Web Topic Summarization

Josef Steinberger[1]; Karel Ježek[1]; Martin Sloup[1]

[1] Department of Computer Science and Engineering, University of West Bohemia in Pilsen
Univerzitní 8, Pilsen 306 14, Czech Republic
e-mail: jstein@kiv.zcu.cz; jezek_ka@kiv.zcu.cz;msloup@student.zcu.cz

Abstract

In this paper, we present our online summarization system of web topics. The user defines the topic by a set of keywords. Then the system searches the Web for the relevant documents. The top ranked documents are returned and passed on to the summarization component. The summarizer produces a summary which is finally shown to the user. The proposed architecture is fully modular. This enables us to quickly substitute a new version of any module and thus the quality of the system's output will get better with module improvements. The crucial module which extracts the most important sentences from the documents is based on the latent semantic analysis. Its main property is independency of the language of the source documents. In the system interface, one can choose to search a news site in English or Czech. The results show a very good search quality. Most of the retrieved documents are fully relevant, only a few being marginally relevant. The summarizer is comparable to state-of-the-art systems.

Keywords: Information retrieval; searching; summarization; latent semantic analysis

1. Introduction

Searching the web has played an important role in human life in the past couple of years. A user either searches for specific information or just browses topics which interest him/her. Typically, a user enters a query in natural language, or as a set of keywords, and a search engine answers with a set of documents which are relevant to the query. Then, the user needs to go through the documents to find the information that interests him. However, usually just some parts of the documents contain query-relevant information. A benefit to the user would be if the system selected the relevant passages, put them together, made it concise and fluent, and returned the resulting text. Moreover, if the resulting summary is not relevant enough, the user can refine the query. Thus, as a side effect, summarization can be viewed as a technique for improving querying.

Our aim is to apply the following step after retrieval of the relevant documents. The set of documents is summarized and the resulting text is returned to the user. So, basically, the key work is done by the summarizer. In the past we created a single-document summarizer which extracted the most important sentences from a single source document [1]. The core of the summarizer was covered by latent semantic analysis (LSA – [2]). Now, we are experimenting with its extension to process multiple documents – a cluster of documents concerning the same topic. Several new problems arise here. For example, because the documents are about the same topic, they can contain similar sentences. We have to ensure that the summary does not contain this type of redundancy.

In this paper, we present the SWEeT system (Summarizer of WEb Topics). A user enters a query in the system. That query should describe the topic he would like to read about (e.g. "George Bush Iraq War"). The system passes the query to a search engine. It answers with a set of relevant documents sorted by relevance to the query. Top *n* documents, where *n* is a parameter of the system, are then passed to our summarizer, the core of the system. The created summary is returned to the user, together with references to the searched documents that can help him to get more details about the topic.

The structure of the paper is as follows. In Section 2, a quick overview of our SWEeT approach is presented. We then go deeper into the technical details (Section 3). We describe the architecture of the system and then we briefly mention the function and approach of each module. Then, in Section 4, we discuss the evaluation results, which can give an idea of the searching and summarizing quality. Moreover, we show a couple of resulting summaries and system screenshots. In the end, we discuss our vision of the system's further extensions and improvements.

2. Approach Overview

Until we go into more technical details, we will explain the approach firstly in a simple way. After the user submits a query it is passed to a search engine. It answers with a set of relevant documents. Their contents, together with some additional information, e.g. date of publication, are extracted and passed on to the summarizer.

The first task for the summarizer is to extract the most important sentences from the set of documents. Our approach follows what has been called a term-based strategy: find the most important information in the document(s) by identifying its main terms, and then extract from the document(s) the most important information (i.e., sentences) about these terms [3]. Moreover, to reduce the dimensionality of the term space, we use the latent semantic analysis [2], which can cluster similar terms and sentences into 'topics' on the basis of their use in context. The sentences that contain the most important topics are then selected for the summary. However, in this step, we have to be sure that the summary does not already contain a similar sentence to prevent redundancy. The vector of the sentence that is trying to be included in the summary is compared with those of the sentences already included in the summary by cosine similarity.

After obtaining the summary sentences, we try to remove unimportant clauses from them. (In other words, we perform a second-level summarization.) We designed a set of knowledge-poor features that help in deciding if the most important information contained in a sentence is still present in its compressed version (see Section 3.7, [4]). These features are used by the classifier, which makes a decision on whether the particular clause is/is not important. The shortest of the compressed versions that still contain the main sentence information is selected to substitute the full sentence in the summary. Further, the summary sentences have to be ordered. Our method uses the fact that two sentences that are to appear next to each other in the final summary should be connected by occurrences of the same entities. The last step of our approach is to correct the problematic occurrences of entities brought by extracting sentences without their context. (E.g. there can be a pronoun which the reader could not interpret.) Our approach is to substitute each of these problematic expressions (e.g. *he*) with the full noun phrases (e.g. *president George Bush*) [5].

3. System Architecture

The crucial part of the system is the summarizer. However, state-of-the-art summarization is still far behind human-written summaries. So we designed a modular system to quickly enable us to improve the summarization process (see Figure 1).

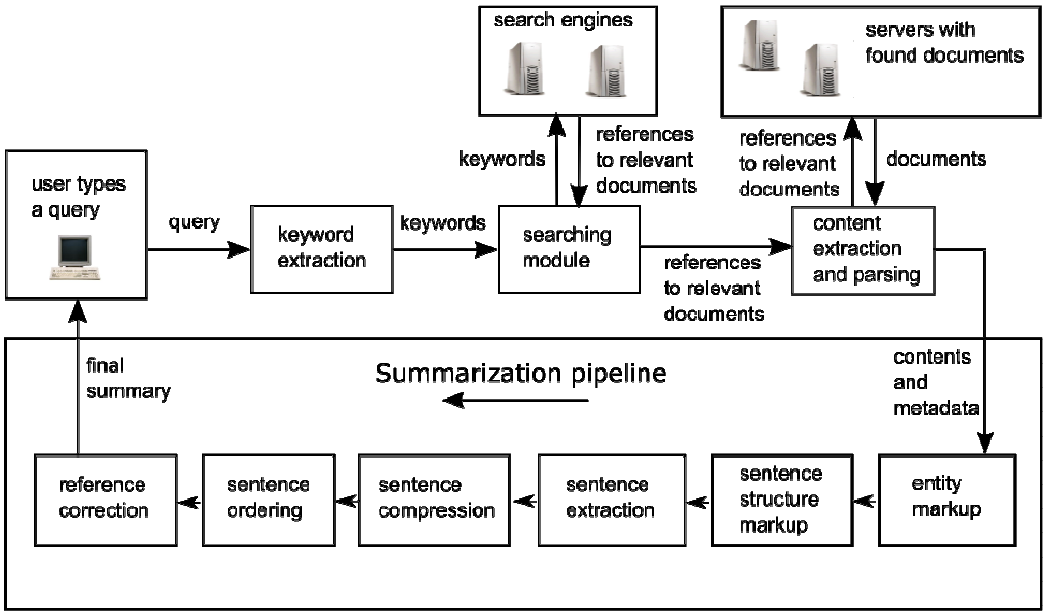

Figure 1: System architecture

The first stage of the process is to pass the query to a search engine. We use the widely used Google search engine. Moreover, the search engine can be easily instructed to search a single domain or a couple of domains. Thus, e.g., we can search just certain news domains to get a summary of a news topic. After getting the

cluster of relevant documents, their source URLs, titles, dates of publication (if available) and the own texts are extracted. The cluster is saved in our designed XML format.

After getting the XML with the searching details the summarization pipeline starts. This pipeline consists of several modules. Their aim is to create the final summary XML node whose content is finally returned as the system answer. The first module annotates entities (e.g., persons, organizations, places) which appear in the text. This would be needed for sentence ordering and entity occurrence correction. Later, we plan to use a complex co-reference resolution system for this task (e.g. Bart [6]). The next module tries to automatically annotate the sentence clause structure which is needed for sentence compression. The sentence extraction module is the main one. Its goal is to select the summary sentences. Our LSA-based method is used here. After this step, the XML file contains the summary node with selected sentences. Then, it is the turn of the sentence compression module, which removes unimportant clauses from the summary sentences. The next module orders the sentences in the summary and the last module corrects the entity occurrences. The last stage takes the content of the XML summary node and presents it to the user. The modules are discussed in the following subsections.

3.1 User Query Processing and Keyword Extraction

The first stage after submitting the query is to extract significant terms from it. The resulting set of keywords is then used in the searching module. For this task we need a list of "stop words", i.e. words that do not carry any information – prepositions, conjunctions, etc. If the module finds a stop word among the query terms, it ignores it. Further, we need to convert the terms into their basic forms (lemmatization). We use a dictionary where for each term we can get a lemma. Thus, we get a set of lemmas that hold the query information that is passed on to the searching module.

3.2 Searching by External Search Engine

The aim of the searching module is to find documents relevant to the query. So far, the system has searched just a single pre-defined domain (for English it is nytimes.com and for Czech it is novinky.cz). We use well-known external search engines to guarantee the highest searching quality. The first one is Google whose performance cannot be doubted. However, we need to search just a single *domain* and thus we use the modifier "site:*domain*". For searching in the Czech news site novinky.cz, we directly use their search engine. It is based on the Seznam engine, one of the most widely used engines on the Czech Web. Thus, good searching quality is also guaranteed in the case of searching the Czech news domain. Nevertheless, the modular architecture enables us to use other search engines as well.

3.3 Content Extraction and Parsing

References to top *n* retrieved documents[1] are passed from the searching module to content extraction and parsing. The documents pointed to by the references are then downloaded and parsed. The parser needs to know what parts of the HTML structure have to be extracted. This cannot be done automatically for any HTML structure. Fortunately, each portal has its own uniform format. We created a simple configuration for each domain in which we run searching. This configuration tells the parser where it should find the title, the date of publication and the own text in the HTML structure. The resulting texts, together with titles and other meta-information, are converted into our own XML format, which is passed, and updated, through the summarization pipeline.

3.4 Entity Markup

Entity markup starts the summarization pipeline. Each module of the pipeline adds some information to the XML data. The first two modules add a marking that is utilized by other modules further down the pipeline. The entity markup module tries to mark all entities that occur in the text (persons, institutions, geographic names, etc.). Here we have to use a natural language parser. This component cannot be language independent. For English we use the Charniak parser [7] and for Czech we use a parser from PDT 2.0 (Prague Dependency Treebank – [8]) which is based on the Collins parser [9]. Both these tools can mark noun phrases (NP) and with a little effort we can get heads of the NPs[2]. From these noun phrases we create co-reference chains. Two NPs are added to the co-reference chain if they contain the same noun. With this approach we can put together phrases

[1] In our experiments we used 10 most relevant documents; however, this constant will be able to be set in the advanced searching settings in the next version of the system.

[2] E.g., the head of the noun phrase "the blue car" is "car".

like “president George Bush”, “Bush”, “the president” or “George”. On the other hand, “the Czech president” and “the U.S. president” will be bound by mistake. In future we plan to use a complex co-reference resolution system [6] that would resolve other anaphoric expressions like pronouns. In the XML data file the entity occurrences are wrapped in tags and the identifier of the entity chain is contained in its attribute. The information about entities is later used in the modules for sentence ordering, reference correction and sentence compression.

3.5 Sentence Structure Markup

After finishing the entity markup, the sentence structure markup follows. Its aim is to identify sentence parts (clauses). For this task we again use the natural language parser’s output. It can derive a sentence tree structure like the one in Figure 2.

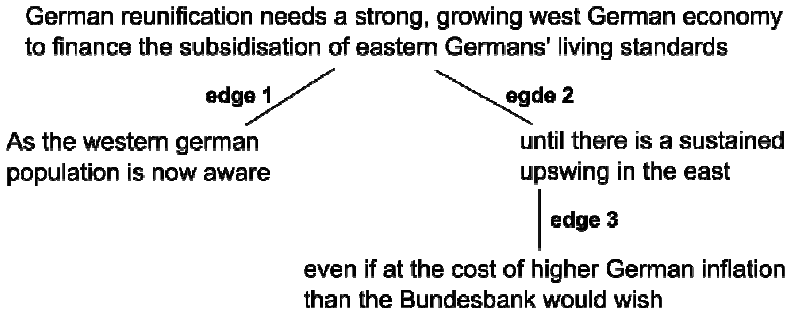

Figure 2: Tree structure of an example sentence.

The knowledge of sentence structure is later used by the sentence compression module. In the XML file, the clauses are wrapped in tags as in the case of entity marking.

3.6 LSA-based Sentence Extraction

This module is the core of the pipeline. It identifies and then extracts the most important sentences from the retrieved documents. The algorithm is based on our LSA-based single-document summarization method [1]. It was extended to work with a set of documents [10].

LSA is a fully automatic mathematical/statistical technique for extracting and representing the contextual usage of words’ meanings in passages of discourse. The basic idea is that the aggregate of all the word contexts in which a given word does and does not appear provides mutual constraints that determine the similarity of meanings of words and sets of words to each other. LSA has been used in a variety of applications (e.g., information retrieval, document categorization, information filtering, and text summarization).

The heart of the analysis in the summarization background is a document representation developed in two steps. The first step is the creation of a term-by-sentence matrix, where each column represents the weighted term-frequency vector of a sentence in the set of documents under consideration. The terms from a user query get higher weight. The next step is to apply Singular Value Decomposition (SVD) to matrix A:

$$A = U \Sigma V^T, \tag{1}$$

where $U = [u_{ij}]$ is an $m \times n$ column-orthonormal matrix whose columns are called *left singular vectors*. $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$ is an $n \times n$ diagonal matrix, whose diagonal elements are non-negative *singular values* sorted in descending order. $V = [v_{ij}]$ is an $n \times n$ orthonormal matrix whose columns are called *right singular vectors*. The dimensionality of the matrices is reduced to r most important dimensions and thus, U is $m \times r$, Σ is $r \times r$ and V^T is $r \times n$ matrix.

From an NLP perspective, what SVD does is to derive the latent semantic structure of the document represented by matrix A: i.e. a breakdown of the original document into r linearly-independent base vectors which express the main ‘topics’ of the document. SVD can capture interrelationships among terms, so that terms and sentences can be clustered on a ‘semantic’ basis rather than on the basis of words only. Furthermore, as demonstrated in [11], if a word combination pattern is salient and recurring in the document, this pattern will be captured and represented by one of the singular vectors. The magnitude of the corresponding singular value indicates the importance degree of this pattern within the document. Any sentences containing this word combination pattern will be projected along this singular vector, and the sentence that best represents this pattern will have the largest index value with this vector. Assuming that each particular word combination pattern

describes a certain topic in the document, each singular vector can be viewed as representing such a topic [12], the magnitude of its singular value representing the degree of importance of this topic.

The method selects for the summary those sentences whose vectorial representation in the matrix $\Sigma \cdot V^T$ has the greatest 'length'. Intuitively, the idea is to choose the sentences with the greatest combined weight across all important topics.

In [10] we proposed the extension of the method to process a cluster of documents written about the same topic. Multi-document summarization is a one step more complex task than single-document summarization. It brings new problems we have to deal with. The first step is again to create a term-by-sentence matrix. In this case we include in the matrix all sentences from the cluster of documents. (On the contrary, in the case of single-document summarization we included the sentences from that document.) Then, we run sentence ranking. Each sentence gets a score which is computed in the same way as when we summarized a single document – vector length in the matrix $\Sigma \cdot V^T$ (LSA score). Now, we are ready to select the best sentences (the ones with the greatest LSA score) for the summary.

However, two documents written about the same topic/event can contain similar sentences and thus we need to solve redundancy. We propose the following process: before adding a sentence to the summary, see whether there is a similar sentence already in the summary. The similarity is measured by the cosine similarity in the original term space. We determine a threshold here. The extracted sentence should be close to the user query. To satisfy this, query terms get a higher weight in the input matrix.

3.7 Knowledge-poor Sentence Compression

Naturally, long sentences with many significant terms are usually selected for the summary. However, they often contain clauses that are unimportant from the summarization point of view. We try to identify these clauses and then remove them. Firstly, we need to create a set of possible compressed forms of each summary sentence. We call them compression candidates (CC). In this step we use the knowledge of sentence structure obtained by the sentence structure markup module (example in Figure 2). If we cut the tree on an edge, we get a compressed sentence (CC) where all subordinate clauses of the edge are removed. And moreover, we can cut the tree more than once - in a combination of edges. In this way we obtain a set of CCs.

After obtaining the set of CCs, we try to select the best candidate within the set. In some of the candidates some important information is removed or even its sense is changed. We designed several features that can help in deciding whether the crucial information is retained or not in the particular candidate[3]. The final decision is left to a two-class classifier. The shortest candidate within the positive ones is selected to substitute the original sentence in the final summary.

3.8 Sentence Ordering

After obtaining sentences (or their compressed versions) which the final summary will consist of, they have to be ordered somehow. Our idea for resolving this problem is that two sentences that occur close to each other should deal with the same entities. The first step is to select the first summary sentence. Each sentence is assigned by a score that describes to what extent it should start the summary. From the entity markup we get entity co-occurrence chains, but moreover, for each chain and document we get one NP that starts the chain in that document. Usually, each entity is introduced in the document with the full NP (e.g. "president George Bush"). Sentences are then scored according to three features – the number of entities occurring in them, the number of entity introductions, and finally, the date of the publication of the document in which the sentence is contained. A sentence from the oldest document is preferred to start the summary. When we have selected the first sentence it is time to select the next one. The sentence that contains the same entities as the previous one is preferred to continue the summary. Thus again, the sentences are scored according to the slightly changed three features: the number of entities occurring in them, where the entities that occur in the previous sentence are emphasized (multiplied by a weight), the number of entity introductions, where again the entities that occur in the previous sentence are emphasized (multiplied by a weight) and finally, the date of the publication of the document in which the sentence is contained. A sentence from the oldest document is preferred as well. This process is repeated until we have ordered all the sentences.

[3] For example, the depth of the removed clause in the clause tree structure can signify how important the clause is (the lower, the less important), or the fall in the LSA score of the CC (compared to the LSA score of the full sentence) can show how important the removed information was. For details, see [4].

3.9 Reference Correction

Anaphoric expressions can only be understood with respect to a context. This means that summarization by sentence extraction can wreak havoc with their interpretation: there is no guarantee that they will have an interpretation in the context obtained by extracting sentences to form a summary, or that this interpretation will be the same as in the original text. For example, a pronoun can occur in the summary without any information about which entity it replaces. Our idea is to replace anaphoric expressions with a full noun phrase in cases where the anaphoric expression could otherwise be misinterpreted. The information marked by the entity marking module is utilized here. However, we need a co-reference resolver. So far we have experimented just with English and the GuiTAR resolver [13]. For details, see [5].

4. Experiments

There are two crucial parts that affect the performance of the system: the quality of searching and the quality of summarization. As for searching, we will present figures showing its accuracy (how many retrieved documents were relevant to the user query and how many were not). We use manual annotations. The quality of the summarization is assessed by the widely-used ROUGE measure [14, 15]. At the end of the section, we present a couple of system summaries and we show system screenshots.

4.1 Searching Results

The following tables demonstrate that with the proposed searching approach we can obtain mostly relevant documents. Just a couple of documents were classified as marginally relevant (i.e., the query terms are mentioned there in the right sense, but the main document's topic is different from the query topic). A few documents were irrelevant (e.g., when we submitted a query about a huge accident on Czech highway D1, the system returned a document about an accident on an Austrian highway). Proper names can increase the accuracy of searching. We analyzed a maximum of the top ten retrieved documents. The results are presented in Table 1 (English queries) and Table 2 (Czech queries).

Query ID	Significant terms in query	Total	Relevant	Marginally relevant	Irrelevant
1	China Olympic games protests	10	10	0	0
2	American radar in Czech Republic	10	8	2	0
3	Independent Kosovo	10	10	0	0
4	Polygamy U.S. sect	10	9	1	0
5	Obama Hillary Clinton president elections	10	10	0	0
6	Soccer stadium security	10	8	0	2
7	Iraq attact U.S.	8	7	1	0
8	Iranian nuclear program	9	8	1	0
9	Mugabe Zimbabwe elections	8	8	0	0
10	Al Queda Osama bin Laden	6	4	2	0
In total		**91**	**82 (90,1%)**	**7 (7.7%)**	**2 (2.2%)**

Table 1: Evaluation of searching quality on English queries

4.2 Summarization Results

Assessing the quality of a summary is much more problematic. The DUC (Document Understanding Conference – [16]) series of annual conferences controls the direction of the evaluation. However, the only fully automatic and widely used method so far is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [14, 15] which compares human-written abstracts and system summaries based on the overlap of n-grams[4].

[4] An n-gram is a subsequence of *n* words from a given text.

Query ID	Significant terms in query	Total	Relevant	Marginally relevant	Irrelevant
1	Peking Čína olympijské hry bojkot	5	4	1	0
2	Americký radar Brdy	10	8	2	0
3	Samostatnost Kosova	9	7	2	0
4	USA polygamní sekta	3	3	0	0
5	Obama Hillary Clinton prezident volby	10	10	0	0
6	Fotbal stadión bezpečnost fanoušci	5	5	0	0
7	Poplatky u lékaře reforma zdravotnictví Julínek	10	9	1	0
8	Daňová reforma	10	7	3	0
9	Hromadná nehoda na dálnici D1	10	7	0	3
10	Sraz neonacistů Praha	9	5	0	4
In total		**91**	**65 (80,3%)**	**9 (11.1%)**	**7 (8.6%)**

Table 2: Evaluation of searching quality on Czech queries

Suppose a number of annotators created manual summaries. The *ROUGE-n* score of a candidate summary (the summary which is evaluated) is computed as follows:

$$ROUGE\ -n = \frac{\sum_{C \in \{manual\ summaries\}} \sum_{n-gram \in C} Count_{match}(n-gram)}{\sum_{C \in \{manual\ summaries\}} \sum_{n-gram \in C} Count(n-gram)}$$

where $Count_{match}(n\text{-}gram)$ is the maximum number of *n*-grams co-occurring in a candidate summary and a manual summary and $Count(gram_n)$ is the number of *n*-grams in the manual summary. Notice that the average *n*-gram ROUGE score, *ROUGE-n*, is a recall metric. It was shown that bigram score *ROUGE-2* and *ROUGE-SU4* (a bigram measure that enables at most 4 unigrams inside bigram components to be skipped [15]) best correlate with the human (manual) system comparison.

We present a comparison of our summarizer with those that participated at DUC 2005 - Tables 3 and 4. Not all of the differences are statistically significant. Therefore, we show by the letters the multiple systems' comparison – the systems that share the same letter (in the last column) are NOT statistically significant. To summarize these tables: in ROUGE-2, our summarizer performs worse than 5 systems and better than 27 systems; however, when we count in significance, none of the systems performs significantly better than ours and 8 of them perform significantly worse. And similarly in ROUGE-SU4, our summarizer performs worse than 5 systems and better than 27 systems; however, when we count in significance, none of the systems performs significantly better than ours and 11 of them perform significantly worse.

4.3 Example summaries

To demonstrate the system output we show two resulting summaries (their desired length is 255 words). One for English and the query: "Al Qaeda and Osama bin Laden" and one for Czech and the query "americký radar Brdy" (American radar Brdy) – Figures 3 and 4.

4.4 System Interface

To get the reader closer to the user interface of the system, we present screen outputs. In Figure 5 there is a page where a user submits a query. The length of the resulting summary can be selected here. In Figure 6 there is a page with searching (and summarization) results. Under the header with the query and the selected summary length, we can see the resulted summary and references to the original documents below.

Summarize ID	ROUGE-2 score	
15	0.0725	A
17	0.0717	A
10	0.0698	A B
8	0.0696	A B
4	0.0686	A B C
SWEeT	**0.06791**	**A B C**
5	0.0675	A B C
11	0.0643	A B C D
14	0.0635	A B C D E
16	0.0633	A B C D E
19	0.0632	A B C D E
7	0.0628	A B C D E F
9	0.0625	A B C D E F
29	0.0609	A B C D E F G
25	0.0609	A B C D E F G
6	0.0609	A B C D E F G
24	0.0597	A B C D E F G
28	0.0594	A B C D E F G
3	0.0594	A B C D E F G
21	0.0573	A B C D E F G
12	0.0563	B C D E F G
18	0.0553	B C D E F G H
26	0.0547	B C D E F G H
27	0.0546	B C D E F G H
32	0.0534	C D E F G H
20	0.0515	D E F G H
13	0.0497	D E F G H
30	0.0496	D E F G H
31	0.0487	E F G H
2	0.0478	F G H
22	0.0462	G H
1	0.0403	H I
23	0.0256	I

Table 3: Multiple comparisons of all peers based on ANOVA of ROUGE-2 recall

Summarize ID	ROUGE-SU4 score	
15	0.1316	A
17	0.1297	A B
8	0.1279	A B
4	0.1277	A B C
10	0.1253	A B C D
SWEeT	**0.12390**	A B C D
5	0.1232	A B C D E
11	0.1225	A B C D E
19	0.1218	A B C D E
16	0.1190	A B C D E F
7	0.1190	A B C D E F
6	0.1188	A B C D E F G
25	0.1187	A B C D E F G
14	0.1176	A B C D E F G
9	0.1174	A B C D E F G
24	0.1168	A B C D E F G
3	0.1167	A B C D E F G
28	0.1146	B C D E F G H
29	0.1139	B C D E F G H
21	0.1112	C D E F G H I
12	0.1107	D E F G H I
18	0.1095	D E F G H I J
27	0.1085	E F G H I J
32	0.1041	F G H I J
13	0.1041	F G H I J
26	0.1023	G H I J K
30	0.0995	H I J K
2	0.0981	H I J K
22	0.0970	I J K
31	0.0967	I J K
20	0.0940	J K
1	0.0872	K
23	0.0557	L

Table 4: Multiple comparisons of all peers based on ANOVA of ROUGE-SU4 recall

Even as American officials portrayed the case as mainly a Canadian operation, the arrests so close to the United States border jangled the nerves of intelligence officials who have been warning of the continuing danger posed by small "homegrown" extremist groups, who appeared to operate without any direct control by known leaders of Al Qaeda. These fighters include Afghans and seasoned Taliban leaders, Uzbek and other Central Asian militants, and what intelligence officials estimate to be 80 to 90 Arab terrorist operatives and fugitives, possibly including the Qaeda leaders Osama bin Laden and his second in command, Ayman al-Zawahri. In recent weeks, Pakistani intelligence officials said the number of foreign fighters in the tribal areas was far higher than the official estimate of 500, perhaps as high as 2,000 today. The area is becoming a magnet for an influx of foreign fighters, who not only challenge government authority in the area, but are even wresting control from local tribes and spreading their influence to neighboring areas, according to several American and NATO officials and Pakistani and Afghan intelligence officials. Some American officials and politicians maintain that Sunni insurgents have deep ties with Qaeda networks loyal to Osama bin Laden in other countries. Hussein's government, one senior refinery official confided to American soldiers. In fact, money, far more than jihadist ideology, is a crucial motivation for a majority of Sunni insurgents, according to American officers in some Sunni provinces and other military officials in Iraq who have reviewed detainee surveys and other intelligence on the insurgency.

Figure 3: Example English summary. Result for the query: "Al Qaeda and Osama bin Laden"

Rozhovory Spojených států s českou vládou o umístění radaru by mohly být završeny na bukurešťském summitu NATO na počátku dubna, s Poláky by chtěl Washigton dohodu uzavřít do konce volebního období amerického prezidenta George Bushe, tedy do konce roku. Plán Američanů umístit v Brdech protiraketový radar a v Polsku sila s obrannými raketami vyvolává od počátku odpor ruských představitelů. "Naše velká síla neznamená, že si můžeme dělat, co chceme a kdy chceme," řekl McCain a dodal: "Musíme naslouchat (různým) názorům a respektovat kolektivní vůli našich demokratických spojenců."Republikánský kandidát uvedl, že součástí skupiny nejvyspělejších států G8 by měly být demokratické země včetně Indie a Brazílie. Podle informací z ruských médií nabízí Američané Rusům možnost inspekcí objektů systému v ČR a Polsku, omezení možností radaru tak, aby nemohl sledovat ruský vzdušný prostor a slibují, že rakety do sil v Polsku neumístí do té doby, než bude zjevné hrozící nebezpečí. Poté, co před několika dny v Moskvě američtí ministři zahraničí a obrany Condoleezza Riceová a Robert Gates předložili oficiálně zatím nezveřejněné návrhy mající ruské obavy rozptýlit, se zřejmě ruská strana s existencí systému smířila. Nejlepší způsob, jak uklidnit ruské obavy z evropských prvků americké protiraketové obrany, by ale podle něj bylo vůbec radar v ČR a sila pro antirakety v Polsku nestavět. Informace z Moskvy potvrzuje nedávné tvrzení předsedy ČSSD, že dohoda Ruska a USA o protiraketové obraně je na spadnutí. To je vítězstvím Ruska, které ovšem nechtělo americký radar v ČR a sila s obrannými raketami v Polsku.

Figure 4: Example Czech summary. Result for the query: "americký radar Brdy" (i.e., American radar construction in Czech Republic-Brdy)

Figure 5: SWEeT's query form

Figure 6: SWEeT's result

5. Conclusion and Future Work

Pilot experiments show the solid quality of system summaries. The future version of the system will enable advance searching where the user will be able to select domains that will be searched, he will be able to select a summarizer and set it up. After that, we will work on multilingual processing. The system will search in various languages. The terms will be indexed by the EuroWordNet (EWN) thesaurus [17, 18] in an internal EWN format – Inter Lingual Index (ILI). As a result the system's answer will be multilingual. If the user understands more languages, he will get to know what is written about the topic in different countries/languages. And moreover, because the same terms in different languages would be linked, the summarizer can use all documents together to decide what is important in the topic. The proposed modular architecture has several advantages. We can easily change the search engine or the summarizer or any of its modules. Our summarizer is based on LSA, which works just with the context of words and thus is not dependent on any particular language. We perform experiments with both Czech and English queries. Another possible function of the system can be knowledge-poor question answering. When a user enters a question, the answer should be found in the summary. So far, the basic version of the system has been stable, however, some of the modules are still in the experimental stage and there are many things to be improved.

Acknowledgement

This research was partly supported by project 2C06009 (COT-SEWing).

References

[1] Steinberger, J., and Ježek, K. Text summarization and singular value decomposition. In *Proceedings the 3rd International Conference on Advances in Information Systems, Lecture Notes in Computer Science 2457*, Springer-Verlag Berlin Heidelberg, p. 245–254, 2004.

[2] Landauer, T. K., and Dumais, S. T. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. In *Psychological Review*, 104, p. 211–240, 1997.

[3] Hovy, E., and Lin, C. Automated text summarization in SUMMARIST. In *ACL/EACL Workshop on Intelligent Scalable Text Summarization*, Madrid, Spain, 1997.
[4] Steinberger, J., and Tesař, R. Knowledge-poor Multilingual Sentence Compression. In *Proceedings of 7th Conference on Language Engineering*, Cairo, Egypt, p. 369-379, 2007.
[5] Steinberger, J., Poesio, M., Kabadjov, M.A., and Ježek, K. Two Uses of Anaphora Resolution in Summarization. In *Special Issue of Information Processing & Management on Summarization*, volume 43, issue 6, Elsevier Ltd., p. 1663-1680, 2007.
[6] Versley, Y., Ponzetto, S.P., Poesio, M., Eidelman, V., Jern, A., Smith, J., Yang, X., and Moschitti, A. BART: A Modular Toolkit for Coreference Resolution. To appear in *ACL'08*.
[7] Charniak, E. A maximum-entropy-inspired parser. In *Proceedings of NAACL*. Philadelphia, 2000.
[8] *The Prague Dependency Treebank 2.0*. http://ufal.mff.cuni.cz/pdt2.0/
[9] Collins, M. *Head-Driven Statistical Models for Natural Language Parsing*. PhD Dissertation, University of Pennsylvania, 1999.
[10] Steinberger, J., and Křišťan, M. LSA-Based Multi-Document Summarization. In *Proceedings of 8th International PhD Workshop on Systems and Control, a Young Generation Viewpoint*, Balatonfured, Hungary, p. 87-91, 2007.
[11] Berry, M.W., Dumais, S.T., and O'Brien, G.W. Using Linear Algebra for Intelligent IR. In *SIAM Review*, 37(4), 1995.
[12] Ding, C.H.Q. A probabilistic model for latent semantic indexing. In *Journal of the American Society for Information Science and Technology*, 56(6), p. 597–608, 2005.
[13] Poesio, M., Kabadjov, M.A. A general-purpose, off-the-shelf anaphora resolution module: implementation and preliminary evaluation. In *Proceedings of LREC*, Lisbon, Portugal, 2004.
[14] Lin, C., and Hovy, E. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of HLT-NAACL*, 2003.
[15] Lin, Ch. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of the workshop on text summarization branches out*, Barcelona, Spain, 2004.
[16] Document Understanding Conference Past Data. http://www-nlpir.nist.gov/projects/duc/data.html
[17] EuroWordNet thesaurus. http://www.illc.uva.nl/EuroWordNet/
[18] Michal Toman, Josef Steinberger, Karel Ježek: Searching and Summarizing in Multilingual Environment. In *Proceedings of the 10th International Conference on Electronic Publishing*, p. 257–265, FOI Commerce, Bansko, Bulgaria, 2006.