Detecting Similarity Between Text documents using N-gram Analysis and Sentence Similarity

NATURAL LANGUAGE PROCESSING CS6320.001
Charu Arora (cxa150730@utdallas.edu)
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas
Richardson, Texas-75080

ABSTRACT

Today, a colossal amount of digital data is generated and there is a plethora of sources. This increase in data, has spiked the level of plagiarism with it. Currently, plagiarism detection is based mainly at string level. This research paper sheds light on a method of detecting similarity between text documents using natural language processing techniques. This method is capable of detecting not only true copy documents but also detects lightly paraphrased sentences within documents.

1. INTRODUCTION

With the advancements in the technological world and the development of Internet, accessing any information from around the world has become an easy task. Some people intentionally or unintentionally misrepresent this information as their own work without giving appropriate acknowledgement. This phenomenon is defined as Plagiarism. In the academic domain, this is considered an offence and has become more prevalent with the use of electronic sources. Thus, detecting plagiarism has become the top priority especially for the academic institutions.

Plagiarism is of different types. Some of the basic types include:

- True copy (Also known as copy – paste, where word to word is copied directly from the data source or data sources.)
- Slight paraphrasing (Use of synonyms from thesaurus to display a someone else's slightly altered text as their own piece of work.)
- Heavy paraphrasing (This includes changing the sentence type by combining two sentences, changing the order of the sentence, changing the part of speech)

There are a number of different plagiarism detection techniques that have developed over time. With the currently available tools, it is easy to detect true copy but detecting paraphrasing is still an on-going research. MOSS (Measure of Software Similarity), developed at Stanford, detects similarity in program codes written in C, C++, JAVA etc. Another software that detects plagiarized content within text documents is called Turnitin. A major limitation with this software is that it can only detect plagiarism in documents when they are an exact copy of the source or they contain exact sentences from different sources. Copied sentences that are altered using a thesaurus or those that are cleverly paraphrased can still achieve a very low percentage of plagiarism using Turnitin.

Therefore, it is not a very efficient way of detecting true plagiarism which makes efficient plagiarism detection a challenging and a daunting task.

The aim of this paper is to use Natural Language Processing (NLP) techniques to detect plagiarism in text documents and thus improve the accuracy of the existing techniques. The next section discusses some of the methodologies currently being used. Following that, Section 3 will focus on the methodology chosen in this paper to detect plagiarism and presents its results in Section 4. The limitations and future work are elaborated in Section 5 and finally conclusions are drawn in the last section.

2. PREVIOUS WORK

Here, some of the methods are discussed which are currently being used in detecting plagiarism.

- In [4], Lukashenko uses Euclidean distance to calculate the similarity between two documents. If the two documents are identical, their score is zero. Another method that they used is Cosine Similarity, which is similar to the scalar product of document vectors divided by their lengths.

- In [5], Shivakumar uses word frequency occurrences to compare new documents with their already existing documents that have been registered.

- Clough [7] used charts such as the dotplots to visualize and detect plagiarism. They used the plots to display the density of overlapping between two sentences.

- Turnitin is currently the best commercial tool, recommended by Joint Information Systems Committee in 2010, used by numerous educational institution to detect and prevent plagiarism. Due to its limitations, that are, not being able to detect paraphrasing and inaccessibility to all the data sources available on the web [6], it is not efficient.

- A few researchers like Clough (2003) and Ceska (2009) have tried to incorporate basic Natural Language Techniques to detect plagiarism. Clough stated that, NLP techniques can help improve the accuracy by detecting paraphrased sentences. Ceska used techniques like tokenization, stop- word removal, lemmatization, number replacement and synonym recognition in their research. Even though these techniques did provide a slightly better accuracy of detecting plagiarism, no substantial improvement was noticed as compared to the basic methods.

3. METHODOLOGY

This section discusses about the natural language preprocessing techniques and the similarity detection algorithm.

3.1 PRE-PROCESSING TECHNIQUES

The basic pre-processing techniques used in this experiment is as follows:

- Sentence Segmentation

  The document is split into sentences and processing the document is done line by line.

- Tokenization

  The sentence is tokenized which means that each word, punctuation and symbols are split to get tokens. For example, the sentence ‘I can write.’ Is converted to a list with 4 elements, ‘I’, ‘can’, ‘write’ and ‘.’.

- Lowercase

  Each letter in the document is converted to lowercase for generalized matching. For example, ‘The’ is converted to ‘the’.

- Stop-word removal

  Removing functional words such as articles, pronouns, prepositions and determiners. For example, ‘a’, ‘an’, ‘the’, ‘such’, ‘for’, ‘do’.

- Removal of Punctuation

  All the symbols and punctuation are removed from each of the sentence. For example, ‘?’,’(‘, ‘)’, ‘.’.

- Lemmatization

  All the words are transformed to their dictionary base to generalize the analysis. For example, ‘saddest’ is transformed to ‘sad’.

  All of these techniques were applied in combination to the documents [3]. These processed documents were then analyzed using the following algorithms:

- N-gram Similarity
- Sentence Similarity
- Longest Common Subsequence.

The method described in this experiment checks for similarity between two documents. Also, an assumption is made that it is known which document is the original document.

3.2 COMPARING METHODOLOGIES

3.2.1 N-gram similarity

Initially, n-grams are calculated for each of the two documents before pre-processing the text. In this experiment, ‘n’ has been checked for values from 1 to 5. An example of a trigram is:

Sentence -

"I love reading books."

Trigrams –

- "I", "love", "reading"
- "love", "reading", "books"
- "reading", "books", "."

Manning & Schutze use the Jaccard method to compare trigrams.

$$J(A, B) = \frac{|\,\mathrm{S(A)} \cap S(B)|}{|\,\mathrm{S(A)} \cup S(B)\,|}$$

where S(A) and S(B) represent the set of trigrams in the original and the copied document. The numerator is the intersection of the set of trigrams between the two documents while the denominator consists of all the trigrams in both documents. [1]

In this method, I have slightly altered this metric. Here the numerator remains the same, but the normalization is done based on the set with the minimum number of trigrams. This is a better measure for document pairs with varied lengths.

$$T(A, B) = \frac{|\,\mathrm{S(A)} \cap S(B)|}{|\,\min\,(\mathrm{S(A)}, S(B))\,|}$$

This measure was used not only for trigrams, but also to calculate the bigram, 4-gram and 5-gram similarity.

3.2.2 Sentence Similarity

When a sentence is paraphrased, generally, either the order of the words is changed or synonyms are used. For example,

Sentence –
"Last night, I went grocery shopping with my friends."

Paraphrased Sentence –
"I went grocery shopping yesterday with my friends."

When we use the above method, since the order of the words are changed and synonyms are used, the value of the similarity coefficient drops significantly. In this method, after pre-processing both the documents, each sentence from the copied document is compared with every sentence in the original document. It finds the intersection of words between two sentences and then compares the length of the two sentences to find the similarity between them. For a more generalized comparison, the synset from WordNet is used to find synonyms of all the words in both sentences and find their intersection.

3.2.3 Longest Common Subsequence

After pre-processing the text, without removing the stop words, the longest common subsequence included in both the documents is computed. This technique helps in finding documents that are a true copy of the original text.

4. EXPERIMENTAL RESULTS

For this experiment, the corpora developed by Clough and Stevenson is used. It consists of 100 documents out of which 5 are original documents, which are passages taken from Wikipedia. 95 of the remaining documents are classified as follows:

- True Copy
- Light paraphrasing
- Heavy paraphrasing
- Non-plagiarism

The following results are displayed after trimming the data furthermore due to limitations discussed in the next section. The n-gram similarity of each of these documents with the original document is shown below.

	1-gram	**2-gram**	**3-gram**	**4-gram**	**5-gram**	**Similarity Coefficient**
True Copy	.52	.93	.98	.98	.98	.96
Light Paraphrasing	.50	.79	.73	.64	.56	.49
Heavy Paraphrasing	.26	.22	.11	.07	.05	.07
Non plagiarized	.16	.05	.03	0	0	.02

The table above displays the data before pre-processing the document.

	1-gram	**2-gram**	**3-gram**	**4-gram**	**5-gram**	**Similarity Coefficient**
True Copy	.98	.44	.23	.14	.08	.96
Light Paraphrasing	.67	.15	.03	.08	.0	.49
Heavy Paraphrasing	.60	.13	.05	.02	.01	.07
Non plagiarized	.47	.08	.01	0	0	.02

The table above displays the data after pre-processing the document as a whole (without sentence segmentation).

4.1 IMPACT OF PRE-PROCESSING TECHNIQUES

It is observed that stop word removal has a huge impact on the n-gram similarity. Reduction in the n-grams is directly related to the removal of function words. The accuracy, especially for value of $n>2$, significantly drops. Lemmatization also has the same impact on n-gram similarity.

To improve this accuracy, n-gram similarity is done at the sentence level and not as a whole document. The sentence similarity, uses pre-processed techniques to remove punctuations, function words. Each of this sentence in the copied document is compared with every sentence from the original document.

Based on the number of words that intersect, including synonym matching, and the length of the two sentences, the percentage similarity is computed. This is then averaged, based on the comparison with all the sentence to compute the percentage similarity for the whole document. The final column in the two tables represent this value.

5. DISCUSSION

Using NLP techniques when comparing two sentences, it gives a better accuracy on the similarity of the statement but when computed as a whole, can decrease its accuracy. Even though the results shown above look promising, they do have some limitations.

For the Sentence Similarity method, it performs an $n*m$ comparison, where n is the number of sentences in the copied document and m is the number of sentenced in the original document. If the document consists of many sentences, this algorithm can be time consuming. The above method, even though it provides a better accuracy, is only good for short documents. Also, since synonym sets from WordNet are used, there is also a dependency there. Only those synonyms that are available in WordNet, can be matched. For example,

Sentence 1:
'This chocolate is known to be the best chocolate in the world.'

Sentence 2:
'Lindt is called as one of the user's favorite chocolates.'

The word 'known', when lemmatized changed to 'know' and the word 'called' from sentence 2 changes to 'call'. Even though they are clear synonyms based on the two sentences, they are not matched using WordNet.

6. CONCLUSION

In this paper, we were able to detect plagiarism in true copy documents and also documents that were slightly paraphrased. NLP techniques such as lemmatization, punctuation and stop word removal at the sentence level improves our accuracy of plagiarism.

In the future, high end NLP techniques can be incorporated to further improve the accuracy of detecting similarity.

Also, using machine learning to train the software with the writing style of a particular person, such as training on the person's vocabulary usage, common spelling mistakes, distribution of words, frequency of words and average sentence length, can help predict if a text has been written by the same person [8].

From our results and limitations, it can be concluded that plagiarism detection will only improve with NLP techniques but as of now, humans still need to intervene to judge plagiarized cases.

7. REFERENCES

1. Chong, Miranda, Lucia Specia, and Ruslan Mitkov. "*Using natural language processing for automatic detection of plagiarism."* Proceedings of the 4th International Plagiarism Conference (IPC 2010), Newcastle, UK. 2010.

2. Clough, P., & Stevenson, M. (2009). "*Developing a corpus of plagiarised short answers.*" Language Resources and Evaluation, LRE 2010. (To be published)

3. Ceska, Zdenek, and Chris Fox. "*The influence of text pre-processing on plagiarism detection.*" Association for Computational Linguistics, 2011.

4. R. Lukashenko, V. Graudina, and J. Grundspenkis. "*Computer-based Plagiarism Detection Methods and Tools: an Overview.*" In Proc. of CompSysTech, pages 1–6, 2007.

5. Shivakumar, N. and Garcia-Molina, H. (1995) "*SCAM: A Copy Detection Mechanism for Digital Documents.*" In: 2nd International Conference in Theory and Practice of Digital Libraries (DL 1995), June 11-13, 1995, Austin, Texas.

6. Batane, T. (2010). "*Turning to Turnitin to Fight Plagiarism among University Students*." Educational Technology & Society, 13 (2), 1–12.

7. Clough, P. (2003). "*Old and new challenges in automatic plagiarism detection.*" National Plagiarism Advisory Service, (February edition), (pp. 391-407).

8. Clough, P. (2000). "*Plagiarism in Natural and Programming Languages: An Overview of Current Tools and Technologies.*" Technical Report CS-00-05, University of Sheffield, UK.