International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-2, December 2019

Cross-Language Plagiarism Detection Based on CLAD Method

Jasurbek Atadjanov, Boburbek Atadjanov

Abstract: ***This paper describes the cross-language plagiarism detection method CLAD (Cross-Language Analog Detector) between test document and indexed documents. The main difference of this method from existing versions is the detection of plagiarism among multiple languages not only two languages. While translating terms, it used the dictionary-based machine-translation method. CLAD's working process consists of document indexing and detection process phases. In this paper, we will describe both of these phases.***

Keywords: stemming text, search information from document, cross-language plagiarism detection

I. INTRODUCTION.

Plagiarism, usage of the work by done another person without proper acknowledgment to the original source, is an infamous problem in academic society. There are several methods, system, and services which help to detect plagiarism by machine [1]. As a result, the development of machine translation of the text has posed the problems of detecting cross-language plagiarism (CLPD) [2-4].

The main problem for CLPD during the translation of the text, it is important to determine exactly what type of word translation is used in the document. For example, in Russian, there is word "человек" in English it can be translated "person", "human", "individual", and "man". So, if we translate this word as "person" during the plagiarism detection process, but the document uses "man" version, during comparison the text we can get an incorrect result. If we check all synonym versions of the word, then we will have performance issues.In this paper, we will describe CLAD (Cross-Language Analog Detector) method which used to detect similarity score between documents which are in different or same natural language. CLAD used "Bag of words analysis" model to determine the similarity of two documents [5-7]. In this method plagiarism detection process consists of converting the document into plain text, parsing text, analyzing words morphologically (stemming), analyzing words lexically (detecting and removing stop-words), normalizing synonym forms, translating words (dictionary-based machine-translation method), comparing the bag of words.

Revised Manuscript Received on December 30, 2019.
***** Correspondence Author
Jasurbek Atadjanov*, is a software developer at Uztelecom. Tashkent University of Information Technologies (TUIT), Tashkent, Uzbekistan. j.atadjanov@gmail.com
Bobur Atadjanov, is a software developer at Uztelecom. Tashkent University of Information Technologies (TUIT), Tashkent, Uzbekistan. bobur.atadjanov@gmail.com

The main difference of this method from existing ones is that it can detect check plagiarism more than two natural language. To avoid compiling a dictionary for each pair of languages, the main language is selected in this method. If two documents are in a different language, first their bag of words are translated into the main language and then compared.

'jComporator' system was developed by this CLAD method and the detection quality of this system is directly related to synonyms and dictionary databases. The database structure of the jComporator system is designed by the NoSQL mechanism [8-10]. In the document indexing process, it was used Apache Lucene system which developed by Apache Software Foundation [11]. To store the rest of the information (dictionary, synonyms, reports, and etc.), MySQL was used. MySQL is an open-source relational database management system (RDBMS) [12].

II. RELATED WORKS

This section provides an overview of related works that deal with the detection of cross-language plagiarism. Work by Vera Danilova (2013) showed methods of cross-language plagiarism detection between documents. It described the process of comparing documents that are written in different natural languages [13]. All only considered an algorithm for determining plagiarism between two languages. In addition, a synonym for the form of words is not considered in these algorithms. In the paper by Zaid Alaa, Sabrina Tiun, and Mohammedhasan Abdulameer (2016) the method of cross-language method documents in Arabic and English was described. The paper also showed a comparison of documents considering the synonymity of words [2].

In an interesting paper [14], Daniele Anzelmi and colleagues report the SCAM (Standard Copy Analysis Mechanism) algorithm which is a relative measure to detect overlapping by making comparison on a set of words that are common between test document and registered document. To compare documents, taking into account the synonym forms of words, this algorithm suggests checking each synonym form. In this case, the total number of operations will be calculated using the following formula

$$s = \sum_{i=1}^{l} c_i \qquad (1)$$

Here, l - count of words in document, c_i - count of synonym forms i - word, s - total operations number. The total number of the comparison operations will be even greater if we use algorithms of the class shilling [15].

$$s = \prod_{i=1}^{l} c_i \qquad (2)$$

The paper [11] introduced a cross-language plagiarism system for English-translated copies of Spanish document's detection. Their system was comprised of three stages; namely translation detection, internet search and report generation.

There are several systems, which can detect document plagiarism by using web search engines, like AntiPlagiarism.NET, Advego Plagiatus, Unplag, Grammarly, Copyscape. Also, there are Unicheck, Turnitin, PlagTracker, Антиплагиат, PlagScan system and services which work on their own database [16-19].

III. IMPLEMENTATION

As discussed above, document plagiarism detection consists of (1) document indexing, (2) similarity checking phases. In this paper, we will describe plagiarism detection process for Uzbek, English, and Russian documents. To describe this method, the main language is chosen as English. If a document is in Uzbek or Russian during the indexing process, its terms will be translated into English.

IV. DOCUMENT INDEXING PROCESS

In this phase, we will describe the process of inserting a document into the database. The following figure illustrates the document indexing process.

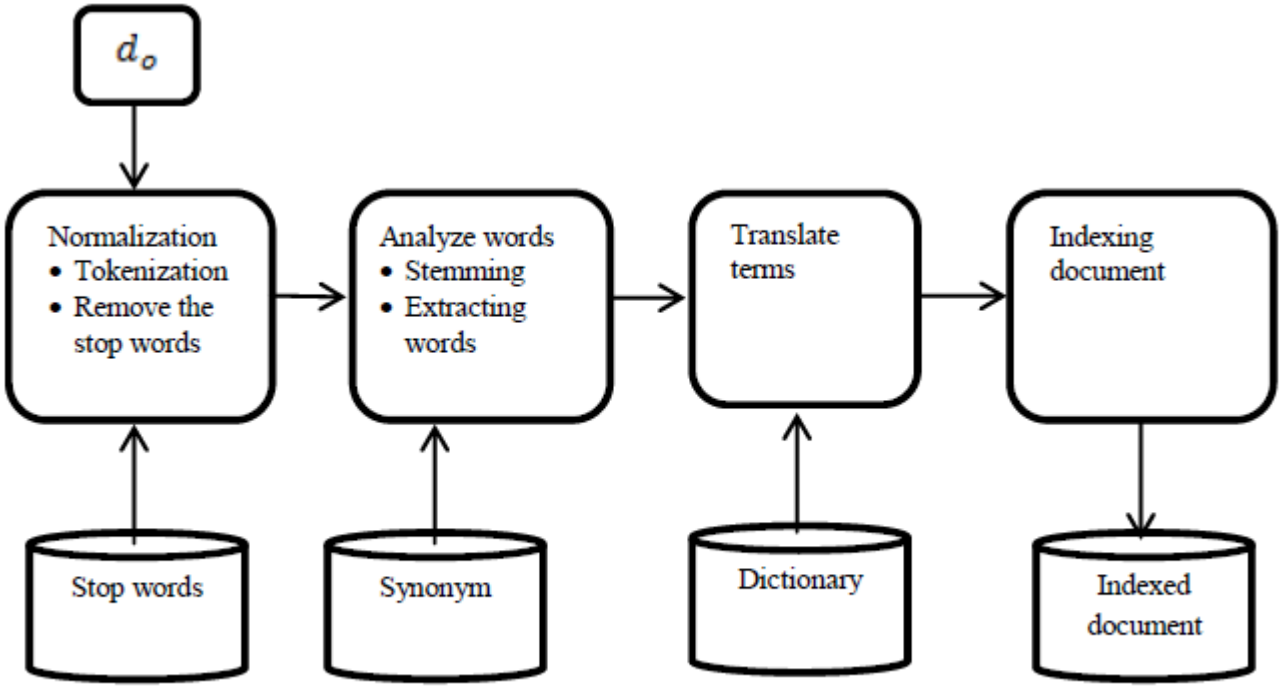

Fig. 1. Document indexing

A. Document Normalization

The document normalization phase consists of (1) content analyzing, (2) tokenization (3), and stop word removal steps. The main aim of this phase is to prepare the original document's dataset for similarity comparisons with other texts.

Content analysis is consists of retrieving simple text (words, themes, or concepts) from digital files in different formats. In this step, we can use Apache Tika toolkit. The Apache Tika™ toolkit supports extracting metadata and text from more than a thousand different fil

e types (such as PPT, XLS, and PDF) [20]. After this step document which in any file formats will be converted into plain text format, in Table-1 it is given the result of this step.

Table 1. Result of content analyzing step

Source text in HTML format (Input)	Plain text (Output)
This paper describes the cross-language plagiarism detection method CLAD (Cross-Language Analog Detector) between test document and indexed documents. The main difference of this method from existing versions is the detection of plagiarism among multiple languages not only two languages. It is one part of mathematics and computer science, which consists of methods, algorithms and a class of problems	This paper describes the cross-language plagiarism detection method CLAD (Cross-Language Analog Detector) between test document and indexed documents. The main difference of this method from existing versions is the detection of plagiarism among multiple languages not only two languages. It is one part of mathematics and computer science, which consists of methods, algorithms and a class of problems

Tokenization is the process of converting text into elements (words, phrases, symbols, or other meaningful elements) called tokens, and condition process of documents will be based on a set of these tokens. There are a number of algorithms for document tokenization. In this algorithm, we used tokenization using Regular Expressions (sometimes called a rational expression) [21]. There are given regular expressions that parse text into a collection of words.

$$\sim [A - Z] . * [. , : ! ? ;] (? = \backslash s | \$) \sim s \quad (3)$$

Table-2 shows how the tokenization has been done each word and component, including the stop words and special characters.

Table 2. Result of tokenization

Plain text (Input)	Words collection (Output)
This paper describes the cross-language plagiarism detection method CLAD (Cross-Language Analog Detector) between test document and indexed documents. The main difference of this method from existing versions is the detection of plagiarism among multiple languages not only two languages. It is one part of mathematics and computer science, which consists of methods, algorithms and a class of problems	"this", "paper", "describes", "the", "cross", "language", "plagiarism", "detection", "method", "CLAD", "cross", "language", "analog", "detector", "between", "test", "document", "and", "indexed", "documents", "main", "difference", "of", "this", "method", "from", "existing", "versions", "is", "the", "detection", "of", "plagiarism", "among", "multiple", "languages", "not", "only", "two", "languages", "it", "is", "one", "part", "of", "mathematics", "and", "computer", "science", "which", "consists", "of", "methods", "algorithms", "and", "a", "class", "of", "problems"

It is known, that every natural language has **stop words**, which used inside of sentence to relate words to each other. There is no single universal list of stop words used by all-natural language processing tools; and indeed, not all tools even use such a list. The next step consists of removing stop words from collection words. The list of the English stop words that has been used in this study is a default English stop words list, and is a well-known list used by many researchers, including [26]. In Table-3 it is displayed text after removing stop words step.

Table 3. Removing stop words

Words collection (Input)	Bag of words (Output)
"this", "paper", "describes", "the", "cross", "language", "plagiarism", "detection", "method", "CLAD", "cross", "language", "analog", "detector", "between", "test", "document", "and", "indexed", "documents", "main", "difference", "of", "this", "method", "from", "existing", "versions", "is", "the", "detection", "of", "plagiarism", "among", "multiple", "languages", "not", "only", "two", "languages", "it", "is", "one", "part", "of", "mathematics", "and", "computer", "science", "which", "consists", "of", "methods", "algorithms", "and", "a", "class", "of", "problems"	"paper", "describes", "cross", "language", "plagiarism", "detection", "method", "CLAD", "cross", "language", "analog", "detector", "test", "document", "indexed", "documents", "main", "difference", "method", "versions", "detection", "plagiarism", "multiple", "languages", "part", "mathematics", "computer", "science", "methods", "algorithms", "class", "problems"

B. Analyze words

In this step, we will detect the morphological root of the word and in information technology; this process is **stemming** [23]. There are a number of algorithms for stemming words in natural languages. There are many algorithms for stemming words in natural languages like Snowball Framework [24, 25].

During the stemming process we can use algorithms from Snowball Framework. This framework has the algorithms to stemming about 20 languages. Unfortunately, Snowball Framework does not have any stemming algorithm for Uzbek language. In [26] Uzbek language suffixes categorization was described, which can help us to build the stemmer algorithm for Uzbek language by using Snowball Framework. We can see the result of the stemming process for our words in Table-4.

Table 4. Result of stemming step

Words collection (Input)	Stemmed words collection (Output)
"paper", "describes", "cross", "language", "plagiarism", "detection", "method", "CLAD", "cross", "language", "analog", "detector", "test", "document", "indexed", "documents", "main", "difference", "method", "versions", "detection", "plagiarism", "multiple", "languages", "languages", "part", "mathematics", "computer", "science", "methods", "algorithms", "class", "problems"	"paper", "describe", "cross", "language", "plagiarism", "detect", "method", "CLAD", "cross", "language", "analog", "detect", "test", "document", "index", "document", "main", "difference", "method", "version", "detect", "plagiarism", "multiple", "language", "language", "part", "math", "compute"

C. Extracting words

In this phase, we convert stemmed words into a formal form. In this form, every element of the document will consist of stemmed word and term frequency words in the document.

$$D_j = \left((d_1, n_1), (d_2, n_2), .., (d_p, n_p)\right) \quad (4)$$

$$\forall d_i \notin H_j \quad (5)$$

Here, j - the natural language of D document, H_j - the collection of stop words j natural language, d_i - i the term on the text, n_i - the number of occurrences d_i term in the text. In Table 5 it was shown the structure documents in (4) version. This table also described synonym forms of every word.

Table 5. Term Frequency of the document's words

Term	Term Frequency	Synonym forms	Count of synonym forms
computer	3	PC, laptop, electronic brain	3
algorithm	2	algorism, logic	2
math	1	-	0
science	1	knowledge, ology, study	3
data	1	information, evidence, facts, material	3

D. Target form of term synonyms

The purpose of displaying synonym forms of terms above is to explain comparison process to the reader considering synonymity. Before comparing terms, this algorithm detects the target synonym form of every term. In this case, we use the following data structure.

Here, **word** – table to store a list of words and its terms, **class** – table to store word category and target word pointer, **relation** – table to store word and its category relation. After retrieving a set of terms, we get target versions of every term based on this data structure.

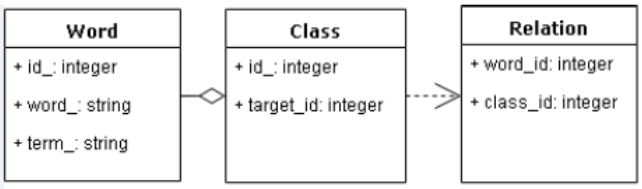

Fig 2. Target form of terms detection data structure.

In this step, every term will be replaced its target synonym form. For example, for "person", "human", "individual", and "man" terms we have marked as target version **"person"**. In this case, during compare documents or storing database it will be used **"person"** term for "human", "individual", and "man" terms. This operation will be done for every language document, so we should have a synonym database for every natural language which we using in plagiarism detection process, not only for English.

According to this operation, every term will be converted into its target word. As a result, using one comparison operation we can check all existing forms of term synonyms. In this case, total comparison operations count will be equal **count of terms**. This is the main difference in this algorithm than existing algorithms like SCAM and shilling. The total count of comparison operations is shown for the SCAM algorithm in (1) and in (2) showed shilling class algorithms. After this step, document in (3) form will be like as the following:

$$D = ((d_{12}, n_1), (d_{22}, n_2), .., (d_{p2}, n_p)) \quad (6)$$

Here, d_{i2} - the d_i term's target synonym form, $\forall d_i \in D\, \omega(d_i) = d_{i2}$, $\omega(d_i)$ - the function which detects the target version of d_i term. After it we combine this bag of words, this phase consists of detecting duplicate terms and using one of them and summarize term`s the number of occurrences. In this case, we can display (6) form in the following:

$$D = ((d_{13}, n_{11}), (d_{23}, n_{21}), .., (d_{k3}, n_{k1})) \quad (7)$$

$$\forall d_{i3} \neq d_{j3}, i \neq j \quad (8)$$

Here, n_{i1} - the number of occurrences d_{i3} term in the (6) form.

E. Translate terms

The CLAD uses Dictionary-based machine-translation method to translate terms from Uzbek or Russian to English [25]. The following figure illustrates database structure to store a dictionary which using the translating process.

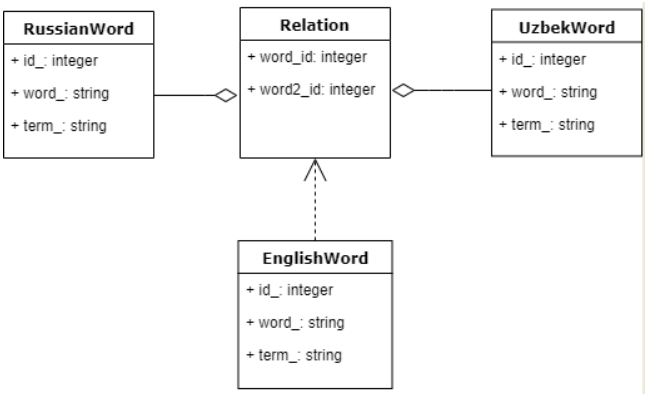

Fig. 3. Dictionary structure to translate terms

During translate we use terms in (6) form. As discussed above in the document in English we will skip this section.

$$\forall d_{i3}\ \omega(trans(d_{i3})) = d'_{i3} \quad (9)$$

Here, $trans(d_i)$ - the translation function of d_i term into English, $\omega(d_i)$ - the function which detects the target version of d_i term. After this steps, we can describe *D* document in the following form.

$$D = ((d_{13}, d'_{13}, n_{11}), (d_{23}, d'_{23}, n_{21}), .., (d_{k3}, d'_{23}, n_{k1})) \quad (10)$$

F. Indexing document

In this section, we describe how to store *D* document in the database. Every document will be converted into (10) before storing it into the database. As a database, it was used Apache Lucene. The documents the following parameters will be stored in Apache Lucene:

- Document title;
- Document author(s);
- Natural language which document is written;
- Document elements in (10) forms;

During this step we using Apache Lucene's **IndexWriter** class [11].

V. DETECTION PROCESS

In this phase, we will describe the process of plagiarism detection process by index document. The following figure illustrates the document indexing process.

As illustrated in Figure 3, in the phase consists of document normalization, analyze words, and translate terms sections which familiarly with the Document Indexing phase. That's why we will not repeat these steps, and we believe that this T document is presented as follows.

$$T = ((t_1, t'_1, m_1), (t_2, t'_2, m_2), ..., (t_p, t'_p, m_p)) \quad (11)$$

Here, t_i - i term of T document, t'_i - translated form of t_i term, m_i - the number of occurrences t_i term.

A. Retrieval of candidate documents

In this section, we will describe how to find documents from the Apache Lucene database according to t'_i terms of T document. In this case we can use **IndexSearcher** class of Apache Lucene [11]. To easy describe condition process we show similarity checking process between two D and T documents.

B. Comparing documents

Our algorithm detects similarity between two documents based on the set of terms both documents have. In other words, similarity of both documents is calculated considering similarity (10) and (11) objects. At first, we will calculate the weight of both documents.

$$N = \sum_{i=1}^{p} n_{i1} \qquad (9)$$

$$M = \sum_{i=1}^{s} m_i \qquad (10)$$

Here, N - the weight of the D document, M - the weight of document T. Next step, we will get the list of words that exists in both D and T documents by intersection set of their words.

$$[d'_{13}, d'_{23}, ..., d'_{k3}] \cap [t'_1, t'_2, ..., t'_p] = [x_1, x_2, .., x_k] \qquad (11)$$

$$X = ((x_1, n_1, m_1), (x_2, n_2, m_2), ..., (x_k, n_k, m_k)) \qquad (12)$$

Here x_i - the term in the *D* and *T* documents, n_i - the number of recurrences, x_i term in the D document, m_i - the number of occurrences x_i term in the *T* document. The similarity degree of *D* document to *T* document will be calculated as the following formula.

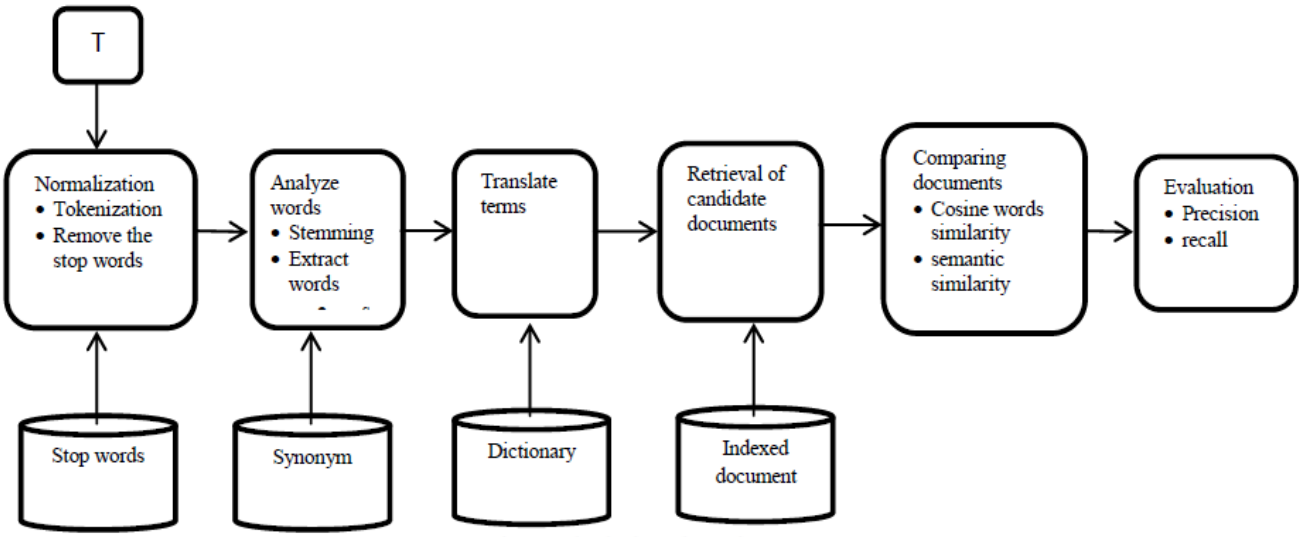

Fig. 4. Plagiarism detection process

$$dt = \sum_{i=1}^{k} \frac{n_i \cdot m_i}{N^2} \qquad (13)$$

The similarity degree of *T* document to *D* document will be calculated as the following formula.

$$td = \sum_{i=1}^{k} \frac{n_i \cdot m_i}{M^2} \qquad (14)$$

From (13) and (14) we can calculate the total similarity degree of both documents. It will be calculated using (17) formula.

$$sim(D,T) = \max(dt, td) \qquad (15)$$

Through executing all steps for documents which retrieved from indexed database, we will have the collection of documents that are similar to *D* document.

VI. EXPERIMENTAL RESULTS

The proposed model of this study was programmed with Java programming language. The objective of the proposed model is to detect CLPD throughout indexed documents. During the test process, we got a file in Russian from indexed database (**we marked it file1.doc**) and replaced its some words with synonym forms, and marked it **file2.doc**. Next, some paragraphs' location was changed in file1.doc and new **file3.doc** file was generated. Afterwards, we translated file.doc into Uzbek and saved this file with **file4.doc** name. These three files (file2.doc, file3.doc, and file4.doc) were given to experts and asked them to rate plagiarism degree between source file (file1.doc).

After receiving answers from experts, we took the arithmetic mean values according to their results (fiel2.doc – 84%, file3.doc – 79%, file4.doc – 50%). In the Figure-3 it was given experiment result with diagram version.

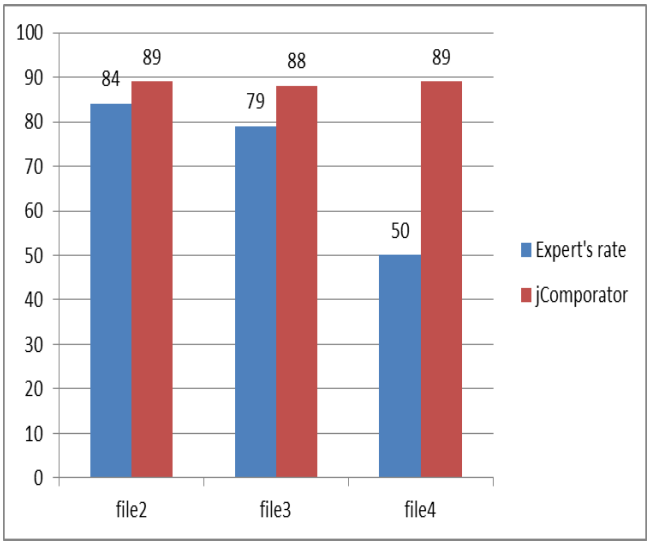

Fig. 5. Experiment result

The result of the experiment showed that in file4html we get the really bad results because there were used homonyms words and our algorithm did not give the right result. But the rest of the files its results are closer to the result of experts.

VII. CONCLUSION

This paper has presented CLAD method which cross-language plagiarism detection process among indexed documents. This method was implemented into jComporator information system, which detects document plagiarism, and tested on Tashkent University of Information Technologies named after Muhammad Al-Khwarizmi in 2013-2014 years. In this process, systems helped to detect a number of plagiarism documents.

ACKNOWLEDGEMENT

This research project is funded by Information Systems branch of Uztelecom Stock Company.

REFERENCES

1. Macdonald, R., & Carroll, J. (2006). Plagiarism—a complex issue requiring a holistic institutional approach. Assessment & Evaluation in Higher Education, 31(2), 233–245. doi:10.1080/02602930500262536 https://www.tandfonline.com/doi/abs/10.1080/02602930500262536
2. Zaid Alaa, Sabrina Tiun, and Mohammedhasan Abdulameer (2016) Cross-language plagiarism of arabic-english documents using linear logistic regression. Journal of Theoretical and Applied Information Technology 10 th January 2016. Vol.83. No.1 https://ukm.pure.elsevier.com/en/publications/cross-language-plagiarism-of-arabic-english-documents-using-linea
3. Potthast, Martin; Barrón-Cedeño, Alberto; Stein, Benno; Rosso, Paolo (2011), "Cross-Language Plagiarism Detection" (PDF), Language Resources and Evaluation, 45 (1): 45–62, doi:10.1007/s10579-009-9114-z, ISSN 1574-020X, archived from the original (PDF) on 26 November 2013, retrieved 7 October 2011 https://www.researchgate.net/publication/225146236_Cross-Language_plagiarism_detection
4. Potthast, Martin; Barrón-Cedeño, Alberto; Eiselt, Andreas; Stein, Benno; Rosso, Paolo (2010), "Overview of the 2nd International Competition on Plagiarism Detection", Notebook Papers of CLEF 2010 LABs and Workshops, 22–23 September, Padua, Italy (PDF), archived from the original (PDF) on 3 April 2012, retrieved 7 October 2011 https://www.researchgate.net/publication/221159542_Overview_of_the_2nd_International_Competition_on_Plagiarism_Detection
5. Si, Antonio; Leong, Hong Va; Lau, Rynson W. H. (1997), "CHECK: A Document Plagiarism Detection System", SAC '97: Proceedings of the 1997 ACM symposium on Applied computing (PDF), ACM, pp. 70–77, doi:10.1145/331697.335176, ISBN 978-0-89791-850-3 http://www.cs.cityu.edu.hk/~rynson/papers/sac97.pdf
6. Dreher, Heinz (2007), "Automatic Conceptual Analysis for Plagiarism Detection" (PDF), Information and Beyond: The Journal of Issues in Informing Science and Information Technology, 4: 601–614, doi:10.28945/974 https://www.researchgate.net/publication/230854738_Automatic_Conceptual_Analysis_for_Plagiarism_Detection
7. Muhr, Markus; Zechner, Mario; Kern, Roman; Granitzer, Michael (2009), "External and Intrinsic Plagiarism Detection Using Vector Space Models", PAN09 - 3rd Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse and 1st International Competition on Plagiarism Detection (PDF), CEUR Workshop Proceedings, 502, pp. 47–55, ISSN 1613-0073, archived from the original (PDF) on 2 April 2012. http://ceur-ws.org/Vol-502/paper9.pdf
8. Leavitt, Neal (2010). "Will NoSQL Databases Live Up to Their Promise?" (PDF). IEEE Computer. https://dl.acm.org/citation.cfm?id=1731109
9. Mohan, C. (2013). History Repeats Itself: Sensible and NonsenSQL Aspects of the NoSQL Hoopla (PDF). Proc. 16th Int'l Conf. on Extending Database Technology. https://openproceedings.org/2013/conf/edbt/Mohan13.pdf
10. Fowler, Martin. "No sql Definition". many advocates of NoSQL say that it does not mean a "no" to SQL, rather it means Not Only SQL https://martinfowler.com/bliki/NosqlDefinition.html
11. Erik H., Otis G., Michael McC. Lucene in Action – Covers Apache Lucene v.3.0// Manning Publications.-486p.,2009 y. https://epdf.pub/lucene-in-action-second-edition.html
12. Frank, Mike. "Announcing General Availability of MySQL 8.0". blogs.oracle.com. Retrieved 10 October 2019. https://worddisk.com/wiki/MySQL/
13. Vera Danilova (2013), Cross-Language Plagiarism Detection Methods. Proceedings of the Student Research Workshop associated with RANLP 2013, pages 51–57, Hissar, Bulgaria, 9-11 September 2013. https://pdfs.semanticscholar.org/7b3d/8c516a22b8b38b48d2767685df4393e22665.pdf
14. Daniele Anzelmi, Domenico Carlone, Fabio Rizzello,Robert Thomsen, D. M. Akbar Hussain (2011) Plagiarism Detection Based on SCAM Algorithm // Proceedings of the International MultiConference of Engineers and Computer Scientists 2011 Vol I? IMECS 2011, March 16-18, 2011 Hong Kong http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.421.2867
15. Bretag, T., & Mahmud, S. 2009. A model for determining student plagiarism: Electronic detection and academic judgement. Journal of University Teaching & Learning Practice, 6(1). Retrieved from http://ro.uow.edu.au/jutlp/vol6/iss1/6
16. Macdonald, R., & Carroll, J. 2006. Plagiarism—a complex issue requiring a holistic institutional approach. Assessment & Evaluation in Higher Education, 31(2), 233–245. doi:10.1080/02602930500262536 https://www.tandfonline.com/doi/abs/10.1080/02602930500262536
17. Lancaster, Thomas. 2003. Effective and Efficient Plagiarism Detection (PhD Thesis), School of Computing, Information Systems and Mathematics South Bank University. https://www.researchgate.net/publication/228729388_Effective_and_Efficient_Plagiarism_Detection
18. Roy, Chanchal Kumar;Cordy, James R. 2007. "A Survey on Software Clone Detection Research". School of Computing, Queen's University, Canada.
19. Salha M. Alzahrani, Naomie Salim, and Ajith Abraham, Senior Member. 2012. Understanding Plagiarism Linguistic Patterns, Textual Features, and Detection Methods. IEEE Transactions on systems, man, and cybernetics—part c: Applications and Reviews, Vol. 42, no. 2, March 2012
20. Chris A. Mattmann and Jukka L. Zitting. 2011. Tika In Action. ISBN 9781935182856.- 256 p. 2011 y.
21. Ruslan Mitkov. 2003. The Oxford Handbook of Computational Linguistics. Oxford University Press. p. 754. ISBN 978-0-19-927634-9.
22. Tong, Simon, Uri Lerner, Amit Singhal, Paul Haahr, and Steven Baker. 2012. "Locating meaningful stopwords or stop-phrases in keyword-based retrieval systems." U.S. Patent 8, issued July 3, 2012. PP.214-385.
23. Lovins, Julie Beth. 1968. "Development of a Stemming Algorithm" (PDF). Mechanical Translation and Computational Linguistics. 11: 22–31.

24. Popovič Mirko, Willett Peter. 1992. The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data, Journal of the American Society for Information Science, Volume 43, Issue 5 (June), pp. 384–390
25. Ruslan Mitkov. 2003. The Oxford Handbook of Computational Linguistics. Oxford University Press. p. 754. ISBN 978-0-19-927634-9.
26. Atadjanov J.A. — Models of Morphological Analysis of Uzbek Words // Кибернетика и программирование. – 2016. – № 6. – С. 70 - 73. DOI: 10.7256/2306-4196.2016.6.20945 URL: https://nbpublish.com/library_read_article.php?id=20945
27. Uwe Muegge (2006), "An Excellent Application for Crummy Machine Translation: Automatic Translation of a Large Database", in Elisabeth Grafe (2006; ed.), Proceedings of the Annual Conference of the German Society of Technical Communicators, Stuttgart: tekom, 18–21.

AUTHORS PROFILE

Jasur Atadjanov is a software developer at Uztelecom. He has more than 17 years of experience in software development. He earned a Bachelor of Science at Urganch State University (USU) and a Master's Degree at Tashkent University of Information Technologies (TUIT). He also gained a Ph.D. degree at TUIT through thesis on 'Automated Library System' based on which 15 academic articles were published. Besides, he created 'ARMAT' software based on the research work and implemented it in more than 60 higher education institutions. Moreover, he is head of the Billing System Development department at Uztelecom. In Uztelecon he developed a billing system for the telecommunication area. Nowadays it is used for all regional branches of the company for more than 1.5 mln clients. Nowadays, he is working on creating a multilanguage plagiarism detection system and has published the article "Cross-Language Plagiarism Detection Based on CLAD Method" concerning the outcome achievements.

Bobur Atadjanov is a software developer at Uztelecom. He has more than 10 years of experience in software development. He earned a Bachelor of Science at Tashkent University of Information Technologies (TUIT) through a thesis on 'Cryptography by DIA Matrixes' and a Master's Degree at the same university through thesis on 'Software Development by Grid Computing'. His honorable positions in Republican Subject Olympiads can be attributed to his academic skills. As for work experience, he participated in several corporate projects including Armat.uz (Corporate electron library), LarStat(universal parser for CDR files), v3.esud.uz(an online platform for the judicial system) and the like. His research interest is the Cross-Platform plagiarism system.