Comparing Three Plagiarism Tools (Ferret, Sherlock, and Turnitin)

Mitra Shahabi[1] *mitra.shahabi@ua.pt*
Department of Language and Culture
University of Aveiro
Aveiro, 3800-356, Portugal

Abstract

An attempt was made to carry out an experiment with three plagiarism detection tools (two free/open source tools, namely, Ferret and Sherlock, and one commercial web-based software called Turnitin) on Clough-Stevenson's corpus [1] including documents classified in three types of plagiarism and one type of non-plagiarism. The experiment was toward Extrinsic/External detecting plagiarism. The goal was to observe the performance of the tools on the corpus and then to analyze, compare, and discuss the outputs and, finally to see whether the tools' identification of documents is the same as that identified by Clough and Stevenson.

Keywords: Plagiarism detection tool, Ferret, Sherlock, Turnitin, Clough-Stevenson's corpus.

1. INTRODUCTION

Plagiarism, defined as the act of using others´ ideas and words in a text document without acknowledging the sources, is one of the most increasing issues in academic communities especially for the higher education institutions [2]. The existence of Internet and online search engines has advanced the international collaboration in education but at the same time it also has raised the plagiarism opportunity. Nowadays, pre-written essays are accessible online through the websites, essay banks or paper mills. This technology can be misused by the students and lead them to plagiarism.

Motivated by the plagiarism problem, a field namely plagiarism detection arises. Both the academic and commercial communities put their effort to detect plagiarism [1]. Plagiarism analysis can be distinguished as intrinsic and extrinsic analysis [3]. In intrinsic analysis, the aim is to detect plagiarism within the document (i.e. the source does not to be identified); whilst in extrinsic analysis, the aim is to detect plagiarism across documents (i.e. comparing suspicious documents with their potential sources).

Plagiarism detection methods in natural language originate from diverse areas such as file comparison, information retrieval, authorship attribution, file compression, and copy detection. These approaches work well to handle text with minimal alterations such as word-for-word plagiarism. However, they still have problems in detecting paraphrasing plagiarism, plagiarism of ideas, and cross-lingual plagiarism where the text is altered significantly [1]. The academic and commercial communities are still in the process of delivering a better plagiarism detection solution; see for example the three competitions on plagiarism detection in the recent years: PAN´09, PAN´10, and PAN´11. PAN´11 was held in conjunction with 2011 CLEF conference [4]; eleven plagiarism detection were evaluated based on the third revised edition of the PAN plagiarism corpus PAN-PC-11. Figure 1 shows the overview of important corpus parameters [4].

Comparing the detection performance measures of plagdet, precision, recall, and granularity of

[1] PhD student in Translation with scholarship from Fundação para a Ciência e a Tecnologia (FCT) (Portugal), with reference number SFRH/BD/60210/2009

the detectors[2], Grman and Raven [5] was known as the best-performing detector and Grozea and Popescu [6] and Oberreuter et al. [7] were known as the second and the third best-performing tools, respectively (cited in [4]).

Document Statistics

Document Purpose		Plagiarism per Document		Document Length	
source documents	50%	hardly (5%-20%)	57%	short (1-10 pp.)	50%
suspicious documents		medium (20%-50%)	15%	medium (10-100 pp.)	35%
– with plagiarism	25%	much (50%-80%)	18%	long (100-1000 pp.)	15%
– without plagiarism	25%	entirely (>80%)	10%		

Plagiarism Case Statistics

Obfuscation		Case Length	
none	18%	short (<150 words)	35%
paraphrasing		medium (150-1150 words)	38%
– automatic (low)	32%	long (>1150 words)	27%
– automatic (high)	31%		
– manual	8%		
translation ({de, es} to en)			
– automatic	10%		
– automatic + manual correction	1%		

FIGURE 1: A screenshot of the corpus statistics for 26 939 documents and 61 064 plagiarism cases in the PAN-PC-11.

In comparison with the performance reported in PAN´09 and PAN´10, a PAN´11 shows a drop in the plagdet performance; this result has been attributed to an increased detection difficulty [4].

There are different plagiarism detection tools among which we can refer to Turnitin, Glatt, Eve2, Wordcheck, CopyCatchGold, and so on [8; 9; 10; 11; 12; 13].

The tools performance is usually based on two methods, statistical, semantical, or both. However, the statistical method are better welcomed since they are easily applicable

In this study, an extrinsic plagiarism detection experiment was conducted. The applied detection tools were using three tools Ferret [14], Sherlock [15] and Turnitin, which is an online service created by iParadigms, LLC. The rest of this document will explain the details of the tools and corpus, discussion of the experiment results, and conclusion of the experiment.

2. THE COURPUS

In this study, the freely available Clough-Stevenson's corpus [1] was applied. The corpus consists of answers to five short questions on a variety of topics in Computer Science field. The five short questions are:

1. What is inheritance in object oriented programming?

[2]

$$prec(S,R) = \frac{1}{|R|}\sum_{r\in R}\frac{|\bigcup_{s\in S}(\mathbf{s}\sqcap\mathbf{r})|}{|\mathbf{r}|},\ rec(S,R) = \frac{1}{|S|}\sum_{s\in S}\frac{|\bigcup_{r\in R}(\mathbf{s}\sqcap\mathbf{r})|}{|\mathbf{s}|},$$

$$gran(S,R) = \frac{1}{|S_R|}\sum_{s\in S_R}|R_s|,\quad plagdet(S,R) = \frac{F_1}{\log_2(1+gran(S,R))}$$

S: the set of plagiarism in the corpus; R: the set of detection reported by a plagiarism detector for the suspicious document; F1: the equally weighted harmonic mean of precision and recall. Plagdet is the combination of the other three measures.

2. Explain the PageRank algorithm that is used by the Google search engine.
3. Explain the Vector Space Model that is used for Information Retrieval.
4. Explain Bayes Theorem from probability theory.
5. What is dynamic programming?

To simulate plagiarism, for each question, a suitable entry in Wikipedia which contains the answer to the question was selected as the source document. In order to represent a variety of different degrees of plagiarism, participants were asked to answer the question using one of the following models (pp. 7-8):

Near Copy: Participants were asked to answer the questions by performing copy-and-paste action from the relevant Wikipedia entry of 200-300 words without any instruction about which parts of the article to copy.

Light Revision: Participants were asked to answer the questions by performing copy-and-paste action from the relevant Wikipedia entry and they may alter it in some basic ways such as substituting words and phrases with synonyms and also paraphrasing. However, they are not allowed to alter the order of information found in the sentences.

Heavy Revision: Participants were asked to answer the questions by performing copy-and-paste action from the relevant Wikipedia entry and instructed to rephrase the text without any constraint about how to alter the text.

Non-plagiarism: Learning materials such as lecture notes or textbooks sections that are relevant with the questions were provided to the participants. They were asked to answer the questions by using their own knowledge including what they had learned from the materials provided. Participants were allowed to look at other materials but Wikipedia to answer the questions.

Accordingly, the corpus consists of 100 documents: five Wikipedia entries and 95 answers provided by 19 participants. A breakdown of the number of answers in the corpus can be seen in Table 1. The average length of file in the corpus is 208 words and 113 tokens. 59 of the files are written by native English speakers and the remaining 36 files by non-native speakers.

Category	Learning Task					Total
	A	B	C	D	E	
Near Copy	4	3	3	4	5	19
Light Revisions	3	3	4	5	4	19
Heavy Revisions	3	4	5	4	3	19
Non-plagiarism	9	9	7	6	7	38
Total	19	19	19	19	19	95

TABLE 1: Corpus breakdown

3. THE PLAGIRISM DETECTION TOOLS

Plagiarism detection tools are useful in terms of detecting and also preventing plagiarism. Since there are many tools available now, one should be wise on selecting it according to their need. And also, as plagiarism detection software only gives suggestion to the user about the suspicious documents, further analysis should be done by human as well as the final decision.

For this study, the three plagiarism detection tools Ferret, Sherlock, and Turnitin were compared and analyzed. The systems detect plagiarism based on the statistical methods of matching n-gram words (adjacent `words´ of input), between the texts. The comparison is carried out between all the documents, i.e. every document is compared with every other document. As the

tools read the documents they extract all n-grams of the two documents under the comparison and then match them. Afterwards, they calculate the rate of documents similarity based on the following formula, where A is "the set of n-grams extracted from one of the documents and B is the set of n-grams from the comparing document by [16].

$$Similarity = \frac{Number\ of\ common\ trigrams}{Total\ number\ of\ trigrams} = \frac{A \cap B}{A \cup B}$$

3.1. Ferret

Ferret is a freely available standalone plagiarism detection system developed at the University of Hertfordshire. It runs on Windows environment and very easy to install and run. File formats that Ferret can process are .txt, .rtf, .doc and .pdf. The algorithm is written in C++. Ferret takes a set of documents, converts each text into reference number, set of characteristic trigrams. It compares every text with each other based on counting the number of distinct trigrams similar between the texts, and produces a list of file-pairs together with the similarity scores that ranked from the most similar pair to the least similar one. This count is used to calculate the resemblance measure, as the number of similar trigrams in a pair of documents, divided by the total number of different trigrams in the pair. Ferret manifests the scores of similarity precisely like 0.90991. The numbers were rounded for sake of being simplified for analysis; in this case, for example, it was taken as 0.91. The system allows user to select any pair of texts and do further investigation as they will be displayed side by side with similar paragraphs highlighted (similar parts in blue and different parts in black). See the Figures 2 and 3.

3.2. Sherlock

Sherlock is a free and open source plagiarism detection program for essays, computer source code files, and other kinds of textual documents in digital form. It turns the texts into digital signatures to measure the similarity between the documents. A digital signature is a number formed by turning several words (3 by default) in the input into a series of bits and joining those bits into a number.

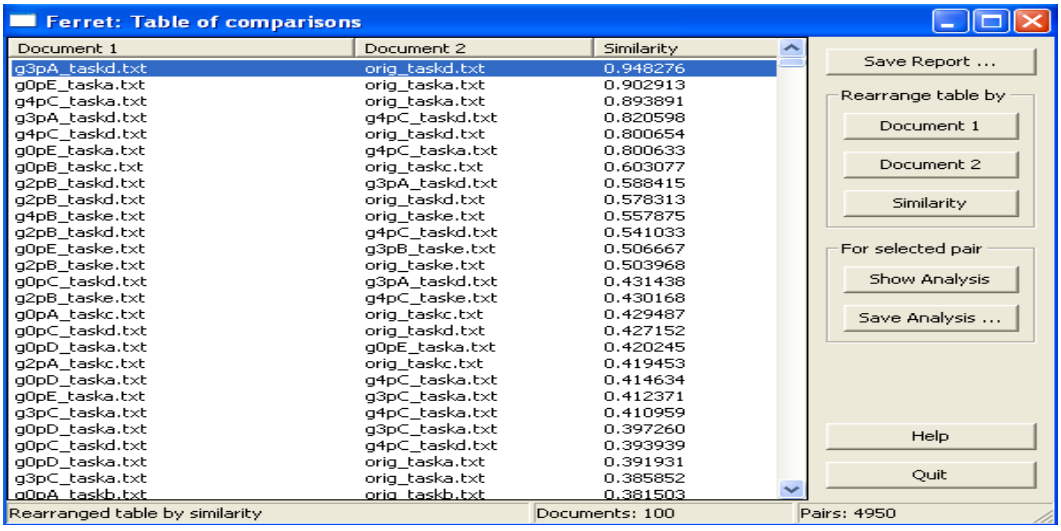

FIGURE 2: A screenshot of Ferret showing a table of comparison

Sherlock is written in C programming language (Fig. 4) and needs to be compiled before being installed either on Unix/Linux or Windows. It is a command-line program and it does not have a graphical user interface. Executing a "sherlock *.txt" command will compare all the text files in the current directory and produce a list of file-pairs together with the similarity percentage (Fig. 5). This output list is not ordered by the similarity percentage.

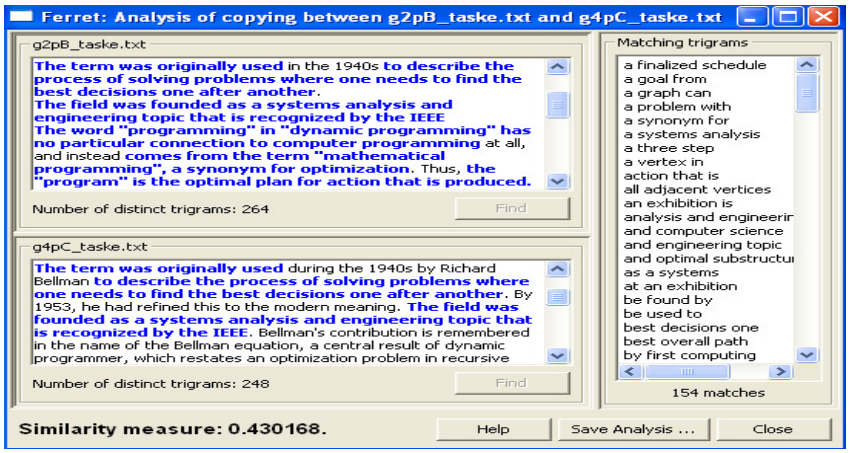

FIGURE 3: A screenshot of Ferret showing the analysis of copying between two texts

Important point to be noted when analyzing the output of Sherlock is the fact that 100% score does not imply that the files are identical because the Sherlock program actually throws away some data randomly in the process in order to simplify and speed up the match.

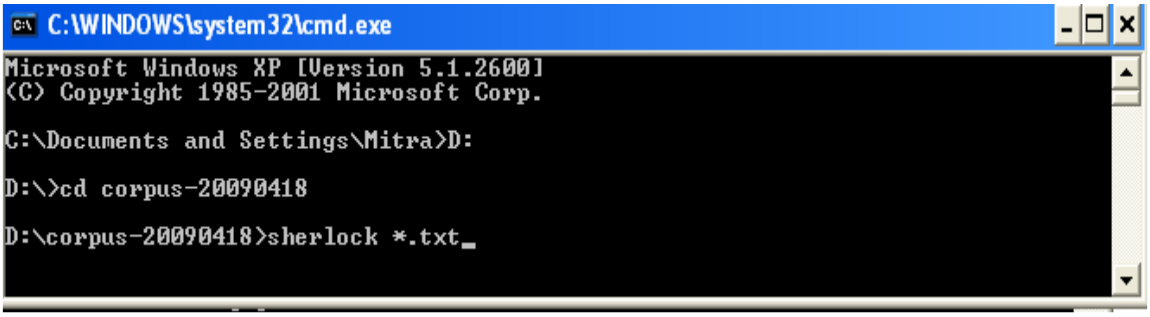

FIGURE 4: A screenshot of Sherlock showing a command-line

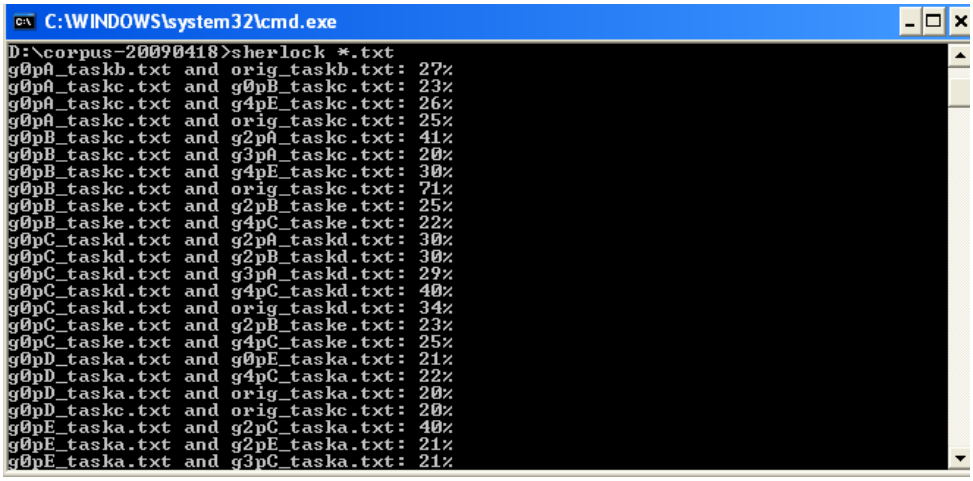

FIGURE 5: A screenshot of Sherlock showing the results the similarity of the compared documents

There are four command-line options giving a possibility to change the numbers in the command line in order to see different performance results.

a) *-t threshold%.* The system is showing the files with similarities of 20% by default; the higher this threshold the more similar files are printed.
b) *-z zerobits*. The 'granularity' of the comparison is 4 by default but it can be changed from 0 to 31. However, it should be noticed that the higher this number, the less exact the comparison will be but the faster, and vice versa.
c) *-n number_of_words*. The default for the system is 3 words (3-gram) form one digital signature. We can change the number of words (min 1, max 7); the higher the number the slower but more exact the process however "the less likely they are to co-occur in both texts" (Specia, 2010), and vice versa.
d) *-o outfile*. It is to store the different results, acquired by making some changes in the aforementioned program options, in the same folder that the corpus exists.

Example: sherlock -t 80% -z 3 -n 2 -o results.txt *.java (see Fig. 4).

With Sherlock, it is not possible to see what parts of the compared documents are similar. It is only possible to see the rate of similarity of the documents in question in percentage (see Fig. 5).

3.3. Turnitin

Turnitin is a web-based subscription plagiarism detection service, maintained by a company named iParadigms. To use this service, user simply has to log on to Turnitin website without any other installation. Turnitin detects material copied from the Internet and also cross-checking of submitted essays within a task as well as other text documents in the database. Every submitted essay is added to the database and will be used in the future when other essay is submitted. Turnitin offers a free restricted trial account that allows user to submit five text documents over 30 days period. In this trial account, access to the Turnitin database is not given.

In Turnitin, we cannot have, like Ferret, both the documents in one window to see the similarities of the compared texts. The only document that is shown is the suspicious text; the parts similar to the other document appear in red the distinct parts are in black color (see Fig. 6).

FIGURE 6: A screenshot of Turnitin showing the results the similarity of the compared documents

4. METHODOLOY

The experiment was carried out with the three tools on the corpus. The present study did not cover all the results reported by the three tools; the focus was only on the results of comparison between the students' documents (tasks a to e) and their related original sources (original a to e). The results of comparison between the student' documents, or in case of Turnitin the comparison

with other sources, were left.

For Sherlock, *t* (threshold) was changed from 0.20 to 0.00 in order to make the tool compatible with Ferret and Turnitin which report the similarities from 0.00.

After analyzing the differences and similarities between the three tools, the goal was to find whether or not their outputs match the classification of the tasks presented by Clough and Stevenson.

As the outputs of all systems appeared in numbers, the Clough-Stevenson's classification of documents (Appendix B) was also needed in numbers; hence, the mean similarity between the documents and the Wikipedia articles illustrated by Clough and Stevenson [1] (p.14) was used for this purpose. See Figure 7.

Category	$c_n(A,B)$ for n-gram					lcs_{norm}
	1	2	3	4	5	
Near copy	0.95	0.89	0.85	0.81	0.78	0.88
Light revision	0.87	0.70	0.56	0.46	0.39	0.76
Heavy revision	0.81	0.52	0.34	0.26	0.21	0.58
Non-plagiarised	0.63	0.23	0.05	0.01	0.00	0.41

FIGURE 7: mean similarity between the documents and the Wikipedia articles illustrated by Clough and Stevenson

5. ANALYSIS AND DISCUSSION

Appendix A shows the results of all the three systems along with Clough-Stevenson's classification of the documents Ferret and Sherlock, in most cases, reported the results more or less the same, but Turnitin's outputs in many cases were greatly different from the other two, usually showing a higher percentage of similarities (Appendix A). In order to investigate the reason, The system´s 'analysis part' was checked to see the overlapped parts of the two documents in order to examine whether or not the tools have matched the compared documents properly. It could be realized only with Ferret and Turnitin, because as aforementioned before Sherlock has the drawback of not providing a graphical user interface showing the two documents with the overlapped and distinct parts; it just reports the percentages results.

It was discovered that Turnitin performs quite well and it is Ferret that does not show the expected percentage, because it considers the longer text (for this corpus, the longer is always the source [1]) as the base and then looks how much of this text is overlapped by the shorter text and the result is shown as the percentage of similarity between the two documents[3], i.e. if the suspicious document is, for example, 100% similar to the original document but its size covers only 40% of the original source, Instead of reporting 100% plagiarism, Ferret reports 40% plagiarism.

Regarding the fact that Ferret and Sherlock reported a quite similar output it was speculated that Sherlock, probably, performs like Ferret. And because of the problems addressed to Ferret and Sherlock, the comparison was only made between the Turnitin's output and Clough-Stevenson's classification.

Analyzing the data in Appendix A, it was discovered that out of 95 documents, Turnitin identified 61 documents similar to and 34 documents different from Clough-Stevenson's classification of documents. Table 2 illustrates these 34 cases. The system acted properly for all the non-plagiarized tasks; the outputs match with Clough-Stevenson's. The differences up to 0.20 between Turnitin outputs and Clough-Stevenson's classification of the documents was ignored since, for Clough-Stevenson's classification of the texts, the mean similarity was considered for

[3] It is infact the shorter text which must be checked how much of it has been covered by the original text.

comparison; however, for Turnitin's the exact percentage of similarity was taken into account.

Table 2 shows the documents whose rate of plagiarism has been wrongly reported by Turnitin. The figures in blue indicate 0.40 ≤ 0.20 differences between the results of the system and the Clough and Stevenson´s; and the reds signify a considerable difference (≥ 0.40) between them.

	Document 1	Document 2	Clough -Stevenson (mean similarity)	Turnitin
3	g0pA_taskc.txt	orig_taskc.txt	0.56	0.85
4	g0pA_taskd.txt	orig_taskd.txt	0.34	0.00
11	g0pC_taska.txt	orig_taska.txt	0.34	0.00
15	g0pC_taske.txt	orig_taske.txt	0.56	0.89
17	g0pD_taskb.txt	orig_taskb.txt	0.56	0.76*
18	g0pD_taskc.txt	orig_taskc.txt	0.34	0.58
21	g0pE_taska.txt	orig_taska.txt	0.56	0.99
22	g0pE_taskb.txt	orig_taskb.txt	0.34	0.66
27	g1pA_taskb.txt	orig_taskb.txt	0.34	0.00
28	g1pA_taskc.txt	orig_taskc.txt	0.56	0.26
29	g1pA_taskd.txt	orig_taskd.txt	0.85	0.34
34	g1pB_taskd.txt	orig_taskd.txt	0.56	0.35
35	g1pB_taske.txt	orig_taske.txt	0.85	0.50
36	g1pD_taska.txt	orig_taska.txt	0.56	0.34
42	g2pA_taskb.txt	orig_taskb.txt	0.34	0.00
43	g2pA_taskc.txt	orig_taskc.txt	0.56	0.78*
44	g2pA_taskd.txt	orig_taskd.txt	0.85	0.31
48	g2pB_taskc.txt	orig_taskc.txt	0.34	0.00
49	g2pB_taskd.txt	orig_taskd.txt	0.56	0.93
51	g2pC_taska.txt	orig_taska.txt	0.85	0.66*
54	g2pC_taskd.txt	orig_taskd.txt	0.34	0.56
55	g2pC_taske.txt	orig_taske.txt	0.56	0.00
62	g3pA_taskb.txt	orig_taskb.txt	0.34	0.00
68	g3pB_taskc.txt	orig_taskc.txt	0.34	0.00
69	g3pB_taskd.txt	orig_taskd.txt	0.56	0.30
75	g3pC_taske.txt	orig_taske.txt	0.56	0.78*
78	g4pB_taskc.txt	orig_taskc.txt	0.34	0.61
79	g4pB_taskd.txt	orig_taskd.txt	0.56	0.82
84	g4pC_taskd.txt	orig_taskd.txt	0.34	0.97
85	g4pC_taske.txt	orig_taske.txt	0.56	0.91
86	g4pD_taska.txt	orig_taska.txt	0.56	0.28
91	g4pE_taska.txt	orig_taska.txt	0.34	0.00
92	g4pE_taskb.txt	orig_taskb.txt	0.56	0.93
93	g4pE_taskc.txt	orig_taskc.txt	0.85	0.32

TABLE 2: The differences between Turnitin's output and Clough-Stevenson's classifications[4].

In order to simplify the results, the wrong outputs are presented below in table 3, in the way

[4] As this table has been, in fact, extracted from the table in Appendix A, the numbers in the left column seem out of order.

Clough and Stevenson classified the texts (near-copy, heavy revision, light revision, and non-plagiarism). If the differences between the Turnitin´s figures are higher than 20 it is an indication of changing the level of the text in the classification of texts; that is, the text with 0.56 rate of plagiarism, being classified by Clough and Stevenson as highly revised, was seen in turnitin´s outputs with 0.35 rate of plagiarism; so it was reported as a lightly revised text in turnitin´s results. It can be concluded that the blue colors mean texts with one level higher or lower than the real classification of the text, and the red color identifies two levels higher or lower than the accurate position; except for light revision and near-copy texts which the difference of the rate of their plagiarism is ≤0.30); they have been marked with (*). The differences are summarized in table 3 below.

	Turnitin´s			
Clough & Stevenson´s	Near-copy	High revision	Light revision	Non-plagiarism
Near-copy	–	–	–	–
High revision	8	–	4	1
Light revision	1	5	–	7
Non-plagiarism	–	3	1	–

TABLE 3: Differences of texts classification between Turnitin´s outputs and Clough and Stevenson´s classification of texts

As noticed in table 3, the noises produced by Turnitin are as follows: 8 highly revised texts were reported near-copy, 4 were reported lightly revised, and 1 as non-plagiarized; one lightly revised texts was reported near-copy, 5 were reported as highly revised, and 7 as non-plagiarized; and 3 non-plagiarized texts were reported as highly revised and 1 as lightly revised.

Regarding the fact that Ferret and Sherlock reported a quite similar output it was speculated that Sherlock, probably, performs like Ferret. Although they did not report the results in a manner expected (like Turnitin), their outputs were evaluated in terms of precision and recall (Table 4). Only file-pairs of answer and source within the same task were included for the evaluation.

As noticed in table 4, both Ferret and Sherlock give a perfect precision score for all cases, starting from similarity score 0.1 for Ferret and similarity percentage 10% for Sherlock, which means all captured documents are indeed plagiarism. However, both systems give a very low recall score when thresholds are set very high (0.5 for Ferret and 50% for Sherlock). As the thresholds are set lower, the recall scores are getting higher. At similarity score threshold 0.1, recall score of Ferret is 0.68421053 where 39 out of 57 cases of plagiarism detected. At similarity percentage threshold 10%, recall score of Sherlock is 0.57894737 where 33 out of 57 cases of plagiarism detected.

Ferret			Sherlock		
	Precision	Recall		Precision	Recall
≥ 0.5	1	0.14035088	≥ 50%	1	0.10526316
≥ 0.4	1	0.19298246	≥ 40%	1	0.1754386
≥ 0.3	1	0.31578947	≥ 30%	1	0.22807018
≥ 0.2	1	0.45614035	≥ 20%	1	0.36842105
≥ 0.1	1	0.68421053	≥ 10%	1	0.57894737

TABLE 4: Precision and Recall of Ferret and Sherlock Outputs

In Ferret output, the majorities of the file-pairs captured with similarity score ≥ 0.2 are near copy and light revision plagiarism. Among 26 suspicious documents, only four of them are categorized as heavy revision and all these four texts are written by non-native English speakers. The file-pairs captured with similarity score 0.1-0.2 vary between near copy, light revision, and heavy

revision plagiarism. All of the non-plagiarism answers have similarity score below 0.03. There are also three heavy revision plagiarism texts within this range as well as one near copy plagiarism text written by a non-native English speaker but he/she claims a very good knowledge of the question topic and the question is perceived as a not difficult one. There is only one document that contains plagiarism but has zero similarity score against its original.

Similar to Ferret, in Sherlock output, the majorities of the file-pairs captured with similarity percentage ≥ 30% are near copy and light revision plagiarism. Only one of 13 suspicious documents is a heavy revision and it is written by a non-native English speaker. The rest of the heavy plagiarism answers have similarity percentage below 30% along with the other near copy and light revision plagiarism. There are 15 texts which have similarity percentage between 1%-9% and three of them are non-plagiarism. Setting the threshold to 1% will give 45 (out of 57) documents that contains plagiarism with different degrees and three non-plagiarism documents, which implies there are 12 documents that actually contains plagiarism but assigned a similarity percentage of zero.

6. CONCLUSION

In this paper it was tried to reveal some strengths and weaknesses of three plagiarism detection tools, namely, Sherlock, Ferret, and Turnitin. They were compared according to their features and performances. The criterion for selecting these tools for this study was to discover how the easily available or free/open source tools are performing and at the end which of them can be considered the best. Since one of the advantages of open source tools is that we can improve them in order to meet our goals. It appeared that Ferret and Sherlock, in most cases, produce the same results in plagiarism detection performance; however, Turnitin reported the results with great difference from the other two tools: It showed a higher percentage of similarities between the documents and the source. After investigating the reason (just checked with Ferret and Turnitin, cause Sherlock does not provide a view of the two documents with the overlapped and distinct parts), it was discovered that Turnitin performs quite acceptable and it is Ferret that does not show the expected percentage; it considers the longer text (for this corpus the longer is always the source) as the base and then looks how much of this text is overlapped by the shorter text and the result is shown as the percentage of similarity between the two documents, and this leads to wrong results. Therefore, there is always a need for human intervention to make a lot of effort to check if the output reports a real percentage of plagiarism. From this it can be also speculated that Sherlock does not manifest the results properly. Although they did not report the results in a manner expected (like Turnitin), their outputs were evaluated in terms of precision and recall.

Both Ferret and Sherlock give a perfect precision score for all cases, which means all captured documents are indeed plagiarism. However, both systems give a very low recall score when thresholds are set very high. As the thresholds are set lower, the recall scores are getting higher.

7. Future work

A change in Ferret system program can probably solve the problem of giving wrong percentage, because its problem seems just in giving the non-intended percentage, and it works well in matching the 3-grams. One negative point of Sherlock is the user interface; It does not have a graphical user interface, i.e., it does not manifest the content of the texts in condition we need to analyze the output. It is very important that a user be able to easily compare the parts that are marked similar. For this purpose it is better that the tool displays the comparing files next to each other with highlighting similar parts

The reliability of the Clough-Stevenson's corpus, as the only base of evaluation, is also questionable.

8. REFERENCES

1. P. Clough and M. Stevenson. “Developing a Corpus of Plagiarized Short Answers, Language Resources and Evaluation: Special Issue on Plagiarism and Authorship Analysis, In Press.” Internet: http://ir.shef.ac.uk/cloughie/resources/plagiarism_corpus.html#Download, Sep. 10, 2009 [Oct. 12, 2011].

2. G. Judge. “Plagiarism: Bringing Economics and Educations Together (With a Little Help from IT).” Computers in Higher Economic Review, vol. 20(1), pp. 21-26, 2008.

3. B. Stein and S. Meyer zu Eissen. “Near similarity search and plagiarism analysis,” in From Data and Information Analysis to Knowledge Engineering, M. Spiliopoulou et al., EDs. Springer, 2006, pp. 430-437.

4. M. Potthast et al. “Overview of the 3rd international competition in plagiarism detection”: notebook for PAN at CLEF 2011, in Notebook Papers of CLEF 2011 LABs and Workshops, 19-22 Sep., Amsterdam, The Netherlands, 2011.

5. J. Grman and R. Ravas. “Improved implementation for finding text similarities in large collection of data: notebook for PAN at CLEF 2011, in Notebook Papers of CLEF 2011 LABs and Workshops, 19-22 Sep., Amsterdam, The Netherlands, 2011.

6. C. Grozea and M. Poescu. “The encoplot similarity measure for automatic detection of plagiarism”: notebook for PAN at CLEF 2011, in Notebook Papers of CLEF 2011 LABs and Workshops, 19-22 Sep., Amsterdam, The Netherlands, 2011.

7. G. Oberreuter, G. L´Huillier, S. Apíos, and J. D. Velasquez. “Approaches for intrinsic and external plagiarism detection”: notebook for PAN at CLEF 2011, in Notebook Papers of CLEF 2011 LABs and Workshops, 19-22 Sep., Amsterdam, The Netherlands, 2011.

8. M. Delvin. “Plagiarism detection software: how effective is it? Assessing Learning in Australian Universities.” Internet: http://www.cshe.unimelb.edu.au/assessing learning/docs/PlagSoftware.pdf, 2002 [Sep. 23, 2012].

9. T. Lancaster and F. Culwin. “A review of electronic services for plagiarism detection in student submissions.” the Teaching of Computing, Edinburgh, 2000. Internet: http://www.ics.heacademy.ac.uk/events/presentations/317_Culwin.pdf, 2000 [Oct. 01, 2012].

10. T. Lancaster and F. Culwin. “Classifications of Plagiarism Detection Engines.” *ITALICS*, vol. 4 (2), 2005.

11. H. Maurer, F. Kappe, and B. Zaka. “Plagiarism – A Survey.” *Journal of Universal Computer Sciences*, vol. 12 (8), pp. 1050 – 1084, 2006.

12. C. J. Neill and G. Shanmuganthan. “A Web – enabled plagiarism detection tool.” *IT Professional*, vol. 6 (5), pp. 19 – 23, 2004.

13. C. Lyon, R. Barrett and J. Malcolm. “A theoretical basis to the automated detection of copying between texts and its practical implementation in the Ferret plagiarism and collusion detector,” in Proc. The Plagiarism: Prevention, Practice and Policies Conference, 2004.

14. R. Pike. “The Sherlock Plagiarism Detector.” Internet: http://www.cs.su.oz.au/~scilect/sherlock, 2007 [Oct. 04, 2011].

15. J. Malcolm and P. Lane. “Efficient Search for Plagiarism on the Web.” *Kuwait*, vol. 1, pp. 206-211, 2008.

APPENDICES

APPENDIX A: The results shown by the three systems & Clough and Stevenson´s mean similarity of documents

	Documents		Plagiarism detection tools			
	Document 1	Document 2	Clough-Stevenson (mean similarity)	Ferret	Sherlock	Turnitin
1	g0pA_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
2	g0pA_taskb.txt	orig_taskb.txt	0.85	0.38	0.27	1.00
3	g0pA_taskc.txt	orig_taskc.txt	0.56	0.42	0.25	0.85
4	g0pA_taskd.txt	orig_taskd.txt	0.34	0.06	0.00	0.00
5	g0pA_taske.txt	orig_taske.txt	0.05	0.00	0.00	0.00
6	g0pB_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
7	g0pB_taskb.txt	orig_taskb.txt	0.05	0.01	0.00	0.00
8	g0pB_taskc.txt	orig_taskc.txt	0.85	0.60	0.71	0.74
9	g0pB_taskd.txt	orig_taskd.txt	0.56	0.22	0.16	0.58
10	g0pB_taske.txt	orig_taske.txt	0.34	0.11	0.15	0.49
11	g0pC_taska.txt	orig_taska.txt	0.34	0.05	0.00	0.00
12	g0pC_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
13	g0pC_taskc.txt	orig_taskc.txt	0.05	0.00	0.00	0.00
14	g0pC_taskd.txt	orig_taskd.txt	0.85	0.42	0.34	0.97
15	g0pC_taske.txt	orig_taske.txt	0.56	0.18	0.15	0.89
16	g0pD_taska.txt	orig_taska.txt	0.85	0.39	0.19	1.00
17	g0pD_taskb.txt	orig_taskb.txt	0.56	0.08	0.02	0.76
18	g0pD_taskc.txt	orig_taskc.txt	0.34	0.22	0.20	0.58
19	g0pD_taskd.txt	orig_taskd.txt	0.05	0.00	0.00	0.00
20	g0pD_taske.txt	orig_taske.txt	0.05	0.00	0.00	0.00
21	g0pE_taska.txt	orig_taska.txt	0.56	0.90	0.81	0.99
22	g0pE_taskb.txt	orig_taskb.txt	0.34	0.10	0.05	0.66
23	g0pE_taskc.txt	orig_taskc.txt	0.05	0.00	0.00	0.00
24	g0pE_taskd.txt	orig_taskd.txt	0.05	0.00	0.00	0.00
25	g0pE_taske.txt	orig_taske.txt	0.85	0.18	0.13	1.00
26	g1pA_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
27	g1pA_taskb.txt	orig_taskb.txt	0.34	0.02	0.00	0.00
28	g1pA_taskc.txt	orig_taskc.txt	0.56	0.10	0.00	0.26
29	g1pA_taskd.txt	orig_taskd.txt	0.85	0.18	0.12	0.34
30	g1pA_taske.txt	orig_taske.txt	0.05	0.01	0.00	0.00
31	g1pB_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
32	g1pB_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
33	g1pB_taskc.txt	orig_taskc.txt	0.34	0.14	0.03	0.32
34	g1pB_taskd.txt	orig_taskd.txt	0.56	0.09	0.03	0.35
35	g1pB_taske.txt	orig_taske.txt	0.85	0.22	0.16	0.50
36	g1pD_taska.txt	orig_taska.txt	0.56	0.09	0.05	0.34
37	g1pD_taskb.txt	orig_taskb.txt	0.85	0.11	0.10	0.88
38	g1pD_taskc.txt	orig_taskc.txt	0.05	0.00	0.00	0.00
39	g1pD_taskd.txt	orig_taskd.txt	0.05	0.02	0.06	0.00
40	g1pD_taske.txt	orig_taske.txt	0.34	0.02	0.02	0.00
41	g2pA_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
42	g2pA_taskb.txt	orig_taskb.txt	0.34	0.07	0.03	0.00
43	g2pA_taskc.txt	orig_taskc.txt	0.56	0.41	0.47	0.78
44	g2pA_taskd.txt	orig_taskd.txt	0.85	0.22	0.25	0.31
45	g2pA_taske.txt	orig_taske.txt	0.05	0.00	0.00	0.00
46	g2pB_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
47	g2pB_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
48	g2pB_taskc.txt	orig_taskc.txt	0.34	0.00	0.07	0.00
49	g2pB_taskd.txt	orig_taskd.txt	0.56	0.57	0.58	0.93

50	g2pB_taske.txt	orig_taske.txt	0.85	0.50	0.38	1.00
51	g2pC_taska.txt	orig_taska.txt	0.85	0.34	0.45	0.66
52	g2pC_taskb.txt	orig_taskb.txt	0.05	0.01	0.00	0.00
53	g2pC_taskc.txt	orig_taskc.txt	0.05	0.02	0.00	0.00
54	g2pC_taskd.txt	orig_taskd.txt	0.34	0.15	0.16	0.56
55	g2pC_taske.txt	orig_taske.txt	0.56	0.04	0.05	0.00
56	g2pE_taska.txt	orig_taska.txt	0.34	0.31	0.26	0.30
57	g2pE_taskb.txt	orig_taskb.txt	0.56	0.13	0.02	0.62
58	g2pE_taskc.txt	orig_taskc.txt	0.85	0.00	0.00	0.78
59	g2pE_taskd.txt	orig_taskd.txt	0.05	0.00	0.00	0.00
60	g2pE_taske.txt	orig_taske.txt	0.05	0.01	0.00	0.00
61	g3pA_taska.txt	orig_taska.txt	0.05	0.01	0.03	0.00
62	g3pA_taskb.txt	orig_taskb.txt	0.34	0.06	0.02	0.00
63	g3pA_taskc.txt	orig_taskc.txt	0.56	0.27	0.22	0.55
64	g3pA_taskd.txt	orig_taskd.txt	0.85	0.94	0.62	1.00
65	g3pA_taske.txt	orig_taske.txt	0.05	0.00	0.00	0.00
66	g3pB_taska.txt	orig_taska.txt	0.05	0.00	0.00	0.00
67	g3pB_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
68	g3pB_taskc.txt	orig_taskc.txt	0.34	0.07	0.03	0.00
69	g3pB_taskd.txt	orig_taskd.txt	0.56	0.08	0.00	0.30
70	g3pB_taske.txt	orig_taske.txt	0.85	0.24	0.19	1.00
71	g3pC_taska.txt	orig_taska.txt	0.85	0.38	0.14	0.99
72	g3pC_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
73	g3pC_taskc.txt	orig_taskc.txt	0.05	0.00	0.00	0.00
74	g3pC_taskd.txt	orig_taskd.txt	0.34	0.11	0.00	0.52
75	g3pC_taske.txt	orig_taske.txt	0.56	0.09	0.00	0.78
76	g4pB_taska.txt	orig_taska.txt	0.05	0.01	0.00	0.00
77	g4pB_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
78	g4pB_taskc.txt	orig_taskc.txt	0.34	0.27	0.21	0.61
79	g4pB_taskd.txt	orig_taskd.txt	0.56	0.28	0.17	0.82
80	g4pB_taske.txt	orig_taske.txt	0.85	0.55	0.41	0.93
81	g4pC_taska.txt	orig_taska.txt	0.85	0.90	0.77	0.89
82	g4pC_taskb.txt	orig_taskb.txt	0.05	0.00	0.00	0.00
83	g4pC_taskc.txt	orig_taskc.txt	0.05	0.00	0.00	0.00
84	g4pC_taskd.txt	orig_taskd.txt	0.34	0.80	0.85	0.97
85	g4pC_taske.txt	orig_taske.txt	0.56	0.36	0.40	0.91
86	g4pD_taska.txt	orig_taska.txt	0.56	0.09	0.10	0.28
87	g4pD_taskb.txt	orig_taskb.txt	0.85	0.00	0.00	0.93
88	g4pD_taskc.txt	orig_taskc.txt	0.05	0.01	0.00	0.00
89	g4pD_taskd.txt	orig_taskd.txt	0.05	0.00	0.00	0.00
90	g4pD_taske.txt	orig_taske.txt	0.34	0.15	0.08	0.51
91	g4pE_taska.txt	orig_taska.txt	0.34	0.01	0.00	0.00
92	g4pE_taskb.txt	orig_taskb.txt	0.56	0.35	0.35	0.93
93	g4pE_taskc.txt	orig_taskc.txt	0.85	0.16	0.26	0.32
94	g4pE_taskd.txt	orig_taskd.txt	0.05	0.00	0.00	0.00
95	g4pE_taske.txt	orig_taske.txt	0.05	0.02	0.04	0.00

APPENDIX B: The Clough-Stevenson's classification of the level of plagiarism (Plg.) in documents

Documents	Plg.	Documents	Plg.	Documents	Plg.	Documents	Plg.
g0pA_taska.txt	non	g0pE_taske.txt	cut	g2pB_taskd.txt	light	g3pC_taskc.txt	non
g0pA_taskb.txt	cut	g1pA_taska.txt	non	g2pB_taske.txt	cut	g3pC_taskd.txt	heavy
g0pA_taskc.txt	light	g1pA_taskb.txt	heavy	g2pC_taska.txt	cut	g3pC_taske.txt	light
g0pA_taskd.txt	heavy	g1pA_taskc.txt	light	g2pC_taskb.txt	non	g4pB_taska.txt	non
g0pA_taske.txt	non	g1pA_taskd.txt	cut	g2pC_taskc.txt	non	g4pB_taskb.txt	non
g0pB_taska.txt	non	g1pA_taske.txt	non	g2pC_taskd.txt	heavy	g4pB_taskc.txt	heavy
g0pB_taskb.txt	non	g1pB_taska.txt	non	g2pC_taske.txt	light	g4pB_taskd.txt	light
g0pB_taskc.txt	cut	g1pB_taskb.txt	non	g2pE_taska.txt	heavy	g4pB_taske.txt	cut
g0pB_taskd.txt	light	g1pB_taskc.txt	heavy	g2pE_taskb.txt	light	g4pC_taska.txt	cut
g0pB_taske.txt	heavy	g1pB_taskd.txt	light	g2pE_taskc.txt	cut	g4pC_taskb.txt	non

g0pC_taska.txt	heavy	g1pB_taske.txt	cut	g2pE_taskd.txt	non	g4pC_taskc.txt	non
g0pC_taskb.txt	non	g1pD_taska.txt	light	g2pE_taske.txt	non	g4pC_taskd.txt	heavy
g0pC_taskc.txt	non	g1pD_taskb.txt	cut	g3pA_taska.txt	non	g4pC_taske.txt	light
g0pC_taskd.txt	cut	g1pD_taskc.txt	non	g3pA_taskb.txt	heavy	g4pD_taska.txt	light
g0pC_taske.txt	light	g1pD_taskd.txt	non	g3pA_taskc.txt	light	g4pD_taskb.txt	cut
g0pD_taska.txt	cut	g1pD_taske.txt	heavy	g3pA_taskd.txt	cut	g4pD_taskc.txt	non
g0pD_taskb.txt	light	g2pA_taska.txt	non	g3pA_taske.txt	non	g4pD_taskd.txt	non
g0pD_taskc.txt	heavy	g2pA_taskb.txt	heavy	g3pB_taska.txt	non	g4pD_taske.txt	heavy
g0pD_taskd.txt	non	g2pA_taskc.txt	light	g3pB_taskb.txt	non	g4pE_taska.txt	heavy
g0pD_taske.txt	non	g2pA_taskd.txt	cut	g3pB_taskc.txt	heavy	g4pE_taskb.txt	light
g0pE_taska.txt	light	g2pA_taske.txt	non	g3pB_taskd.txt	light	g4pE_taskc.txt	cut
g0pE_taskb.txt	heavy	g2pB_taska.txt	non	g3pB_taske.txt	cut	g4pE_taskd.txt	non
g0pE_taskc.txt	non	g2pB_taskb.txt	non	g3pC_taska.txt	cut	g4pE_taske.txt	non
g0pE_taskd.txt	non	g2pB_taskc.txt	heavy	g3pC_taskb.txt	non		