913043	0.745713043	0.581913043
Aurelian Walls	0.351043478	0.740663478	0.491043478
Castel Sant'Angelo	0.599286957	0.740476957	0.599286957
Museum of the Ara Pacis	0.710621739	0.736409739	0.501321739
Column of Marcus Aurelius	0.568521739	0.736353739	0.498521739
Mausoleum of Augustus	0.567217391	0.733537391	0.574217391
Quirinal Palace	0.564956522	0.730156522	0.564956522
Theatre of Marcellus	0.5662	0.728341	0.5501
Altare della Patria	0.035217391	0.728217391	0.035217391
Circus Maximus	0.46826087	0.72726087	0.46826087
Arch of Constantine	0.562347826	0.726147826	0.576347826
Forum Augustum	0.538391304	0.725571304	0.538391304
Piazza Navona	0.727547826	0.725447826	0.555347826

matics[10] and a portal of Food[11]. Portals can have sub-portals, thus defining a hierarchy. Then if the page of a POI p_i is included in a portal whose topic is a TOI t_j (this is the case for example for *Art, History*, and *Religion*), it means that p_i is very pertinent to TOI t_j. If this is the case, we add the value $\pi_{i_{t+1},j} = 9$ for the computation of the pertinence score of p_i.

Popularity computation. In this step we assign to each POI a *popularity score*. In order to estimate the popularity of a POI we adopt a very simple approach: we count the number of languages in which the Wikipedia page of that POI exists. This number is then normalized in the range [0, 1], dividing it for the maximum number of languages over all POIs. The resulting value pop_i is the popularity score of poi p_i. Notice that this value is not related to TOIs.

Computation of the POIs Descriptions. The relevance score $R_i[j]$ of POI p_i with respect to TOI t_j is obtained by combining the pertinence score $\Pi_i[j]$ and the popularity score pop_i as follows $R_i[j] = \alpha \cdot \Pi_i[j] + (1 - \alpha) \cdot pop_i$, where α is a parameter in the range [0, 1] that allows us to weight differently the contribution of the pertinence score and of the popularity score. The description of POI p_i is the array R_i.

For example, Table 2 reports the descriptions of 20 POIs in the city of Rome with respect to the three TOIs *Art, History*, and *Religion*. The table is ordered according to the decreasing order of relevance with respect to *History*.

5. Experiments and Implementation

In this section we present experiments and implementations of the techniques of Section 4. We begin by reporting the results of an experiment on the crawling procedure that constructs the concept network. This experiment aims at estimating the number of hops that a crawler needs to perform in order to reach all TOIs (Section 5.1), which is related with both the effectiveness and the efficiency of the step that constructs the concept network. Second, we perform an experimental evaluation to have an indication about the accuracy of the descriptions computed by Cicero (Section 5.2). Finally, we describe a proof of concept implementation of a simple content-based recommender system that is based on Cicero (Section 5.3).

[10] https://en.wikipedia.org/wiki/Portal:Mathematics.
[11] https://en.wikipedia.org/wiki/Portal:Food.

5.1. Properties of the Concept Network

The concept network construction procedure described in Section 4 should include all TOIs and all POIs in a connected graph. At the same time, the number of hops that are needed to construct the network should be bounded by a, possibly small, constant. It is natural to wonder whether the number of POIs affects the number of hops to be performed to construct an effective concept network. Namely, one could expect that the smaller is the number of POIs and tag values from which the crawling procedure begins, the larger becomes the number of hops needed to include all TOIs. We therefore ran an experiment to validate the following hypothesis:

- **H1:** The number of hops to reach a TOI increases as the number of POIs of a region decreases.

In order to execute the experiment, we considered the set P of the 1537 POIs in the city of Rome and the following set T of 9 TOIs: $T = \{$*Ancient Rome, Archaeology, Architecture, Art, Design, History, Medieval, Religion, Renaissance*$\}$. We extracted at random subsets of POIs of P with varying sizes. More precisely, we considered 10 different cardinalities, ranging from 10 to 100 in steps of 10; for each cardinality, we generated at random 5 subsets of P. For each TOI $t \in T$ and for each subset of P, we ran the concept network construction procedure of Section 4 and measured the following quantities:

- The number h of hops needed to include t in the concept network.
- The number h' of hops needed to guarantee that for every POI p there exists at least one path in the network connecting p (or a tag value of p) to t; clearly $h' \geq h$.

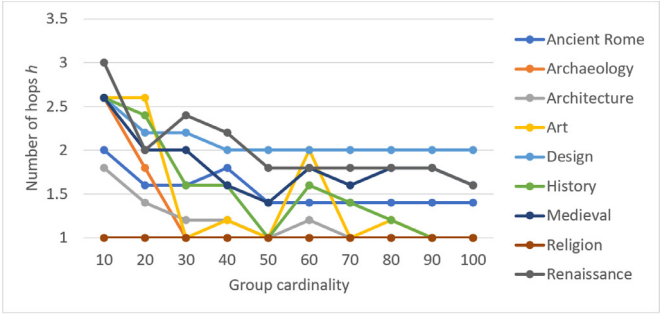
Fig. 7(a) shows the experimental results for quantity h and Fig. 7(b) shows the experimental results for quantity h'. It is interesting to observe that the number of hops is very small in both charts and that, surprisingly, it is not much affected by the cardinality of the groups. While this result partly contradicts the experimental hypothesis, it also suggests that, in practice, a crawling procedure can reach the wanted TOIs with 2-3 hops even for territories with a limited number of POIs.

We also remark that the number of hops needed to include the TOIs in the network is not an upper bound on the length of the shortest paths. Namely, while the crawler executes a BFS and thus it proceeds by following BFS-tree edges in the Wikipedia graph, the paths connecting TOIs to POIs may include edge not in the BFS-tree. For each group of POIs we also measured the maximum graph theoretic distance d between POIs and TOIs, that is the length of the longest among the shortest paths that connect POIs to each TOI. Fig. 7(c) reports the average value of d for the different cardinalities. For example, while three hops are sufficient to guarantee that there is a path from every POI to each TOI, there can be shortest paths whose average length is 8.

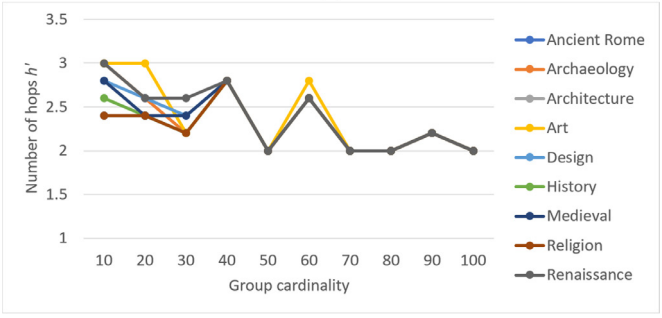
5.2. Experimental Evaluation of Cicero

We now present an experimental study whose goal is to evaluate the effectiveness of the POIs description computed by Cicero. In particular, our experiments are guided by the following experimental hypotheses.

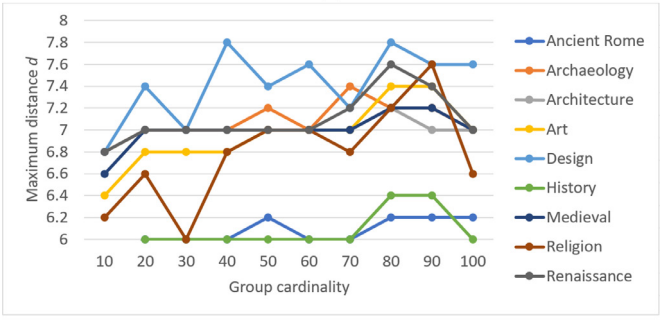
- **H1:** System Cicero computes effective POIs descriptions. Specifically, the ranking of the POIs with respect to a given set of TOIs is comparable with that provided by a ground truth.
- **H2:** The effectiveness of Cicero is not affected by the size of the set of POIs considered. In particular, the effectiveness of the system is about the same when applied on a big and famous city and when applied on a smaller and less famous town.



(a)



(b)



(c)

Fig. 7. (a) Number of hops h needed to include each TOI in the concept network. (b) Number of hops h' needed to guarantee that each TOI can be reached from all POIs. (c) Maximum distance d between POIs and each TOI.

In what follows we describe the experimental process that we employed, the dataset used in our experiments, the construction of the ground truth, and the results that we obtained.

Experimental process. We compared a solution computed by Cicero with a ground truth, that is, with a solution computed by some authoritative source for the given region. Specifically, we first sort the POIs of a certain region by the relevance scores computed by Cicero with respect to each TOI in a given set, thus obtaining for each TOI t a list of POIs sorted from the most relevant to the least relevant with respect to t. Each list is then compared with an analogous list provided by the ground truth.

Dataset construction. We considered two geographical regions that correspond to two Italian cities having different sizes and popularity: Rome and Perugia. Also, we considered the following three TOIs: *Religion, Art* and *History*. Although, ideally, all POIs of the two cities should be ranked, we limited the list of evaluated POIs because we used a ground truth constructed by humans (see also the next paragraph) and thus it was important to limit the required effort. Specifically, for each city and for each TOI we selected a subset $P_{a,\,b}$ (with $a \in \{Rome, Perugia\}$ and $b \in \{Religion, Art, History\}$) of the available POIs. The number of POIs in each $P_{a,\,b}$ is 25 when $a = Rome$ and 10 when $a = Perugia$ (the different number of POIs is justified by the different size of the two cities). The POIs in each set have been selected as the top 25 or the top 10 in the sorted list computed with Cicero. The sets were manually checked to guarantee that the selected POIs were reasonably well known, which was the case.

Table 3
Percentage error of the computed ranking with respect to the ground truth.

	Religion	Art	History
Rome	28%	29%	32%
Perugia	20%	33%	29%

Ground truth construction. In order to construct our ground truth, we looked for an authoritative source for each of the two cities. After considering some possible alternatives, we decided to rely on people with a solid knowledge of the two cities. In particular, we recruited 12 people for Rome and 8 people for Perugia. Among the 12 people selected for Rome there were 7 professional touristic guides, while the pool of people selected for Perugia included 5 people whose job is related to tourism (museum directors, public administrators in the area of touristic promotions, etc.). The remaining people (both for Rome and Perugia) were people who know the city and visited it several times. We gave to each person the three sets $P_{a,\,b}$ relative to the city for which the person was recruited and asked her to assign to each POI $p_i \in P_{a,\,b}$ a score between 1 and 5 to indicate how "important" is to visit POI p_i for a tourist interested in the TOI b. For each set $P_{a,\,b}$ we computed the average score obtained by each POI in the set and rank the POIs accordingly. Our ground truth consists therefore of six ordered lists $l_{a,\,b}$ of POIs (one per city and per TOI).

It might be worth remarking that we also considered alternative approaches for computing the ground truth. Unfortunately, none of them met our needs. We considered using online travel recommender systems, but most of them allows one to search for an itinerary but do not give a ranking of all (or most) POIs of a region. We also considered Web portals that "classify" the POIs of a region by topic. These websites provide lists of POIs to visit if one is interested in art, history, etc., but the POIs in these lists are not ranked by relevance with respect to a TOI. Finally, we considered systems where POIs are ranked based on the ratings given by the users. In this case however, the rating is not related to a set of TOIs.

Results and discussion. For each city a and for each TOI b, we compared the list $l_{a,\,b}$ of the ground truth and the list $l'_{a,b}$ of the same set of POIs ranked according to the scores computed by Cicero. As a measure of the error made by our method with respect to the ground truth, we counted the number of pairs of POIs $\langle p_j, p_k\rangle$ such that p_j is before p_k in $l_{a,\,b}$ and after p_k in $l'_{a,b}$. In other words, we measured the number of pairs of POIs that have different relative positions in the two lists. We express this error as a percentage of the total number of pairs, i.e., $\frac{m(m-1)}{2}$ where m is the number of POIs in the two lists. Table 3 shows the results for each city and for each TOI. The error is always around 30% and there is no significant difference between the two cities and the three TOIs.

In order to better interpret the numbers in Table 3 we executed another experiment. We asked to two other people to fill the same questionnaire used by the experts to define the ground truth. We then measured the error made by these two people with respect to the ground truth. The idea was to compare the behavior of our system with two humans with a good knowledge of the two cities. The data are shown in Table 4.

Concerning Rome, the percentage errors of the two users are similar to those automatically computed (with better values for our technique in 4 cases over 6). About Perugia, our technique has lower percentage error in most cases. The most divergent values are those for *Religion* (where our technique outperforms User 1

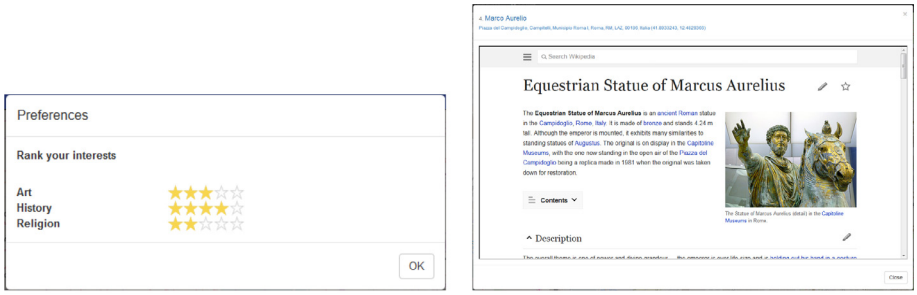


Fig. 8. The CRS system: (a) Choosing the users' interests. (b) Wikipedia page preview of a POI.

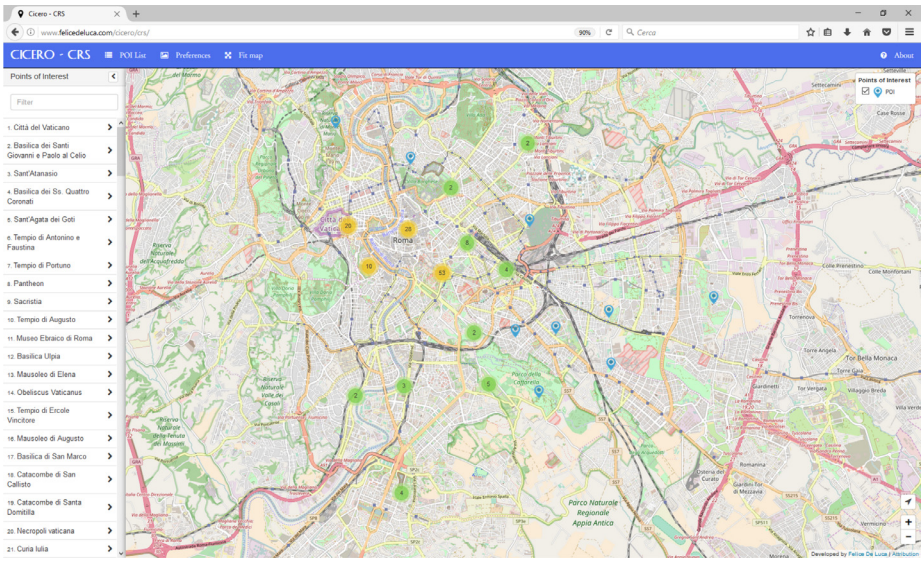


Fig. 9. The CRS system: list and map of ranked POIs.

Table 4
Percentage error of the ranking computed by two people with respect to the ground truth.

		Religion	Art	History
User 1	Rome	35%	33%	34%
	Perugia	45%	16%	38%
User 2	Rome	27%	35%	30%
	Perugia	23%	18%	29%

by 25%) and *Art* (where both users outperformed our technique by about 15%).

In summary, while the relatively limited size of the collected experimental data does not allow to draw statistically significant conclusions, the outcome of the experiments does however provide a good evidence in favor of both **H1** and **H2**. Concerning **H2**, we remark that Perugia has fewer POIs, which are less popular and for which there is less information in the on-line data sources used by our technique.

5.3. Proof of concept

The Knowledge Base constructed by Cicero is meant to be used in a content-based recommender system. For this reason, we implemented a proof-of-concept system, called `Cicero Recommender System` (CRS)[12].

The system can be accessed within the GUI of Cicero by simply following the link "Use in CRS". By interacting with CRS, a user is requested to fill a form (see Fig. 8(a)) to indicate her interests with respect to a set of TOIs that have been used by Cicero when creating the Knowledge Base for the region of interest. The user's interests are described by a vector I, where the component $I[j]$ is a number (in the range [0, 1]) indicating the interest of the user for the TOI t_j. To compute the ranking of the POIs to be returned to the user, CRS computes for each poi $p_i \in P$ a score $s_i = \sum_{j=1}^{n} I[j]R_i[j]$, where $R_i[j]$ is the relevance score of POI p_i with respect to the TOI t_j as stored in the Knowledge Base. The list of POIs ranked by the values s_i $(i = 1, 2, \ldots, m)$ is shown to the user together with their position on a map (see Fig. 9). The user can look at the details of each POI by selecting it either in the list or on the map. When a POI is selected, a pop-up shows basic information about the POI (its name, its position in the rank, its address and its geographic position) and a preview of the Wikipedia page, if any (see Fig. 8(b)). The user can also filter the list of POIs based on their names.

The system CRS can be used either after an execution of Cicero using the output as a knowledge base, or with a set of cities (currently Rome and Perugia) whose corresponding knowledge bases have already been computed by Cicero. For both these cities the TOIs available to express the user's interests are *Art*, *History*, and *Religion*. We remark that the two cities and the set of TOIs are the same as those used in the experimental analysis described in Section 5.2.

[12] http://www.felicedeluca.com/cicero/crs/.

6. Conclusions and Future Work

We studied the problem of designing the Content Analyzer of a content-based travel recommender system. We described and implemented a technique to automatically compute a description of the POIs of one or more geographic regions using publicly available sources of information (OpenStreetMap and Wikipedia). We performed an experimental evaluation against a human-created ground truth. Even though these experiments do not constitute the ultimate aim of the present work, they however provide a good approximation of such effectiveness. Possible future work and further developments of our research are:

- Design a larger experimental analysis so to obtain statistical evidence about the effectiveness of Cicero.
- Build and experiment a more sophisticated version of CRS, for example by including the automatic computation of itineraries, and the possibility to access it by the means of context-aware mobile apps.
- Design and experiment new strategies to compute POIs relevance scores. For example, we find it interesting to study how to adapt techniques of Explicit Semantic Analysis (ESA) (Egozi et al., 2011) to the context of travel recommender systems.
- Integrate more on-line data sources to enrich the amount of information associated with each POI. For example, Wikipedia is not suited for ranking POIs with respect to TOIs related to entertainment, such as *Gastronomy* or *Events*.
- Study how to integrate Cicero with techniques that estimate the popularity of a POI by using sentiment analysis on social networks.

References

Banerjee, S., Ramanathan, K., & Gupta, A. (2007). Clustering short texts using wikipedia. In *Proceedings of the 30th annual international acm sigir conference on research and development in information retrieval, SIGIR '07* (pp. 787–788). New York, NY, USA: ACM. doi:10.1145/1277741.1277909.

Batet, M., Moreno, A., Sánchez, D., Isern, D., & Valls, A. (2012). Turist@: Agent-based personalised recommendation of tourist activities. *Expert Systems with Applications, 39*(8), 7319–7329. doi:10.1016/j.eswa.2012.01.086.

Binucci, C., Didimo, W., Liotta, G., Montecchiani, F., & Sartore, M. (2013). TRART: A system to support territorial policies. In J. A. Botía, & D. Charitos (Eds.), *Proceedings of the 9th international conference on intelligent environments: 17* (pp. 629–634). IOS Press. doi:10.3233/978-1-61499-286-8-629. Ambient Intelligence and Smart Environments

Borrás, J., Moreno, A., & Valls, A. (2014). Intelligent tourism recommender systems: A survey. *Expert Systems with Applications, 41*(16), 7370–7389. doi:10.1016/j.eswa.2014.06.007.

Braunhofer, M., Elahi, M., Ricci, F., & Schievenin, T. (2013). *Context-aware points of interest suggestion with dynamic weather data management*. In Z. Xiang, & I. Tussyadiah (Eds.) (pp. 87–100)). Cham: Springer International Publishing doi:10.1007/978-3-319-03973-2_7.

Brilhante, I. R., Macedo, J. A., Nardini, F. M., Perego, R., & Renso, C. (2015). On planning sightseeing tours with TripBuilder. *Information Processing & Management, 51*(2), 1–15. doi:10.1016/j.ipm.2014.10.003.

Castillo, L., Armengol, E., Onaindía, E., Sebastiá, L., González-Boticario, J., Rodríguez, A., … Borrajo, D. (2008). SAMAP: An user-oriented adaptive system for planning tourist visits. *Expert Systems with Applications, 34*(2), 1318–1332. doi:10.1016/j.eswa.2006.12.029.

Cenamor, I., de la Rosa, T., Núñez, S., & Borrajo, D. (2017). Planning for tourism routes using social networks. *Expert Syst. Appl., 69*, 1–9. doi:10.1016/j.eswa.2016.10.030.

Didimo, W., & Montecchiani, F. (2014). Fast layout computation of clustered networks: Algorithmic advances and experimental analysis. *Inf. Sci., 260*, 185–199. doi:10.1016/j.ins.2013.09.048.

Egozi, O., Markovitch, S., & Gabrilovich, E. (2011). Concept-based information retrieval using explicit semantic analysis. *ACM Trans. Inf. Syst., 29*(2), 8:1–8:34. doi:10.1145/1961209.1961211.

Gabrilovich, E., & Markovitch, S. (2009). Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Intell. Res. (JAIR), 34*, 443–498. doi:10.1613/jair.2669.

Gavalas, D., Kasapakis, V., Konstantopoulos, C., Pantziou, G., Vathis, N., & Zaroliagis, C. (2015). The ecompass multimodal tourist tour planner. *Expert Systems with Applications, 42*(21), 7303–7316. doi:10.1016/j.eswa.2015.05.046.

Gavalas, D., & Kenteris, M. (2011). A web-based pervasive recommendation system for mobile tourist guides. *Personal and Ubiquitous Computing, 15*(7), 759–770. doi:10.1007/s00779-011-0389-x.

de Gemmis, M., Lops, P., Musto, C., Narducci, F., & Semeraro, G. (2015). Semantics-aware content-based recommender systems. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 119–159). Springer. doi:10.1007/978-1-4899-7637-6_4.

Győrödi, R., Győrödi, C., & Dersidan, M. (2013). An extended recommendation system using data mining implemented for smart phones. *International Journal of Computers & Technology, 11*(3), 2360–2372.

Huang, Y., & Bian, L. (2009). A Bayesian network and analytic hierarchy process based personalized recommendations for tourist attractions over the internet. *Expert Systems with Applications, 36*(1), 933–943. doi:10.1016/j.eswa.2007.10.019.

Jarmasz, M., & Szpakowicz, S. (2003). Roget's thesaurus and semantic similarity. In N. Nicolov, K. Bontcheva, G. Angelova, & R. Mitkov (Eds.), *Recent advances in natural language processing iii, selected papers from RANLP 2003, borovets, bulgaria: 260* (pp. 111–120). John Benjamins, Amsterdam/Philadelphia. Current Issues in Linguistic Theory (CILT)

Leacock, C., & Chodorow, M. (1998). Combining Local Context and WordNet Similarity for Word Sense Identification. In C. Fellbaum (Ed.), *Wordnet: An electronic lexical database.* (pp. 265–283). MIT Press.

Lee, C.-S., Chang, Y.-C., & Wang, M.-H. (2009). Ontological recommendation multi-agent for Tainan city travel. *Expert Systems with Applications, 36*(3, Part 2), 6740–6753. doi:10.1016/j.eswa.2008.08.016.

Lim, K. H., Chan, J., Leckie, C., & Karunasekera, S. (2015). Personalized tour recommendation based on user interests and points of interest visit durations. In *Proceedings of the 24th international conference on artificial intelligence*. In *IJCAI'15* (pp. 1778–1784). AAAI Press.

Lucas, J. P., Luz, N., Moreno, M. N., Anacleto, R., Figueiredo, A. A., & Martins, C. (2013). A hybrid recommendation approach for a tourism system. *Expert Systems with Applications, 40*(9), 3532–3550. doi:10.1016/j.eswa.2012.12.061.

Martínez Santiago, F., Ariza López, F., Montejo-Ráez, A., & Ureña López, A. (2012). GeOasis: A knowledge-based geo-referenced tourist assistant. *Expert Systems with Applications, 39*(14), 11737–11745. doi:10.1016/j.eswa.2012.04.080.

Meehan, K., Lunney, T., Curran, K., & McCaughey, A. (2013). Context-aware intelligent recommendation system for tourism. In *Pervasive computing and communications workshops (percom workshops), 2013 ieee international conference on* (pp. 328–331). doi:10.1109/PerComW.2013.6529508.

Montejo-Ráez, A., Perea-Ortega, J. M., García-Cumbreras, M. A., & Martínez-Santiago, F. (2011). Otiüm: A web based planner for tourism and leisure. *Expert Systems with Applications, 38*(8), 10085–10093. doi:10.1016/j.eswa.2011.02.005.

Moreno, A., Valls, A., Isern, D., Marin, L., & Borrás, J. (2013). SigTur/E-Destination: Ontology-based personalized recommendation of tourism and leisure activities. *Engineering Applications of Artificial Intelligence, 26*(1), 633–651. doi:10.1016/j.engappai.2012.02.014.

Niaraki, A. S., & Kim, K. (2009). Ontology based personalized route planning system using a multi-criteria decision making approach. *Expert Systems with Applications, 36*(2, Part 1), 2250–2259. doi:10.1016/j.eswa.2007.12.053.

Ponzetto, S. P., & Strube, M. (2007). Knowledge derived from wikipedia for computing semantic relatedness. *J. Artif. Intell. Res. (JAIR), 30*, 181–212. doi:10.1613/jair.2308.

Rada, R., Mili, H., Bicknell, E., & Blettner, M. (1989). Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man, and Cybernetics, 19*(1), 17–30. doi:10.1109/21.24528.

Ruotsalo, T., Haav, K., Stoyanov, A., Roche, S., Fani, E., Deliai, R., … Hyvönen, E. (2013). SMARTMUSEUM: A mobile recommender system for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web, 20*, 50–67. doi:10.1016/j.websem.2013.03.001.

Savir, A., Brafman, R., & Shani, G. (2013). Recommending improved configurations for complex objects with an application in travel planning. In *Proceedings of the 7th acm conference on recommender systems*. In *RecSys '13* (pp. 391–394). New York, NY, USA: ACM. doi:10.1145/2507157.2507196.

Staab, S., Werthner, H., Ricci, F., Zipf, A., Gretzel, U., Fesenmaier, D. R., … Knoblock, C. A. (2002). Intelligent systems for tourism. *IEEE Intelligent Systems, 17*(6), 53–64. doi:10.1109/MIS.2002.1134362.

Taieb, M. A. H., Aouicha, M. B., & Hamadou, A. B. (2013). Computing semantic relatedness using wikipedia features. *Knowledge-Based Systems, 50*, 260–278. doi:10.1016/j.knosys.2013.06.015.

Umanets, A., Ferreira, A., & Leite, N. (2014). GuideMe a tourist guide with a recommender system and social interaction. *Procedia Technology, 17*, 407–414. doi:10.1016/j.protcy.2014.10.248.

Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Oudheusden, D. V. (2011). The City Trip Planner: An expert system for tourists. *Expert Systems with Applications, 38*(6), 6540–6546. doi:10.1016/j.eswa.2010.11.085.

Wang, W., Zeng, G., & Tang, D. (2011). Bayesian intelligent semantic mashup for tourism. *Concurrency and Computation: Practice and Experience, 23*(8), 850–862. doi:10.1002/cpe.1676.

Wu, Z., & Palmer, M. (1994). Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on association for computational linguistics*. In *ACL '94* (pp. 133–138). Stroudsburg, PA, USA: Association for Computational Linguistics. doi:10.3115/981732.981751.

Yang, W.-S., & Hwang, S.-Y. (2013). iTravel: A recommender system in mobile peer-to-peer environment. *Journal of Systems and Software, 86*(1), 12–20. doi:10.1016/j.jss.2012.06.041.