行政院國家科學委員會專題計畫期中報告

計畫名稱：大學生之網路相關著作權觀念、態度、行為與

寫作剽竊研究(2/3)

執行期限：97年8月1日至98年7月31日

計畫編號：NSC96-2520-S-009-003-MY3

主持人：周 倩

執行單位：國立交通大學教育研究所

Plagiarism Detection using N-gram Occurrence Statistics Based on ROUGE

Chien-Ying Chen [1], Jen-Yuan Yeh[2], Hao-Ren Ke[1, 3]

[1] Institute of Information Management, National Chiao Tung University,
No. 1001, University Rd., Hsinchu 300, Taiwan
[2] Department of Computer Science, National Chiao Tung University,
No. 1001, University Rd., Hsinchu 300, Taiwan
[3] University Library, National Chiao Tung University,
No. 1001, University Rd., Hsinchu 300, Taiwan
andy1629.iim95g@g2.nctu.edu.tw, jyyeh@cis.nctu.edu.tw, claven@lib.nctu.edu.tw

Abstract. With the arrival of Digital Library Era, control of information flow is nearly impossible; the lack of control leads to free usage of any digital information available. Plagiarism occurs when users fail to credit the original owner for the borrowed content. Two main approaches to plagiarism detection are fingerprinting and term occurrence, whose performances are affected when copied content has been modified. We propose the application of ROUGE in plagiarism detection, which had been successfully exploited in the evaluation of automatic summaries, in hope of overcoming the deficit suffered by fingerprinting-based and term occurrence-based methods. We evaluated the performance of ROUGE-based methods with a self-created corpus and empirically determined an ideal setting for each method.

Keywords: plagiarism detection; ROUGE; n-gram co-occurrence statistics

1. Introduction

Looking up plagiarism in some of the dictionaries, one will find different definitions. Though slightly different from one another, these definitions convey an identical idea – plagiarism is the use of other people's work/idea as one's own without crediting the original owner. This kind of behavior is equivalent to stealing. Nevertheless, cases of plagiarism are still being reported in classes and even in academic research. Maurer et al. [5] described the policies against plagiarism in some of the most prestigious universities and how each of the schools handles such misconduct; some of these universities were seeing increasing number of reported plagiarism cases, including Web plagiarism, in recent years that ranged from 2003 to 2006. Not only does plagiarism violate copyright regulation, but also influence the quality of education and research. Knowledge is accumulated through learning and thinking, and school assignments force students to learn and think during the process of completing the assignments. However, plagiarism deprives students of undergoing such process as they spend less time to think. Even if students do read the content before they plagiarize, they are most likely to forget about the content faster than those who genuinely do their work. In the academic research domain, no new discoveries will be made if the researchers only reuse existing information. Collberg et al. [1] focused on self-plagiarism and argued that self-plagiarism causes new but similar papers to be published without contributing to the overall advancement of academic research.

Although there are different plagiarism detection approaches, each method has its pros and cons. One common weakness is the vulnerability to text modification which can be achieved through addition, deletion and substitution of words, and also change of sentence structure or word order. In this research, we propose a prototype of a system based on ROUGE [3], which calculated n-gram occurrence statistics to evaluate the quality of a candidate summary with regard to one or more reference summaries. The n-gram co-occurrence statistics in ROUGE was originally adopted from BLEU [6] which focused on evaluation of the quality of machine translation. In both research, the more similar a candidate text is to a reference text, the better the qualities of summary and translation. The same concept may be applied to plagiarism detection – the more similar a candidate text is to a reference text, the more likely that plagiarism has occurred. Moreover, two methods in ROUGE, longest common subsequence (LCS) and skip-bigram, may work on certain types of plagiarism. In general, the proposed methods should be able to conquer most of the text alteration strategies mentioned earlier. We will discuss in more detail about related work in Section 2, methodology in Section 3, experiments and evaluation in Section 4, and conclusion in Section 5.

2. Related Works

Until now, quite a considerable amount of research has focused on plagiarism detection. Fig. 1 provides an overview about the development of plagiarism detection. The classification is derived from the taxonomy in [7]. As Fig. 1 indicates, the plagiarism detection methods can be categorized into three main categories: fingerprinting, term occurrence, and style analysis.

Fingerprinting can be considered as the most widely adopted approach in plagiarism detection. The origin of this method, as suggested by previous studies, is attributed to the work done by Udi Manber [4]. In that research, Manber aimed to find out similar documents in a database. The research was based on Rabin Fingerprint scheme, which was applied to generate a unique identity (fingerprint) for each document. Rabin fingerprint scheme or hash function as often used interchangeably, can transform a sequence of substring into an integer. And a good scheme/function should generate the same integer for the same substring; on the other hand, it should generate different integer for each unique substring to ensure consistency and avoid undesirable collisions of fingerprints.

Term occurrence is probably the most intuitive approach because lexical words contain explicit information of the text and they can be analyzed to determine the similarity between two documents. One assumption is that the more terms both documents have in common, the more similar they are. N-gram co-occurrence statistics in ROUGE can be classified under term occurrence; therefore, term occurrence has been applied to a range of studies such as automatic evaluation of summaries and automatic evaluation of machine translation mentioned earlier, as well as common information retrieval problems like clustering and categorization. Due to a common purpose between the aforementioned studies and plagiarism detection, i.e. determining similarities between documents, application of term occurrence in plagiarism detection seems promising.

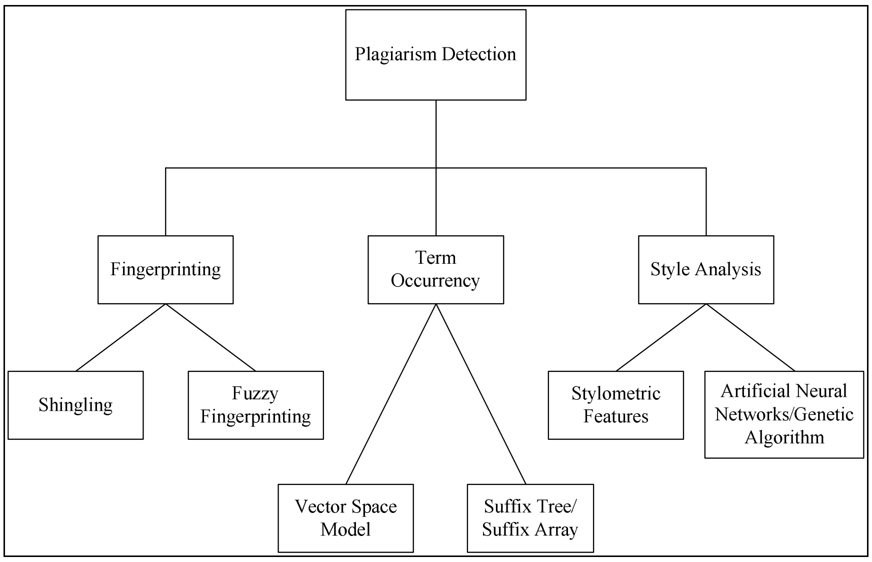

Fig. 1. Classification of Major Plagiarism Detection Approaches and derived methods

Style analysis is the most special approach to plagiarism, because it focuses more on implicit information than explicit information of the texts. The basic principle behind style analysis is that every author has his/her own writing style, which should remain consistent throughout the text, and that the characteristics of each style is hard to manipulate or imitate, making the plagiarized portion of work to stand out in the text implicitly [2]. Although style analysis may not need a reference corpus, it needs to be trained to learn about rules of writing. Hence, various artificial neural networks (ANNs) and genetic algorithms (GAs) have been applied to analyze style and authorship [2]. The trained ANNs or GAs will be able to recognize the style of a particular author and therefore articles written by the author.

3. Methodology

Having discussed about existing plagiarism detection methods, the methods proposed in this research will be discussed next. The purpose of this research is to provide a framework of a plagiarism detection tool.

The methods are based on the foundation of n-gram co-occurrence statistics at sentence level. N-gram co-occurrence statistics can detect verbatim copy as good as fingerprinting, and when N=1, the method is immune from change of sentence order. By including longest common subsequence and skip-bigram, we hope to overcome problems caused by addition and deletion of original text. Fig. 2 shows the detection procedures.

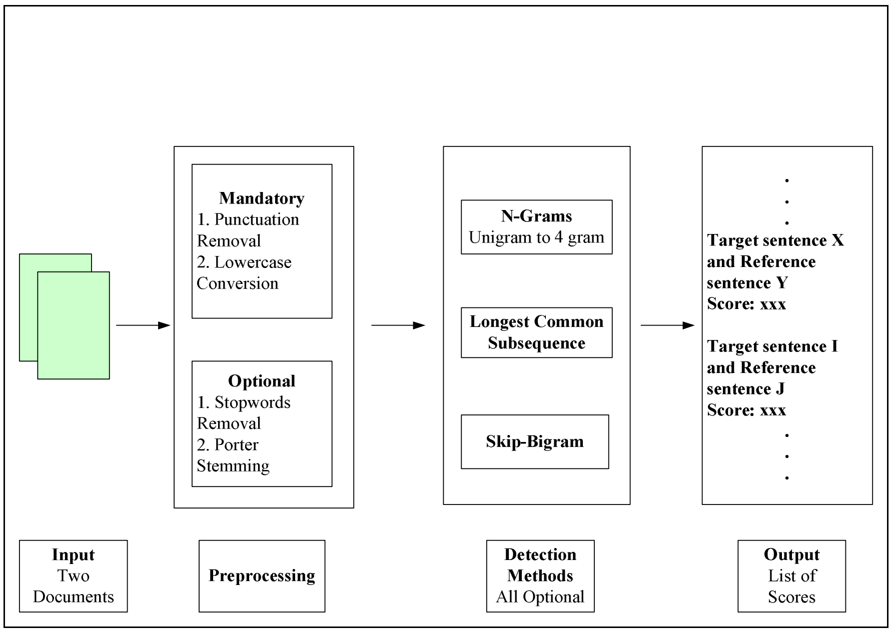

Fig. 2. Detection Procedures of the Proposed Tool

3.1 Preprocessing

Before two documents can be compared with each other, steps are taken to process the content of the documents. There are a total of four preprocessing steps as oftenly used in information retrieval studies, which include punctuation removal, lowercase conversion, stopwords removal, and Porter stemming.

3.2 ROUGE-N

ROUGE-N in this research includes unigram to 4-gram. We will use unigram as an example to explain how ROUGE-N works. Each token in a sentence is a unigram. Before comparing the sentences, every unique unigram and its number of occurrence(s) in the sentence will be recorded for every sentence. Beginning with the first sentence of the reference document, every unique unigram is compared with all unique unigrams in every sentence of the candidate document, followed by the second sentence of the reference document and so on and so forth. Overall, all reference sentences will be compared with all candidate sentences for a total of $M \times N$

times, where M and N are the number of sentences in the reference and candidate documents respectively.

The number of overlapping unigram(s) between two sentences, one from the reference document and the other from the candidate document, will be counted. The overlapping total, numerator of Equations (1) and (2), is divided by the length of the reference sentence and length of the candidate sentence separately in order to calculate recall and precision. We take the smaller number of occurrence of the overlapping unigrams in the two sentences as the numerator. Such modification is called *clipping* in BLEU [6]. Clipping is to prevent exaggerated precision score in certain cases. Fig. 3 is an example from BLEU, in which the word, *the*, appears in both the reference and candidate sentences two times and seven times respectively, if according to Equation (1) without the *clipping* mechanism, the precision score contributed by this word will be 7/7, which is clearly exaggerated. However, if the score is clipped it becomes 2/7, which is more reasonable.

$$\underset{\substack{1\le u\le y\\ 1\le v\le z}}{R-N}(S_u^R,S_v^C) = \frac{\sum_{\text{n-gram} \in S_u^R \text{ and } S_v^C} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum_{\text{n-gram}' \in S_u^R} \text{Count}(\text{n-gram}')} \qquad (1)$$

$$\underset{\substack{1\le u\le y\\ 1\le v\le z}}{P-N}(S_u^R,S_v^C) = \frac{\sum_{\text{n-gram} \in S_u^R \text{ and } S_v^C} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum_{\text{n-gram}' \in S_v^C} \text{Count}(\text{n-gram}')} \qquad (2)$$

Candidate Sentence:
the the the the the the the
Reference Sentence:
the cat is on the mat

Standard Precision: 7/7
Clipped Precision: 2/7

Fig. 3. Example of the Clipping Mechanism

N-gram (including unigram) score is expressed as Equation (3) below:

$$\underset{\substack{1\le i\le M\\ 1\le j\le N}}{F-N}(S_i^R,S_j^C) = \frac{2*R-N*P-N}{R-N+P-N} \qquad (3)$$

The comparison procedure for n-grams (from two to four) is the same as unigram. The differences are the comparing unit, n-grams, and the number of n-grams.

3.3 Longest Common Subsequence (LCS)

LCS is the longest in-sequence string of matched tokens between two sentences. In unigram matching, the position of matched token is not a constraint. As long as a unigram co-occurs in both sentences, it will contribute to the similarity between two sentences. Although LCS is also based on matching unigrams, it only considers matched tokens that form the longest in-sequence subsequence of the reference sentence. In other words, even if a unique unigram is in both the reference and candidate sentences, but if it is out of order with other matched tokens, it is not included in the LCS and will not contribute to the LCS score (See Fig. 4 for example). The numerator of both Equations (4) and (5) is the LCS between a reference sentence and a candidate sentence. Equation (6) is the averaged LCS score between two sentences.

Candidate sentence 1: Police kill the gunman

Candidate sentence 2: The gunman kills police

Reference sentence: Police killed the gunman

The LCS between Reference sentence and Candidate sentence 1 is *police the gunman* while the LCS between Reference sentence and Candidate sentence 2 is *the gunman, excluding police*. The first pair of sentences shows the skipping nature of LCS and the second pair of sentences shows the in-sequence rule that bounds LCS.

Fig. 4. Example of LCS

$$R-LCS(S_v^C,S_u^R)=\frac{LCS(S_v^C,S_u^R)}{\sum_{\text{unigram}\in S_u^R}\text{Count (unigram)}} \qquad (4)$$

$$P-LCS(S_v^C,S_u^R)=\frac{LCS(S_v^C,S_u^R)}{\sum_{\text{unigram}\in S_v^C}\text{Count (unigram)}} \qquad (5)$$

$$F-LCS(S_j^C,S_i^R)=\frac{2*(R-LCS)*(P-LCS)}{(R-LCS)+(P-LCS)} \qquad (6)$$

3.4 Skip-Bigram

Skip-bigram is a variation of bigram. The difference is the formation of bigrams. For skip-bigram, bigrams are formed not only by consecutive tokens, but also by other in-sequence tokens within the window. Skip distance, *d*, has to be set before finding skip-bigrams in the sentence. Skip distance is the maximum number of tokens in between any two combining tokens. When skip distance is determined, we can find all possible skip-bigrams in a sentence starting from the first token. Let $w_1, w_2, \ldots, w_n$ be the sentence. At w_1, w_1 can form a bigram with every token up to w_4, in case of d=2, followed by w_2, which will form a bigram with every token up to w_5. The entire process stops when the last skip-bigram is formed by w_{n-1} and w_n. Fig. 5 shows an example of ski-bigram, and Equations (7) to (9) illustrate how skip-bigram score is calculated.

For a given sequence: Andy eats an apple

When d=2, skip-bigrams generated will be as follows:

Start with the first word *andy*, it can form a bigram with the furthest token, *apple*, and followed by *eats* and *an* respectively. When *andy* has formed bigrams with all possible tokens, *eats* will form bigrams with *an* and *apple*. Finally, the last skip-bigram is *an apple*.

There are a total of six skip-bigrams by the sequence above when d=2. They are as shown:

andy eats, *andy an*, *andy apple*, *eats an*, *eats apple*, *an apple*

Fig. 5. Example of Skip-bigrams Formation

$$R-Skip(S_u^R, S_v^C) = \frac{SKIP2_{clip}(S_u^R, S_v^C)}{(p-d-1)(d+1)+\sum_{r=0}^{d-1} d-r} \tag{7}$$

$$P-Skip(S_u^R, S_v^C) = \frac{SKIP2_{clip}(S_u^R, S_v^C)}{(q-d-1)(d+1)+\sum_{r=0}^{d-1} d-r} \tag{8}$$

$$F-Skip(S_j^C, S_i^R) = \frac{2*(R-Skip)*(P-Skip)}{(R-Skip)+(P-Skip)} \tag{9}$$

In situations where people insert or delete words from an original sentence, or change tenses from past tense to past perfect tense and vice versa, pure bigram has minimal use; this is because the bigrams will not be the same between reference and candidate sentences. As skip-bigram allows gaps, it has higher chance of producing the correct bigrams to match with the ones in the reference sentence.

4 Experiments and Evaluation

4.1 Data Set

In the field of plagiarism detection, there is not a standard and valid plagiarism corpus that is publicly available yet. There are a number of works that used news corpus such as the Reuters News corpus for their evaluation, while a small number of works used research articles corpora that are managed by the university and therefore only accessible by the university members. The remaining choice is to make one's own plagiarism corpus, which usually is relatively small due to limited resources. This research adopted the last approach instead of using a news corpus because even though news content is often reused, modification of this nature may not be able to represent plagiarism.

The data set used in this research is called the *abstract* set. It was based on the observation that abstracts of some papers are actually formed by sentences taken from the main text. Such characteristic may be utilized to simulate the plagiarism scenario by treating the abstract as the candidate of plagiarism and the main text as the source being plagiarized. It consists of 978 pairs of annotated sentence pairs, among which 32 pairs have been annotated as plagiarism.

4.2 Experiments

The goal of the experiments is to determine the performance of each method in recognizing plagiarism instances under different thresholds. Since the effectiveness of the proposed methods in detecting plagiarism were unknown, all methods were tested using the *abstract* set with different thresholds, in multiples of 10 starting from 0 with the maximum threshold being 100 because F-measures were multiplied by 100 and treated as percentage. Every pair of sentences received a score from each method and the score was compared with the threshold. The comparison would yield four different outcomes namely true positive (TP), false positive (FP), true negative (TN), and false negative (FN). If the score was larger than the threshold and the pair was annotated as *plagiarism*, then it would be considered as the method had correctly identified a plagiarism instance and TP would be recorded; on the other hand, a FP would be recorded instead if the pair was annotated otherwise.

By looking at the problem from an information retrieval perspective, the number of plagiarism pairs was the number of relevant documents. Therefore the performance of each method could be evaluated in terms of recall, precision, and F-measure. These three measures are represented by Equations (10) to (12) respectively.

$$Recall = \frac{TN}{TN + FP} \qquad \textbf{(10)}$$

$$Precision = \frac{TP}{TP + FP} \qquad \textbf{(11)}$$

$$F - measure = \frac{2 \times Recall \times Precision}{Recall + Precision} \qquad (12)$$

By running the methods with the *abstract* set, each method under a specific preprocessing setting would generate a table like Table 1. There are four possible settings: both stopwords and stemming are applied (SW+SM), stopwords are removed (SW), stemming is applied (SM), and neither stopwords nor stemming are applied (No Pre).

Table 1. Experimental Results of Unigram for the *Abstract* Set

Threshold	TP	FP	TN	FN	Recall	Precision	F-measure
0	32	946	0	0	1	0.03272	0.063366
10	31	884	62	1	0.96875	0.03388	0.06547
20	31	701	245	1	0.96875	0.04235	0.081152
30	30	381	565	2	0.9375	0.072993	0.13544
. . .							
100	6	0	946	26	0.1875	1	0.315789

4.3 Evaluation

The primary goal of the evaluation was to determine and recommend a desirable setting for each method. To recommend a setting for each method, comparison between the performances of the same method under different settings was necessary.

Comparison was made easier by creating Table 2 and generating **Fig. 6**. By visualizing the results (**Fig. 6**), the characteristics and performance of each setting were relatively clearer than just by looking at the numbers. Nevertheless, the lines were close sometimes and more detailed information was needed to make the right judgment. In this case, F-measure with No Pre - F(No Pre) and F-measure with SM - F(SM) had the top two highest F-measures at threshold=50 while F(SW+SM) and F(SW) formed smoother curves on the graph. By only looking at threshold≥50 as in **Fig. 7**, F(SW+SM) and F(SW) obviously performed better than F(SM) and F(No Pre); hence the choices were cut down to two: F(SW+SM) and F(SW). Even with only two choices, it was hard to decide which preprocessing yielded better results. Therefore, we referred to data in Table 1 to see the number of TPs, FPs, TNs, and FNs. When the number of TPs was substantial, lower number of FPs was our top priority because less time will be spent to filter out FPs. By going through the same step for every method, we came up with Table 3.

Table 2. F-measures of Unigram under Different Settings

Threshold	F(SW+SM)	F(SW)	F(SM)	F(No Pre)
0	0.063366	0.063366	0.063366	0.063366

10	0.064854	0.07721	0.066184	0.069264
20	0.094512	0.133333	0.084584	0.098257
30	0.221374	0.31016	0.172308	0.219608
40	0.396694	0.45098	0.419355	0.490566
50	0.638889	**0.67692**	**0.6875**	**0.70968**
60	**0.65385**	0.666667	0.625	0.625
70	0.653061	0.638298	0.439024	0.439024
80	0.4	0.4	0.358974	0.358974
90	0.222222	0.222222	0.27027	0.27027
100	0.171429	0.117647	0.060606	0.060606

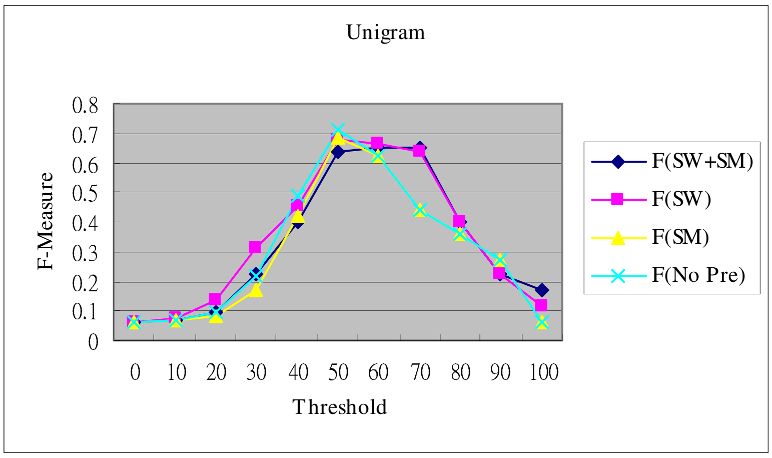

Fig. 6. Line Graph of Unigram under Different Settings

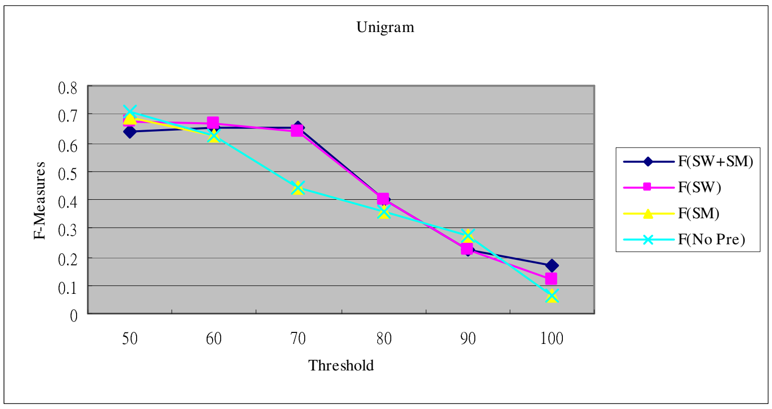

Fig. 7. Partial Graph of **Fig. 6**

Table 3. Empirically Recommended Setting for Each Method

Methods	**Recommended Setting**
Unigram	Stopwords, Threshold=60
Bigram	No Preprocessing, Threshold=40
Trigram	Stemming, Threshold=30
4-gram	Stopwords, Threshold=30
Skip-bigram	Stopwords & Stemming, Threshold=30
LCS	Stopwords & Stemming, Threshold=50

5. Conclusion

We implemented ROUGE on plagiarism detection. There are a total of six methods, unigram to 4-gram, LCS and skip-bigram. Through experiments and observation of the results, we empirically determined a setting for each method. Further experiments should be carried out to determine the value of each method in plagiarism detection. Meanwhile, WordNet [8] may be added into the tool to help detect plagiarism involving substitution of words.

References

1. Collberg, C., Kobourov, S., Louie, J., & Slattery, T. SPlaT: A System for Self-Plagiarism Detection. In *Proceedings of IADIS International Conference WWW/Internet 2003*, vol. 1, 508 -- 514. Algarve, Portugal (2003)
2. Dierderich, J. Computational Methods to Detect Plagiarism in Assessment. *Information Technology Based Higher Education and Training* pp. 147--154 (2006)
3. Lin, C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out*, Post-Conference Workshop of ACL 2004, pp. 74-81. Barcelona, Spain (2004)
4. Manber, U. Finding Similar Files in a Large File. In *Proceedings of the USENIX Winter 1994 Technical Conference*, pp. 2--2. San Francisco, California (1994)
5. Maurer, H., Kappe, F., & Zaka, B. Plagiarism – A Survey. *Journal of Universal Computer Science*, vol. 12, no. 8, pp.1050 -- 1084 (2006)
6. Papineni, K., Roukos, S., Ward, T., & Zhu, W. -J. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311--318. Philadephia, USA (2002)
7. Stein, B., & Meyer Zu Eissen, S. Near Similarity Search and Plagiarism Analysis. *Data and Information Analysis to Knowledge Engineering*, vol. 10, 430 –437 (2006)
8. WordNet: <http://wordnet.princeton.edu/>