Semantic Search in Tabular Structures

Aleksander Pivk[1], Matjaž Gams[1] and Mitja Luštrek[1,2]

[1] Department of Intelligent Systems, Jozef Stefan Institute,
Jamova 39, SI-1000 Ljubljana, Slovenia

[2] Department of Computer Science, University of Alberta,
Edmonton, Alberta, Canada T6G 2E8

E-mail: *{aleksander.pivk, matjaz.gams, mitja.lustrek}@ijs.si*

Keywords: tabular structures, ontology learning, semantic web, query answering

Received: November 18, 2005

The Semantic Web search aims to overcome the bottleneck of finding relevant information using formal knowledge models, e.g. ontologies. The focus of this paper is to extend a typical search engine with semantic search over tabular structures. We categorize HTML documents into topics and genres. Using the TARTAR system, tabular structures in the documents are then automatically transformed into ontologies and annotated to build a knowledge base. When posting queries, users receive responses not just as lists of links and description extracts, but also enhanced with replies in the form of detailed structured data.

Povzetek: Razvili smo metode semantičnega spleta za iskanje informacij v tabelah.

1 Introduction

The World Wide Web has in the years of its exponential growth become a universal repository of human knowledge and culture, thus enabling an exchange of ideas and information on a global level. The tremendous success of the Internet is based on its usage simplicity, efficiency, and enormous market potential [4].

The success of the World Wide Web is countervailed by efforts needed to search and find relevant information. The search of interesting information turned out to be a difficult, time-consuming task, especially due to the size, poor structure and lack of organization of the Internet [3, 7, 31]. A number of approaches appeared in the last decade with a common objective to improve searching and gathering of information found on the Web.

One of the first solutions to cope with the information overload was search engines. In order for a search engine to function properly and to return relevant and satisfactory answers to user queries, it must conduct two important tasks in advance. The first task is to crawl the Web and gather, following the hyperlinks, as many documents as possible. The second task deals with document indexing [12, 35], hyperlink analysis, document relevance ranking [20, 23] and high dimensional similarity searches [13, 19].

Once these tasks are completed, a user may post queries to gain answers. User queries, unless they include highly selective keywords, tend to match a large number of documents, because they do not contain enough information to pinpoint most highly relevant resources [28]. They may sometimes even miss the most relevant responses, because of no direct keyword matches, but the most common disadvantage of this approach is that an engine might return thousands of potentially interesting links for a user to manually explore. The study described in [17] showed that the number of keywords in queries is typically smaller than three, which clearly cannot sufficiently narrow down the search space. In principle, this can be seen as the problem of users of search engines themselves. Reducing the ambiguity of search requests can be achieved by adding semantics, i.e. by making computers better 'understand' both the content of the web pages and the users' intentions, which can help to improve search results even with a request of a very limited size. In addition, search engines favor largest information providers due to name-branding and time optimization. To overcome this bottleneck, the Semantic Web search, where information is structured in a machine interpretable way, is a natural step forward, as originally envisioned by Tim Berners-Lee [4].

Moving to a Semantic Web, however, requires the semantic annotation of Web documents, which in turn crucially depends on some sort of automatic support to facilitate this task. Most information on the Web is presented in semi-structured and unstructured documents, i.e. loosely structured natural language text encoded in HTML, and only a small portion represents structured documents [2, 12]. A semi-structured document is a mixture of natural language text and templates [2, 12]. The lack of metadata that would precisely annotate the structure and semantics of documents and ambiguity of natural language in which these documents are encoded makes automatic computer processing very complex [9, 21].

Tabular structures (i.e. tables or lists) are incorporated into semi-structured documents and may have many different forms and can also differ substantially even if they represent the same content or data [14, 16].

Here we consider tabular structures as tables and lists which are described by a particular tag in HTML, i.e. `<table>` represents a table, where `<ul>`, `<ol>` and `<dl>` stand for different list types. A simple example of a table, found in a Web document, is represented in Figure 1.

Accommodation	Location	Type	Period	Cost
Bungalow GABER	Rogla	Apartment	Winter	1260,-
			Spring	1150,-
			Summer	1090,-
			Autumn	850,-
Hotel AREH	Pohorje	Single Room	Winter	590,-
			Rest	490,-
		Double Room	Winter	840,-
			Rest	690,-
Pension MARTIN	Pohorje	Apartment	Winter	1190,-
			Autumn	1090,-
			Rest	960,-
		Double Room	Winter	890,-
			Autumn	790,-
			Rest	730,-

Figure 1: A simple table example belonging to a tourism domain.

Understanding of their content is crucial when it comes to finding and providing explicit answers to user queries. The table understanding task demands efficient handling of tabular structures and automatic transformation of structures into explicit semantics, which enables further reasoning about the data within tables. But both, a table structure comprehension task and a semantic interpretation task exceed in complexity the corresponding linguistic task [16].

The paper is structured as follows. In Section 2 we first introduce a simplified view of the ALVIS search engine followed by a detailed description and an algorithm of our extensions. Section 3 describes the current state of our approach and its evaluations. After presenting the related work in Section 4 we give concluding remarks in Section 6.

2 ALVIS - Semantic search engine extensions

The basic idea for attaining a semantic search engine is to automatically enrich the indexed web pages with semantics, in our case in the form of ontologies. This is the key subject of the EU IST-project ALVIS (Superpeer Semantic Search Engine, IST-1-002068-STP), which represents the framework for our contribution. The general architecture of ALVIS is presented in Figure 2. Our major contribution is represented as two bold boxes on the right-hand side of the figure.

The ALVIS project (`www.alvis.info/alvis`) conducts research in the design, use and interoperability of topic-specific search engines with the goal of developing an open source prototype of a distributed, semantic-based search engine. ALVIS facilitates semantic retrieval and incorporates pre-existing domain ontologies using facilities for import and maintenance. The distributed design is based on exposing search objects as resources and on using implicit and automatically generated semantics (not just ontologies) to distribute queries and merge results. Because semantic expressivity and interoperability are competing goals, developing a system that is both distributed and semantic-based is the key challenge: research involves both the statistical and linguistic format of semantic internals, and determining the extent to which the semantic internals are exposed at the interface.

The problems that were identified and served as a motivation factor while developing the system and methodology are: (a) tabular structures present a general schema for data presentation due to their good visual comprehension, hence a great deal of data is hidden within them, (b) multiple table representations for the same content/data are very likely, (c) manual annotation does not scale in general, (e) tables and lists, due to their richer structure, support the discovery of interdependent information, whilst a typical keyword or link-analysis based search usually fails in this area [27], and (f) ontology learning from arbitrary text has so far had limited success [6, 9].

The whole process of semantic search over tabular structures consists of two subprocess: (a) transformation of tabular structures found on the Web into the knowledge base, described in subsection 2.1 and shown in Figure 3, and (b) user query processing, described in subsection 2.2 and shown in Figure 6.

2.1 Tabular structures transformation

The overall analysis, transformation, and formalization of tabular structures can be graphically observed in Figure 3. It roughly corresponds to the bold boxes in Figure 2 and directly to Figure 4. Here we will shortly present each of the five major steps with the obtained results given in section 3:

(1) Categorization of Web documents
  (a) into Topics
  (b) into Genres
(2) Filtering of proper tabular structures from documents
(3) Transformation of arbitrary tabular structures into formal representations
(4) Automatic ontology generation from formalized tabular structures
(5) Knowledge base construction

Figure 4: Tabular structure formalization procedure.

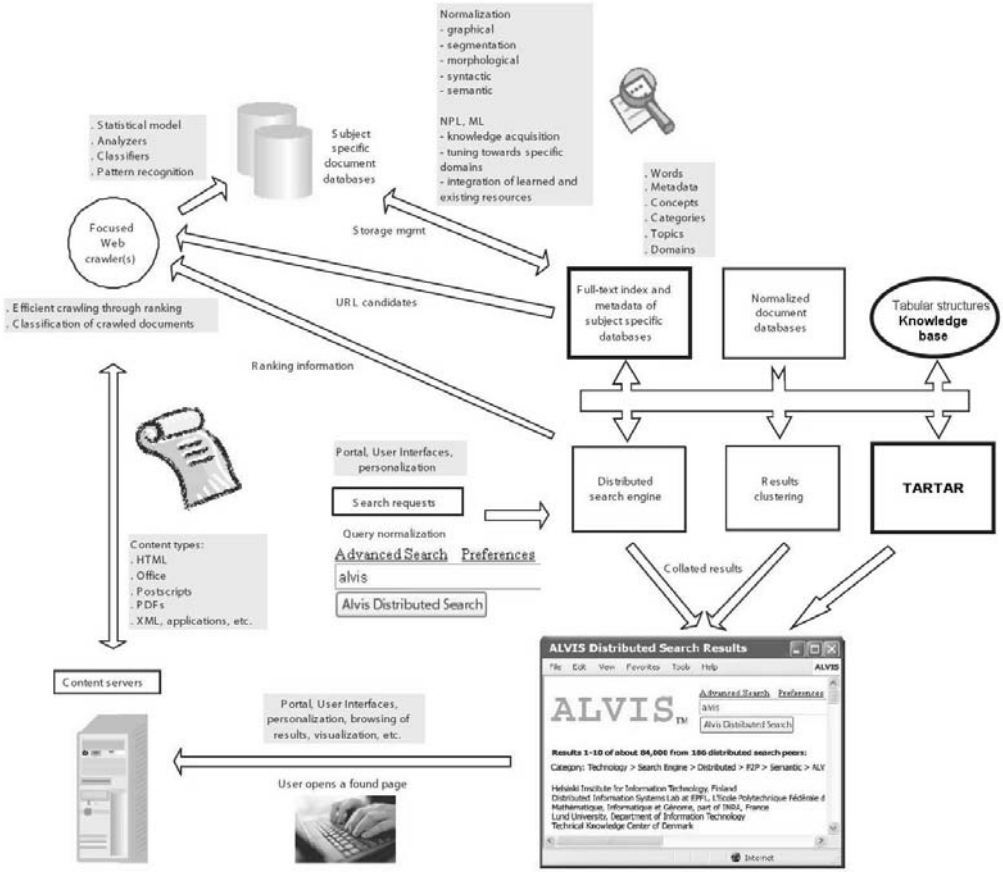

Figure 2: Performance architecture of ALVIS search engine.

1) Categorization of Web documents

Clustering is a technique for classification of Web documents into groups within which interdocument similarity is large compared to the similarity between documents chosen from different groups, and enables taxonomy design, e.g. topic hierarchy construction. The technique assists fast similarity search, while groups of similar documents significantly narrow the search space.

1.a) Topics

Work on topic classification has been utilized by our colleagues Mladenić et al. [22]. They used machine learning techniques on a document collection gathered from Yahoo to reconstruct Yahoo's manually built topic taxonomy. Documents are represented as feature-vectors that include word sequences, where the learning approach depends on feature subset selection and results in a combination of a set of independent classifiers. The whole categorization process belongs to the bold box (Full-text index...) in Figure 2.

1.b) Genres

Another type of categorization that we propose is categorization into genres. According to Merriam-Webster Online Dictionary, a genre is a category of artistic, musical, or literary composition characterized by a particular style, form, or content. Editorial and poem, for example, are genres characterized by style and form, while science fiction is a genre characterized by content. For our purpose, we will concentrate on the first kind of genres, because the second kind are already in large part covered by topical categorization. In information retrieval, genres should ideally be orthogonal to topics, because this best divides the search space and reduces the number of search hits. Traditional genres often do not satisfy this requirement, but in the selection of genres we use and in the preparation of training data for the genre classifier, we tried to be as close to the ideal as possible.

We chose 20 genres one can expect to find on the World Wide Web: some of them Web-specific, such as FAQ and community, others traditional, such as journalistic and poetry. The next step was the preparation of the training data for the genre classifier. We gathered Web pages returned by the most popular queries according to Google, random Web pages and Web pages specifically sought to belong to the less common genres. Each Web page was manually categorized by two annotators and in case of disagreement checked by the authors.

Finally, a suitable classification algorithm needed to be selected. Most algorithms require extraction of stylistic

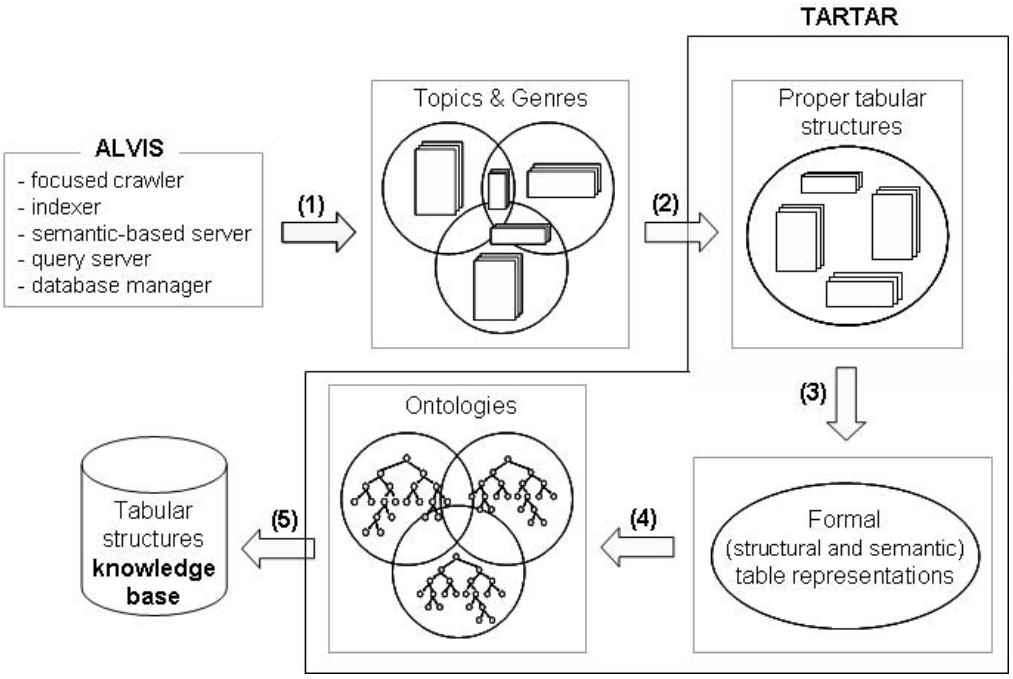

Figure 3: The transformation of tabular structures found on the Web into the knowledge base.

features from the Web pages. Due to the complexity and irregularity of HTML found on the Web, this is a time-consuming and error-prone procedure that can compromise the robustness of the classifier. Therefore we chose prediction by partial matching (PPM) classification algorithm, which can accept raw HTML as the input. This algorithm is derived from data compression techniques and essentially learns the way to most efficiently compress the Web pages belonging to each genre. An uncategorized page is then compressed with each genre-specific compressor in turn and categorized into the genre associated with the most effective compressor. Initial tests have been performed by Bratko and colleagues [5], proving successful in a related problem setting.

2) Filtering of proper tabular structures

Tabular structures appear in Web documents within specific tags, thus enabling their easy discovery. Unfortunately, it has been shown that at least 70% of such structures are not used for presentation of data but for better visual and layout purposes [8]. To solve this problem, we adopted a solution developed by Wang et al. [34]. It is based on machine learning techniques, in particular decision trees and support vector machines. In the learning process, several types of features are used, such as structural, content, and word/text features.

3) Formalization of tabular structures

This essential part concentrates on the analysis of tabular structures aiming to exploit their partial structure and cognitive modeling habits of humans. Understanding of table contents requires table-structure comprehension and semantic interpretation. The outcome of applying the method is a knowledge frame encoded in an F-Logic representation language [18].

The most comprehensive and complete model for the analysis and transformation of tables found in literature is Hurst's [15], which is also adopted here. The model analyzes tables along graphical, physical, structural, functional, and semantic dimensions. Our approach stepwise instantiates the last four dimensions.

In the first step, corresponding to the physical dimension, a table is extracted, cleaned, canonicalized and transformed into a regular matrix form.

In the second step, the goal is to detect the table structure. This is a very complex task, since there is a huge number of table layout variations. The three most important sub-tasks are: (a) to determine table reading orientation(s), which is discovered by measuring the distance/similarity among cells and hence among rows and columns (features given at the end of the paragraph); (b) to dismember a table into logical units and further into individual regions, in a way that regions consist of only attribute cells, e.g. 'Location' in Figure 1, or instance cells, e.g. 'Rogla'; (c) to resolve the table type, which must belong to the one of five pre-defined types (one- or two-dimensional, or three types of complex tables). For these purposes several heuristics, i.e. token type hierarchy, and measures are used, i.e. value similarity (regression), value features (character/numeric ratio, mean, variance, standard deviation), data frames (e.g. regular expressions for recognizing dates), and string patterns.

In the third step, the functional table model (FTM) is constructed. FTM is represented as a directed acyclic graph, which rearranges table regions in a way to exhibit

an access path for each individual cell. After finalizing the FTM construction, its recapitulation is carried out with a goal to minimize the model.

Finally, in the fourth step we deal with the discovery of semantic labels for table regions, where WordNet [11] lexical ontology and GoogleSets service are employed. These semantic labels serve as annotations of FTM nodes and are later also used within outgoing formal structures as can be observed in Figure 5.

A very detailed description of this particular part can be found in [30].

4) Automatic ontology generation

Tables with different structures that are categorized into the same topic/genre clusters and had been individually transformed into knowledge frames, could now be merged into a single cluster ontology. In this way, a generalized representation that semantically and structurally describes captured tables within each cluster would be created.

Our approach to frame matching and merging is multifaceted, which means that we use all evidence at our disposal to determine how to match concepts. In using this evidence we look not only for direct matches as is common in most schema matching techniques, but also indirect matches. The most relevant matching techniques are the following: (a) label matching which depends on WordNet features and string distance metrics; (b) value similarity, where matching is based on value characteristics; (c) expected values by using constant value recognizers in data frames; (d) constraints that include keys of the table, functional relationships, one-to-one correspondences, and subset/superset relationships; and (e) structural context, where features such as proximity, node importance measured by in/out-degree, and neighbor similarity help match object sets.

Once the mappings among frames have been discovered, the actual merging is executed. Sometimes two frames are directly fused by simply merging corresponding nodes and edges. Often, however, merging induces conflicts that must be resolved, where several different approaches for conflict resolution are used. Conflict resolution will not be discussed in this paper due to length limits. Finally, after all frames of a particular topic/genre cluster are incorporated, the outcome reflects an appropriate domain ontology, where the concepts are arranged into a directed acyclic graph, where the arcs represent the relations among concepts and are labeled with relation type.

Figure 5 illustrates a simple example of the ontology creation process by merging two same-cluster tables; the second table is shown in Figure 1. Both tables deal with hotel price information, but the naming convention and contents are different. The matching of the frame concepts and object sets are based on several techniques: Label Matching and Structure help determine the concept matching, while Value Similarity and Expected Values apply to price, room type, and period. The domain ontology is presented as a frame, but its concepts are further classified and the

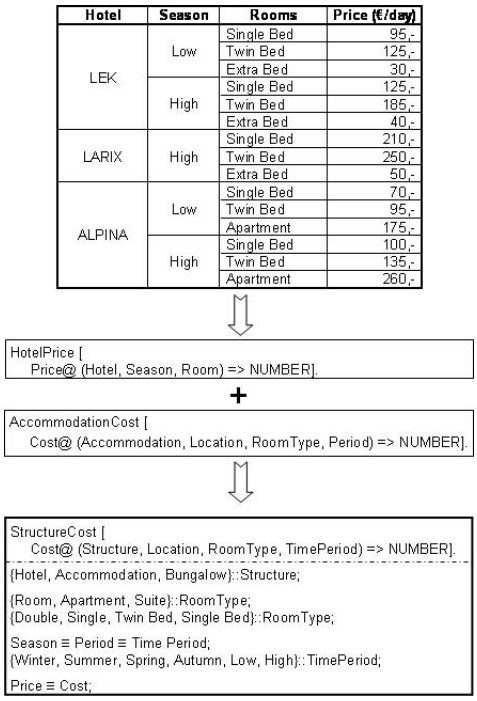

Figure 5: A simple example of the ontology creation process.

classification encoded in F-Logic notation. For example, concepts 'Hotel' and 'Accommodation', according to the WordNet, share a common hypernym 'Structure'; values 'Double/Single Room', 'Apartment', and 'Suite' become concepts and are hierarchically organized under a 'RoomType' concept; the same thing happens to season periods. Also note that concepts 'Price' and 'Cost' share the same synset in WordNet, which is in the ontology described by an equivalent operator (≡) among them. In this way a common ontology covering both frames is generated, which enables annotation and query answering, the later through the knowledge base.

We have explored issues of frame matching for ontology generation in [29], where further information can be found.

5) Knowledge base construction

By constructing multiple domain ontologies, where each one corresponds to a particular topic/genre cluster, the final step of our algorithm is evident in a construction of a knowledge base. The knowledge base is created by formalizing the table contents according to the newly generated formal structures. Together with the inference engine it will be used for providing replies to user queries.

Steps 2, 3, 4, and 5 are all implemented within a system named TARTAR (Transforming ARbitrary TAbles into fRames) as part of the PhD dissertation [25] and later extended for ALVIS. Further information regarding TARTAR can be found in [27, 30], while the open source code is freely available and can be downloaded at `http://dis.`

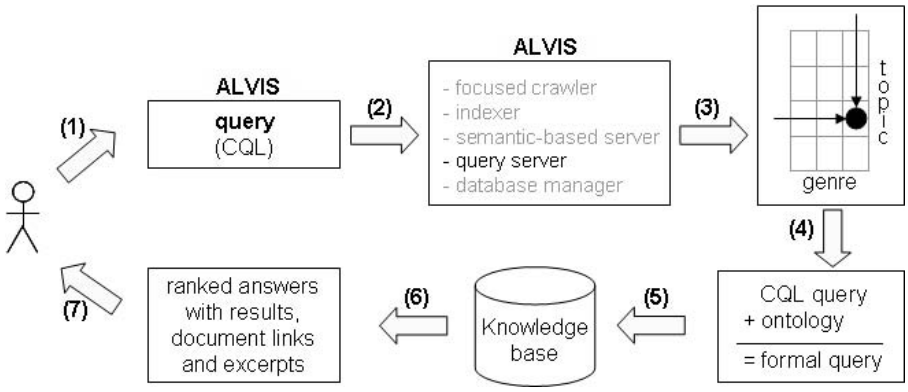

Figure 6: A seven-step user query processing schema.

ijs.si/sandi/work/tartar/.

Each of these algorithmic steps is important in the overall process and benefits search in terms of accuracy and reduced complexity.

2.2 Tabular structures querying

Many search systems permit the query to contain single words, phrases, and word inclusions and exclusions. Instead of exclusion, a safer way is to rate each document for how likely it is to satisfy the user's information need, to sort in decreasing order of this score, and present the results in the ranked list. Search engines differ in how they perform vector-space ranking.

Our motivation is to extend the ALVIS search engine with our tabular structure handling approach, where formal knowledge models and semantics decrease ambiguity and hence better determine similarity among a query and a document. In addition, they provide semantically relevant replies to user queries.

The execution of a user query is performed along the following seven steps, as shown in Figure 6 (note the corresponding numbers):

1. *Querying*: users provide queries using CQL [1] (Common Query Language). CQL is a formal language for representing queries to information retrieval systems such as Web indexes, bibliographic catalogs and museum collection information and has been adopted in ALVIS. The CQL design objective is that queries be human readable and human writable, and that the language be intuitive while maintaining the expressiveness of more complex languages such as SQL. Simplicity and intuitiveness of expression is achieved by combining with the richness of Z39.50's type-1 query [32]. This process corresponds to the 'Query requests' box in Figure 2.

2. *Resolution*: the ALVIS query server resolves the CQL query and extracts its key terms. This step corresponds to the 'Distributed search engine' box in Figure 2.

3. *Categorization*: extracted key terms are used to determine potentially interesting categories, topics and genres in particular, where detailed search should be executed. Categorization is perform as described in subsection 2.1 1) and is utilized by Mladenić and colleagues [22] and Bratko and colleagues [5]. The purpose of searching only within certain topics and genres is to significantly narrow the search space and consequently speed up the whole replying process, which is of particular importance when querying the knowledge base of tabular structures. This corresponds to the left-upper bold box (Full test index ...) in Figure 2.

4. *Mapping*: as described in subsection 2.1 (4), each cluster possesses its own automatically generated ontology. The ontology concepts and relations are used together with the extracted key terms in order to reformulate posted CQL queries into new, formal queries. These queries are encoded in the F-Logic formal language, same as the knowledge base. This corresponds to the TARTAR component in Figure 2.

5. *Inference*: The new formal queries, obtained in a previous step, are posted against the knowledge base using an appropriate inference engine, which returns plausible results. This part has been presented in the paper by Pivk and colleagues [30].

6. *Ranking*: ALVIS ranking component ranks, according to its ranking function, the results obtained from the knowledge base with the set of documents that match the CQL query. The ranked results are sent to the output creation module.

7. *Results*: ranked results and document excerpts are gathered and incorporated into a document, which is returned to the user. Note that this document involves also the results inferred from the knowledge base, which represent semantically aware answers to a posted query.

Here we present processing of two simple example queries that relate to the content of tables shown in

Figure 1 and 5. For simplicity, we will first present a query in a natural language, followed by individual step descriptions. Natural language interface is also being developed, but not directly as part of the ALVIS system. The two demonstrating examples are:

A) SHOW THE ACCOMMODATIONS THAT OFFER APARTMENTS.

1. Q="`accommodation=* and type=apartment`";
2. Key terms of Q, such as accommodation, type, and apartment would determine possible categories, i.e. "tourism:skiing:slovenia", and a genre, such as "commercial/shopping".
3. An example of an F-logic query where for mapping purposes the ontology shown in Figure 5 is used:

   ```
   FORALL S,L <- EXISTS A,T,P
   A:StructureCost[Cost@(S,L,
   apartment,T)->P].
   ```

4. The query Q posted against the knowledge base returns the following three results gained from the table examples:

   - S=Bungalow GABER, L=Rogla;
   - S=Pension MARTIN, L=Pohorje;
   - S=Hotel ALPINA, L=?;

5. - 7. The results are ranked and returned to the user as a Web document, as shown in Figure 7. Some results are upgraded with tables, where a table consists of all semantically relevant facts found in the particular document. These results are in the output document marked as 'Reply #' link, where # represents the relevance number of a document introducing the facts. Such examples can be observed from the first two results in Figure 7.

B) SHOW ALL PRICES OF DOUBLE ROOMS IN HOTELS ON POHORJE, SLOVENIA.

1. Q="`price=* and accommodation=hotel and hotel=* and place=pohorje and type='double room'`";
2. Holds a similar description as in the previous example;
3. ```
   FORALL S,T,P <- EXISTS A
   A:StructureCost[Cost@(S,pohorje,
   double room,T)->P and
   isa_(S,hotel)].
   ```
   and
   ```
   FORALL S,T,P <- EXISTS A
   A:StructureCost[Cost@(S,pohorje,
   twin bed,T)->P and isa_(S,hotel)].
   ```
4. - S=Hotel AREH, T=Winter, P=840;
   - S=Hotel AREH, T=Rest, P=690;
5. - 7. Similar as in the previous example.

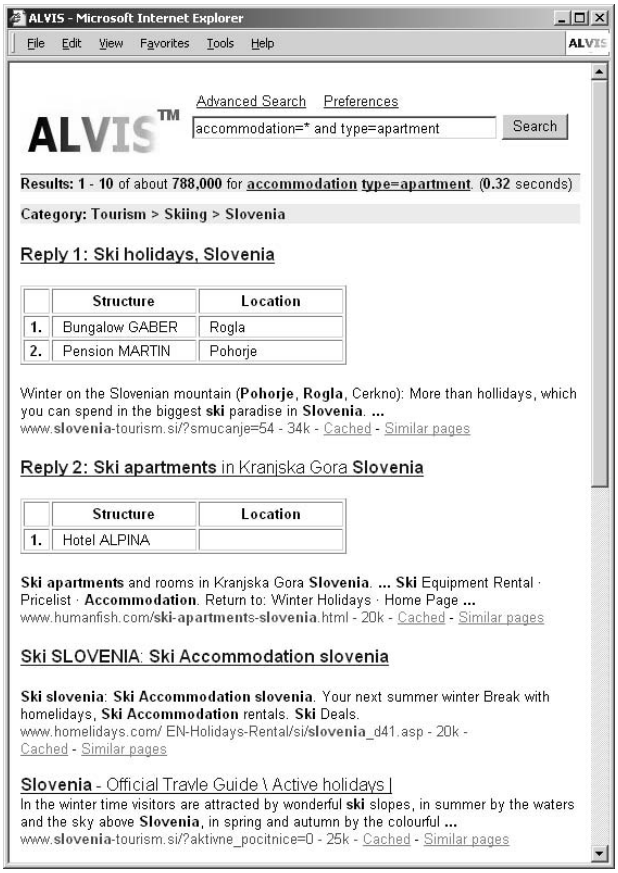

Figure 7: A simulated graphical representation of ranked results, requested in example A).

3 Current state and evaluations

Here we will shortly present the individual state-of-the-art of the described system components and some evaluations. More detailed descriptions are given in the referenced papers.

- *Topic categorization* is based on a hierarchical topic structure. For each topic, a naive Bayesian classifier is built. For an uncategorized document, each of the classifiers estimates the probability of it belonging to a given topic. Experimental evaluation on real world data shows that the proposed approach gives good results. The best performance was achieved with features selected based on a feature scoring measure known from information retrieval called Odds ratio using relatively small number of features [22].
- *Genre categorization*: For this purpose, we created a dataset of over 1500 Web pages manually categorized into 20 genres. Each genre has at least 70 examples. Work on classification is still in progress, but initial tests have been performed by Bratko and colleagues [5] using prediction by partial matching (PPM) classification algorithm [10], which proved very successful in related problem setting. Upon completion of the work, the dataset will be made publicly
   ```

available.

- *Filtering of proper tabular structures*: To solve this problem, we adopted a solution developed by Wang et al. [34], which is based on machine learning techniques, as presented in section 2.1 (2). The classification accuracy reaches almost 95%, meaning that nearly no human intervention is required [25, 26].
- *Formalization of tabular structures*: The empirical evaluation is performed from efficiency and applicability perspectives, where the dataset consisted of over 700 test tables, belonging to two different domains, $t$ourist and $g$eopolitical. First, the efficiency E of the approach is measured according to the portion of correctly transformed tables into knowledge frames reaching 84.52% ($E = \frac{1}{2}(E_t + E_g) = \frac{1}{2}(\frac{289}{369} + \frac{313}{345})$). The more detailed results description is omitted due to its complexity and the lack of space but can be found in [26, 25]. Second, the applicability is shown by querying the content of tables encoded in the knowledge base, where it is shown that returned answers are true and complete in 100% of cases [30].
- *Automatic ontology generation*: Initial tests of the ontology generation process have been shown in [29], indicating that the process is feasible but pointing out that the conflict resolution still needs some improvements. Further tests are still in progress.
- *Knowledge base construction*: This step cannot be evaluated separately since it depends on the results of previous two tasks and merely formalizes the data found within tables. Anyway, the study in [30] showed that by querying the content of tables encoded in the knowledge base, the returned answers are true and complete in all cases.

The system consisting of the described parts is not yet fully implemented. While individual parts have already been tested and achieved the expected performance, the functionality of the integrated system is yet to be verified.

4 Related work

In this paper we have covered several important topics of modern information handling, which can be basically split into three main areas: document categorization, ontology learning, and analysis of tabular structures. We will not present each of these areas thoroughly, since these are given in respective papers, but will rather show their main achievements.

An important portion of information retrieval [3, 31] is focused to the document categorization tasks, where several different techniques and methods have been used, ranging from machine learning techniques such as Bayesian learning, SVM, or (hierarchial) clustering, to natural language processing, graph theory, and combination of these approaches.

Genre categorization is most commonly performed by parsing the documents to extract a set of stylistic features, which vary significantly from author to author, and then using one of the numerous classification algorithms: regression, SVM, decision trees, naive Bayes, clustering. Less common are character-base methods, which use raw text as the input. The best classification accuracy in this group is shown by Peng et al. [24] reaching 86% when classifying documents into 10 different genres. Prediction by partial matching (PPM), the method which we are going to employ, also belongs to this group. Finally, there are visual methods, but these are typically used on scanned documents represented as bitmaps and are not well suited for use in a search engine.

A very recent systematic overview of related work on table recognition, transformation, and inferences can be found in [36]. Most of the work in this area has been done on table detection and recognition addressing several types of document encodings, mostly plain text files, images, and HTML documents. Work performed on textual tables and images was mainly oriented towards table detection, row labeling, and cell classification [36]. Work on HTML tables was extended to indexing relation detection, cell/row/table merging or splitting, classification [34], and other approaches aiming at the deep understanding of table structure [36]. Table extraction methods have also been applied in the context of question answering and ontology learning. A similar proposal to ours was introduced by Tijerino et al. [33]. They presented only a vision for a system that would be able to generate ontologies from arbitrary tables or table-equivalents. Their approach consists of a four-step methodology which includes table recognition and decomposition, construction of mini ontologies, discovery of inter-ontology mappings, and merging of mini-ontologies. For the purpose of semantics discovery the approach is multifaceted, meaning they use all evidence at their disposal (i.e. Wordnet, data frames, named entities, etc.). Since the paper, unlike ours, presents only a vision, no evaluation is provided.

5 Conclusion and Future Work

ALVIS represents a major attempt to create a semantic search engine. For this purpose, it introduces several novel approaches and mechanisms. Our approach is novel in the sense that it is the first to address the whole process of semantically aware information discovery and query answering by analyzing information presented in tabular structures. By putting together the components, such as two-aspect document categorization, detection, filtering, transformation and formalization of tabular structures, a search engine gets enhanced with new capabilities in a scalable way.

Each of the major formalization algorithmic steps is important in the overall process and benefits in terms of accuracy and reduced complexity. We anticipate that the system

will also benefit in terms of scalability and speed, but cannot show or discuss that in details since it has not been fully implemented and tested yet.

Regarding success rate of HTML input tabular transformations into formal semantic descriptions, the proposed approach reaches nearly 85%. Besides the transformation the approach also enables annotation of the original resources with generated formal descriptions.

Although tabular semantic approach by itself may not be the ultimate research goal, and although its success relies on specific properties of tables, i.e. implicit relationships, it might give some indications regarding semantic search from plain text. We are aware that our approach and the ontology learning text processing approaches tackle and cope with different problems, thus making direct comparison nearly impossible, we still assume that our results show great potential for wide acceptance.

Future research in the tabular structure analysis area is aiming into two main directions. Firstly, the focus will be to incorporate some machine learning techniques, in particular for: (a) the classification of tabular structures into the (predefined) table type classes, and (b) the improved and more scalable discovery and division of tables into logical units and regions. This will enable better grounds for the ongoing transformation process. Secondly, our methodology is domain and document type independent, but has been implemented to cover HTML tables only. Therefore two important extensions need to be provided in the future: (a) the extensions to cover other document types, i.e. PDF, excel, etc., and (b) to enable a framework for a simple inclusion of the other domain-dependent knowledge models. With these extensions comprehended, we anticipate that the approach will be used to semantically enrich a variety of legacy data and will support the conversion of the existing Web into a Semantic Web.

Acknowledgement

This work has been supported by the EU IST-projects ALVIS (Superpeer Semantic Search Engine, IST-1-002068-STP) and SEKT (Semantically Enabled Knowledge Technologies, IST-2004-506826). The research was also supported by Slovenian Ministry of Education, Science and Sport, and by Marie Curie Fellowship of the European Community program 'Host Training Sites', mediated by FZI and AIFB at University of Karlsruhe, Germany. Thanks to all our colleagues for participating in the evaluation of the system as well as to the reviewers for useful comments on the paper.

References

[1] Z39.50 International Maintenance Agency. CQL - Common Query Language, 2004. `http://www.loc.gov/z3950/agency/zing/cql/`.

[2] A. Antonacopoulos and J. Hu. *Web Document Analysis: Challenges and Opportunities*. World Scientific, 2004.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[4] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 2001(5), 2001.

[5] A. Bratko and B. Filipic. Exploiting structural information for semi-structured document categorization. *Information Processing & Management*, 42(3):679–694, 2006.

[6] P. Buitelaar, P. Cimiano, and B. Magnini, editors. *Ontology Learning from Text: Methods, Applications and Evaluation*, volume Frontiers in Artificial Intelligence and Applications. IOS Press, 2005.

[7] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kauffman, 2002.

[8] H. Chen, S. Tsai, and J. Tsai. Mining tables from large scale HTML texts. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, pages 166–172, 2000.

[9] P. Cimiano, L. Schmidt-Thieme, A. Pivk, and S. Staab. Learning taxonomic relations from heterogeneous evidence. In P. Buitelaar, P. Cimiano, and B. Magnini, editors, *Ontology Learning from Text: Methods, Applications and Evaluation*, pages 56–71. IOS Press, 2005.

[10] John G. Cleary and Ian H. Witten. A comparison of enumerative and adaptive codes. *IEEE Transactions on Information Theory*, 30(2):306–315, 1984.

[11] C. Fellbaum. *WordNet, an electronic lexical database*. MIT Press, 1998.

[12] W.B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. Prentice Hall, 1992.

[13] A. Gionis, P. Indyk, and R. Motwani. Similarity Search in High Dimensions via Hashing. In *VLDB*, pages 518–529, 1999.

[14] J. Hu, R. Kashi, D. Lopresti, G. Nagy, and G. Wilfong. Why table ground-truthing is hard? In *Proceedings of the 6th International Conference on Document Analysis and Recognition*, pages 129–133, 2001.

[15] M. Hurst. Layout and language: Beyond simple text for information interaction - modelling the table. In *Proceedings of the 2nd International Conference on Multimodal Interfaces*, 1999.

[16] M. Hurst. Layout and language: Challenges for table understanding on the web. In *Proceedings of the International Workshop on Web Document Analysis*, pages 27–30, 2001.

[17] B. Jansen, A. Spink, and J. Bateman. Searchers, the subjects they search, and sufficiency: A study of a large sample of excite searchers. In *Proceedings of the World Conference on the WWW, Internet and Intranet (WebNet-98)*, pages 913–928. AACE Press, 1998.

[18] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42:741–843, 1995.

[19] J.M. Kleinberg. Two Algorithms for Nearest-neighbor Search in High Dimension. In *ACM Symposium on Theory of Computing*, pages 599–608, 1997.

[20] J.M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632, 1999.

[21] A. Maedche. *Ontology Learning for the Semantic Web*. Kluwer Academic Publishers, 2002.

[22] D. Mladenič and M. Grobelnik. Feature Selection on Hierarchy of Web Documents. *Decision Support Systems*, 35:45–87, 2003.

[23] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998. `http://www-db.stanford.edu/~backrub/pageranksub.ps`.

[24] F. Peng, S. Dale, and W. Shaojun. Language and Task Independent Text Categorization with Simple Language Models. In *Proceedings of the Human Language Technology Conference of the ACL*, pages 189–196, Edmonton, Canada, 2003.

[25] A. Pivk. *Automatic Generation of Ontologies from Web Tabular Structures*. PhD Thesis (in Slovene), University of Maribor, Slovenia, 2005.

[26] A. Pivk. Automatic Ontology Generation from Web Tabular Structures. *AI Communications*, 19(1):83–85, 2006.

[27] A. Pivk, P. Cimiano, and Y. Sure. From Tables to Frames. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2–3):132–146, 2005.

[28] A. Pivk and M. Gams. Domain-dependant information gathering agent. *Expert Systems with Applications*, 23:207–218, 2002.

[29] A. Pivk and M. Gams. Construction of domain ontologies from tables. In *Proceedings of the 8th International Multi-Conference on Information Society (IS'05)*, pages 615–619, 2005.

[30] A. Pivk, Y. Sure, P. Cimiano, M. Gams, V. Rajkovic, and R. Studer. Transforming Arbitrary Tables into Logical Form with TARTAR. *Data & Knowledge Engineering*, accepted, 2006.

[31] F. Sebastiani, editor. *Advances in Information Retrieval*, Proceedings of the 25th European Conference on IR Research (ECIR 2003), Lecture Notes in Computer Science, Vol. 2633, Pisa, Italy, April 14-16, 2003. Springer.

[32] M. Taylor and M. Cromme. Searching very large bodies of data using a transparent peer-to-peer proxy. In *Proceedings of DEXA 2005 Workshop*, pages 1049–1053. IEEE Computer Society, 2005.

[33] Y.A. Tijerino, D.W. Embley, D.W. Lonsdale, and G. Nagy. Ontology generation from tables. In *Proceedings of 4th International Conference on Web Information Systems Engineering (WISE'03)*, pages 242–249, 2003.

[34] Y. Wang and J. Hu. A machine learning based approach for table detection on the web. In *Proceedings of the 11th International Conference on World Wide Web*, pages 242–250. ACM Press, 2002.

[35] I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: Compression and Indexing Documents and Images*. Multimedia Information and Systems. Morgan Kaufmann, 1999.

[36] R. Zanibbi, D. Blostein, and J.R. Cordy. A survey of table recognition: Models, observations, transformations, and inferences. *International Journal of Document Analysis and Recognition*, 7(1):1–16, 2004.