Interactive Topic Graph Extraction and Exploration of Web Content

Günter Neumann and Sven Schmeier

Abstract In the following, we present an approach using interactive topic graph extraction for the exploration of web content. The initial information request, in the form of a query topic description, is issued online by a user to the system. The topic graph is then constructed from N web snippets that are produced by a standard search engine. We consider the extraction of a topic graph to be a specific empirical collocation extraction task, where collocations are extracted between chunks. Our measure of association strength is based on the pointwise mutual information between chunk pairs which explicitly takes their distance into account. This topic graph can then be further analyzed by users so that they can request additional background information with the help of interesting nodes and pairs of nodes in the topic graph, e.g., explicit relationships extracted from Wikipedia or those automatically extracted from additional Web content as well as conceptual information of the topic in form of semantically oriented clusters of descriptive phrases. This information is presented to the users, who can investigate the identified information nuggets to refine their information search. An initial user evaluation shows that our approach is especially helpful for finding new interesting information on topics about which the user has only a vague idea or no idea, at all.

1 Introduction

Today's web search is still dominated by a document-perspective: a user enters one or more keywords that represent the information of interest and receives a ranked

Günter Neumann
German Research Center for Artificial Intelligence GmbH (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, e-mail: neumann@dfki.de

Sven Schmeier
German Research Center for Artificial Intelligence GmbH (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany e-mail: schmeier@dfki.de

list of documents. This technology has been shown to be very successful, because it very often delivers concrete documents or web pages that contain the information the user is interested in. The following aspects are important in this context: 1) Users basically have to know what they are looking for. 2) The documents serve as answers to user queries. 3) Each document in the ranked list is considered independently.

If the user only has a vague idea of the information in question or just wants to explore the information space, the current search engine paradigm does not provide enough assistance for these kind of searches. The user has to read through the documents and then eventually reformulate the query in order to find new information. Seen in this context, current search engines seem to be best suited for "one-shot search" and do not support content-oriented interaction.

In order to overcome this restricted document perspective, and to provide more interactive searches to "find out about something", we want to help users with the web content exploration process in two ways:

1. We consider a user query as a specification of a topic that the user wants to know and learn more about. Hence, the search result is basically a graphical structure of the topic and associated topics that are found.
2. The user can interactively explore this topic graph, in order to either learn more about the content of a topic or to interactively expand a topic with newly computed related topics.

In the first step, the topic graph is computed on the fly from the a set of web snippets that has been collected by a standard search engine using the initial user query. Rather than considering each snippet in isolation, all snippets are collected into one document from which the topic graph is computed. We consider each topic as an entity, and the edges between topics are considered as a kind of (hidden) relationship between the connected topics. The content of a topic are the set of snippets it has been extracted from, and the documents retrievable via the snippets' web links.

The topic graph is then displayed either in a standard Web browser or on a mobile device (in our case an iPad). By just selecting a node, the user can either inspect the content of a topic (i.e, the snippets or web pages) or activate the expansion of the graph through an on the fly computation of new related topics for the selected node.

In a second step, we provide additional background knowledge on the topic which consists of 1) explicit relationships that are either generated from an online Encyclopedia (in our case Wikipedia) or automatically extracted from the web snippets through learned relation extraction rules, and 2) additional conceptual information of the topic in form of semantically oriented clusters of descriptive phrases automatically extracted from the Web.

The background knowledge approaches are based on and driven by specific patterns, e.g., the structure of infoboxes, in case of Wikipedia, predefined seed relations in the case of learned relation extraction rules or predefined patterns like "X is a Y" in the case of concept extraction. In contrast, the topic graph extraction process is much less pattern dependent. This means, that the topic graph extraction component is much more flexible in dealing with dynamic changes of web content, whereas

background knowledge components are much more fixed and stable with respect to possible changes.

This way the user can explore in a uniform way both new information nuggets and background information nuggets interactively. Fig. 1 summarizes the main components and the information flow.

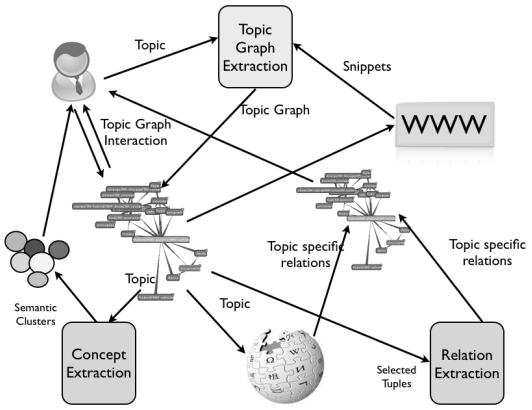

Fig. 1 Blueprint of the proposed system.

By selecting a node from a newly extracted topic graph, the user can request information from new topics on basis of previously extracted information. In such a dynamic information extraction situation, the user expects real–time performance from the underlying technology. The requested information cannot simply be pre–computed, therefore most of the relevant information has to be extracted online relative to the current user request. That is why we assume that the relevant information can be extracted from a search engine's *web snippets* and hence avoid the costly retrieval and processing time for huge amounts of documents. Of course, direct processing of web snippets also poses certain challenges for the Natural Language Processing (NLP) components.

Web snippets are usually small text summaries which are automatically created from parts of the source documents and are often only in part linguistically well–formed, cf. [17]. Thus the NLP components are required to possess a high degree of robustness and run–time behavior to process the web snippets in real–time. Since, our approach should also be able to process Web snippets from different languages, the NLP components should be easily adaptable to many languages. Finally, no restrictions to the domain of the topic should be pre–supposed, i.e., the system should be able to accept topic queries from arbitrary domains. In order to fulfill all these requirements, we are favoring and exploring the use of shallow and highly data-

oriented NLP components. Note that this is not a trivial or obvious design decision, since most of the current prominent information extraction methods advocate deeper NLP components for concept and relation extraction, e.g., syntactic and semantic dependency analysis of complete sentences and the integration of rich linguistic knowledge bases like Word Net.

The paper is organized as follows. In the next section, we illustrate the core concepts of our approach with an operating example. We then compare our work with that of others in section 3. The major components of our approach are described in more detail in the sections 4, 5, and 6. In section 7, we present and discuss the results of an initial user evaluation, Section 8 concludes the paper and outlines some areas for future research.

2 Running Example

A prototype application of our approach has been implemented as a mobile touchable application for online topic graph extraction and exploration of web content. The system has been implemented for operation on an iPad.

The following screenshots show some results for the search query "Justin Bieber" running on the current iPad demo–app. At the bottom of the iPad screen, the user can select whether to perform text exploration from the Web (via button labelled "i–GNSSMM") or via Wikipedia (touching button "i–MILREX"). The Figures 2, 3, 4, 5 show results for the "i–GNSSMM" mode, and Fig. 6 for the "i-MILREX" mode. General settings of the iPad demo-app can easily be changed. Current settings allow e.g., language selection (so far, English and German are supported)[1] or selection of the maximum number of snippets to be retrieved for each query. The other parameters mainly affect the display structure of the topic graph.

An entity can be ambiguous, i.e., two (or more) individuals can be referred to by the same name. For example, the name "Jim Clark" can refer to the well–known car racing driver or to the founder of Netscape but it can refer to even more individuals. Proper disambiguation of named entities is still a difficult problem that has not been resolved.[2]

So far, the extracted topic graph does not provide direct help for NE disambiguation, i.e., it is possible that the topic graph merges information from different individuals which cannot be easily distinguished. As an initial approximate solution towards NE disambiguation in the context of our approach, we provide an additional operator which we call the concept extractor (CE). It receives the label of a triggered (touched) node and searches the Web for descriptive phrases. All descriptive phrases that are found are then clustered on the basis of latent semantic similarity. It is possible (although not in all cases) to induce from the resulting clusters whether a NE

[1] Actually, both languages are only supported in the i–GNSSMM mode. In the case of the i–MILREX mode, we currently only support the English Wikipedia.

[2] Consult, for example, the web page http://nlp.uned.es/weps/ for more information about the problem space.

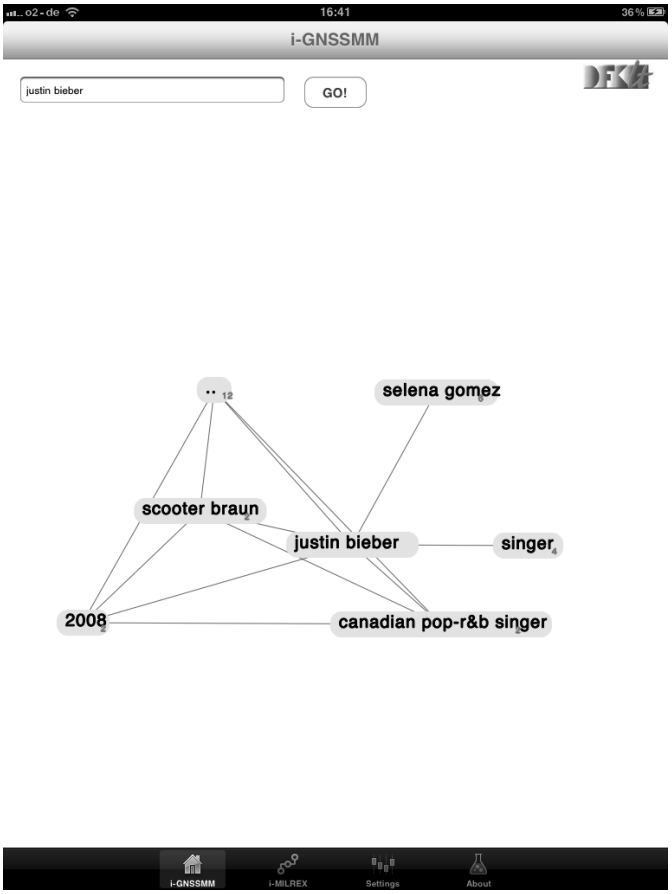

Fig. 2 The topic graph computed from the snippets for the query "Justin Bieber". The user can double touch on a node to display the associated snippets and web pages. Since a topic graph can be very large, not all nodes are displayed. Nodes, which can be expanded are marked by the number of hidden immediate nodes. A single touch on such a node expands it, as shown in Fig. 3. A single touch on a node that cannot be expanded adds its label to the initial user query and triggers a new search with that expanded query.

might refer to several individuals. Here is the output of this module for the name "Jim Clark" (we are using this example, because it nicely illustrates our approach; note that the clusters and their labels are computed completely automatically and unsupervised, see sec. 5 for more details):

———— Cluster [PRESIDENT]————-

Jim Clark, who is the president of the Arizona Western Heritage Foundation, did a fantastic job of putting the whole event together.

———— Cluster [MOTOR]————-

1963: Scotland's Jim Clark became the youngest world motor racing champion

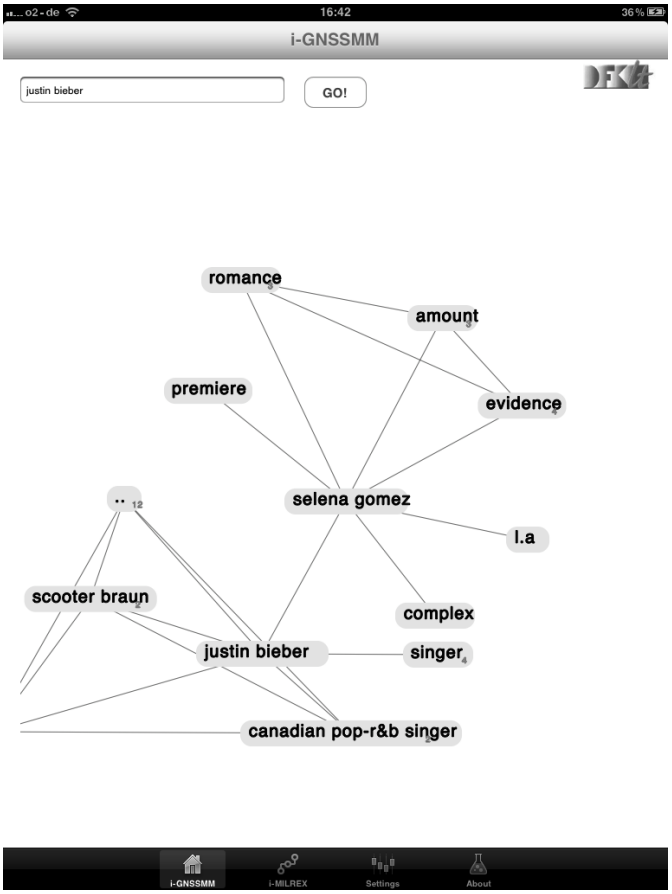

Fig. 3 The topic graph from Fig. 2 has been expanded by a single touch on the node labelled "selena gomez". Double touching on that node triggers the display of associated web snippets (Fig. 4) and the web pages (Fig. 5).

———— Cluster [INDIANAPOLIS]————————-

1965 - Jim Clark becomes first foreigner in 49 years to win Indianapolis 500 car race

———— Cluster [GRAPHICS]————————-

Andreessen partnered with Jim Clark, who a decade earlier had graduated from Stanford University in California to start up Silicon Graphics.

———— Cluster [GRAND]————————-

Jim Clark became world champion grand prix driver in 1963 and 1965 and was the first non-American to win the Indianapolis 500 for nearly 50 years.

Jim Clark was a driver of such towering ability that he had few serious rivals during his seven years in grand prix racing.

———— Cluster [FORMULA]————————-

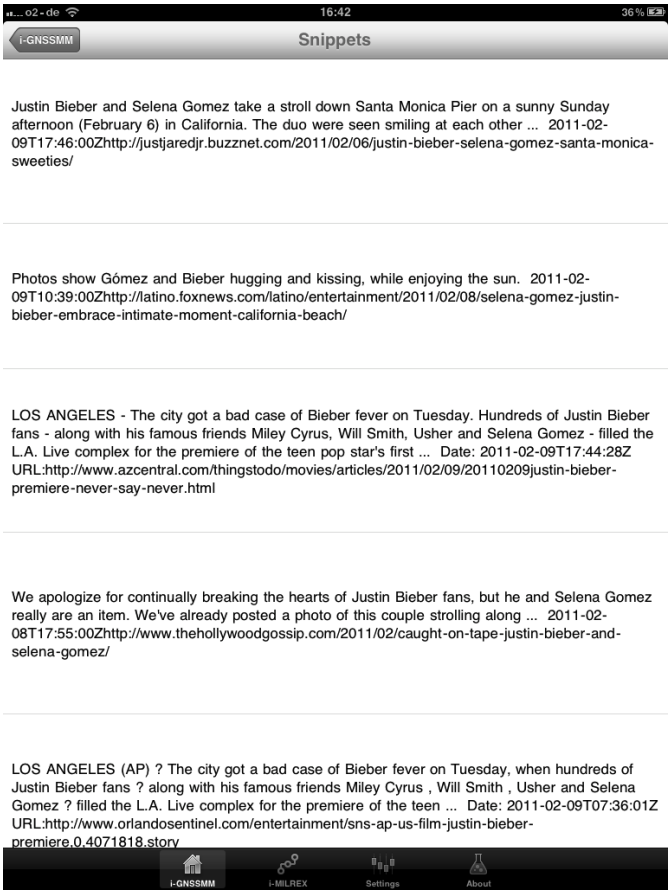

Fig. 4 The snippets that are associated with the node label "selena gomez" of the topic graph from Fig. 3.In order to go back to the topic graph, the user simply touches the button labeled iGNSSMM on the left upper corner of the iPad screen.

> Jim Clark was a racing legend who racked up two Formula One World Championships, won some 25 Grand Prix races and even competed in NASCAR before his untimely death in 1968.
>
> ———— Cluster [DRIFT]————-
>
> Jim Clark was a great exponent of the three wheel drift in his Lotus Cortina, campaigning the Lotus to win the British Touring Car Championship in 1964.

Although this initial clustering does not actually disambiguate named entities, it already provides hints that there are several different individuals named "Jim Clark", viz. the president of the Arizona Western Heritage Foundation, the Netscape founder (as a start up of Silicon Graphics) and the racing car driver. A major problem so far is that often "senses" are distributed across different clusters (e.g., in the case of the

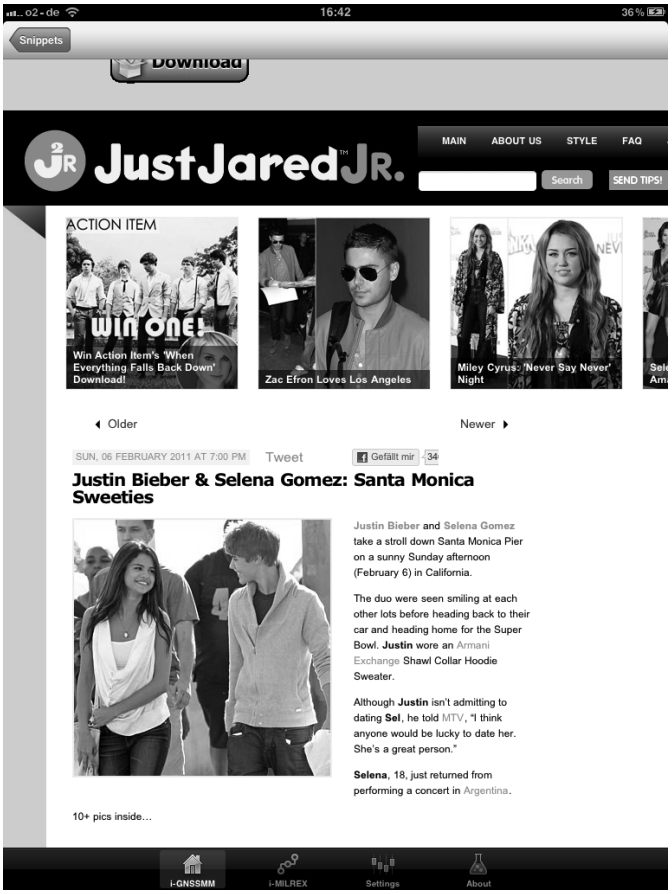

Fig. 5 The web page associated with the first snippet of Fig. 4. A single touch on that snippet triggers a call to the iPad browser in order to display the corresponding web page. The left upper corner button labeled "Snippets" has to be touched in order to go back to the snippets page.

racing car driver), so that a clear identification of the possible senses is not currently possible.

One last step in our system is the discovery of hidden relations between concepts (see Fig. 6)[3]. The Relation Extraction (RE) component can extract such information and add it to the topic graph by labeling its edges. Currently, it uses background knowledge from Wikipedia infoboxes. For missing relationships, i.e., if the desired relations are not contained in the infoboxes, we analyze the previously retrieved snippets based on relation extraction models that have already been learnt. Note that the component RE is the only module that requires some offline effort by the user, namely a small set of domain-independent relations and a small relation hierarchy,

[3] The screenshots shows relations retrieved from Wikipedia infoboxes only. The component for detecting missing relationships is not yet integrated in the running system

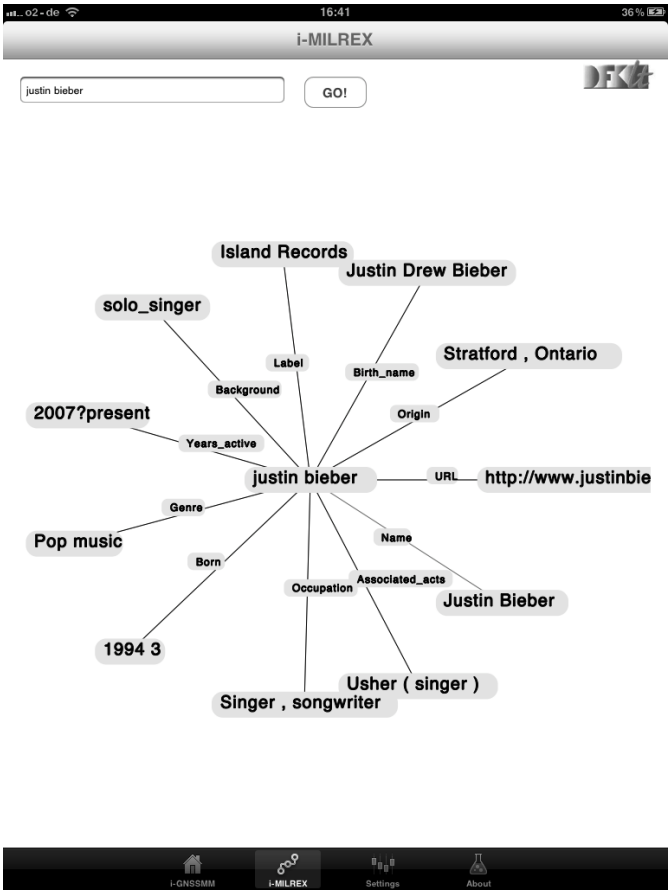

Fig. 6 If mode "i–MILREX" is chosen then text exploration is performed based on relations computed from the info–boxes extracted from Wikipedia. The central node corresponds to the query. The outer nodes represent the arguments and the inner nodes the predicate of a info–box relation. The center of the graph corresponds to the search query.

see section 6. However, since we are using a minimally supervised machine learning component equipped with sophisticated inference mechanisms, the integration of new relations into the systems is simple and can be done by non-linguists.

3 Related Work

Our approach is unique in the sense that it combines interactive topic graph extraction and exploration with recently developed technology from text mining, informa-

tion extraction and question answering methods. As such, it learns from and shares ideas with other search results. The most relevant ones are briefly discussed below.

Collocation Extraction We consider the extraction of a topic graph as a specific *empirical collocation extraction task*. However, instead of extracting collocations between words, which is still the dominating approach in collocation extraction research (e.g., [2]), we are extracting collocations between chunks, i.e., word sequences. Furthermore, our measure of association strength takes into account the distance between chunks and combines it with the PMI (pointwise mutual information) approach [24].

Concept Extraction During the last years, the problem of finding definitions for specific concepts (the *definiendum*) has been addressed by Question Answering Systems (QASs) in the context of the Text REtrieval Conference (TREC) and the Cross Language Evaluation Forum (CLEF). In TREC, QASs answer definition questions in English, such as "*What is a quasar?*", by extracting as far as possible non-redundant descriptive information ('nuggets') about the *definiendum* from the AQUAINT corpus.

In order to discover definition utterances, definition QASs usually align sentences with surface patterns in the target corpus at the word and/or the part-of-speech level [14]. Hence, the probability of matching sentences increases as the size of the target collection grows, and accordingly, performance improves substantially [15]. Along with surface patterns, definition QASs take advantage of wrappers around online resources, WordNet glossaries and Web snippets [5]. In addition, QASs, like Google, have also shown that definition Web–sites are a fertile source of descriptive information in English, concretely, in providing answers to 42 out of 50 TREC–2003 questions [5]. However, Web snippets have not yet been proven to be a valuable source of descriptive phrases.

QASs usually tackle redundancy by: (a) randomly removing one sentence from every pair that shares more than 60% of their terms [14], or (b) filtering out candidate sentences by ensuring that their cosine similarity to all previously selected utterances is below a certain threshold. It is also worth while to remark that definition QASs have not yet made effort to deal with disambiguation of the different senses of the *definiendum*.

Web Information Extraction Web Information Extraction (WIE) systems have recently been able to extract massive quantities of relational data from online text. The most advanced technologies are algorithmically based on Machine Learning methods taking into account different granularities of linguistic feature extraction, e.g., from PoS-tagging to full parsing. The underlying methods of the learning strategies for these new approaches can range anywhere from supervised or semi-supervised to unsupervised. Currently, there is a strong tendency towards semi-supervised and, more recently, unsupervised methods.

For example, [20] presents an approach for unrestricted relation discovery that is aimed at discovering all possible relations from texts and presents them as tables. [21] has further developed this approach to what he calls on-demand information extraction . Major input to the system is topic based in form of keywords that are used to crawl relevant Web pages. The system then uses dependency parsing as a major

tool for identifying possible relational patterns. The candidate relation instances are further processed by specialized clustering algorithms. A similar approach has been developed by [7] who further combines this approach with advanced user interaction.

Another approach of unsupervised IE has been developed by Oren Etzioni and colleagues, cf. [1]; [8]; [25]. They developed a range of systems (e.g., KnowItAll, Textrunner, Resolver) aimed at extracting large collections of facts (e.g., names of scientists or politicians) from the Web in an unsupervised, domain-independent, and scalable manner. In order to increase performance, specific Machine Learning based wrappers have been proposed for extracting subclasses, lists, and definitions.

Relation Extraction The bottleneck of Etzionis and his colleagues work is that they focus on the extraction of unary relations, although they claim these methods should also work in relations with greater arity. In a recent paper, [19] presented URES, an unsupervised Web relation extraction system, that is able to extract binary relations, on a large scale, e.g., CEO_of, InventorOf, MayorOf reporting precision values in the upper 80ies. Furthermore, [6] presents a method that is able to handle sparse extractions.

Bunescu and Mooney [4] propose binary relation extraction based on Multiple Instance Learning (MIL). The process starts with some positive and negative instances of a relation, retrieves documents or snippets matching those instances and builds positive and negative training sets for a MIL algorithm. The generated model is then used to classify whether a text contains the relation or not. Our RE component (cf. section 6) is based on a similar idea but with a MIL algorithm specialized on extracting relations using snippets and extensions to n-ary relations. Systems for extracting n-ary relations usually use parsers in combination with bootstrapping methods. See for example, the approaches presented by Greenwood and Stevenson[12], Sudo et al. [23], McDonald et al. [18].

4 Topic Graph Extraction

The core idea of our topic graph extraction method is to compute a set of chunk–pair–distance elements for the N first web snippets returned by a search engine for the topic Q, and to compute the topic graph from these elements.[4] In general for two chunks, a single chunk–pair–distance element stores the distance between the chunks by counting the number of chunks in–between them. We distinguish elements which have the same words in the same order, but have different distances. For example, (Peter, Mary, 3) is different from (Peter, Mary, 5) and (Mary, Peter, 3).

We begin by creating a document S from the N-first web snippets so that each line of S contains a complete snippet. Each textline of S is then tagged with Part–of–Speech using the SVMTagger [13] and chunked in the next step. The chunker

[4] For the remainder of the paper N=1000. We are using Bing (http://www.bing.com/) for Web search.

recognizes two types of word chains. Each chain consists of longest matching sequences of words with the same PoS class, namely noun chains or verb chains, where an element of a noun chain belongs to one of the extended noun tags[5], and elements of a verb chain only contains verb tags. We finally apply a kind of "phrasal head test" on each identified chunk to guarantee that the right–most element only belongs to a proper noun or verb tag. For example, the chunk "a/DT british/NNP formula/NNP one/NN racing/VBG driver/NN from/IN scotland/NNP" would be accepted as proper NP chunk, where "compelling/VBG power/NN of/IN" is not.

Performing this sort of shallow chunking is based on the assumptions: 1) noun groups can represent the arguments of a relation, a verb group the relation itself, and 2) Web snippet chunking needs highly robust NL technologies. In general, chunking crucially depends on the quality of the embedded PoS tagger. However, it is known that PoS tagging performance of even the best taggers decreases substantially when applied on web pages [11]. Web snippets are even harder to process because they are not necessary contiguous pieces of texts, and usually are not syntactically well-formed paragraphs due to some intentionally introduced breaks (e.g., denoted by ... betweens text fragments). On the other hand, we want to benefit from PoS tagging during chunk recognition in order to be able to identify, on the fly, a shallow phrase structure in web snippets with minimal efforts. The assumption here is that we can tolerate errors caused by the PoS tagger because of the amount of redundancies inherent in the web snippets.

The chunk–pair–distance model is computed from the list of noun group chunks.[6] This is done by traversing the chunks from left to right. For each chunk c_i, a set is computed by considering all remaining chunks and their distance to c_i, i.e., $(c_i, c_{i+1}, dist_{i(i+1)})$, $(c_i, c_{i+2}, dist_{i(i+2)})$, etc. We do this for each chunk list computed for each web snippet. The distance $dist_{ij}$ of two chunks c_i and c_j is computed directly from the chunk list, i.e., we do not count the Position of ignored words lying between two chunks.

The motivation for using chunk–pair–distance statistics is the assumption that the strength of hidden relationships between chunks can be covered by means of their collocation degree and the frequency of their relative Positions in sentences extracted from web snippets; cf. [10] who demonstrated the effectiveness of this hypothesis for web–based question answering. We are also making use of chunk–pair–distance statistics in the concept extractor, see sec. 5.

Finally, we compute the frequencies of each chunk, each chunk pair, and each chunk pair distance. The set of all these frequencies establishes the The chunk–pair–distance model CPD_M. It is used for constructing the topic graph in the final step. Formally, a topic graph $TG = (V, E, A)$ consists of a set V of nodes, a set E of edges, and a set A of node actions. Each node $v \in V$ represents a chunk and is

[5] Concerning the English PoS tags, "word/PoS" expressions that match the following regular expression are considered as extended noun tag: "/(N(N|P))|/VB(N|G)|/IN|/DT". The English Verbs are those whose PoS tag start with VB. We are using the tag sets from the Penn treebank (English) and the Negra treebank (German).

[6] Currently, the main purpose of recognizing verb chunks is to improve proper recognition of noun groups. The verb chunks are ignored when building the topic graph.

labeled with the corresponding PoS tagged word group. Node actions are used to trigger additional processing, e.g., displaying the snippets, expanding the graph etc.

The nodes and edges are computed from the chunk–pair–distance elements. Since, the number of these elements is quite large (up to several thousands), the elements are ranked according to a weighting scheme which takes into account the frequency information of the chunks and their collocations. More precisely, the weight of a chunk–pair–distance element $cpd = (c_i, c_j, D_{ij})$, with $D_{i,j} = \{(freq_1, dist_1), (freq_2, dist_2), ..., (freq_n, dist_n)\}$, is computed based on PMI as follows:

$$PMI(cpd) = log_2((p(c_i, c_j)/(p(c_i) * p(c_j))) \\ = log_2(p(c_i, c_j)) - log_2(p(c_i) * p(c_j))$$

where relative frequency is used for approximating the probabilities $p(c_i)$ and $p(c_j)$. For $log_2(p(c_i, c_j))$ we took the (unsigned) polynomials of the corresponding Taylor series[7] using $(freq_k, dist_k)$ in the k-th Taylor polynomial and adding them up:

$$PMI(cpd) = (\sum_{k=1}^{n} \frac{(x_k)^k}{k}) - log_2(p(c_i) * p(c_j))$$

$$, where\ x_k = \frac{freq_k}{\sum_{k=1}^{n} freq_k}$$

The visualized topic graph TG is then computed from a subset $CPD'_M \subset CPD_M$ using the m highest ranked cpd for fixed c_i. In other words, we restrict the complexity of a TG by restricting the number of edges connected to a node.

5 Concept Extraction

By Concept Extraction (CE for short), we mean the identification and clustering of descriptive sentences for the node of a topic graph that has been selected by the user. The particular approach that we are following is based on our own work on searching for definitional answers on the Web, using surface patterns, cf. [9]. We consider CE to be a member of the actions A_n associated to a node n with label w_n (a word or word group) such that CE is automatically evaluated with the input "define:w_n" which can be paraphrased as the definition question: "What is w_n ?".

We will now summarize the major steps exploited by CE, see [9] for more details. CE aims at finding answers to definition questions from Web snippets. The major advantages are that it: (a) avoids downloading full-documents, (b) does not need specialized wrappers that extract definition utterances from definitional Web–sites, and (c) uses the redundancy provided by Web snippets to check whether the information is reliable or not. CE achieves these goals by rewriting the query in

[7] In fact we used the polynomials of the Taylor series for $ln(1+x)$. Note also that k is actually restricted by the number of chunks in a snippet.

such a way that it markedly increases the probability of aligning well-known surface patterns with web snippets. Matched sentences are therefore ranked according to three aspects: (a) the likelihood of words to belong to a description, (b) the likelihood of words to describe definition facets of the word being defined, and (c) the number of entities in each particular descriptive sentence. For this ranking purpose, CE takes advantage of a variation of Multi-Document Maximal Marginal Relevance and distinguishes descriptive words by means of Latent Semantic Analysis (LSA), cf. [16].

5.1 Potential Sense Identification

An important feature of CE is a module that attempts to group descriptive utterances by potential senses, checking their correlation in the semantic space supplied by LSA. Note that this means that CE can also be used for disambiguation of the concept in question by clustering the extracted facts according to some hidden semantic relationship. Currently, we are assuming that final disambiguation is done by the user, but we are also exploring automatic methods, e.g., by taking the complete topic graph into account.

There are many-to-many mappings between names and their concepts. On the one hand, the same name or word can refer to several meanings or entities. On the other hand, different names can indicate the same meaning or entity. To illustrate this, consider the next set S of descriptive utterances recognized by the system:

1. John Kennedy was the 35th President of the United States.
2. John F. Kennedy was the most anti-communist US President.
3. John Kennedy was a Congregational minister born in Scotland

In these sentences, "*US President John Fitzgerald Kennedy*" is referred to as "*John Kennedy*" and "*John F. Kennedy*", while "*John Kennedy*" also indicates a Scottish congregational minister. In the scope of this work, a *sense* is one meaning of a word or one possible reference to a real-world entity.

CE disambiguates senses of a topic δ by observing the correlation of its neighbors in the reliable semantic space provided by LSA. This semantic space is constructed from the term-sentence matrix M (by considering all snippets as a single document, and each snippet as a sentence), which considers δ as a *pseudo-sentence* which is weighted according to the traditional *tf-idf*. CE builds a dictionary of terms W from normalized elements in the snippet document S, with uppercasing, removal of html-tags, and the isolation of punctuation signs. Then CE distinguishes all possible unique *n-grams* in S together with their frequencies. The size of W is then reduced by removing *n-grams*, which are substrings of another equally frequent term. This reduction allows the system to speed up the computation of M as UDV' using the *Singular Value Decomposition*. Furthermore, the absence of syntactical information of LSA is slightly reduced by taking strong local syntactic dependencies into account, following our approach described in 4.

5.2 Experiments

A baseline system was implemented in which 300 snippets were retrieved by processing the input query (the topic in question) using the same query processing module as the one used in CE. The baseline splits snippets into sentences and accounts for a strict matching of the topic in question. In addition, a random sentence from a pair that shares more than 60 % of its terms, and sentences that are a substring of another sentence were discarded. The baseline and CE were then tested with 606 definition questions from the TREC 2003/2001 and CLEF 2006/2005/2004 tracks.

Overall, CE consistently outperformed the baseline. The baseline discovered answers to 74% of the questions and CE up to 94%. For 41.25% of the questions, the baseline found one to five descriptive sentences, whereas CE found 16 to 25 descriptive sentences for 51.32% of the questions. More specifically, results show that CE finds nuggets (descriptive phrases) for all definition questions in the TREC 2003 set, contrary to some state-of-the-art methods, which found nuggets for only 84%. Furthermore, CE finds nuggets for all 133 questions in TREC 2001 question set, in contrast with other techniques, which found a top five ranked snippet that conveys a definition for only 116 questions within the top 50 downloaded full documents.

Concerning the performance of the sense disambiguation process, CE was able to distinguish different potential senses for some topic δs, e.g., for "*atom*", the particle–sense and the format–sense. On the other hand, some senses were split into two separate senses, e.g., "*Akbar the Great*", where "*emperor*" and "*empire*" indicated different senses. This misinterpretation is due to the independent co-occurrence of "*emperor*" and "*empire*" with δ, and the fact that it is unlikely that they share common words. In order to improve this, some external sources of knowledge are necessary. This is not a trivial problem, because some δs can be extremely ambiguous like "*Jim Clark*", which refers to more than ten different real-world entities. CE recognized the racing car driver, the Netscape founder and the president of the Arizona Western Heritage Foundation. Independently of that, we found that entities and the correlation of highly closed terms in the semantic space provided by LSA can be important building blocks for a more sophisticated strategy for the disambiguation of δ.

6 Relation Extraction and Background Knowledge

In the previous sections we extracted concepts and collected descriptive information like definitions and further explanations. Now we want to take a closer look at the relationships holding between those concepts, i.e. we want to semantically label the edges of our topic graph.

In our approach we divide the task into two different but complementary steps:

(1) In order to provide query specific background knowledge we make use of Wikipedia's infoboxes. These infoboxes contain facts and important relationships related to articles. We also tested DBpedia as a background source [3]. However, it

turned out that currently it contains too much and redundant information. For example, the Wikipedia infobox for "Justin Bieber" contains eleven basic relations whereas DBpedia has fifty relations containing lots of redundancies. In our current prototype, we followed a straightforward approach for extracting infobox relations: We downloaded a snapshot of the whole English Wikipedia database (images excluded), extracted the infoboxes for all articles if available and built a Lucene Index running on our server. We ended up with 1.124.076 infoboxes representing more than 2 million different searchable titles. The average access time is about 0.5 seconds. Currently, we only support exact matches between the user's query and an infobox title in order to avoid ambiguities. We plan to extend our user interface so that the user may choose different options. Furthermore we need to find techniques to cope with undesired or redundant information (see above). This extension is not only needed for partial matches but also when opening the system to other knowledge sources like DBpedia, newsticker, stock information and more.

(2) In case of important relations missing in the infoboxes we try to extract them from the previously retrieved snippets.[8] As mentioned earlier snippets often do not contain complete sentences but only parts of them or they may contain dots which means parts of the text have been omitted. Hence the relation extraction algorithm should not rely on techniques that require more than very shallow linguistic analysis.

Bunescu and Mooney [4] perform a minimal supervised process for extracting binary relations on snippets (see also section 3). The approach we describe here is an extension to this approach that relies on a new multiple instance learning algorithm specially developed for binary relation extraction and its extension to n-ary relations.

6.1 Binary Relation Extraction

This process starts with some positive and negative instances of a relation created by search queries from a standard search engine like Google or Bing. For the relation *origin(person, country)* one could formulate the following set of positive examples:

Example 1.

- Arnold Schwarzenegger * * * * * * * Austria (1375)
- Albert Einstein * * * * * * * Germany (622)
- Dirk Nowitzki * * * * * * * Germany (323)
- Detlef Schrempf * * * * * * * Germany (311)
- Brigitte Bardot * * * * * * * France (163)

and negative examples could be:

- Berlusconi * * * * * * * Germany (1375)

[8] For "Jim Clark", e.g., wikipedia's infoboxes do not provide information for the relations: birthplace, place_of_death, or cause_of_death.

- Franz Beckenbauer * * * * * * * USA (622)
- Helmut Kohl * * * * * * * England (323)

The numbers in brackets denote the number of snippets found by the search engine - in this case we used the Bing search engine. Unfortunately you cannot count on all positive examples to be really positive for the envisaged relation. For example, Arnold Schwarzenegger could have visited his parents in Austria, so in that case the relation would be *visit(person1, person2, country)*. In the negative results there is probably no snippet expressing a person's origin. Thus for this example we construct 5 bags of "pseudo" positive snippets and label them as positive, and 3 bags of negative snippets which are labeled as negative.

In the next step we apply a new multiple instance learning algorithm optimized for relation extraction to the positive and negative bags. This algorithm is specially designed to work with positive and negative bags of examples including problems which occur because of incomplete knowledge about the labels of single training examples. A bag is labeled positive if at least one example can be labeled positive and negative if all examples in the bag are negative. Our learning algorithm works as follows:

1. For each bag, merge all instances into one document and build a feature vector for this document. The feature vector may consist either of words and tokens occurring in the text or of letter n-grams. A combination of both is also possible.
2. For each feature vector compute the norm according to its size:

$$norm(bag_i) = \sqrt{\sum |x_{k_i}|} \tag{1}$$

   where x_{k_i} *is the* $k-th$ *document in the* $i-th$ *bag*
3. Weight the features in the vectors according to their number of occurrences in the bag:

$$w_{training}(x_{k_i}) = \#x_k \in bag_i \tag{2}$$

   The learning phase has now been completed and no further computations are necessary in this step, thus, making algorithm performance very fast.
4. Compute a leave-one-out calculation to identify false positive examples. This is done by removing such an instance from the feature vector of the bag and classifying it:

   - Build a feature vector of the example to be classified: The preprocessing should be the same as in the learning phase
   - Weight the features according to their relevances in the categories. This gives the mutual information gain for each token and each bag:

$$w'(x_{k_i}) = \max(0, \frac{\#bag_i + 1}{\sum(x_k \in bag_i)}) \tag{3}$$

$$w(x_{k_i}) = w'(x_{k_i}) * w_{training}(x_{k_i}) \tag{4}$$

   - Compute the classification relevances for each bag:

$$\forall\ bag_i\ \ relevance(bag_i) = \sum \frac{w(x_{k_i}) * \kappa_i}{norm(bag_i) * length(x)} \tag{5}$$

where κ_i denotes a smoothing factor for the bags. The

- The classification is done according to the category with the highest relevance

For the set of false positive examples identify the features with the highest relevance as they have the highest impact on the incorrect classification. These are removed from the feature vectors of the positive bags and added to the feature vector of one of the negative bags. Afterwards perform the steps (1) and (2) again.

5. Recompute the leave-one-out calculation using the newly weighted bags and relabel false negative examples, i.e. examples that are labeled as positive but classified as negative. As side effect, the number of false positives inside the positive bags is also reduced automatically.
6. Repeat the process from step (2) until the process converges.

The main properties of this classification method are simplicity, fastness and robustness even in unbalanced or noisy training data. The classification result has reliable confidence values which have been created by computing a final leave one out classification for all data in the training set. Furthermore it allows direct control of the mutual information gain of each feature in the feature space.

The result of the learning process on these bags is a model that can be used to classify unseen snippets and if they are classified as positive, to extract the relation predicted by the model. For the example 3 above, the Precision–Recall graph (cf. Fig. 7) shows our success. The curve shows an F-measure of about 0.76. With this method we have successfully trained several models for different relations (Table 1).

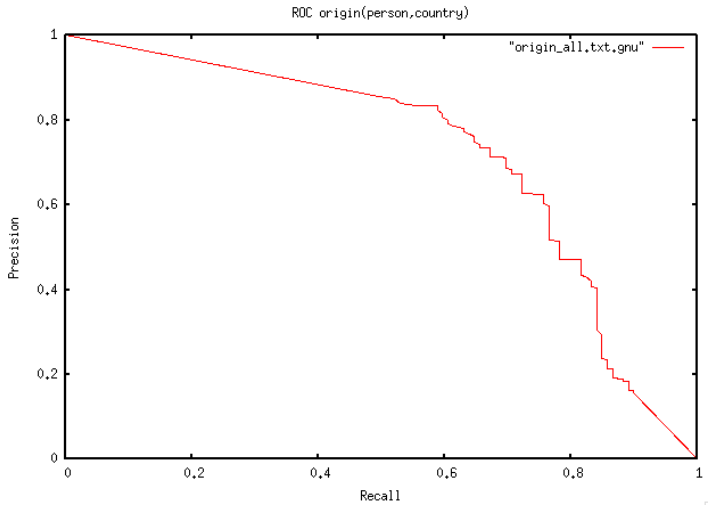

Fig. 7 Precision–Recall graph for origin(person,country)

Table 1 F-Measures for different relations

Relation	# train bags positive	# train bags negative	F–measure
birthplace(person,city)	5 (1490 snippets)	3 (362 snippets)	0.86
origin(person,country)	5 (2794 snippets)	3 (2320 snippets)	0.76
place_of_death(person,city)	5 (2466 snippets)	3 (951 snippets)	0.68
cause_of_death(person,cause)	5 (2533 snippets)	3 (569 snippets)	0.58
married_with(person,person)	4 (388 snippets)	2 (1025 snippets)	0.87
father_of(person,person)	4 (1849 snippets)	3 (939 snippets)	0.72

6.2 Inferring Binary and N-ary Relations

Learning of n-ary relations using the described technique is not possible for quite a few reasons. The main problem in learning n-ary relations, especially for n>3, is to create enough training data by formulating useful search queries. For some relations, it might be possible to generate positive training data but it is very difficult to formulate search queries that deliver meaningful negative examples and to do so requires a good understanding of the learning technique itself. Furthermore, whenever the argument types of the relation contain times and dates, e.g. *birth(person, city, date)*, money amount, e.g. *company_acquisition(company1, company2, money amount)*, or just numbers, it is impossible to find examples for the negative training bags. But even for binary relations, we may need an additional technique because although the learning algorithm relies on a minimal amount of supervision, it still requires some background work like finding adequate search queries or training of an optimal model, i.e. determining the desired precision/recall rate.

McDonald et al. [18] have shown that factoring n-ary relations into a set of binary relations, and reconstructing the instantiated binary relations into n-ary again by using maximal cliques in the relation graphs has been successful on a large set of biomedical data. In contrast our solution infers n-ary relations from binary ones by using manually constructed relation hierarchies (see Fig. 8). Starting from the binary relations using the approach described in the previous section (squared boxes) the relation hierarchy offers the possibility to infer n-ary relations (parallelograms) as well as new binary relations (rounded-corner boxes). Note that relations may only be derived if the binary relations (squared boxes) have been successfully extracted. The resulting relation candidates are then validated by generating a new search request containing the matched arguments of the relation with *AND*, and sending it to the search engine. If the number of search results is above a certain threshold we conclude that the inferred relation is validated. For argument types like times and dates, amounts of money, etc.[9] we use an extended relation hierarchy as shown in (see Fig. 9) because they never occur as arguments of our original or derived relations. Again we utilize search queries for validation of the resulting relation candidates.

[9] The classification of NP chunks to argument types like times and dates is currently done by using simple regular expressions.

However, general evaluation for this technique is very difficult as it depends heavily on the instances of the relations themselves and their existence on the Web. So we need to define new evaluation strategies for this special kind of relation extraction (see section 8). Generally speaking, the success of the n-ary relation extraction as described here depends on two basic factors:

1. Confidence of the classified binary relations: With higher confidence figures of these relations the probability of the inferred n-ary and binary relations increases.
2. Verification on the Web: In our experiments we observed that the precision rate of inferred relations increases if they can be found in different places on the Web, i.e. if they are well supported by the Web. We also experimented for n-ary relations (n>3) with high Web support using classification models possessing higher recall and lower precision for their underlying binary relations (this means with lower confidences). This resulted in increased recall with constant precision rates for the n-ary relations.

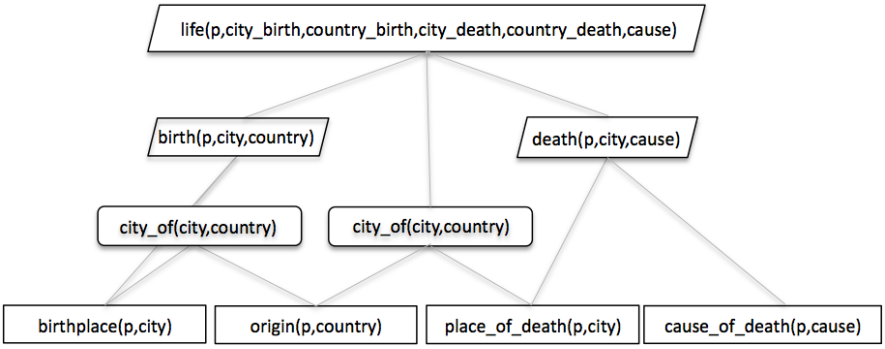

Fig. 8 An example for a relation hierarchy. The squared boxes show relations extracted from the snippets. The rounded-corner boxes show inferred binary relations, the parallelograms show inferred n-ary relations

7 Evaluation

For an initial evaluation we had 20 testers: 7 came from our lab and 13 from non–computer science related fields. 15 persons had never used an iPad before. After a brief introduction to our system (and the iPad), the testers were asked to perform three different searches (using Google, i–GNSSMM and i–MILREX) by choosing the queries from a set of ten themes. The queries covered definition questions like *EEUU* and *NLF*, questions about persons like *Justin Bieber*, *David Beckham*, *Pete Best*, *Clark Kent*, and *Wendy Carlos* , and general themes like *Brisbane*, *Balancity*,

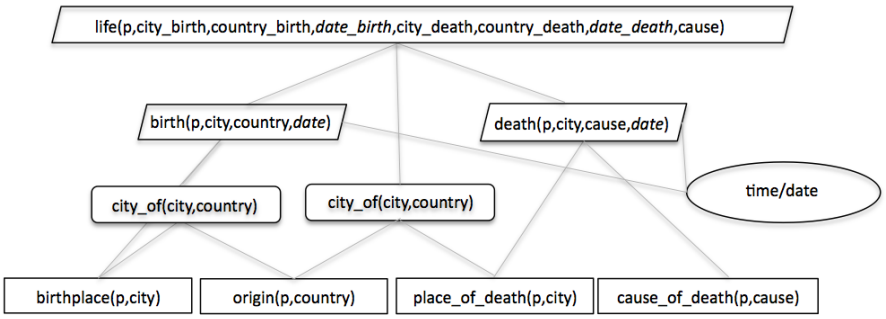

Fig. 9 New relations are created by attaching the topic "date".

and *Adidas*. The task was not only to get answers on questions like "Who is ..." or "What is ..." but also to acquire knowledge about background facts, news, rumors (gossip) and more interesting facts that come into mind during the search. Half of the testers were asked to first use Google and then our system in order to compare the results and the usage on the mobile device. We hoped to get feedback concerning the usability of our approach compared to the well known internet search paradigm. The second half of the participants used only our system. Here our research focus was to get information on user satisfaction of the search results. After each task, both testers had to rate several statements on a Likert scale and a general questionnaire had to be filled out after completing the entire test. Table 2 and 3 show the overall result.

Table 2 Google

#Question	v.good	good	avg.	poor
results first sight	55%	40%	15%	-
query answered	71%	29%	-	-
interesting facts	33%	33%	33%	-
suprising facts	33%	-	-	66%
overall feeling	33%	50%	17%	4%

Table 3 i-GNSSMM and i-MILREX

#Question	v.good	good	avg.	poor
results first sight	43%	38%	20%	-
query answered	65%	20%	15%	-
interesting facts	62%	24%	10%	4%
suprising facts	66%	15%	13%	6%
overall feeling	54%	28%	14%	4%

The results show that people in general prefer the result representation and accuracy in the Google style. Especially for the general themes the presentation of web snippets is more convenient and more easy to understand. However when it comes to interesting and suprising facts users enjoyed exploring the results using the topic graph. The overall feeling was in favor of our system which might also be due to the fact that it is new and somewhat more playful.

The replies to the final questions: *How successful were you from your point of view? What did you like most/least;? What could be improved?* were informative and contained positive feedback. Users felt they had been successful using the system. They liked the paradigm of the explorative search on the iPad and preferred touching the graph instead of reformulating their queries. The presentation of background facts in i–MILREX was highly appreciated. However some users complained that the topic graph became confusing after expanding more than three nodes. As a result, in future versions of our system, we will automatically collapse nodes with higher distances from the node in focus. Although all of our test persons make use of standard search engines, most of them can imagine to using our system at least in combination with a search engine even on their own personal computers.

8 Conclusion and Future Work

Above, we presented an approach of interactive topic graph extraction for exploration of web content. The initial information request is issued online by a user to the system in the form of a query topic description. The topic query is used for constructing an initial topic graph from a set of web snippets returned by a standard search engine. At this point, the topic graph already displays a graph of strongly correlated relevant entities and terms. The user can then request further detailed information for parts of the topic graph in the form of definitions and relations from the Web and Wikipedia infoboxes.

A prototype of the system has been realized on the basis of a mobile touchable user interface for operation on an iPad. We believe that our approach of interactive topic graph extraction and exploration, together with its implementation on a mobile device, helps users explore and find new interesting information on topics about which they have only a vague idea or even no idea at all.

The next steps will be the development of a sophisticated evaluation of n-ary relation extraction and systematic field tests for complete system cycles under different user profiles. For this, we also plan to improve the concept disambiguation by implementing a stronger integration of concept and relation extraction. Furthermore, we plan to explore contextual user interaction, e.g., by taking into account all available information of the neighboring nodes and edges of the parts of the topic graph selected by the user. A particular challenge will be the exploration of multi–lingual and cross–lingual methods, which support comparison and merging of extracted information from Web snippets for different languages.

Acknowledgements The presented work was partially supported by grants from the German Federal Ministry of Economics and Technology (BMWi) to the Theseus project (FKZ: 01MQ07016).

References

1. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M.S., Etzioni, O.: *Open Information Extraction from the Web*. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp 2670–2676 (2007)
2. Baroni, M., Evert, S.: *Statistical methods for corpus exploitation*. In A. Lüdeling and M. Kytö (eds.), Corpus Linguistics. An International Handbook, Mouton de Gruyter, Berlin (2008)
3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: *DBpedia - A crystallization point for the Web of Data*. Web Semantics: Science, Services and Agents on the World Wide Web 7 (3): 154165 (2009)
4. Bunescu, R.C., Mooney, R.J.: *Learning to extract relations from the web using minimal supervision*. In Proceedings of ACL'07, pp 576–583 (2007)
5. Cui, H., Kan, M.Y., Chua T.S., Xiao, J.: *A comparative study on sentence retrieval for definitional question answering*. SIGIR Workshop on Information Retrieval for Question Answering (IR4QA), Sheffield, UK (2004)
6. Downey, D., Schoenmackers, S., Etzioni, O.: *Sparse Information Extraction: Unsupervised Language Models to the Rescue*. In Proceedings of ACL, pp 696–703 (2007)
7. Eichler, K., Hemsen, H., Löckelt, M., Neumann, G., Reithinger, N.: *Interactive Dynamic Information Extraction*. In Proceedings of KI'2008, Kaiserslautern, pp 54–61 (2008)
8. Etzioni, O.: *Machine reading of web text*. In Proceedings of the 4th international Conference on Knowledge Capture, Whistler, BC, Canada, pp 1-4 (2007)
9. Figueroa, A., Neumann, G., Atkinson, J.: *Searching for definitional answers on the web using surface patterns*. IEEE Computer 42(4), pp 68–76 (2009)
10. Figueroa, A., Neumann, G.: *Language Independent Answer Prediction from the Web*. In proceedings of the 5th FinTAL, Finland (2006)
11. Giesbrecht, E., Evert, S.: *Part-of-speech tagging - a solved task? An evaluation of PoS taggers for the Web as corpus*. In proceedings of the 5th Web as Corpus Workshop, San Sebastian, Spain (2009)
12. Greenwood, M.A., Stevenson, M.: *Improving semi-supervised acquisition of relation extraction patterns*. In Proceedings of the Workshop on Information Extraction Beyond The Document, Sydney, pp 12–19 (2006)
13. Giménez, J., Màrquez, L.: *SVMTool: A general PoS tagger generator based on Support Vector Machines*. In proceedings of LREC'04, Lisbon, Portugal (2004)
14. Hildebrandt, W., Katz, B., Lin, J.: *Answering Definition Questions Using Multiple Knowledge Sources*. In Proceedings HLT-NAACL, pp 49–56 (2004)
15. Joho, H., Liu, Y.K., Sanderson, M.: *Large Scale Testing of a Descriptive Phrase Finder*. In Proceedings 1st Human Language Technology Conference, San Diego, CA, pp 219–221 (2001)
16. Landauer, T., McNamara, D., Dennis, S., Kintsch, W.: *Handbook of Latent Semantic Analysis*. Lawrence Erlbaum (2007)
17. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press (2008)
18. McDonald, R., Kulick, S., Pereira, F., Winters, S., Jin, Y., White, P.: *Simple algorithms for complex relation extraction with applications to biomedical IE*. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp 491–498 (2005)
19. Rosenfeld, B., Feldman, R.: *URES: an unsupervised web relation extraction system*. In Proceedings of the COLING/ACL on Main Conference Poster Sessions, Sydney, Australia, pp 667–674 (2006)

20. Shinyama, Y., Sekine, S.: *Preemptive information extraction using unrestricted relation discovery*. In Proceedings of the Proceedings of the Human Language Technology Conference of the NAACL, Main Conference, New York City, USA, pp 304–311 (2006)
21. Sekine, S.: *On-demand information extraction*. In Proceedings of the COLING/ACL, Sydney, Australia, pp 731–738 (2006)
22. Soubbotin, M.M.: *Patterns of Potential Answer Expressions as Clues to the Right Answers*. In Proceedings of the TREC-10 Conference, NIST (2001), Gaithersburg, Maryland, pp 293–302 (2001)
23. Sudo, K., Sekine, S., Grishman, R.: *An improved extraction pattern representation model for automatic IE pattern acquisition*. In Proceedings of ACL, pp 224–231 (2003)
24. Turney, P.D.: *Mining the web for synonyms: PMI-IR versus LSA on TOEFL*. In Proceedings of the 12th European Conference on Machine Learning. Freiburg, Germany, pp 491-502 (2001)
25. Yates, A.: *Information Extraction from the Web: Techniques and Applications*. Ph.D. Thesis, University of Washington, Computer Science and Engineering (2007)