The Sixth International Conference on e-Learning (eLearning-2015), 24-25 September 2015, Belgrade, Serbia

PLAGIARISM DETECTION IN HOMEWORK ASSIGNMENTS AND TERM PAPERS

DARKO PUFLOVIĆ
University of Niš, Faculty of Electronic Engineering, darkopuflovic@gmail.com

LEONID STOIMENOV
University of Niš, Faculty of Electronic Engineering, leonid.stoimenov@elfak.ni.ac.rs

Abstract: *Student obligations imply writing a large number of homework assignments and term papers. Usually, they are submitted in electronic form. Checking papers for plagiarism isn't an easy task. Quantity prevents teachers and professors to check all of them by hand. Therefore, there is a need for a system that will perform this task automatically. This paper describes principles behind one such system. Text contained in papers written by students, but also ones that can be found on internet, is converted in n-gram models, which are kept, and used later for comparison with newly generated ones. Potential application of this system is at the Faculty of Electronic Engineering in Niš, Department of Computer Science, where it can be used to check student papers, written in Serbian language.*

Keywords: *Plagiarism detection, n-gram models, Statistical language models*

1. INTRODUCTION

Appearance of computers changed the way education works. The possibilities that computers offer, supported entire process of learning with the help of advanced technologies that enable easier and faster understanding of complex concepts, as well as further progress. This aspect of education is called e-learning - type of education that uses a variety of technologies to deliver skills and knowledge. Education can be provided inside classrooms, or can be distributed through online classes, using different means of delivering content such as internet, CD-ROM, etc. [1].

Schools in the past, where classes are performed by exposing the facts, are complemented by contents that use technology, providing more information and creating a stronger connection between theory and practice.

New approach brought with it major changes in terms of working with students and evaluating their progress. One such problem is plagiarism. As an addition to the evaluation of student work, it is necessary to be sure that the work is genuine and not just a copy from a plenty of works which can be found on the internet. This task is complicated by itself, and becomes even more difficult when one considers the number of papers to be reviewed and all possible sources.

That is why there is huge need for systematized, computer aided approach, which would ensure that this task is performed automatically, or at least help teacher to find a part of text which is probably plagiarized.

Similar systems exist and can be found online. They use a large number of different approaches, and each of them is specialized in specific kind of plagiarism. The vast majority of them check papers against internet as source for comparison. The comparison results are much more accurate if the comparison is done against another document. In the next section, we'll talk about different approaches that these systems use and in chapter 3 we will discuss the approach that we used to solve this problem.

2. RELATED WORK

There are several approaches to plagiarism detection [2, 3, 4, 5, 6]. Most of them are used for texts written in the English language, but have the option to disable word lemmatization (the algorithmic process of determining the lemma for a given word) [7]. When lemmatization is disabled, results are not so accurate, because of same words that are written in various forms. On the other hand, comparison without lemmatization enables the use of these tools on documents written in other languages.

In plagiarism detection process there are several methods used:

- String matching [8] – is a method that uses basic search algorithms for finding parts of text that overlap. This method is rarely used because of its great need of computational power and storage. That makes this method bad for comparison of larger amounts of documents.
- Citation analysis [9] – finds all citations in the documents and similarities between them. More similar citations means that there is a greater likelihood that two documents share same content. This method is good for comparison of scientific papers, but it's not suitable for plagiarism detection in other kinds of papers due to lack of text content verification.
- Stylometry [10] – trying to establish a certain style that authors use in writing papers. The similarities in styles can be indication that document is plagiarized.
- Bag of words analysis [11] uses vector spaces to represent every word in document. Words are then compared with one of various methods that can be used to compare vectors.

- Fingerprinting [12] – transforms documents into n-gram models. Each n-gram is like a fingerprint of that document part. They can be compared by sentence, paragraph or entire document, and performance and memory usage are much better then in string matching method. Also, bag of word analysis may be similar to this approach if words are represented by unigram models (n-gram of length 1).

Plagiarism detection using n-gram models [13] is flexible and allow user to choose length of n-gram used in process. Based on length, n-gram models are divided into unigrams (n-gram containing one word), bigrams (composed of two words), trigrams (three words), etc. Along with words [14], n-gram models may be made of characters [15], lengths of words [16], or some other value which represents that part of sentence and allows comparison between them. Parts of sentences with same meaning should have the same values for n-grams at the appropriate places to maximize chance of finding plagiarised parts of text.

An example of plagiarism detection tool is StringSearch[1] which uses fast searching algorithms implemented in Java and can be used for string matching. Another example of available tools is CitePlag[2]. CitePlag uses citation analysis and can compare document against another document that user uploaded or against one of many documents contained in large database from which user can select one. Plagiarism Checker[3] is a representative of the group of tools that uses n-gram models to check documents against plagiarism. It's written in Python and can read document files (in *.docx* format). Models are used in combination with Google Search API.

3. OUR APPROACH FOR PLAGIARISM DETECTION

Tools that are described in the previous chapter are either made for the English language or are language neutral. Documents that our plagiarism checker should check are written in Serbian language. English based tools are not good for this task because they transform words in lemmas, but using rules for English language, which gives poor results for documents written in Serbian language. Therefore there is a need to create a new tool that transforms text using the rules for Serbian language and makes n-gram models out of transformed text.

Models that are created this way can be compared using different similarity measures. Probability of occurrence of elements in the n-gram model can also be displayed as result.

Lemmatization

Process of lemmatization can be done in multiple ways. Best two approaches are the use of morphology dictionary and stemmer [17, 18].

Complete morphology dictionary of all words in Serbian language is not available online, and making one is not an easy task. In addition to standard dictionary, it should contain all forms of given word. The lack of such resource is a major issue for text mining in Serbian documents, but in its absence, stemmer is good replacement. The combination of these two approaches is also possible.

Stemmer can be made using stemming rules. Better rules provide better results. For plagiarism detection, original word is not important to get, but different lemmas should be represented by different stems. In some cases, it's difficult to find right stem, because different words can be written the same, depending on the context in which the word is used.

Stemmer used in our application gives good results in most cases. It consists of a large number of rules (suffixes), which should be cut from the end of word.

After transformation of all words in the document it is necessary to convert document into n-gram model. Whole document can be transformed in one model, or it can be done on sentence or paragraph level. This is achievable by choosing whether sentence ending characters should be replaced by special character that represents n-gram stop character or will be simply ignored.

Another important thing in this step is to choose whether stop words will be removed from text [19]. Application have list of stop words, that can be changed, and their deletion can improve comparison results, because they are not of great importance for the process of plagiarism detection. Stop words can be inserted into copied text, because they don't alter the meaning of a sentence. That is even bigger reason why their removal is important, so that comparison is not affected in any way.

Everything described before depends on language resources, so that its realization differs from the other tools mentioned in the previous chapters.
Models that were obtained in this way can be of different lengths. Documents to be compared should be the same length.

Comparing n-gram models

Resulting n-gram models have large number of different n-grams, but a lot of them will be similar. The frequency of n-gram elements shows the distribution of words in document. Comparison of the occurrence of all n-gram elements represents similarity between those two documents.

The n-gram elements can be treated as words in bag of words analysis. Every element can be represented as vector in vector space. This raises the possibility to compare n-gram models in same manner. One of similarity measures that can be used for this comparison is cosine similarity.

[1] *https://github.com/johannburkard/StringSearch*
[2] *http://www.sciplore.org/2013/citeplag-got-a-makeover-2*
[3] *https://github.com/architshukla/Plagiarism-Checker*

Cosine similarity is similarity measure in vector space that measures the cosine of an angle between them to determine how similar they are [20]. This measure works for any number of dimensions in vector space that represents the document. Another way to calculate this value is the use of dot product:

$$Cosine(A,B) = \cos(\theta) = \frac{A * B}{||A|| * ||B||} \quad (1)$$

A * B represents dot product of vectors A and B, and ||A|| is represented as $\sqrt{a_1{}^2 + a_2{}^2 + \cdots + a_n{}^2}$

Documents can be also represented as sets of n-gram elements. In that case, the measures of similarity that are used are adapted to work with sets. Some measures on sets used are:

- Jaccard similarity [21]

$$Jaccard(A,B) = \frac{|A \cap B|}{|A \cap B| + |A \Delta B|} = \frac{|A \cap B|}{|A \cup B|} \quad (2)$$

- Sørensen–Dice similarity [22]

$$Sorensen - Dice(A,B) = \frac{2|A \cap B|}{2|A \cap B| + 1|A \Delta B|} = \frac{2|A \cap B|}{|A| + |B|} \quad (3)$$

- Anderberg's similarity [22]

$$Anderberg(A,B) = \frac{|A \cap B|}{|A \cap B| + 2|A \Delta B|} = \frac{|A \cap B|}{|A \cup B| + |A \Delta B|} \quad (4)$$

- Sokal-Sneath similarity [22]

$$Sokal - Sneath(A,B) = \frac{2|\overline{A \Delta B}|}{|[n]| + |\overline{A \Delta B}|} \quad (5)$$

- Hamming similarity [22]

$$Hamming(A,B) = \frac{|A \cap B| + |\overline{A \cup B}|}{|A \cap B| + |\overline{A \cup B}| + |A \Delta B|} = 1 - \frac{|A \Delta B|}{|[n]|} \quad (6)$$

- Roger-Tanimoto similarity [22]

$$R - T(A,B) = \frac{|A \cap B| + |\overline{A \cup B}|}{|A \cap B| + |\overline{A \cup B}| + 2|A \Delta B|} = \frac{|[n]| - |A \Delta B|}{|[n]| + |A \Delta B|} \quad (7)$$

- Hamann similarity [22]

$$Hamann(A,B) = \frac{2|\overline{A \Delta B}|}{|[n]|} - 1 = \frac{|[n]| - 2|A \Delta B|}{|[n]|} \quad (8)$$

- Russel-Rao similarity [22]

$$Russel - Rao(A,B) = \frac{|A \cap B|}{|[n]|} \quad (9)$$

- Faith similarity [22]

$$Faith(A,B) = \frac{|A \cap B| + |\overline{A \Delta B}|}{2|[n]|} \quad (10)$$

Where |A| represents number of elements in vector A

$$|A \Delta B| = (A \cup B) \backslash (A \cap B) \quad (11)$$

$$\overline{A \cup B} = [n] \backslash (A \cup B) \quad (12)$$

[n] represents superset of sets A and B.

Some similarity measures described before return the same results. The reason why we still use them is the way they calculate those results. The smaller the set after operations, the less need for memory and processing power is needed to calculate results. Choosing a good similarity measure is important for performance.

For example, cosine similarity for vectors A and B:

A = [1 0 2 6], B = [7 9 3 0],

would be calculated like this:

$$\frac{1 * 7 + 0 * 9 + 2 * 3 + 6 * 0}{\sqrt{1^2 + 0^2 + 2^2 + 6^2} * \sqrt{7^2 + 9^2 + 3^2 + 0^2}} \approx 0.715 \quad (13)$$

For other similarity measures vectors could be:

A = [a b c d], B = [b c e f].

Jaccard similarity:

$$\frac{|[b,c]|}{|[a,b,c,d,e,f]|} = \frac{2}{6} \approx 0.33 \quad (14)$$

Sørensen–Dice similarity:

$$\frac{2|[b,c]|}{|[a,b,c,d]| + |[b,c,e,f]|} = \frac{4}{8} = 0.5 \quad (15)$$

Anderberg's similarity:

$$\frac{|[b,c]|}{|[a,b,c,d,e,f]| + |[a,d,e,f]|} = \frac{2}{10} = 0.2 \quad (16)$$

Types of n-gram models

Our application uses three types of n-gram models:

- word model
- character model
- length model

Word model [14] is most commonly used in plagiarism detection. Unlike other models, word model can be used to make connection between words in sentence. This is possible because this model type is not paying attention to separators, but only words and order in which they appear in sentence.

Character model [6, 15] divides documents in characters which are then used to create model. An advantage of this approach is its ability to compare the use of separators, which can be useful when comparing the styles of writing of two authors. Downside of this model is the memory required for storage and number of n-gram elements that should be compared with another document.

Length model [16] is similar to word model, but instead of words, this model uses their lengths. This is a big improvement in terms of memory usage, but a major drawback in precision of the system, because every word

with the same length is replaced by the same number, so that a large number of words that can have a completely different meaning can be interpreted in the same way.

Results depending on n-gram length

There is a big difference in the results depending on length of n-gram that is used to create models. Short models can be used to find any similar words in documents. This can be useful if the comparison is based on finding words that have same frequency of occurrence. If this is the case, best value for length of n-gram is one (unigram model). The longer the element of n-gram model, the relationship between words in sentence becomes more important. The downside of long elements is that it's very difficult to find matches between two documents, because the goal in that case is to find long combination of same words in exactly the same order. Best results are obtained if elements have length between three to seven words.

In case of length models, best results are achieved by using elements of length between 5 and 8. Too short elements provide a large number of false positives, while too long elements have the same problem, but also the problem that word models have.

Character models give best results in case of very large elements. The characters and separators are important in this type of comparison, so longer model means greater precision. Longer models in this case mean faster and less memory-intensive comparison. Best results in this type of model are obtained using lengths between 10 and 20.

Representation of results

Similarity measures which are described in chapter 3.2. provides results that can be presented in different ways. Our application uses three different methods:

- similarity expressed as a percentage (Fig. 1)
- similarity presented in form of graph (Fig. 2)
- marking similar n-grams in documents (Fig. 3)

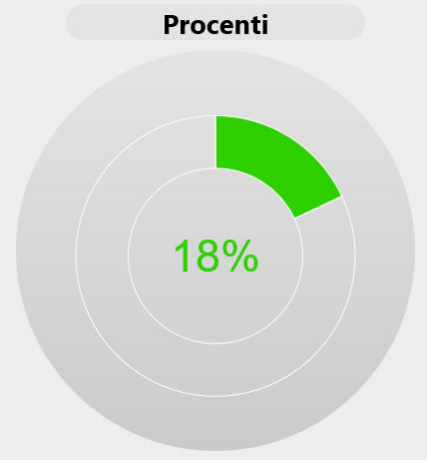

Figure 1: Percentage similarity

In case of similarity expressed as a percentage, n-gram models of two documents are compared using one of similarity measures. Every similarity measure gives results that are represented as decimal number between 0 and 1. This result can be easily transformed into percentages.

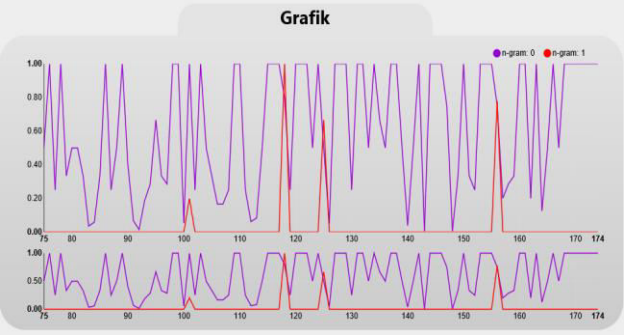

Figure 2: Similarity represented as a graph

Similarity presented in form of graph is not using similarity measures, but only the probabilities of occurrence of n-gram element in a document. Each document is represented by different line in graph, and every value is again between 0 and 1. If the value is equal to 0, the element doesn't appear in document. For all other values element is present in document and similarity between value of element in that document and value of same element in other document speaks about similarity between those two elements.

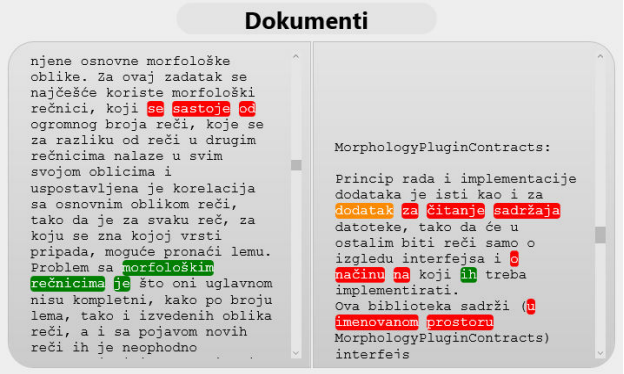

Figure 3: Similar n-grams in documents

Method of marking similar n-grams in documents works in the same manner as the similarity represented as a graph. Probabilities for each n-gram element in documents are compared between similar elements. Results are presented in 3 different colours, depending on difference in probability values. Green colour represents big difference, which probably means that the similarity of those two elements is small. Orange colour is used for closer values and red for very similar ones. Parts of text that are not highlighted contain elements that are not present in another document.

The idea behind realization of such system is the need for a systematic approach to solving the problem of plagiarism at the Faculty of Electronic Engineering (Department of Computer Science) in Niš. Implemented system should enable the creation of student work repository that will continue to be used for comparison with the incoming new works. This should largely facilitate the evaluation process.

4. CONCLUSION AND FUTURE WORK

In this paper we presented our approach to plagiarism detection in text documents written in Serbian language using n-gram models.

Text from documents can be transformed in 3 different types of n-gram models: word, character and length. Depending on the type which is used, text can be lemmatized or not. In case of word models that is best approach. This step is done using stemmer.

Plagiarism detection is most successful if stop words are removed from text. That is done using dictionary of stop words. Text is divided in words, characters or word lengths, but n-gram models are created out of their collection. Collections can be made out of words from one sentence or whole document. That depends on the way separators are interpreted. If sentence separators are replaced with special sentence ending characters then n-gram models are created out of sentences and if not, models are created from whole text.

Finally, n-gram models are ready for comparison. Comparison is done in multiple ways. One of them is the use of vector spaces and similarity measures. This can be done on level of sentence, paragraph or whole document. The result in this case is represented by number between 0 and 1 that can be transformed into percentage.

Another way of comparison can be done using probability of n-gram element occurrence in n-gram model. Results in this case can be represented in two different ways, as a graph or as a text with highlighted words or characters that are likely plagiarised.

Our plagiarism checker gives good results, but there are parts that can be changed, to make results even better. Biggest problem with current system is lemmatization. Resources for Serbian language are scarce.

Also, morphology dictionary is one of the resources that are needed to make the system much more accurate. Another solution could be better stemmer. Stemming rules can be difficult to write. Even better solution would be to use stemmer that has the ability to reconstruct original lemma.

Another part of application that can be revisited is n-gram model itself. The current approach to creating models is good, but can be more memory friendly if other data structures are used for storage [23]. That would make a big difference in memory usage in case of really long documents and large lengths of n-gram models. In the case when the length of the model is very large, words, characters or lengths (especially in the case of words and characters) are repeated often in n-gram elements. Approach using linked lists and hash table would allow parts of elements to be preserved once, and instead of them model could use only pointer.

ACKNOWLEDGMENT

Research presented in this paper was funded by the Ministry of Science of the Republic of Serbia, within the project "Technology Enhanced Learning in Serbia", No. III 47003.

LITERATURE

[1] D. Randy Garrison, *E-Learning in the 21st Century: A Framework for Research and Practice*, Routledge, New York, 2011.

[2] Sven Meyer zu Eissen, Benno Stein, *Intrinsic Plagiarism Detection* (Springer, London, 2006).

[3] Antonio Si, Hong Va Leong, Rynson W. H. Lau, *A document plagiarism detection system,* Proceedings of the 1997 ACM symposium on Applied computing, 1997.

[4] Bela Gipp, Norman Meuschke, Joeran Beel, *Comparative evaluation of text (and citation) based plagiarism detection approaches using guttenplag*, Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries, 2011.

[5] Martin Potthast, Alberto Barrón-Cedeño, Benno Stein, Paolo Rosso, *Cross-language plagiarism detection*, Springer, 2010.

[6] Efstathios Stamatatos, *Intrinsic Plagiarism Detection Using Character n-gram Profiles*, Proceedings of the SEPLN'09 Workshop, 2009.

[7] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, *Introduction to Information Retrieveal*, Cambridge University Press, 2008.

[8] Stéphane Ducasse, Oscar Nierstrasz, Matthias Rieger, *On the effectiveness of clone detection by string matching,* Journal of Software Maintenance and Evolution: Research and Practice, 2006.

[9] Bela Gipp, Jöran Beel, *Citation based plagiarism detection: a new approach to identify plagiarized work language independently,* Proceedings of the 21st ACM conference on Hypertext and hypermedia, 2010.

[10] Hermann Maurer, Frank Kappe, Bilal Zaka, *Plagiarism - A Survey*, Journal of Universal Computer Science, vol. 12, no. 8, 2006.

[11] M. Zechner, M. Muhr, R. Kern, M. Granitzer, *External and Intrinsic Plagiarism Detection Using Vector Space Models*, Proceedings of the SEPLN'09 Workshop, 2009.

[12] Yurii Palkovskii , Alexei Belov , Irina Muzika, *Exploring Fingerprinting as External Plagiarism Detection Method*, CLEF 2010, 2010.

[13] Alberto Barrón-Cedeño, Paolo Rosso, *On Automatic Plagiarism Detection Based on n-Grams Comparison*, Springer, 2009.

[14] Ido Dagan, Lillian Lee, Fernando C. N. Pereira, *Similarity-Based Models of Word Cooccurrence Probabilities*, Springer, 1999.

[15] Paul McNamee, James Mayfield, *Character N-Gram Tokenization for European Language Text Retrieval*, Springer, 2004.

[16] Alberto Barrón-Cedeño, Chiara Basile, Mirko Degli Esposti, Paolo Rosso, *Word Length n-Grams for Text Re-use Detection*, Springer, 2010.

[17] Kimmo Kettunen, Tuomas Kunttu, Kalervo Järvelin, *To stem or lemmatize a highly inflectional language in a probabilistic IR environment?*, Journal of Documentation, 2005.

[18] Vlado Kešelj, Danko Šipka, *A suffix subsumption-based aproach to building stemmers and lemmatizer for highly inflectional languages with sparse resources*, INFOthecha, 2008.

[19] Prasha Shrestha, Thamar Solorio, *Using a Variety of n-Grams for the Detection of Different Kinds of Plagiarism*, CLEF 2013, 2013.

[20] Jun Ye, *Cosine similarity measures for intuitionistic fuzzy sets and their applications*, Elsevier, 2010.

[21] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, Supachanun Wanapu, *Using of Jaccard Coefficient for Keywords*, Proceedings of the International MultiConference of Engineers and Computer Scientists 2013, Hong Kong, 2013.

[22] János Podani, *Introduction to the Exploration of Multivariate Biological Data*, Backhuys Publishers, 2000., pp. 55-110.

[23] Daniel Robenek, Jan Platoš, Václav Snášel, *Efficient In-memory Data Structures for n-grams Indexing*, DAtabases, TExts, Specifications and Objects (DATESO), 2013., pp. 48–58.