MULTI-AGENT SYSTEM FOR SIMILARITY SEARCH IN STRING SETS

Katarzyna Harężlak, Michał Sala

***Abstract**: The aim of the paper is to present the assumptions and the architecture of the system for searching similarity in string sets. During the research all the required steps of a procedure of text documents processing which includes text extraction, pruning, stemming and lemmatization were analysed. Models of a text documents' description and the method of creating a vector of features were developed as well. This vector consists, inter alia, of chosen words and the number of their occurrences. The process of the text analysis is supported by a set of various dictionaries. These are Stop-words, Domain and Lemma dictionaries and all of them were considered in the context of the Polish language. Because the Lemma dictionary is supposed to consist of many entries, the efficient method of its access optimisation was elaborated. Various measures used for calculating degree of a text documents similarity were studied too. Moreover, the method for determining the quality of user queries and text documents adjustment were proposed.*

The system was realized in accordance with the idea of multi-agent systems. Its functionality is ensured by the set of agents acting on the basis of separate threads. In the research, tests of the system work efficiency were also performed.

***Keywords**: agent systems, text similarity search*

***ACM Classification Keywords**: I.7 Document And Text Processing*

Introduction

Knowledge is an inseparable, continuously extended, element of the modern life. Fast, regarding various areas, science development results in providing us with many books, articles and web information sources which, in the era of widespread use of computers and global networks, makes access to knowledge unlimited. For this reason, while searching for information to understand a given issue better, many sources should be analysed. Making this process useful and effective entails the need of creating methods ensuring fast access to particular information.

The simplest solutions, based on metadata or patterns searching, have two disadvantages. The result of their action consists of too much information, which does not match a given pattern – in the worst case scenario potentially valuable documents can not include words defined in the search criteria. Next drawback is a possibility of important information loss – this situation can take place when a searched document is determined by too small or inappropriate set of metadata. What is more, two different papers indexed by the same set of words can be classified as similar documents whereas in reality refer to different areas.

More advanced systems use more precise analysis including text comparison and classification [Bollacker, 1998, Aggarwal 2001]. In this case the notion of documents' similarity, relying on defining correspondence degree between documents being compared, has been introduced. Owing to this rate, the problems of obtaining too large result set and possibility of information loss can be reduced. The text semantic similarity is mainly determined on the basis of documents' contents analysis and comparison. Considering this idea on the high abstraction level two main methods can be distinguished. The first one uses text matching to point out the same part of documents. In this case the similarity is determined by the number and quantity of repeated contents. In the second method sets of features, describing text content, are generated, which are subsequently used by functions calculating the distance between two analyzed documents.

Documents written in the natural language, from the computer analysis point of view, feature high redundancy of information. Variety of declensions, words or punctuation marks are useless for methods used for text analysis. Therefore, for their need, digital representation of document's description allowing for mathematical analysis is required. One of the simplest examples is a unigram model, which assumes that documents are described by vectors whose values represent the existence of given words. The effective vector generation, for a particular document, requires carrying out preliminary processing of its content. This activities are responsible for removing unimportant text elements, which have no influence on a text comparison and changing all declensions to basic forms of particular words [Strömbäck, 2005, Dąbrowski, 1978,Smirnov, 2008].

The aim of this paper is to present the assumptions and the architecture of the system for searching similarity in string sets. The system was developed in accordance with the idea of multi-agent systems. Its functionality is ensured by the set of agents acting on the basis of separate threads.

The Multi-Agent Systems

Multi-Agent System – MAS [Bigus, 2001, Wooldridge, 2009] is a system comprising of many intelligent agents which collaborate to solve a given goal. Agents usually are defined as units making autonomous decisions on behalf of a user or a superior system to realize assigned them functions. Their activity rely on continuous observation of objects belonging to a specific environment and choosing, on the basis of the states of these objects, appropriate actions influencing the environment.

Agents are characterized, dependently on their applications, by various features. From this paper point of view these are reactivity, autonomy, cooperation and coordination.
It means that agents have to observe their environments and respond to occurring changes, individually decide which activities should be taken and exchange needed resources to complete chosen actions and coordinate their execution.

There are many frameworks supporting agent system building but in case of presented research C# language and .NET class were used.

The System Architecture

The vision of the system, with taking idea of agent system into account, is presented in the figure 1a. It consists of three elements: the **Environment**, **Superior Layer** and **Agents**, divided into two categories. The first category comprises of Processing Agents, which produce data used by the second group in the process of documents comparison. They observe an environment detecting presence of source documents to be analysed. In case of existence of such document, agent removes it from the environment and prepares for further analysis. After that document, as a processed one, is returned to the environment. This is an impulse activating agents from the second category. They assess a documents in terms of meeting search criteria and, in case of success, add document to the result set. Regardless of reacting to environment ‘s changes, agents can communicate between one another to synchronize their activities.

The Superior Layer is the application, which defines user interface and starts, on user’s request, the process of documents’ analysis. This operation entails defining source of documents to analyse and providing a set of search conditions defined by a user. The above mentioned elements, in conjunction with processed documents, constitute the last component in the system vision **- the Environment**. The illustrative model of the main functionality of the system is presented in the figure 1b.

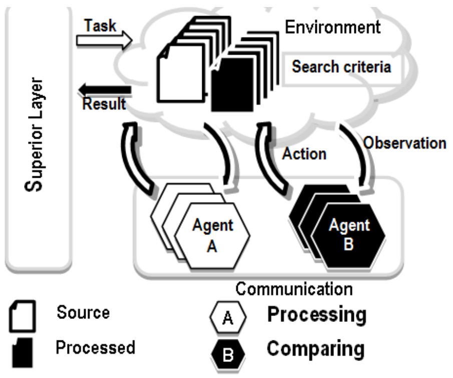


Fig.1a. The vision of the system

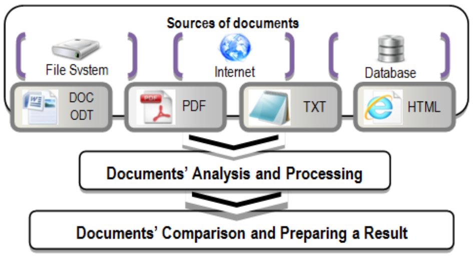

Sources of documents
File System
Internet
Database
DOC ODT
PDF
TXT
HTML
Documents' Analysis and Processing
Documents' Comparison and Preparing a Result

Fig.1b. The illustrative model of the main functionality of the system

The System Modules

The system was divided into few modules (Fig. 2). The **Main** one controls a process of starting the system, its work and results presentation. This element includes implementation of the user interface enabling access to all functions of the system. Its duty is to create agents' threats and to assign them appropriate objects gathering data on agents' states. In the Main module an instance of the environment class is created, which is passed to all newly created agents. In addition, among functionality of this module can be found:

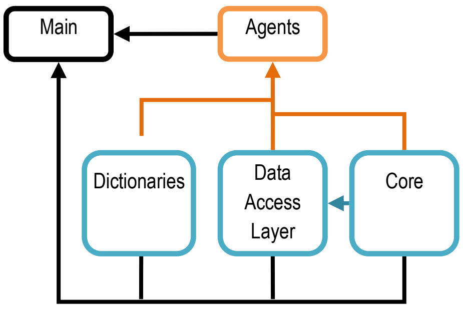


Fig. 2. The system modules

- loading and modifying domain dictionaries,
- changing user requests to lemma strings
- loading documents from particular sources (file system, Internet), database or xml files,
- saving processed documents to the database or xml files.

The **Agents** module includes implementation of agents processing and comparing documents. Processing Agents comprise logic facilitating such operation like pruning, lemmatization and counting numbers of lemma occurrences, searching and counting patterns in an analysed document. Agents of the second type, apart from comparing processed documents, are responsible for determining a quality of documents and search criteria matching.

The **Core**, **Dictionaries** and **Data Access Layer** constitute auxiliary modules responsible respectively for delivering the basic structures and functionality, as well as providing access to the set of dictionaries and sources of documents to be processed. The schema of modules collaboration is presented in the figure 3a and 3b.

Documents processing

During the research presented in the paper for documents analysis the n-gram model was used [Cavnar 1994, Palus, 2011]. In accordance with this model numbers of occurrences of given words sequences are stored in form of vectors of features representing characteristics of documents. Features in vectors are represented by a collection of keys and values corresponding to them.

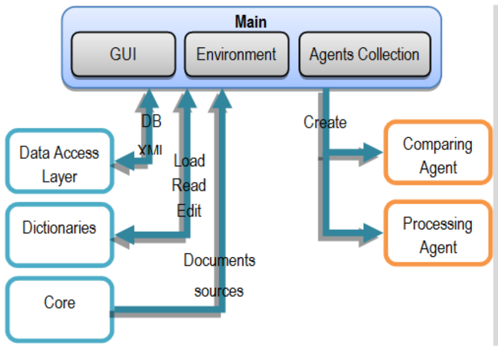


Fig.3a. The Main module and its dependencies

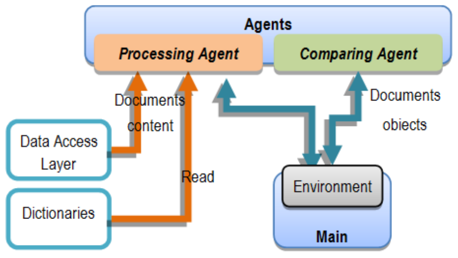


Fig.3b. Agents collaboration

Defining vectors of features, requires going through document contents with simultaneous calculating number of occurrences of particular words or phrases. However this process is performed in documents whose contents have been transformed by operations of lemmatization and pruning earlier [Borycki, 2002, Nguyen, 2009]. These are procedures, in which word's declinations are replaced by a common term, which, dependently on a given method, is a lemma or a core of a word. A lemma is basic form of a word, while core does not have to be proper word and is developed as a result of a inflectional transformations. If it is taken into account, that word, being result of the lemmatization process, is the same word, which is searched in the analyzed document, defining a vector of features can be performed in the same step.

In both procedures, three types of dictionaries – the **Stop-words**, **Domain** and **Lemma** one – play a significant role. First of them is used to remove from a document all unimportant elements like "but", "and", "why" or "whatever". Content of the Domain dictionary allows for finding common contents of documents and classifying them in terms of belonging to a particular field. This dictionary is divided into as many files as domain is to be analysed and can be complemented by a user, also with usage of the system suggestions, based on current analysis of documents. For defining last of the aforementioned dictionaries – the Lemma one, including lemma of polish words – text form of content of the Morfologik dictionary, was used [Morfologik, 2011].

All dictionaries are stored in the file system and during the system start they are loaded into appropriate structures. In case of the Stop-words dictionary it is a simple list of elements, whereas structures of the Domain and the Lemma dictionaries are divided into sections. Units of the first of these two dictionaries represent particular fields, while division of the lemma dictionary is based on the common prefix of a word.

Searching user criteria refers to the problem of searching patterns in character strings. There are many algorithms which can be used to solve this problem. Among them well known algorithms like *Naive*, *Boyer-Moore* or *Knutt-Morris-Pratt* one can be mentioned [Cormen, 2001]. Two last algorithms are effective solutions for text pattern searching, but they are characterized by one feature, which can be considered as a disadvantage. It is an ability to match at most one sequence in one run and, as a consequence, there is the lack of possibility of searching for many patterns concurrently. To remove this inconvenience the solution relaying on creating an index, which points to, for each searched sequence, all places of occurrences of its first elements, was proposed. Navigating through the index entries enables finding particular patterns quickly. However, preparing such index requires scanning a document to find proper positions. It can be done, like in the case of counting words, during lemmatization phrase. The idea of the index is presented in the figure 4.

Calculating number of occurrences of a phrase is more complicated task then in case of a single word. At the beginning, a set of all possible sequences of words belonging to search criteria must be determined. Such sequences can consists of few words of length between 2 and n. Moreover, in case of phrases existing in Polish language, the order of words doesn't affect meaning of a phrase. For this reason all possible sequences without repetition of size between *2* and *n* from set of size *k* have to be taken into consideration.

In the presented research described problem was solved by a use of an algorithm building appropriate tree, in which all but one (root) nodes represent words and each path in the tree, with start point in the root element, constitutes one of a pattern sequences.

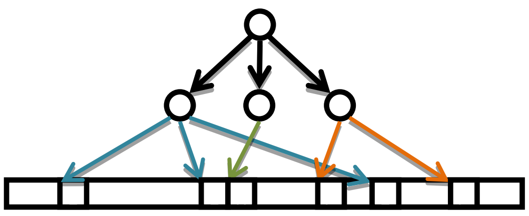


Fig. 4. The example of an index pointing occurrences of first elements of sequences

Searching phrases in an analyses document begins with finding first occurrence of a first item of the aforementioned index. Then subsequent node in the sequences tree is compared with the consecutive word in a document. Presence of a difference means that pattern was not found in that place of a document. Such operation is repeated for all sequences beginning with the same word of a pattern and for all words beginning other pattern sequences.

Documents Comparison

One of the main goals of the presented system is to deliver a tool for comparing and assessing similarity of text documents. As it was described in the previous sections, the system element responsible for providing such functionality is, equipped with appropriate structures and logic, the Comparing Agent. For determining documents matching it uses vectors of features defined earlier and some distance and proximity measures like Hamming or Euclidean one [Deza, 2009]. However, these measure had to be modified for the purpose of this research, because of an existence of one disadvantage – the difficulty to transform values returned by them to a percentage scale, which reflects a documents conformance better. In case of similarity measures it can be difficult to determine a value, which reflects complete documents matching. Likewise, using distance measures, a values describing maximal difference between documents is not known.

Obtaining results in a percentage scale requires introducing some changes to basic formulas of previously mentioned measures. Assuming following symbols:

P – documents distance measure in percentage scale,

n – number of features,

u, v – vectors of features of two documents

the modified Hamming measure is defined as follows:

$$P = \frac{1}{n}\sum_{k=1}^{n}\frac{|u_k - v_k|}{\max(u_k, v_k)}100\% \tag{1}$$

According to this formula the percentage ratio of dissimilarity, for each pair of documents, is calculated. To represent it in a form of a function determining documents similarity in direct way, next modification should be applied (Equ. 2)

$$similarity = \left(1 - \frac{1}{n}\sum_{k=1}^{n}\frac{|u_k - v_k|}{\max(u_k, v_k)}\right)100\% \tag{2}$$

In order to improve the quality of documents comparison, features characterizing documents are differentiated by various weights, to strengthen meaning features representing search criteria. What is more weights of phrases are higher, because their occurrence is more unique than simple words. The final formula for calculating similarity is defined by the equation 3:

$$similarity = \left(1 - \frac{1}{m}\sum_{k=1}^{n} \frac{w_k |u_k - v_k|}{\max(u_k, v_k)}\right) 100\% \tag{3}$$

where:

w – vector weights of features,

w_k – weight of k feature,

m = $\sum_{k=1}^{n} w_k$ – sum of weights.

The rule of weighted features was applied to other distance and proximity measures, and in form of the following equations, were embedded into the Comparing Agent logic :

1. weighted Hamming distance measure

$$P(u,v) = \sum_{k=1}^{n} w_k . |u_k - v_k| \tag{4}$$

2. weighted Euclidean distance measure

$$P(u,v) = \sqrt{\sum_{k=1}^{n} w_k . |u_k - v_k|} \tag{5}$$

3. proximity measure based on number of occurrences of search words

$$P(u,v) = \sum_{k=1}^{n} w_k u_{k.} v_k \tag{6}$$

4. proximity measure based on values of features

$$P(u,v) = \sum_{k=1}^{n} w_k . \min(u_{k.} v_k) \tag{7}$$

5. proximity measure based on values of features, taking into account also the documents characterized by low factors

$$P(u,v) = \sum_{k=1}^{n} w_k . \frac{\min(u_k, v_k)}{\max(N(u), N(v))}, \quad where \ N(x) = \sum_{i=1}^{n} w_i . x_i \tag{8}$$

The Result Quality

The result quality is defined as a value determining, for a given document, the degree of matching the search criteria. It is calculated on the basis of the same vectors of features, which are used in the process of comparing documents. Using weighted Hamming measure and assuming that user condition vector is set to zero, unambiguous values of a quality of documents conformance can be obtained. The higher the value the better the criteria met. A percentage representation of a result requires its linear scaling in the range of 0% and 100% (Equ. 10).

$$quality = \sum_{k=1}^{r} w'_k \left| 0 - v'_k \right| = \sum_{k=1}^{r} w'_k \, u'_k \tag{9}$$

$$quality_{percentage}(i) = \frac{qualilty\,(i)}{\max\limits_{j}\,(quality\,(j))} 100\% \tag{10}$$

where:

r – a number of features belonging to a search criteria,

u' – a vector consisting of document's features corresponding to a search criteria,

w' – weights for u' – a vector of features,

i, j – indexes identifying particular documents.

Determining quality of documents' matching is performed by the Comparing Agent during process of their analysis, but percentage values is generated at the end, after processing all documents, because the highest value of similarity must be calculated first.

The Result Presentation

The effect of the system work is a set of many, compared in pairs, documents with associated quality matching rates. A text form of a result including such large amount of data is difficult to analyse, but the same refers to a graphical representation, which provides a complicated network of linked documents (Fig 5a). This is why few mechanisms were developed to facilitate presentation of obtained results (Fig. 5b):

1. presenting links only for a chosen document,
2. presenting all links with accentuating edges connected to a given node,
3. providing possibility of limiting links description,
4. providing possibility of filtering network on the basis of the lowest similarity rates.

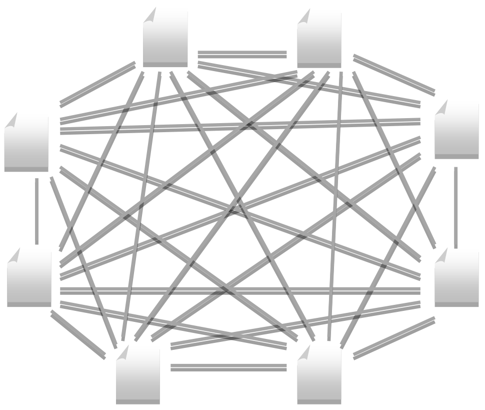


Fig. 5a. Sample network of linked documents

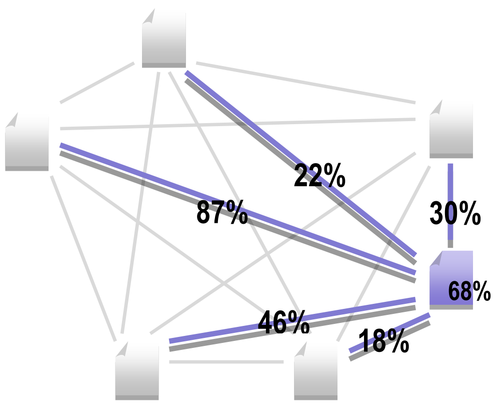


Fig. 5b. The same network with developed mechanisms aplied

Additionally, in the system, the functionality of presenting details regarding similarity rate, after choosing a given link, was developed as well.

Experiments

The developed system was tested in terms of checking how some parameters or methods, can influence its work efficiency. The set of examined elements included the method of storing the Lemma dictionary, the length of searched sequences and methods used for documents comparison.

During the research it was assumed that the Lemma dictionary can be characterized by a large size, so its searching can be crucial for documents processing. Therefore, the decision to divide content of this dictionary into smaller sections was made. However a way of the division and size of a unit had to be examined. As a condition for dictionary splitting, word's prefix was utilized. Performed tests were to show how long such prefix should be. A length of a prefix was determined experimentally by analysis of response time of searching example words.

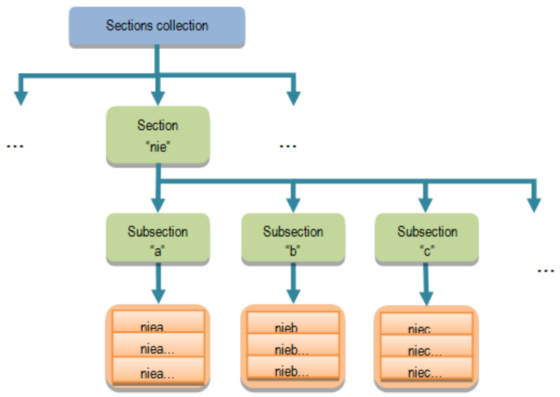


Fig.6. The structure of the lemma dictionary

Obtained findings proved that, in case of Polish language, prefixes of the length of three characters gave the best result but in same kinds of words four characters were needed to reduce search time (Table 1). To improve these outcomes in case of some lemma dictionary sections, additional level, consisting of words having the same character after the prefix, was introduced (Fig. 6).

Table 1. Sample results of experiments – time of lemma dictionary access in milliseconds

Length of the	**1**	**2**	**3**	**4**	Results for prefix consisting of three characters and additional level for words staring with prefix "nie"	
cytryna	55	1	0	2		0
nieznajomy	305	260	263	18		14
tttratwa	23	0	1	4		1

During subsequent tests, influence of the length of a sequence - representing a feature describing document - on time of its processing, was studied. Table 2 includes sample results obtained for 30 documents returned by Google search engine for sequences of length from 1 to 5. It can be noticed that the time of documents processing has a slight upward trend, so the conclusion saying that the length of a feature sequence does not have great impact on total processing time can be drawn.

Table 2. Influence of a sequence length on time of document's processing

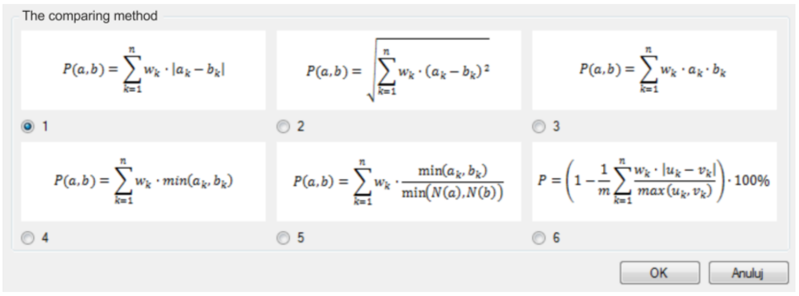
Sequence length - n	**1**	**2**	**3**	**4**	**5**
Test 1, time [ms]	81266	81372	80289	82458	90969
Test 2, time [ms]	82534	83296	82773	82679	95545

The last examined aspect concerned an impact of a chosen comparing method on efficiency of system work. Once more sets of documents obtained from the Internet were used. These documents were analysed by two Processing and three Comparing Agents. Results of two sample tests, consisting of 30 and 10 documents respectively, are shown in the table 3.

Table 3. Influence of a comparing method on time of document's processing

Comparing method	**4**	**5**	**6**	**7**	**8**	**1**
Test 1, time [ms]	59459	51147	53591	54005	48343	51533
Test 2, time [ms]	19221	20581	18634	18373	19519	19174

Numbers of methods visible in the table represent numbers of equations presented in the chapter Documents Comparison. Analysing obtained values, it is difficult to point out the best, in terms of processing efficiency, method. Within a given test, times are comparable, but between tests, in the regard to particular methods, they become contradictory. Therefore, providing user with a possibility to choose the comparing method seems to be good idea (fig. 7).



The comparing method

$$P(a,b) = \sum_{k=1}^{n} w_k \cdot |a_k - b_k|$$

[✓ 1]

$$P(a,b) = \sqrt{\sum_{k=1}^{n} w_k \cdot (a_k - b_k)^2}$$

2

$$P(a,b) = \sum_{k=1}^{n} w_k \cdot a_k \cdot b_k$$

3

$$P(a,b) = \sum_{k=1}^{n} w_k \cdot min(a_k, b_k)$$

4

$$P(a,b) = \sum_{k=1}^{n} w_k \cdot \frac{\min(a_k, b_k)}{\min(N(a), N(b))}$$

5

$$P = \left(1 - \frac{1}{m} \sum_{k=1}^{n} \frac{w_k \cdot |u_k - v_k|}{max(u_k, v_k)}\right) \cdot 100\%$$

6

Fig. 7. The system window allowing for defining the search environment

Conclusion

The main goal of the research was to build the system, which searches documents in defined sources, processes them and shows existing similarities. Additional assumption regarding the system architecture, to be in accordance with idea of the multi-agent systems, was made.

After the analysis of the text processing issue, the document representation based on a vector of features was chosen. Documents' matching, in such case, relies on comparing values of particular features. Achieving this functionality was possible owing to the Lemma dictionary usage. Because this element was expected to be the crucial one in terms of efficiency of the system work, some experiments, examining it, were performed. They made allowances for elaborating solutions optimizing the Lemma dictionary search. These solutions, including the index of sequences of search criteria elements and the Lemma dictionary structure, improved the efficiency of this element analysis.

Experiments performed during the research showed that both length of a search sequence and a method chosen for documents' comparison do not have significant impact on the efficiency of this process.

A few other requirements were also raised with respect to the project, such as the possibility of saving results and their graphic presentation. There were secondary features but also important for the final effect. Their fulfillment allowed to make the application not only universal but also user-friendly.

Bibliography

[Aggarwal 2001] C. C. Aggarwal, P. S. Yu. On Effective Conceptual Indexing and Similarity Search in Text Data in Proceeding ICDM '01 Proceedings of the 2001 IEEE International Conference on Data

[Bigus, 2001] J.P. Bigus, J. Bigus. Constructing Intelligent Agents Using Java, 2nd edn. John Wiley & Sons, Inc., New York, NY, USA. 2001

[Bollacker, 1998] K. D. Bollacker, S. Lawrence, C. L. Giles. CiteSeer: An Autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications. In: Proceeding: AGENTS '98 Proceedings of the second international conference on Autonomous agent, 1998.

[Borycki, 2002] Ł. Borycki, P. Sołdacki. Automatic text classification (Automatyczna klasyfikacja tekstów). 2002, http://www.zsi.pwr.wroc.pl/zsi/missi2002/pdf/s504.pdf, 2011

[Cavnar 1994] B.W. Cavnar, J. M. Trenkle. N-gram-based text categorization. In Proceedings of SDAIR, 1994

[Cormen, 2001] H. Cormen, R. L. Rivest, C. E. Leiserson. Introduction To Algorithms, MIT Press, 200

[Dąbrowski, 1978] M. Dąbrowski, K. Laus-Mączyńska. Methods of Information Search and Classification (Metody wyszukiwania i klasyfikacji informacji), WNT, Warszawa 1978

[Deza, 2009] M. M. Deza, E. Deza Encyclopedia of Distances. Springer-Verlag Berlin Heidelberg 2009

[Morfologik, 2011] Morfologik, http://morfologik.blogspot.com/, 2011

[Nguyen 2009] L. T. Nguyen. Static Index Pruning for Information Retrieval Systems: A Posting-Based Approach. In Proceedings of 7th Workshop on Large-Scale Distributed Systems for Information Retrieval (LSDS-IR'09)

[Palus, 2011] A. Pauls D. Klein. Faster and Smaller N-Gram Language Model. in Proceeding HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, 2011.

[Smirnov, 2008] I. Smirnov. Overview of Stemming Algorithms. DePaul University, http://the-smirnovs.org/info/stemming.pdf, 2011

[Strömbäck, 2005] P. Strömbäck. The Impact of Lemmatization in Word Alignment, Department of Linguistics and PhilologySpråkteknologiprogrammet 2005

[Strömbäck, 2005] P. Strömbäck. The Impact of Lemmatization in Word Alignment, Department of Linguistics and Philology Språkteknologiprogrammet 2005

Authors' Information

***Katarzyna Harężlak** – Silesian University of Technology;*

e-mail: katarzyna.harezlak@polsl.pl

Major Fields of Scientific Research: Distributed and Mobile Databases, Software Engineering, Agent Systems, Social Networks

***Michał Sala** – Silesian University of Technology;*

e-mail: michal.sala@live.com

Major Fields of Scientific Research: Text searching, Agent Systems