Semantic Similarity of Short Texts in Languages with a Deficient Natural Language Processing Support

Bojan Furlan*, Vuk Batanović, Boško Nikolić

School of Electrical Engineering, Department of Computer Engineering and Information Theory, University of Belgrade, 11120 Belgrade, Serbia

Abstract—Measuring the semantic similarity of short texts is a noteworthy problem since short texts are widely used on the Internet, in the form of product descriptions or captions, image and webpage tags, news headlines, etc. This paper describes a methodology which can be used to create a software system capable of determining the semantic similarity of two given short texts. The proposed LInSTSS approach is particularly suitable for application in situations when no large, publicly available, electronic linguistic resources can be found for the desired language. We describe the basic working principles of the system architecture we propose, as well as the stages of its construction and use. Also, we explain the procedure used to generate a paraphrase corpus which is then utilized in the evaluation process. Finally, we analyze the evaluation results obtained from a system created for the Serbian language, and we discuss possible improvements which would increase system accuracy.

Keywords— Linguistic tools for IS modeling, Text DBs, Semantic similarity of words, Similarity of short texts, Corpus-based measures, Paraphrase corpora construction.

1. Introduction

SEMANTIC similarity is a concept by which a metric is given to groups of terms or documents based on the similitude of their meanings. This is a key concept in the understanding of natural languages, for it allows us to make meaningful comparisons and conclusions. That is why it plays such an important role in many fields of artificial intelligence, like automatic categorization and summarization, machine translation, information retrieval etc. The short text semantic similarity (STSS) is a noteworthy problem since short texts are widely used on the Internet, in the form of product descriptions or captions, image and webpage tags, news headlines, etc. STSS also plays an important role in issues related to education and learning, such as the automated testing and grading tasks [1].
There are two main manners of determining STSS – using topological or statistical word-to-word similarity [2]. Topological similarity uses data models which contain information about sets of concepts, and their interrelatedness. Statistical similarity, on the other hand, employs vector state spaces in order to express word correlations extracted from a given text corpus. A great advantage of the statistical tack is that it does not require the existence of any word data models.

For the English language a variety of Natural Language Processing (NLP) resources and tools can be found. Unfortunately, the creation of such resources necessitates significant time and effort, which is why they still do not exist for many languages, rendering many existing English language STSS solutions inapplicable. That lack is especially evident in minor languages and in languages with complex grammar rules.

This is a problem that had to be tackled in our efforts to create an STSS method suitable for Serbian, a highly inflectional language with very limited electronic linguistic resources. Therefore, we decided to base our approach on the statistical similarity method, devising a system-building methodology which would be applicable not only to Serbian, but to other languages with similar issues as well. We named it LInSTSS (Language Independent Short Text Semantic Similarity).

The basic procedure applied when calculating statistical similarity is the construction of a semantic space using the word distribution in the text corpus. In such a space each word has its own context vector, and the semantic similarity of words is represented by the relation of their vectors. This conclusion is a consequence of the distributional hypothesis which claims that words with similar meanings tend to appear in similar contexts. The hypothesis does not imply that words must appear next to one another, but that they should appear alongside the same set of other words.

This paper is organized as follows: in Section 2 we review some related work and analyze existing NLP tools and the prospects of their use in the development of the LInSTSS system. In Section 3 we describe system construction and in Section 4 we give an account and an example of system operation. Section 5 contains an explanation of the evaluation process. We discuss evaluation results of the Serbian language system in Section 6. Finally, in Section 7, we summarize our work and consider future system applications and improvements.

* *Corresponding author. Tel: +381 63 7741059; fax: +381 11 3248681.*
E-mail addresses: *bojan.furlan@etf.bg.ac.rs (B. Furlan)*
bv115045p@student.etf.bg.ac.rs (V. Batanović)
bosko.nikolic@etf.bg.ac.rs (B. Nikolić)

2. Analysis of Available Technologies and Tools

Many quality STSS solutions, both topological and statistical, have already been developed. Yet, few of them are designed in a way that would make them adaptable to languages with a poor NLP support, since they often rely on advanced language-specific processing techniques.

Mihalcea *et al* [2] proposed a method for measuring the semantic similarity of two short texts (sentences or paragraphs) by using corpus-based and knowledge-based measures of word similarity. For each word in the text this method identifies the best match from the opposite text and then includes it in the overall measure of semantic similarity. This approach yields a high F-measure score, but it is computationally demanding and requires the use of a word data model. Islam and Inkpen [3] improved the similarity measure by combining a modified string matching algorithm with a corpus-based measure of semantic similarity. Semantic similarity also plays an important role in the task of recognizing textual entailment (RTE) and shares many features with it (e.g. Wang and Neumann [4]), but, as explained in [5], RTE is an asymmetric task, while determining semantic similarity is not, so a different tack is required.

Li *et al* [6] proposed a method to determine sentence similarity that employs shallow parsing. Noun phrases, verb phrases and preposition phrases are extracted from the given sentences and the final similarity is calculated as a combination of the similarities of these three kinds of phrases. Oliva *et al* [5] combined semantic and syntactic information in their work as well. Syntax is analyzed through a deep parsing process to find the phrases in each sentence. Similarity between concepts that play the same syntactic role is then calculated by using a lexical database. They also experimented with the utilization of psychological plausibility by differently weighing various syntactic roles. Regrettably, parsing tools for Serbian and for many other languages are nonexistent, making the proposed syntactic analysis techniques unfeasible.

Having considered the methods discussed above, we concluded that no existing solutions can be directly applied to the problem of determining STSS in Serbian. Hence, we decided to create our own approach, basing it on a modification of [3] due to the following reasons:

1. It does not use any external knowledge bases (e.g. WordNet), hand-crafted inference rules, or parsing tools, which would be an obstacle when dealing with languages that lack these resources.
2. Accuracy is the percentage of correct identifications made by the system. Since it takes into account both false positive and false negative situations, it is one of the predominant parameters when comparing the qualities of different measures of semantic similarity. There are also others, like precision, recall, and F-measure. For each compared method these parameters were evaluated on the Microsoft Research Paraphrase Corpus (MSRPC), the largest English language paraphrase corpus consisting of 5801 pairs of sentences [7]. Methods that employ text parsing techniques [5, 6] have shown high performance in F-measure and recall, but haven't reached a level of accuracy greater than [3], when evaluated on this corpus.
3. This method does not use the semantic similarity measure alone, but incorporates the string similarity measure as well, so it performs better with different forms of infrequent proper nouns, which is one of the main weaknesses of the knowledge-based measures [8]. While this advantage may not be so prominent in English, it is quite striking in highly inflectional languages that have many word forms. Consequently, evaluation results on English corpora may differ from the results derived for other languages.

In order to construct the STSS system, we had to review existing stemming tools and algorithms of measuring string and semantic similarities. We then ascertained their adaptability to the problem in question, and chose the best available solutions.

Stemming is a transformation by which a word suffix is stripped off without the loss of the word's main semantic content. This procedure can also be viewed as a normalization process in which several morphological variants are mapped to the same form. Stemming, therefore, reduces the number of different words since all the words with the same stem are mapped into a single form. It should be noted that stemmers are the only language-dependent tools used in our approach.

There are numerous stemmers developed for English, among which the Porter stemmer is the most famous one. However, for many languages, including Serbian, such tools are not free, or can be hard to find in the public domain. Kešelj and Šipka [9] proposed a general suffix subsumption-based technique of building stemmers for highly inflectional languages with only sparse resources, which can be applied to Serbian. Their stemmer for Serbian has an experimentally evaluated accuracy of 81.83%.

String similarity is based upon the analysis of lexical matching of words or parts of words. It has been shown [3] that the best system accuracy can be gained by combining the results of string and semantical similarities. Hence, we apply the same practice to calculate string similarity by using three modifications of the LCS (*Longest Common Subsequence*) algorithm and finding the average of their grades. We employ the following LCS modifications:

- **NLCS** – *Normalized Longest Common Subsequence*
- **$MCLCS_1$** – *Maximal Consecutive Longest Common Subsequence starting at character* ***1***
- **$MCLCS_N$** – *Maximal Consecutive Longest Common Subsequence starting at character* ***N***

All three modifications employ a normalization procedure in which their similarity scores are divided by the lengths of the two strings that are compared. This is a strength in comparison to other string similarity methods that do not take into account the length of the shorter string, which, in some situations, can have a noticeable impact on the final grade. Furthermore, the combination of both consecutive and non-consecutive subsequence measures serves to balance out the score.

For measuring the word-to-word semantic similarity we use ready-made corpus processing algorithms from the *S-Space* package [10], which also incorporates many tools for corpus pre- and post-processing. The corpus processing algorithms utilize a co-occurrence matrix in which every row represents a unique word, and every column stands for a unique context. A matrix cell contains the number of occurrences of the row-word in the given column-context. A context can be a document or a region around a word, depending on the algorithm. In case it is a document, context vector dimension will correspond to the total number of documents, whereas in the case of a region-sized context that dimension can, in the worst case, correspond to the total number of different words found in the corpus. In this paper we applied the following algorithms:

- **COALS** – *Correlated Occurrence Analogue to Lexical Semantic* [11]
- **RI** – *Random Indexing* [12]

We chose COALS because it achieves a more consistent accuracy in predicting human similarity judgments than the older algorithms like HAL (*Hyperspace Analogue to Language*) [11]. Furthermore, unlike LSA (*Latent Semantic Analysis*), which expects sets of documents as the input material [13], COALS makes use of an undifferentiated text corpus, and employs a moving window to define word collocations. This makes the size of a COALS co-occurrence matrix almost fixed, unlike an LSA matrix whose dimension is proportional to the number of documents. Therefore, COALS proves to be far more scalable and easy to implement in situations requiring the use of large text corpora.

However, this scalability is achieved by using a computationally demanding technique of word space dimension reduction called SVD (*Singular Value Decomposition*), an algebraic operation which employs matrix factorization and decomposition. This approach is costly in terms of memory consumption, and can be unfeasible for especially large corpora, where initial word space sizes can be huge. For that reason we also considered the RI algorithm, an incremental word space model which does not require a separate dimension reduction phase, making it particularly suitable for big corpora processing.

Finally, to measure the similarity of a word pair, we determine its string and semantic similarity and combine them into a similarity score. These scores are then used to calculate the overall similarity of two text segments. The similarity score for each word pair is weighed by taking into account the specificity of its words. In order to improve the results of our method, we expressed the word specificity by utilizing a normalized term frequency weighing scheme, which gives a higher weight to pairs with more specific words.

3. System Construction

The workflow of our system construction process is shown in Fig. 1. It depicts the stages of semantic space creation.

Corpus acquisition deals with finding a sufficiently large set of texts that could be used to generate a semantic space. Numerous such corpora can be found in English, free of charge, but procuring one in a language like Serbian can prove

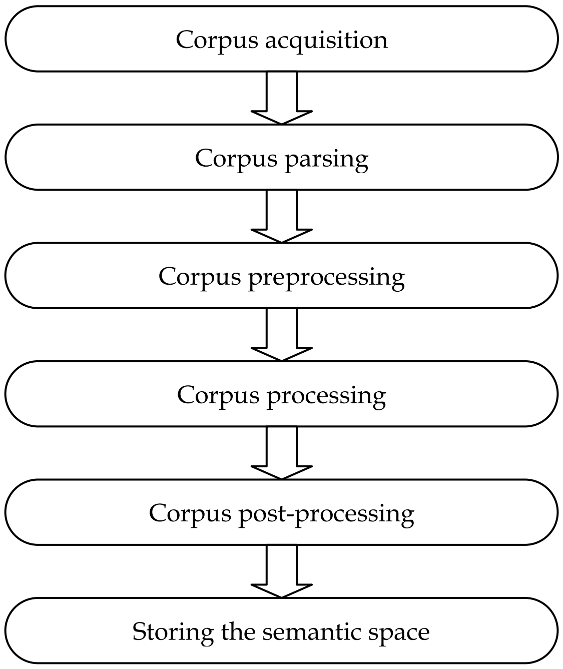

Fig. 1. System construction workflow. It depicts the stages of semantic space creation.

to be a task with limited options. A good, publicly available one was found in the corpus of article abstracts from the Wikipedia in Serbian, which, at the time, comprised 479990 pages totaling over 186 megabytes.

Corpus parsing is necessary so as to remove any superfluous information from further consideration. In our article abstract corpus, the abstracts themselves are enclosed by specific XML tags, but the corpus file also contains other, irrelevant data.

Corpus preprocessing serves to reduce the amount of different words in the corpus, effectively reducing the context vector dimension, and, in effect, the load on computer resources. In the LInSTSS approach preprocessing is done in three steps:

1. Text cleaning – this includes the deletion of all text characters not belonging to the native script of the language in question, the removal of numbers and words that contain numbers, the elimination of punctuation marks and the shifting of all capital letters into lower case.
2. Stop-words removal – stop-words are auxiliary words like prepositions, pronouns, interjections and conjunctions, which carry negligible semantic information, but which are often encountered due to their language function. By removing those words, we decrease the total number of different words in the corpus. The result is that the semantic space is reduced and the accuracy of the semantic algorithms is increased, since the links between semantically important words become more emphasized. The stop-word list utilized in our Serbian language system was formed by gathering the most frequent words from the text corpus [14]. Given that the corpus that we used contains general knowledge taken from an encyclopedia, we expected that the individual word frequencies within it would relatively accurately reflect the general word frequencies in the language itself. The information about word frequencies in the corpus which is gathered in this step is saved for later use in calculating various term frequencies (TFs) for each word.
3. Stemming – the employed corpus of article abstracts is coded in the UTF-8 format and is written partially in Cyrillic and partially in Latin alphabet, since Serbian is a digraphic language that can be written in either script. This posed a problem, since the utilized stemmer demands that input strings be formatted in a special dual1 version of the ASCII coding system in which every character with a diacritic ought to be represented by a combination of two plain ASCII characters. Hence, in order to preserve compatibility with the stemmer module, we had to construct a converter that would process the corpus text into this special coding system, before sending it to the stemmer.

Corpus processing consists of choosing an algorithm for the creation of the semantic space and supplying it with the preprocessed corpus text.

Corpus post-processing deals with the reduction of context vector dimension, i.e. with the reduction of the co-occurrence matrix, which scales down the computational complexity of determining sentence similarity. Each algorithm has its own post-processing routine which is encapsulated within the algorithm, as defined in the *S-Space* package.

A separate part of post-processing involves the calculation of min-max TFs for all words found within the corpus. We obtain the min-max normalization of TF for each word by using the following formula:

$$TF_{\min-\max} = \frac{TF_{\log}}{\max(TF_{\log})} \qquad (1)$$

where TF_{log} is the log-number of times the given word appears in the corpus, and $max(TF_{log})$ is the log-number of times the most frequently occurring word in the corpus appears in it. The TF log-number is calculated as follows:

$$TF_{\log} = -\log\left(\frac{TF_{count}}{n}\right) \qquad (2)$$

where TF_{count} is the occurrence count of the given word in the corpus, and n is the total number of words in the corpus.

Storing the semantic space on a hard drive is necessary in order to avoid recreating the whole semantic space at every program startup. Saving it as a single file is not practical due to poor performance achieved when trying to access a random part of a huge file. Hence, we decided to store the semantic space within a database, which utilizes an index structure, therefore providing an acceptable level of speed when searching and accessing data. The database holds two tables – one for the semantic space built by COALS, and another for the space created by RI. Both tables share the same structure, consisting of key, word, and context vector columns. Context vectors are recorded as long string variables. An additional database table is dedicated to storing the min-max TF values for all corpus words.

4. System Operation

Determining the similarity of texts is done by using a bag-of-words approach based on a modification of [3], in which for each word in the shorter text P we find the most similar matching word in the longer text R. Our modification is inspired by the method proposed in [2], which relies on a similar bag-of-words approach, but uses word specificity to weigh the word similarity. In the text to follow we will first analyze issues related to methods [2] and [3], and then we will introduce our algorithm.

The problem of method [2] is that it has a tendency to overestimate text similarity because it allows multiple words from one text to be paired up with a single word in the other text. For example, let us consider the following text segments:

1. botanical gardens
2. plantation house

According to [2], both words in the first segment would be linked up to the first word (plantation) of the second segment. This would happen because method [2] pairs up each word in the first segment with its most similar word in the second segment, and vice versa, regardless of whether a word has already been paired up. Due to this practice, many dissimilar sentence pairs will be, in fact, deemed similar. To some extent this problem is alleviated in [2] by measuring similarity only between words within the same part-of-speech class. However, as we have already noted, advanced NLP tools like part-of-speech taggers are not available in our case.

On the other hand, method [3] circumvents this issue by eliminating all paired up words from further consideration in the pairing process. In the previous example, the pair (gardens, plantation) would probably have the highest similarity, and would therefore be included in the overall score and then discarded. The similarity score would then be calculated for the remaining pair (botanical, house) after which it would be added to the similarity sum. This approach thus gives a more realistic result of the overall text similarity measure.

Still, method [2] improves its results by taking into account word specificity in the word similarity weighing. Therefore, we have decided to combine these two approaches by utilizing a normalized term frequency weighing scheme. To measure the similarity of a word pair, we first determine its string similarity, and then the semantic one. This technique is then enhanced by using a term frequency ponderation. Particularly, for the ponderation of a word pair similarity score, we employ the following normalization formula:

$$TF_{norm} = 2^{(TF1_{\min-\max} \times TF2_{\min-\max})-1} \tag{3}$$

where $TF1_{min\text{-}max}$ is the min-max TF of the first word, and $TF2_{min\text{-}max}$ is the min-max TF of the second word, as defined in (1). By using this TF normalization technique, we are able to obtain normalized values for each word pair in the range [0.5, 1]. Very common words will have a low TF_{log} value which means that their min-max TFs will be close to 0. This will, in turn, mean that all TF_{norm} values for those words will be close to 0.5. On the other hand, rare words will have a high TF_{log} value, allowing them to have a min-max TF value close or equal to 1. Word pairs which consist of rare words will, thus, have a TF_{norm} value close or equal to 1. In effect, when using normalized TFs in a similarity score ponderation, word pairs made up of rare words will retain their full similarity score, while the scores of word pairs containing common words will be reduced by as much as 50%.

The system operation workflow is shown in Fig. 2. It depicts the stages of determining the numerical similarity score of two given short texts. We will describe the working of the system phase-by-phase, and demonstrate it by utilizing the COALS algorithm on a pair of sentences taken from our Serbian language paraphrase corpus. These sentences and their English translations are shown in Fig. 3.

Text preprocessing begins with the text cleaning procedure described in the *Corpus preprocessing* part of Section 3. It continues with the removal of stop words from the given texts, which are then converted into the aforementioned dual1 coding system. The remaining words from both texts are then stemmed. After this stage, our example sentence pair has the form shown in Fig. 4. We can see that the number of tokens in the first text *P*, which we refer to as its length *m,* is 12, while the length *n* of the second text *R* is 17.

Processing of words appearing in both texts starts with the identification of those words. In our example, those words are the following: podizn, spusxtajucy, platfor, osob, ju, pescacxk, prolaz, terazij. We consider their similarity scores to be equal to their normalized term frequencies, which are calculated using (3) as follows:

$$TF_{norm} = 2^{TF_{\min-\max}{}^{2}-1} \tag{4}$$

We thus treat words appearing in both texts as a word pair which consists of two identical words. We automatically assign a min-max TF value of 1 to words like "podizn" which have not appeared in the corpus material. Table 1 displays the min-max term frequencies and similarity scores of these words. After calculating their similarity scores, we add them up into a similarity sum S_{same}, and then discard these words from further consideration. In our example, the value of the S_{same} sum would be 6.319.

String similarity matrix construction creates a matrix in which every cell is occupied by a numerical value between 0 and 1 representing the string similarity between the column-word and the row-word. The rows of the matrix are used for the remaining words from the first text, while the columns represent the remaining words from the second text. A zero value denotes entirely different string contents, while a value of one indicates a perfect string match. In Section 2 we have presented the approach which we have used to calculate string similarity. Table 2 shows the string similarity matrix which corresponds to the example sentence pair.

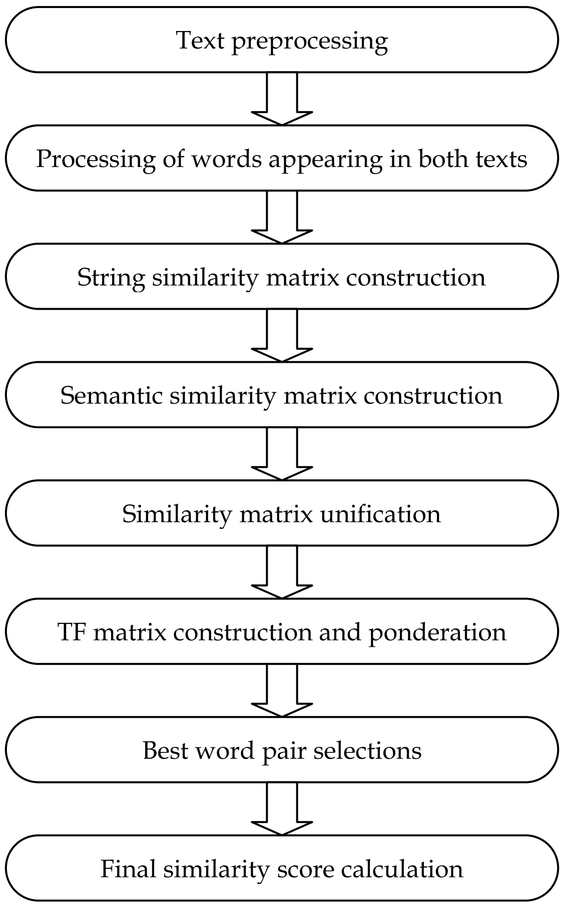
Text preprocessing
Processing of words appearing in both texts
String similarity matrix construction
Semantic similarity matrix construction
Similarity matrix unification
TF matrix construction and ponderation
Best word pair selections
Final similarity score calculation

Fig. 2. System operation workflow. It depicts the stages of determining the numerical similarity score of two given short texts.

Text segment 1:
Podizno-spuštajuća platforma za osobe sa posebnim potrebama je juče puštena u rad u pešačkim prolazima na Terazijama.
A platform stairlift for people with special needs was put into operation yesterday in the pedestrian passageways at Terazije.

Text segment 2:
Osobe sa invaliditetom i svi koji imaju poteškoće sa kretanjem od juče mogu da koriste podizno-spuštajuću platformu u podzemnom pešačkom prolazu na Terazijama u Beogradu.
Since yesterday, people with disabilities and all who have mobility difficulties can use the platform stairlift in the underground pedestrian passageway at Terazije in Belgrade.

Fig. 3. An example sentence pair taken from the Serbian language paraphrase corpus. Original sentences in Serbian are presented alongside their translations into English.

Text segment 1:
podizn spusxtajucy platfor osob posebn potre ju pusx rad pesxacxk prolaz terazij

Text segment 2:
osob invaliditet svi ima potesxkocy kretany ju mo kor podizn spusxtajucy platfor podzemn pesxacxk prolaz terazij beograd

Fig. 4. The example sentence pair after the text preprocessing stage.

Semantic similarity matrix construction creates a matrix in which every cell is occupied by a numerical value between 0 and 1 representing the semantic similarity between the column-word and the row-word. The rows of the matrix are used for the words from the first text, while the columns represent the words from the second text. Similar to the string similarity measurement, a zero value denotes entirely different semantic contents, while a value of one indicates a perfect semantic match. We gain the semantic similarity of words in a pair by calculating the cosine similarity of their context vectors, read from the database. Table 3 shows the semantic similarity matrix obtained by utilizing the COALS algorithm on our example sentence pair.

Similarity matrix unification combines the string and the semantic similarity matrices into one by multiplying their values by a certain ponderation factor and adding them up. Experiments done on our Serbian language system have demonstrated that, in this case, optimal accuracy is attained by using ponderation values of 0.45 and 0.55 for the string and semantic similarity scores, respectively. Table 4 shows the unified similarity matrix which corresponds to our example sentence pair.

The *TF matrix construction and ponderation* phase begins with the creation of a normalized term frequency matrix in which every cell is occupied by a normalized TF value calculated using (3). Table 5 shows the normalized term frequency matrix which corresponds to our example sentence pair. The final similarity matrix, shown in Table 6, is gained by multiplying each cell of the unified similarity matrix with the corresponding cell of the normalized TF matrix.

TABLE 1

MIN-MAX TERM FREQUENCIES AND SIMILARITY SCORES OF WORDS APPEARING IN BOTH TEXTS

Word	podizn	spusxtajucy	platfor	osob	ju	pescacxk	prolaz	terazij
TF$_{min\text{-}max}$	1.000	1.000	0.733	0.548	0.561	0.849	0.643	0.889
Similarity score	1.000	1.000	0.726	0.616	0.622	0.824	0.666	0.865

TABLE 2

STRING SIMILARITY MATRIX

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor	podzemn	beograd
posebn	0.010	0.037	0.000	0.094	0.039	0.055	0.037	0.189	0.016
potre	0.030	0.000	0.000	0.224	0.075	0.066	0.110	0.160	0.047
pusx	0.000	0.055	0.000	0.116	0.000	0.000	0.000	0.035	0.000
rad	0.050	0.000	0.073	0.000	0.079	0.000	0.073	0.031	0.283

TABLE 3

SEMANTIC SIMILARITY MATRIX

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor	podzemn	beograd
posebn	0.000	0.127	0.211	0.000	0.132	0.181	0.195	0.083	0.093
potre	0.000	0.105	0.152	0.000	0.097	0.139	0.106	0.062	0.047
pusx	0.000	0.029	0.054	0.000	0.027	0.034	0.048	0.020	0.043
rad	0.000	0.131	0.191	0.000	0.150	0.200	0.204	0.085	0.198

TABLE 4

UNIFIED SIMILARITY MATRIX

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor	podzemn	beograd
posebn	0.005	0.086	0.116	0.042	0.090	0.124	0.124	0.131	0.058
potre	0.014	0.058	0.084	0.101	0.087	0.106	0.108	0.106	0.047
pusx	0.000	0.040	0.030	0.052	0.015	0.019	0.026	0.027	0.024
rad	0.023	0.072	0.138	0.000	0.118	0.110	0.145	0.061	0.236

TABLE 5

NORMALIZED TERM FREQUENCY MATRIX

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor	podzemn	beograd
posebn	0.711	0.640	0.593	0.707	0.646	0.580	0.620	0.659	0.598
potre	0.756	0.669	0.611	0.751	0.676	0.595	0.644	0.691	0.617
pusx	0.810	0.702	0.632	0.804	0.711	0.613	0.672	0.730	0.640
rad	0.679	0.620	0.580	0.676	0.625	0.569	0.603	0.636	0.585

Best word pair selections start with the final similarity matrix. The goal is to match words across the two texts according to their mutual similarity score. Hence, we search for the highest value within the final similarity matrix, and add it to a similarity sum $S_{different}$. We then remove the row and the column of the matrix to which the selected cell belonged, thereby discarding all other word pairs in which words from the chosen pair appeared. We repeat this procedure until there are no more rows and/or columns left in the matrix. This procedure is demonstrated step-by-step in tables 6-11 where the highest value cell is marked in bold script.

Table 6 - The highest similarity score in our final similarity matrix is the one for the (rad, beograd) word pair. After this first step the $S_{different}$ similarity sum has a value of 0.138.

Table 7 - The highest similarity score in the step 2 matrix is the one for the (posebn, podzemn) word pair. After the second step, the $S_{different}$ similarity sum has a value of 0.224.

Table 8 - The highest similarity score in the step 3 matrix is the one for the (potre, potesxkocy) word pair. After the third step, the $S_{different}$ similarity sum has a value of 0.3.

Table 9 - The highest similarity score in the step 4 matrix is the one for the (pusx, svi) word pair. After the fourth step, the $S_{different}$ similarity sum reaches a final value of 0.328.

The *final similarity score calculation* is performed by utilizing the following formula:

$$S(P,R) = \frac{(S_{same} + S_{different}) \times (m+n)}{2mn} \qquad (5)$$

In other words, the final similarity score *S(P,R)* is gained by summing up the similarity scores of words that appear in both texts (S_{same}) and the scores of word pairs formed from words unique to one of the texts ($S_{different}$). Lastly, this sum is multiplied by a reciprocal harmonic mean function of the lengths of both texts, so as to achieve a final text similarity score between 0 and 1. For our example sentence pair, the final text similarity score will have the following value:

$$S(P,R) = \frac{(6.319 + 0.328) \times (12+17)}{2 \times 12 \times 17} = 0.472$$

As will be shown in section 6, the optimal threshold value for our system is 0.407 when using the COALS algorithm. Therefore, the system would correctly identify the sentence pair from the example as one in which sentences are highly semantically related.

TABLE 6

FINAL SIMILARITY MATRIX

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor	podzemn	beograd
posebn	0.003	0.055	0.069	0.030	0.058	0.072	0.077	0.086	0.035
potre	0.010	0.038	0.051	0.076	0.059	0.063	0.069	0.073	0.029
pusx	0.000	0.028	0.019	0.042	0.010	0.011	0.018	0.020	0.015
rad	0.015	0.045	0.080	0.000	0.074	0.063	0.088	0.039	**0.138**

TABLE 7

BEST WORD PAIR SELECTION – STEP 2

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor	podzemn
posebn	0.003	0.055	0.069	0.030	0.058	0.072	0.077	**0.086**
potre	0.010	0.038	0.051	0.076	0.059	0.063	0.069	0.073
pusx	0.000	0.028	0.019	0.042	0.010	0.011	0.018	0.020

TABLE 8

BEST WORD PAIR SELECTION – STEP 3

Word	invaliditet	svi	ima	potesxkocy	kretany	mo	kor
potre	0.010	0.038	0.051	**0.076**	0.059	0.063	0.069
pusx	0.000	0.028	0.019	0.042	0.010	0.011	0.018

TABLE 9

BEST WORD PAIR SELECTION – STEP 4

Word	invaliditet	svi	ima	kretany	mo	kor
pusx	0.000	**0.028**	0.019	0.010	0.011	0.018

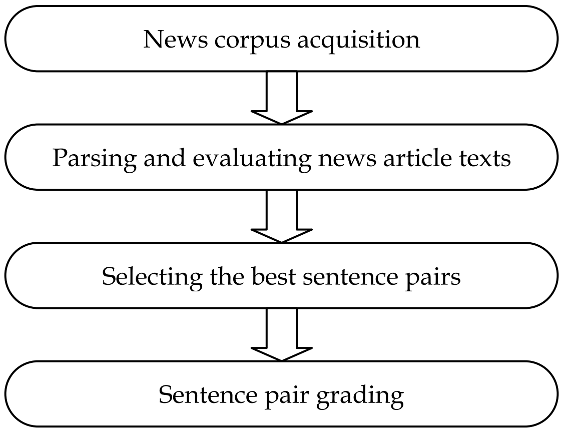
News corpus acquisition
Parsing and evaluating news article texts
Selecting the best sentence pairs
Sentence pair grading

Fig. 5. Evaluation resources creation workflow. It depicts the stages of paraphrase corpus creation.

5. Evaluation Resources

The main issue encountered in system evaluation is finding the work parameter values for which the system achieves maximal accuracy, which is the greatest level of matching between the system-designated similarity scores and the similarity grades that a human would give. In this regard, it is important to determine an optimal system similarity threshold which would serve as a boundary, such that all score values above it could be treated as an assessment of semantic similarity, and all values below it as an appraisal of semantic dissimilitude. The optimal threshold value is one for which the system reaches its maximal possible accuracy. In order to find this threshold, a substantially large set of sentence pairs is needed. Sentences in a pair ought to overlap in their semantic content – some of them should represent actual paraphrases, while others should only be semantically related to an extent.

Nevertheless, a collection of such sentence pairs is usually called a paraphrase corpus. After manually assigning a binary grade to every pair in this corpus, it is possible to compare human-given grades and system similarity scores for each pair and, by using statistical analysis, to determine the optimal threshold value.

As a starting point, we reviewed the insights gained during the construction of the MSRPC [7]. The basic approach to building such a sentence pair corpus lies in the exploitation of a journalistic convention by which the first few sentences of a news article usually contain a short summarization of the article content. Therefore, gathering multiple articles dealing with the same subject matter from different news sources and extracting their starting sentences is a good way of finding sentence pairs that are probably semantically comparable. The workflow of the evaluation resources creation is shown in Fig. 5. It depicts how a paraphrase corpus is created and processed before being put to use.

News corpus acquisition deals with finding a free and publicly available web news archive that allows for easy access to news articles for a desired date. A prime solution for our Serbian language system was the website www.vesti.rs. This site is a news aggregator which collects articles from all major media outlets in Serbia, including not only TV stations and newspapers, but Internet magazines and portals as well. In total, it makes use of over 210 different news sources. We decided to process only the top news stories for each date, because they have the highest probability of being covered by multiple outlets. Moreover, this ensured the procurement of various sentence pairs dealing with all kinds of topics, which prevented the paraphrase corpus from becoming too focused on a single point. We compiled the top news stories from 2010 and from the first seven months of 2011 so as to provide us with enough material for corpus construction.

Parsing and evaluating news article texts is a necessary step after the collection of all news reports. The cleaning and parsing of obtained sentences is a particularly problematic task due to their completely unstructured format, the usage of periods not being limited to sentence endings, and the various forms of unnecessary information which ought to be removed (e.g. information about the news agency, the location of the event, etc.). We assigned several descriptive attributes to every sentence pair that met certain minimal criteria concerning sentence length and the number of semantically significant words they contain. These attributes were used later on to determine the best sentence pair for every article. The attributes we used are: the number of *long* words in the shorter sentence, the number of *long* words in the longer sentence, and the number of *long* words which appear in both sentences, without counting word repetitions. “Long words” is a term we use for those words whose length makes them almost certainly semantically relevant. The minimal length required to deem a word semantically relevant depends, of course, on the language in question. For example, in Serbian there are many five-letter prepositions and personal pronouns. Hence, we concluded that, in this case, we should set a long word minimal length of six letters.

Two poor quality sentence pairs

2.1 Građani Južnog Sudana masovno glasali tokom prvog dana jednonedeljnog referenduma o nezavisnosti te oblasti od vlasti u Kartumu.
The citizens of South Sudan voted en masse during the first day of the one week long referendum on the independence of that region from the Khartoum government.

2.2 Dan uoči referenduma o nezavisnosti južnog Sudana, njegov lider Salva Kir rekao je da ne postoji alternativa mirnoj koegzistenciji između severa i juga.
A day before the referendum on the independence of South Sudan, its leader Salva Kiir said that there is no alternative to a peaceful coexistence between the north and the south.

3.1 Grčka je otkazala naručenih 12,3 miliona doza vakcine protiv novog gripa i traži vraćanje uplaćenog avansa.
Greece has called off the ordered 12.3 million doses of the vaccine against the new flu and seeks to reclaim the money paid in advance.

3.2 Grčke vlasti su otkazale naručenih 12,3 miliona doza vakcine protiv novog gripa A (H1N1) i zatražile vraćanje uplaćenog avansa.
The Greek authorities have called off the ordered 12.3 million doses of the vaccine against the new flu A (H1N1) and have requested a return of the money paid in advance.

Fig. 6. Examples of good and poor quality sentence pairs. Original sentences in Serbian are presented alongside their translations into English.

Selecting the best sentence pairs out of all the pairs coupled to articles is performed by choosing the pair with the highest chances of being the best one for each article. This selection process required the calculation of a numerical quality score for each pair on the basis of its given attributes. An excellent quality sentence pair is one whose sentences provide the same semantic information, but conveyed in wholly different manners. Sentences regarded as being of poor quality are (a) those that are likely semantically diverse, or (b) those that are indeed semantically equivalent, but only due to a great level of string similarity between them. An example of a good quality sentence pair is already shown in Fig. 3. Examples of two poor quality pairs for cases (a) and (b) are given consecutively in Fig. 6. In conclusion, the main goal of the paraphrase corpus creation process is to achieve a suitably high percentage of semantically similar sentence pairs. While doing this, it is important to avoid as much as possible the common examples of semantic equivalence originating in a high level of string matching.

We observed that giving preference to sentences of similar lengths that have around 50% of the same words yields the best results. When sentences in a pair have widely different lengths, the likelihood of them being real paraphrases is reduced. We also noticed that a high percentage of the same words appearing in both sentences greatly increases the chance that one sentence is a mere repetition of the other, with only some slight, semantically irrelevant new information added. On the other hand, a low percentage most often means that the pair is built out of two semantically diverse parts of news articles. Additionally, we assigned a greater weight to shorter sentence pairs, since for them each word plays a proportionately bigger part in the creation of the final score.

Sentence pair grading is done manually. During the grading process, for some sentence pairs it was immediately clear what grade they should be given, but there were also pairs whose semantic information are somewhat similar, but not entirely. Since there were many such occurrences, it was obligatory to establish certain grading guidelines which would ensure the greatest possible uniformity of grading criteria. These guidelines are presented in Fig. 7, in pseudocode form. Furthermore, in this stage we also corrected all typographical and other errors that originated in source text imperfections. Grading was performed by a single human judge, after which a second judge rated a random selection of corpus sentence pairs. This selection's size equaled 30% of the total corpus. The double-checking was done in order to estimate the inter-rater agreement, which is an important parameter since it dictates the upper bound for system accuracy. The inter-rater agreement observed on our control selection of sentence pairs was 78.27%.

Finally, using this procedure to create and prepare evaluation resources, we were able to acquire a Serbian language corpus consisting of 1194 sentence pairs. It includes 553 semantically equivalent pairs, and 641 pairs marked as semantically divergent. Percentage-wise, the semantically equivalent pairs comprise 46.31%, and the semantically divergent 53.69%.

```
if semantic contents of sentences are completely different then
  assign grade 0;
else begin
  remove from consideration differences arising from the use of pronominal and noun phrase
  anaphora;
  if it is unclear whether sentences refer to the same event then
    assign grade 0;
  else if the sentences have the same subject matter but employ different rhetorical structures
  then
    assign grade 0;
  else if the sentences have the same subject matter but emphasize different aspects of it then
    assign grade 0;
  else begin
    if one sentence represents a semantic subset of the other then
    begin
      extract the information present only in the semantically richer sentence;
      if that information is not particularly important then
        assign grade 1;
      else
        assign grade 0;
    end;
    else begin
      compare the subjects, predicates and other important semantic features of both sentences;
      if there is a semantic discrepancy then
        assign grade 0;
      else
        assign grade 1;
    end;
  end;
end;
```

Fig. 7. Guidelines for sentence pair grading. These guidelines are necessary in order to ensure the general uniformity of grading criteria.

6. Evaluation Results

The process of finding an optimal threshold value is initially performed on a larger sentence pair set called a Training data set. The obtained threshold value is afterwards verified on an independent Test data set, and, if it produces a satisfactory accuracy on it as well, then that value is adopted as the optimal one. The Training set for our Serbian language system is made up of 835 randomly selected sentence pairs (70% of the subtotal), while the Test set comprises 359 pairs (30% of the subtotal).

Various accuracies achieved during the scoring of the Serbian paraphrase corpus using a range of threshold values and the COALS algorithm are shown in Table 10. Accuracy represents the ratio of the number of correctly graded sentence pairs and the total number of pairs in the corpus. The threshold value was increased in steps of 0.001, so as to attain the maximal possible accuracy. The highest percentage of correctly identified pairs is gained by using a threshold value of 0.407, which leads to a system accuracy of 76.6%.

The expression True Positives (TP) stands for those sentence pairs which are paraphrases and are correctly identified as such by the algorithm. True Negatives (TN) encompasses dissimilar sentence pairs which are correctly recognized by the algorithm. False Positives (FP) consists of semantically dissimilar sentence pairs erroneously marked as paraphrases. False Negatives (FN) represents sentence pairs which are paraphrases but are incorrectly assessed as semantically different.

On the other hand, accuracies achieved during the scoring of the paraphrase corpus using a range of threshold values and the RI algorithm are shown in Table 11. The optimal threshold value is 0.417, which leads to a system accuracy of 76.6%

Both COALS and RI reach a similar level of accuracy. However, timing tests performed during the evaluation process showed COALS to be consistently faster. This is probably due to the relatively small size of the text corpus utilized for semantic space creation, which prevented RI, an algorithm primarily devised for large corpora, from showing its strong suit in this regard.

In the information retrieval theory, precision (P), recall (R) and F-measure (F) are measures that are used extensively. They are calculated according to the following expressions:

$$P = \frac{TP}{TP+FP} \qquad R = \frac{TP}{TP+FN} \qquad F = \frac{2PR}{P+R}$$

In the context of STSS, precision can be understood as the ratio of the number of correctly identified paraphrase sentence pairs and the total number of pairs marked as paraphrases by the algorithm. Recall represents the ratio between the correctly identified paraphrase sentence pairs and the real number of paraphrase sentence pairs in the corpus. F-measure is computed as the harmonic mean of precision and recall.

Table 12 displays a comparison of the main characteristics of some previously described methods, as well as the approach we propose. The first three rows (1-3) present results of Mihalcea et al. [2], Islam and Inkpen [3], and Li et al. [6], which are evaluated on the Microsoft Research Paraphrase Corpus (MSRPC). The last three rows (4-6) present results evaluated on our Serbian language paraphrase corpus (SRB). In order to perform a comparison, we implemented the steps of the method proposed in [3], but we used COALS and RI for measuring the semantic similarity of a word pair.

We obtained the best results in our implementation of [3] by using COALS, and we therefore used those results as a baseline, shown here in row 4. The last two rows (5-6) contain results of the LInSTSS approach that we propose, in both COALS and RI variants.

The optimal threshold values for COALS and RI gravitate toward 0.4, which is also the case with [6]. Also, our implementation of method [3] evaluated on the SRB corpus has the same optimal threshold value as the original one evaluated on MSRPC.

TABLE 10

AN OVERVIEW OF SENTENCE PAIR SCORES GAINED BY USING THE COALS ALGORITHM

Threshold	Sentences correctly identified by the COALS algorithm					
	Semantically equivalent $\frac{TP}{TP+FN}$		Semantically diverse $\frac{TN}{TN+FP}$		Overall $\frac{TP+TN}{TP+FP+TN+FN}$	
	Training set	Test set	Training set	Test set	Training set	Test set
0.1	100%	100%	0%	0%	46.23%	46.52%
0.2	99.48%	100%	2.67%	2.6%	47.43%	47.91%
0.3	96.11%	97.01%	33.63%	38.02%	62.51%	65.46%
0.4	72.02%	71.86%	75.28%	77.6%	73.77%	74.93%
0.407	70.73%	71.26%	77.28%	81.25%	**74.25%**	**76.6%**
0.5	36.79%	35.93%	96.66%	96.88%	68.38%	68.52%
0.6	8.03%	8.98%	99.78%	100%	57.37%	57.66%
0.7	0.52%	0%	100%	100%	54.01%	53.48%
0.8	0%	0%	100%	100%	53.77%	53.48%
0.9	0%	0%	100%	100%	53.77%	53.48%

TABLE 11

AN OVERVIEW OF SENTENCE PAIR SCORES GAINED BY USING THE RI ALGORITHM

Threshold	Sentences correctly identified by the RI algorithm					
	Semantically equivalent $\frac{TP}{TP+FN}$		Semantically diverse $\frac{TN}{TN+FP}$		Overall $\frac{TP+TN}{TP+FP+TN+FN}$	
	Training set	Test set	Training set	Test set	Training set	Test set
0.1	100%	100%	0%	0%	46.23%	46.52%
0.2	99.48%	100%	1.56%	1.56%	46.83%	47.35%
0.3	97.41%	97.6%	30.29%	33.85%	61.32%	63.51%
0.4	73.58%	73.65%	72.38%	75.52%	72.93%	74.65%
0.417	69.95%	69.46%	77.95%	82.81%	**74.25%**	**76.6%**
0.5	38.08%	37.72%	95.99%	96.35%	69.22%	69.08%
0.6	8.55%	9.58%	99.78%	100%	57.6%	57.94%
0.7	0.52%	0%	100%	100%	54.01%	53.48%
0.8	0%	0%	100%	100%	53.77%	53.48%
0.9	0%	0%	100%	100%	53.77%	53.48%

TABLE 12

A COMPARISON OF THE CHARACTERISTICS OF VARIOUS STSS METHODS

	Method	Optimal threshold	Accuracy	Precision	Recall	F-Measure
1.	Mihalcea et al. (MSRPC)	0.5	70.3%	69.6%	97.7%	81.3%
2.	Islam and Inkpen (MSRPC)	0.6	72.64%	74.65%	89.13%	81.25%
3.	Li et al. (MSRPC)	0.4	70.8%	70.3%	97.4%	81.6%
4.	Baseline (SRB)	0.599	76.04%	**82.93%**	61.08%	70.34%
5.	LInSTSS - COALS (SRB)	0.407	**76.6%**	76.77%	**71.26%**	**73.91%**
6.	LInSTSS - RI (SRB)	0.417	76.6%	77.85%	69.46%	73.42%

Our approach leads to the highest recorded accuracy that is slightly higher than the baseline and which is just a few percentages short of the maximal possible system accuracy, given the inter-rater agreement (78.27%). It also has a significantly higher recall, at the cost of lower precision. The difference between these two measures is reduced in respect to the baseline method, so it improves the overall F-measure value. Also, since we used word specificity to weigh differently the similarity of word pairs, our approach is to some extent strict in assigning the semantic similarity label, which can be noted by comparing values close to the optimal threshold in the *Semantically equivalent* and *Semantically diverse* columns of tables 10 and 11.

7. CONCLUSION

In this paper we described the LInSTSS approach to building a software system for determining the semantic similarity of short texts. This approach is particularly useful for languages where other, alternative system-building methods are not applicable. We discussed the process of constructing such a system, and we showed its operation, as well as the procedure of gathering and preparing resources for its evaluation. Moreover, we presented the results gained from the evaluation of a system which works with texts written in Serbian.

One of the key traits of the proposed system-building methodology is its modularity, which should allow other researchers to easily adjust to the specificities of their own language. We plan to test our approach on another language ourselves, in order to further verify its wide applicability.

Increases in the accuracy of our proposed system can be achieved by using a larger text corpus for semantic space creation. A bigger corpus means a greater number of words within it, which consequently leads to a higher accuracy of the co-occurrence matrix. Another approach to increasing the system's accuracy would be the improvement of preprocessing techniques, achieved chiefly through refinements of the stemmer module. Evaluation quality could be enhanced by an enlargement of the paraphrase corpus which would strengthen the representability of the evaluation results.

ACKNOWLEDGMENT

The work presented here was partially supported by the Serbian Ministry of Education and Science (projects III 44006, III 44009, and III 32047). The authors would also like to thank the anonymous reviewers for their helpful and constructive comments.

REFERENCES

[1] M. Mohler, R. Mihalcea, Text-to-text Semantic Similarity for Automatic Short Answer Grading, European Chapter of the Association for Computational Linguistics (2009) 567-575.

[2] R. Mihalcea, C. Corley, C. Strapparava, Corpus-based and Knowledge-based Measures of Text Semantic Similarity, Proceedings of the National Conference on Artificial Intelligence 21 (1) (2006) 775-780.

[3] A. Islam, D. Inkpen, Semantic Text Similarity Using Corpus-based Word Similarity and String Similarity, ACM Transactions on Knowledge Discovery from Data 2 (2) (2008) 1-25.

[4] R. Wang, G. Neumann, Recognizing Textual Entailment Using Sentence Similarity Based on Dependency Tree Skeletons, ACL-PASCAL Workshop on Textual Entailment and Paraphrasing (2007) 36–41.

[5] J. Oliva, J.I. Serrano, M.D. del Castillo, Á. Iglesias, SyMSS: A Syntax-based Measure for Short-text Semantic Similarity, Data & Knowledge Engineering 70 (4) (2011) 390-405.

[6] L. Li, Y. Zhou, B. Yuan, J. Wang, X. Hu, Sentence Similarity Measurement Based on Shallow Parsing, Fuzzy Systems and Knowledge Discovery 7 (2009) 487–491.

[7] B. Dolan, C. Quirk, C. Brockett, Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources, Proceedings of the 20th International Conference on Computational Linguistics, 2004.

[8] B. Furlan, V. Sivački, D. Jovanović, B. Nikolić, Comparable Evaluation of Contemporary Corpus-Based and Knowledge-Based Semantic Similarity Measures of Short Texts, Journal of Information Technology and Applications 1 (1) (2011) 65-71.

[9] V. Kešelj, D. Šipka, A Suffix Subsumption-based Approach to Building Stemmers and Lemmatizers for Highly Inflectional Languages with Sparse Resources, INFOTHECA – Journal of Informatics and Librarianship 9 (1-2) (2008).

[10] D. Jurgens, K. Stevens, The S-Space Package: An Open Source Package for Word Space Models, Proceedings of the ACL System Demonstrations, Uppsala, Sweden, 2010.

[11] D.L.T. Rohde, L.M. Gonnerman, D.C. Plaut, An Improved Method for Deriving Word Meaning from Lexical Co-Occurrence, Cognitive Psychology 7 (2004) 573-605.

[12] M. Sahlgren, An Introduction to Random Indexing, Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, Copenhagen, Denmark, 2005.

[13] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, R. Harshman, Indexing by Latent Semantic Analysis, Journal of the American Society for Information Science 41 (6) (1990).

[14] R.T. Lo, B. He, I. Ounis, Automatically Building a Stopword List for an Information Retrieval System, 5th Dutch-Belgium Information Retrieval Workshop, Utrecht, Netherlands, 2005.