Using Natural Language Processing for Automatic Detection of Plagiarism

Miranda Chong **Lucia Specia** **Ruslan Mitkov**

Research Group in Computational Linguistics
University of Wolverhampton, UK
{Miranda.Chong | L.Specia | R.Mitkov} @wlv.ac.uk

Keywords: Natural Language Processing, Plagiarism Detection, External plagiarism, Plagiarism

ABSTRACT

Current plagiarism detection tools are mostly limited to comparisons of suspicious plagiarised texts and potential original texts at string level. In this study the aim is to improve the accuracy of plagiarism detection by incorporating Natural Language Processing (NLP) techniques into existing approaches. We propose a framework for external plagiarism detection in which a number of NLP techniques are applied to process a set of suspicious and original documents, not only to analyse strings but also the structure of the text, using resources to account for text relations. Initial results obtained with a corpus of plagiarised short paragraphs have showed that NLP techniques improve the accuracy of existing approaches.

1. INTRODUCTION

The ease of information sharing through the Internet has encouraged searching for literature online, which has resulted in many people, particularly in academic fields, to replicate other people's ideas or work without appropriate acknowledgement. While the prevention of such a problem is a very important direction to be followed especially for educational purposes, the detection of plagiarised cases is also necessary. Over the years many methodologies have been developed to perform automatic detection of plagiarism, including tools for natural language text detection such as Turnitin (iParadigms, 2010) and CopyCatch (CFL software, 2010), and tools for computer programming source code detection such as MOSS (Aiken, 1994).

The detection of plagiarism is not a new research area. Various approaches have been developed to deal with both external and intrinsic plagiarism on written texts (Lukashenko Graudina & Grundspenkis, 2007). External plagiarism detection consists in comparing suspicious plagiarised documents against potential original documents. Intrinsic plagiarism detection, on the other hand, consists in finding plagiarised passages within a document without access to potential original texts.

However, the methods used for plagiarism detection so far are mostly limited to a very superficial level, for example by comparing suspicious texts and original texts at the string level to check the amount of word overlapping across documents (Bull *et al.*, 2001; Badge & Scott, 2009). As a consequence, the accuracy of detection approaches is yet to reach a satisfactory level (Lyon Barrett & Malcolm, 2001). Although recent work (Ceska & Fox, 2009) has targeted pre-processing techniques to generalise documents by replacing words with their base forms, the techniques used are still very limited and no significant improvements were reported. Therefore plagiarism continues to be a growing challenge, affecting many areas - namely education, publishing and even business sectors.

Our aim is to investigate means to improve the accuracy of existing detection approaches by using Natural Language Processing (NLP) technologies. Better approaches to detection could also be used to establish a broader understanding of the importance of correct referencing and encourage the deployment of plagiarism prevention approaches. For this paper, we focus on the challenge of external plagiarism in monolingual text.

The rest of this paper is organised as follows: in Section 2 we present related work on plagiarism detection. In Section 3 we describe the methodology we chose and the experimental settings. In Section 4 we present the results from our experiments. In Section 5 we discuss the strengths and weaknesses we observed. Finally, we draw some conclusions in Section 6.

2. PREVIOUS WORK

In this section we describe the existing detection approaches and investigate related work on how NLP can be beneficial for plagiarism detection.

2.1 Existing approaches

Plagiarism detection approaches have several classifications. According to Lancaster (2003), approaches can be classified by their type of detection methodology, availability of the system, number of documents that the metrics can processes, and complexity of metrics. It has been noted that rather than more accurate multi-dimensional metrics with large structural complexity, pairwise metrics have been most widely applied with superficial complexity. This is due to the trade-off between processing resources and accuracy. The more complex the metrics are, the more processing power is required and it would often take tremendous time and effort even with the aid of powerful computers to perform detection tasks. This is not ideal for users equipped with personal computers.

Although JISC[1] has recommended Turnitin (JISC, 2010), the most successful commercial detection tool, as an educational tool for plagiarism prevention and identification, the user's feedback on the tool were not at all satisfactory, since Turnitin is not able to handle paraphrased texts effectively (Marsh, 2004; Williams, 2009).

The exact algorithms of many commercial tools are not known, whereas the general approaches for existing plagiarism detection researches are mainly non-NLP based. As described in Ceska (2009), these methods included:

1) Relative frequency models, which simply count the occurrences of matches and normalise them by total number of occurrences. These have been employed in early years (Shivakumar & Garcia-Molina, 1996).

2) Dotplot visualisation of matching sequences of words on charts, using dots to display density of overlaps (Clough, 2000).

3) Similarity measures, which calculate the amount of overlaps of subsequences and substrings (Clough, 2003).

4) Document fingerprinting using frequency-based strategies, for example number of occurrence of a word, or structural-based strategies, such as the distribution of a certain occurrence across documents (Hoad & Zobel, 2003).

5) Word pairs metric, which calculates the matching proportion of word sequences in document pairs (Lancaster & Culwin, 2004).

In recent years, a number of similarity metrics have been employed through Vector Space Models, a technique in which the documents are described using vectors where words/terms in the documents are weighted using different strategies, for example Term Frequency and/or Inverse Document Frequency (Manku, Jain & Sarma, 2007; Runeson, Alexandersson & Nyholm, 2007). Another similar technique is the use of Normalised Word Vectors (Dreher, 2007). Additional similarity metrics to measure the distance between the vectors of suspicious and original documents include the Jaccard and Cosine coefficients. Such metrics, along with the Longest Common Subsequence between two documents, were investigated in Lukashenko, Graudina & Grundspenkis (2007).

Advanced structural methods such as Multilevel Text Comparison (Zini *et al.*, 2006), Plagiarism Pattern Checker (Kang, Gelbukh & Han, 2006) and Statistical Language Models have also been used for plagiarism detection. Language Models provide a platform for statistical comparison and manipulation of sequences of tokens, which has been successfully implemented by Barrón-Cedeño & Rosso (2008).

2.2 NLP approaches

NLP involves the processing of human languages by machines. Many fields such as computer-assisted language learning (Chang & Chang, 2004) and extraction of biomedical information (Terol, Martinez-Barco & Palomar, 2006) have already experienced benefits from using NLP. However, it remains an under-explored area for plagiarism detection.

Previous work by Clough (2003), Androutsopoulos & Malakasiotis (2009), and Ceska (2009) pointed out it that would be desirable to apply NLP techniques for plagiarism and that this could yield better accuracies through the detection of paraphrased texts. However, no successful experiments have been performed to show that this is indeed the case.

[1] JISC - Joint Information Systems Committee, is the organisation supporting UK Higher Education.

Recently, Ceska (2009) and Ceska & Fox (2009) applied some pre-processing techniques to improve plagiarism detection accuracy. These included simple heuristics such as replacing numbers by a dummy symbol, removing punctuations, application of basic NLP techniques such as lemmatisation (Section 3.2), removal of irrelevant words and the incorporation of a thesaurus to generalise the words in the texts. While some of the heuristics had a positive impact on the accuracy of their plagiarism detection approach, the use of NLP techniques did not show significant improvement with respect to the basic approach. We believe that this is due to the limitations of both the NLP techniques used and their experimental settings, including the use of a small corpora and inaccurate disambiguation procedures for generalising words.

Runeson, Alexandersson & Nyholm (2007) used shallow NLP text pre-processing to detect duplicate reports, including tokenisation, stemming, stop-word removal (Section 3.3). Although the techniques used were simple, the authors concluded that it was feasible to use NLP approaches to support identification of duplicates.

3. METHODOLOGY

This section provides details about the corpus, text processing techniques and plagiarism detection algorithms used in our experiments.

3.1 Corpus of plagiarised short answers

We concentrate in the detection of plagiarism in written text, more specifically, English short passages. We target "external" plagiarism, where detection is based on the availability of both the suspicious plagiarised texts and the potential original source texts. We used the corpus developed by Clough & Stevenson (2009), which provides samples of plagiarised short passages associated with different levels of plagiarism. It consists of a total of 100 documents containing 5 Wikipedia articles as the *original texts* and 95 *suspicious plagiarised short passages*. The latter were written by students to answer 5 questions, each related to one original document. The answers were based on the original texts (except for non-plagiarised cases), with various degrees of text overlapping, according to the instructions given by the corpus creators. The 4 classes of suspicious documents are classified as follows:

1. Near copy: copy-and-paste from the original text;
2. Light revision: minor alteration of the original text by substituting words with synonyms and performing some grammatical changes;
3. Heavy revision: rewriting of the original by paraphrasing and restructuring;
4. Non-plagiarism: based on participants' own knowledge as the original texts were not given.

The short passages were between 200-300 words and 57 samples were marked as heavy revision (19), light revision (19) or near copy (19) levels whereas the remaining 38 cases were non-plagiarised.

There are other plagiarism detection corpora available such as the METER corpus (Gaizauskas *et al.*, 2001) and the corpus created for the 1st International Competition on Plagiarism Detection (3rd PAN Workshop, 2009). We chose to use the short passages corpus because the other corpora do not provide a detailed description about what levels of plagiarism are represented, thus they are not very suitable for our initial experiments.

3.2 Experimental setup

We propose applying both shallow NLP techniques and advanced NLP techniques during pre-processing stage for identifying plagiarised texts. The experimental framework consists of incorporating conventional techniques, such as comparing texts at the string level together with novel ways to analyse the lexical and syntactic structure of the texts.

Given the original and suspicious documents, we applied different techniques to compare their similarities. As a baseline mechanism we used Ferret (Lyon *et al.*, 2000), which performs trigram comparisons between original and suspicious document pairs, that is, computes the similarity level for such document pair based on the number of matching sequences of three words they have. Ferret had been successfully applied in a number of previous plagiarism detection approaches, such as (Bao, 2006; Bao *et al.*, 2007).

As we will describe in the following sections, besides trigram comparison, we have also computed Language Model similarity scores,

Longest Common Subsequence and Dependency Relations Matching measures. We then used similarity scores given by such techniques as indicators for a Machine Learning algorithm to learn models, to classify a given original-suspicious document pair according to the four levels of plagiarism mentioned in Section 3.1. We followed the evaluation framework suggested in the PAN workshop plagiarism detection competition.

3.3 Pre-processing and NLP techniques

The general pre-processing and NLP techniques used in our experiments are the following:

- Sentence segmentation (Seg)

Split text in the document into sentences and thereby allowing line-by-line processing in the subsequent tests.

- Tokenisation (Tok)

Determine token (words, punctuation symbols, etc.) boundaries in sentences.

- Lowercase (Low)

Substitute every uppercase letters with lowercase to generalise the matching.

- Stop-word removal (Stop)

Remove functional words (articles, pronouns prepositions, complementisers, and determiners) such as “the”, “of”, “a”, “and”.

- Punctuation removal (Pun)

Remove punctuation symbols.

- Part-of-Speech tagging (POS)

Assign grammatical tags to each word, such as “noun”, ”verb”, etc., for detecting cases where words have been replaced, but the style in terms of grammatical categories remains similar.

- Stemming (Stem)

Transform words into their stems in order to generalise the comparison analysis. For example, both “computer” and “computers” are normalised as “computer”, while “product”, “produce”, and “produced” are normalised as “produc”.

- Lemmatisation (Lem)

Transform words into their dictionary base forms in order to generalise the comparison analysis. For example, “produced” is normalised as “produce”.

- Number replacement (Num)

Replace numbers and figures with a dummy symbol in order to generalise the comparison analysis.

We also applied the following syntactic processing techniques:

- Dependency Parsing (Parse)

We use the Stanford parser (Klein & Manning, 2003) to produce the dependency syntactic analysis of each sentence. For example, for the sentence fragment "*a basic concept of Object-Oriented Programming*", the following relations are returned:

```
det(concept, a)
amod(concept, basic)
prep(concept, of)
nn(programming, object-oriented)
pobj(of, programming)
```

where relation identifiers refer to different dependency relations between pairs of words, for example, ‘det’ for *determiner*, ‘amod’ for *modifier*, etc.

- Chunking (Chunk)

We use shallow parsing to identify the constituents in a sentence, such as noun phrase, verb phrase, etc. For example, we generalize parse tree for the sentence fragment "*a basic concept of Object-Oriented Programming*" to keep only the identifiers of the constituents and their structure (through the parenthesis), resulting in the following information:

```
(NP (NP (PP (NP)))
```

where the first noun phrase (NP) constituent covers the complete fragment, consisting of (a) the second NP, which covers "*a basic concept*", and (b) a prepositional phrase (PP), which covers "*of Object-Oriented Programming*". This PP is composed of the last NP, "*Object-Oriented Programming* ".

One or more of these techniques were applied in combination to all original and suspicious texts in the corpus (see second column in Table 1). The pre-processed documents were then analysed by one of the following comparison methodologies (see third column in Table 1), which will be explained in Section 3.4:

a) Trigram Similarity
b) Language Model Probability

c) Longest Common Subsequence
d) Dependency Relations Matching

Table 1. Combinations of pre-processing / NLP techniques and comparison methodologies used

Dataset	*Pre-processing/NLP technique*	*Comparison Methodology*
0	Tok + Low	a
1	Seg + Tok +Low = Baseline	a, b, c
2	Baseline + POS	a
3	Baseline + Stop	a
4	Baseline + Lem	a
5	Baseline + Stop + Lem	a
6	Baseline + Stop + Stem	a
7	Baseline + Stop + Pun	a
8	Baseline + Stop + Pun +POS	a
9	Baseline + Stop + Pun + Num	a
10	Baseline + Stop + Pun + Num + POS	a
11	Chunk	a, b
12	Parse	d

3.4 Comparison methodologies

3.4.1 Trigram similarity measures (a)
Basic tools such as Ferret are based on the comparison of sets of *n* (such as *3*) contiguous words in the original and suspicious document pairs. According to Lancaster & Culwin (2003), standard pre-processing for plagiarism detection includes sentence segmentation, tokenisation and lowercasing. The application of these three techniques (*dataset 1*), along with a trigram similarity metric described below, formed one of the **baselines** for this study. According to experimental results reported by Clough & Stevenson (2009), the use of trigrams is the balance between efficiency and effectiveness; thus we have also adopted trigrams in our experiments.

Overlapping trigrams in sentences can be exemplified as follows:

Original sentence {"This is an example."}

Trigrams {"This", "is", "an"} {"is", "an", "example"} {"an", "example", "."}

In Ferret, the comparison of trigrams is performed using the Jaccard similarity coefficient (Manning & Schutze, 1999):

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

where S(A) and S(B) represent the sets of trigrams in the suspicious and original documents respectively. Their intersection (nominator) represents the set of matching trigrams in those documents, while their union (denominator) represents the set of all trigrams in those documents.

Datasets 2-11 were also processed using Ferret as comparison methodology.

Clough & Stevenson (2009) used a slightly different trigram similarity metric, the containment measure (Broder, 1997):

$$C(A,B) = \frac{|S(A) \cap S(B)|}{|S(A)|}$$

Similar to the Jaccard coefficient, the containment measure calculates the intersecting trigrams, but normalises by the trigrams in the suspicious document only. This measure is more suitable for document pairs with varied document lengths (Broder, 1997), which is the case in this corpus, since the original documents are always longer than the suspicious versions.

The containment measure was used in our experiments with *dataset 0* as another comparative **baseline**.

3.4.2 Language model probability measure (b)
A statistical model of *n*-grams is first computed for the original document, that is, the probabilities of each sequence of 1 to *n* words in the document are computed by counts in the corpus. For example, for a bigram language model, the probability of each bigram consisting of $\{w_{n-1}, w_n\}$ is computed by:

$$P(w_n \mid w_{n-1}) = \frac{count(w_{n-1}, w_n)}{count(w_{n-1})}$$

A language model score for a given suspicious document provides the likelihood of sequences of *n* words in that document, which can be described according to the model of the original document. For example, a bigram score of a suspicious document with *m* words can be computed as the product of the probabilities of all bigrams in that document, as given by the model:

$$P(w_1^m) \approx \prod_{k=1}^{m} P(w_k \mid w_{k-1})$$

In other words, it measures the level of similarity of document pairs by combining the probabilities of which their *n*-grams are the same.

We also computed a variant of the language model probability which normalises the score according to the length (in *m* words) of the suspicious document. It is called perplexity:

$$\frac{1}{m}\log_2 P(w_1^m)$$

Finally, we computed the *out-of-vocabulary rate*, which is the number of words in the suspicious document that have not been seen in the original document.

Using SRILM, a toolkit for language modelling (Stolcke, 2002), we calculated unigrams, bigrams and trigrams language model scores for the basic dataset (*dataset 1*). We also computed such scores for the chunked dataset (*dataset 11*) by using 4-grams and 5-grams probability distributions.

3.4.3 Longest common subsequence (c)
We have applied the Longest Common Subsequence (LCS) technique, which computes the longest sequence of words included in both original and suspicious documents.

We used LCS with *dataset 1*. LCS was applied on each document pair at the sentence level, that is, we computed the longest common subsequence for all pairs of sentences in both documents and checked: 1) the overall longest matching sequence in that document pair; 2) the sum of the longest matching sequence for all sentences normalised by total number of sentences in associated suspicious document; 3) the average length of matching sequences; and 4) the number of matching words in each sentence pair.

3.4.4 Dependency relations matching (d)
Parsed data (*dataset 12*) was used to calculate the relative frequency of matches in terms of dependency relations between two words. Dependency relations provide information describing the syntactic structure of sentences.

The pairs of related words, along with the actual relation in the suspicious document, were compared against those of the original document to check for overlaps between document pairs. The number of overlapping relations was normalised by the number of relations in the suspicious document.

3.5 Machine Learning algorithm

The similarity scores gathered from applying each of the comparison methodologies described in Section 3.4 were used to train a model to classify the documents according to the four plagiarism levels in our corpus. We used the Naïve Bayes classifier, a simple algorithm based on the Bayes' theorem to generate a probabilistic model of the data. Given a number of features $(f_1,\ldots,f_n)$ for a number of training examples pre-classified with their level of plagiarism *c*, the goal of a Naïve Bayes classifier is to learn a model to classify new cases:

$$classify(f_1,\ldots,f_n) = \underset{c}{\operatorname{argmax}}\, p(C=c)\prod_{i=1}^{n} p(F_i=f_i|C=c)$$

In our case these features for each document correspond to the scores resulting from the application of different comparison methodologies to each variant of the dataset, and the classes correspond to the plagiarism level of the document. The classifier then considers such features to choose the most probable classification (or class), which is the one that maximises the probability of such class given the features.

4. EXPERIMENTAL RESULTS

In this section we present the results obtained in our experiments using the machine learning algorithm and the comparison methodologies described in Section 3 with different datasets.

4.1 Feature sets performance comparison

After initial analysis based on the correlation between the type of suspicious document (*near-copy*, *light revision*, *heavy revision* and *non-plagiarism*) and a given type of pre-processing technique plus a comparison methodology, we chose the most appealing combinations for further experiments. We called each combination a "**feature**".

The features are therefore the outcome of the application of either the trigram containment measure, Ferret, LM, LCS or Dependency

Relation Matching to a given pre-processed dataset. In other words, each feature corresponds to a similarity score.

From Ferret we have chosen the baseline dataset (sentence segmentation, tokenisation and lowercase), baseline plus lemmatisation, baseline plus stop-word removal, punctuation removal and number replacement. From the LM models, we have chosen bigram perplexity and trigram perplexity. From all features computed using LCS, we have chosen the sum of the longest common subsequence for all sentences normalised by total number of sentences in associated suspicious document.

Table 2 provides the average similarity scores of the selected features grouped by document type. We observed that for some features the scores did not always decrease linearly from near copy to non-plagiarism as they should. This shows that none of this individual features is sufficient to distinguish the four levels of plagiarism.

Table 2. Average similarity score per type of document according to different features

	Near Copy	*Light Revision*	*Heavy Revision*	*Non-plagiarism*
Trigram Containment Metric	0.5736	0.2477	0.4682	0.0201
Ferret: Baseline	0.3569	0.1488	0.2554	0.0085
Ferret: Baseline + Lem	0.3581	0.1507	0.2574	0.0091
Ferret: Baseline + Stop + Pun + Num	0.3321	0.1142	0.2163	0.0033
LM-Perplexity Bigram	0.0363	0.0509	0.0363	0.0746
LM-Perplexity Trigram	0.0353	0.0501	0.0353	0.0742
LCS	0.3156	0.2883	0.3111	0.2472
Dependency Relations	0.6089	0.2956	0.5085	0.0374

We also computed, in Table 3, Pearson's correlation coefficient scores of the same features described in Table 2. Pearson coefficient computes the average linear dependence between two variables *X* and *Y* based on their mean and standard deviations. In this case, the similarity score and the level of plagiarism, for *n* original-suspicious document pairs:

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{X_i - \bar{X}}{s_X}\right)\left(\frac{Y_i - \bar{Y}}{s_Y}\right)$$

An interesting point to note is that the text pre-processing techniques of stop-word removal, punctuation removal and number replacement has slightly reduced the correlation score with respect to the baseline rather than improving it, whereas the Dependency Parse Relations Matching has achieved the best correlation score.

Table 3. Correlation coefficient scores

Feature tested	*Correlation*
Trigram Containment Measure: Baseline	0.658819672
Ferret: Baseline	0.569792503
Ferret: Baseline + Lem	0.570299807
Ferret: Baseline + Stop + Pun + Num	0.554225471
LM – Bigram Perplexity	0.596717341
LM – Trigram Perplexity	0.598556348
Longest Common Subsequence	0.255367851
Dependency Relations	0.673941000

4.2 Naïve Bayes results

In order to further verify how discriminative the comparison methodology scores and type of datasets are with respect to the type of suspicious document, we trained the Naïve Bayes algorithm using each of the features individually. Based on the results of such analysis, we selected the set of "best features" and used them to learn a model to classify a given suspicious document as belonging to one of the four plagiarism levels. We then evaluated the performance of the set best features resulting from this individual analysis.

We used the Naïve Bayes Classifier implemented in Weka, a machine learning toolkit (Hall et al., 2009). Weka provides a number of machine learning algorithms, along with techniques for feature selection, analysis of the results, etc. Based on the results of the classifier using 10-fold cross-validation, we selected the seven "best" features shown in Table 4.

Table 4. Selected features by performance

- Trigram Containment Measure: Baseline
- Ferret: Baseline + Lem
- Ferret: Baseline + Stop + Pun + Num
- LM - Bigram Perplexity
- LM - Trigram Perplexity
- Longest Common Subsequence
- Dependency Relations

The Naïve Bayes classifier using 10-fold cross-validation has shown very promising performance for the set of best features. Results in Table 5, Table 6, Table 7 and Table 8 refer to using the seven best features shown in Table 4. Table 5 presents the overall accuracy of the classifier with the best features, compared to the best baseline (*dataset 1* using Ferret), and to the set of all features. The accuracy is computed as the number of correctly classified test cases over all number of test cases. As we can see, the set with the best features has returned a considerably higher overall accuracy than the baseline and all features together.

Table 5. Overall accuracy of the Naive Bayes classifier with different feature sets

Naïve Bayes Classifier	*Accuracy*
Best features	70.53 %
Ferret: Baseline	66.32 %
All features	60.00 %

We also computed (Table 6, 7 & 8) Precision, Recall and F-Measure for each class (level of plagiarism), as used in the evaluation framework of the 3rd PAN workshop (2009). Precision is the number of documents correctly identified as belonging to their class, normalised by total number of documents both correctly and incorrectly identified as belonging to that class. Recall is the number of correctly identified documents in a class, normalised by total number of correctly identified documents and those that have not been identified as belonging to that class, but should have been. F-Measure is the harmonic mean of Precision and Recall.

Table 6. Precision, Recall and F-Measure for each type of document obtained with the Naïve Bayes classifier and the set of best features

Class	*Precision*	*Recall*	*F-Measure*
Non	0.881	0.974	0.925
Copy	0.667	0.526	0.588
Heavy	0.55	0.579	0.564
Light	0.5	0.474	0.486
Average	*0.696*	*0.705*	*0.698*

Table 7. Precision, Recall and F-Measure for each type of document obtained with the Naïve Bayes classifier and Ferret Baseline

Class	*Precision*	*Recall*	*F-Measure*
Non	0.884	1	0.938
Copy	0.615	0.421	0.5
Heavy	0.419	0.684	0.52
Light	0.5	0.211	0.296
Average	*0.66*	*0.663*	*0.639*

Table 8. Precision, Recall and F-Measure for each type of document obtained with the Naïve Bayes classifier and the set of all features

Class	*Precision*	*Recall*	*F-Measure*
Non	0.884	1	0.938
Copy	0.333	0.211	0.258
Heavy	0.44	0.579	0.5
Light	0.267	0.211	0.235
Average	*0.561*	*0.6*	*0.574*

We can see from the above tables that the set of best features has outperformed the baseline feature set and all features set in most categories except for minor differences in the non-plagiarised documents category.

Table 9 shows the confusion matrix for the classified documents. In general, we can see that the similarity scores and learning algorithm have distinguished very well the non-plagiarised documents from the remaining (only one case miss-classified as "heavy-revision"). On the other hand, the performance for distinguishing among the three levels of genuine plagiarised documents is yet not satisfactory. This can be explained by the fact that this type of distinction constitutes a harder problem. It also may show evidence that even deeper NLP techniques need to be used.

Table 9. Confusion matrix for classification using best feature set as measure

Expected / *Classified as*	*Non-plag (38)*	*Copy (19)*	*Heavy (19)*	*Light (19)*
Non-plag	**37**	0	1	0
Copy	2	**10**	2	5
Heavy	3	1	**11**	4
Light	0	4	6	**9**

5. DISCUSSION

The results of the experiments presented in this paper are considered satisfactory for classifying the documents in the corpus into two categories: plagiarised and non-plagiarised: 37 out of 38 non-plagiarised documents were correctly classified, and only a few plagiarised documents were classified as non-plagiarised (5 out of 57). Distinguishing among the three different levels of plagiarism has shown to be a much more complex task. However, since we are considering the need of human intervention to judge whether the documents identified as 'plagiarism' are indeed genuine cases and, if necessary, to identify the level of plagiarism, these initial results are already very useful.

Besides the inherent complexity of the task, the relatively low accuracy in distinguishing different levels of plagiarism may be due to some characteristics of the corpus. Particularly, not all participants seemed to have followed the instructions given to them for creating the short passages. For example, some cases annotated as "copy-and-paste" plagiarism actually contained some revised passages, and should therefore have been annotated as "light revision".

As Bao & Malcolm (2006) have pointed out, human intervention is essential to establish plagiarism cases. Our experiment has proven that this is indeed the case. Without final human judgement, no plagiarism detection engine can identify for certain what would be the genuine cases.

This paper has provided a promising framework for plagiarism detection. The main contributions are the use of new NLP techniques, combined via different comparison methodologies, and their integration using a machine learning algorithm. Some of the features tested can be seen as a framework for language independent detection, but our most successful implementation was based on parsing for the English Language. We believe it is possible to apply parser of various languages to achieve similar performance.

Our next goal would be to identify plagiarised texts even if they were paraphrased using different words and structure. Our subsequent step is to improve detection accuracy by incorporating thesaurus to determine lexical relations such as synonymy and textual entailment.

As suggested by Badge & Scott (2009), an accurate plagiarism detection system can be both a detective and preventive tool. It can be a pre-emptive educational tool to prevent inadvertent plagiarism. For instance, students could use the tool to check for potential plagiarised parts which may have been caused due to inappropriate references or lack of awareness.

6. CONCLUSIONS

In this paper our goal was to automatically identify plagiarised texts using NLP techniques.

The paper provides an insight of how NLP techniques are capable of improving current plagiarism detection methodologies, and also invokes further investigation on applying high-level NLP approaches to develop a better methodology.

In our experimental results, we showed that the NLP techniques produced significant improvement in the performance on top of basic detection models. Particularly, the *Dependency Relations* feature demonstrated very promising results to improve the overall detection performance.

Although it has been proven NLP techniques can improve the accuracy of detection tasks, there are other challenges remain, such as multilingual detection, synonymy generalisation (word sense disambiguation) and sentence structure generalisation, which we plan to address in future work.

The final conclusion is that even with more accurate detection methodologies, human intervention will always be required to judge plagiarised cases.

7. REFERENCES

Aiken, A. (1994). MOSS: A System for Detecting Software Plagiarism. Stanford University. [Accessed: 21/3/2010] Available at:
<http://theory.stanford.edu/~aiken/moss/>

Androutsopoulos, I. & Malakasiotis, P. (2009). A Survey of Paraphrasing and Textual Entailment Methods. Technical report, Athens University of Economics and Business, Greece.

Badge, J., & Scott, J. (2009). Dealing with plagiarism in the digital age, University of Leicester, (pp. 1-18).

Bao, J., & Malcolm, J. (2006). Text similarity in academic conference papers. In 2nd International Plagiarism Conference, Northumbria University Press, (pp. 19-21).

Bao, J., Lyon, C., Lane, P., Ji, W., Malcolm, J. (2007). Comparing Different Methods to Detect Text Similarity. Technical Report, University of Hertfordshire.

Barrón-Cedeño, A., & Rosso, P. (2008). Towards the Exploitation of Statistical Language Models for Plagiarism Detection with Reference. In European Conference on Artificial Life, ECAL 2008 PAN Workshop, (pp. 15-19).

Broder, A. (1997). On the resemblance and containment of documents. In Proceedings of the Compression and Complexity of Sequences, IEEE, (pp. 21-29).

Bull, J., Collins, C., Coughlin, E., Sharp, D. (2001). Technical review of plagiarism detection software report. Luton: Computer Assisted Assessment Centre, (pp. 1-36).

Ceska, Z. (2009). Automatic Plagiarism Detection Based on Latent Semantic Analysis. Doctoral Thesis, University of West Bohemia, (pp. 1-128).

Ceska, Z., & Fox, C. (2009). The Influence of Text Pre-processing on Plagiarism Detection. In Recent Advance in Natural Language Processing, RANLP '09. (Poster presentation)

CFL Software Limited (2009). CopyCatch [Accessed: 21/3/2010] Available at: < http://cflsoftware.com/>

Chang, J. S., & Chang, Y. (2004). Computer Assisted Language Learning Based on Corpora and Natural Language Processing: The Experience of Project CANDLE. In Interactive Workshop on Language e-Learning, IWLeL 2004, (pp. 15-23).

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,

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

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

Dreher, H. (2007). Automatic Conceptual Analysis for Plagiarism Detection. In Issues in Informing Science and Information Technology, 4, (pp. 1-14).

Ferret (2009). University of Hertfordshire. [Accessed: 21/3/2010] Available at: <http://homepages.feis.herts.ac.uk/~pdgroup/>

Gaizauskas, R., Wilks, Y., Foster, J., Clough, P., Piao, S., Arundel. J. (2001). METER – Measuring Text Reuse. [Accessed: 21/3/2010] Available at: <http://www.dcs.shef.ac.uk/nlp/meter/>

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. (2009). The WEKA Data Mining Software: An Update. In ACM Special Interest Group on Knowledge Discovery and Data Mining, SIGKDD Explorations, (11)1. (pp.10-18).

Hoad, T., & Zobel, J. (2003). Methods for identifying versioned and plagiarized documents. In Journal of the American Society for Information, 5(3). (pp. 203-215).

iParadigms (2010). Turnitin [Accessed: 21/3/2010] Available at: <http://turnitin.com/>

JISC - Joint Information Systems Committee (2010). Preventing and Detecting Plagiarism. [Accessed: 21/3/2010] Available at: <http://www.jisc-collections.ac.uk/about_collections/publisher_informat inf/coll_jiscfactfile/plagiarism.aspx?>

Kang, N., Gelbukh, A., & Han, S. (2006). Ppchecker: Plagiarism pattern checker in document copy detection. In Text, Speech and Dialogue. Springer, (pp. 661–667).

Klein, D. & Manning, C.D. (2003). Fast Exact Inference with a Factored Model for Natural Language Parsing. In Advances in Neural Information Processing Systems 15 (NIPS 2002), Cambridge, MA: MIT Press, (pp. 3-10).

Lancaster, T., & Culwin, F. (2003). Classifications of plagiarism detection engines. Unpublished internal 2nd draft, Available from South Bank University, London, UK. (pp. 1-16).

Lancaster, T., & Culwin, F. (2004). A Visual Argument for Plagiarism Detection using Word Pairs. In Plagiarism: Prevention, Practice and Policies 2004 Conference, (Vol. April), (pp. 1-14).

Lukashenko, R., Graudina, V., & Grundspenkis, J. (2007). Computer-Based Plagiarism Detection Methods and Tools: An Overview. In ACM International Conference on Computer Systems and Technologies , 54(3), (pp. 203–215).

Lyon, C., Barrett, R., & Malcolm, J. (2001). Experiments in Electronic Plagiarism Detection. [Accessed: 21/3/2010] Available at: <homepages.feis.herts.ac.uk. >

Manku, G. S., Jain, A., & Sarma, A. D. (2007). Detecting near-duplicates for web crawling. International World Wide Web Conference, WWW'07, (pp. 141-149).

Manning, C., & Schutze, H. (1999). Foundations of statistical natural language processing. Reading: MIT Press.

Marsh, B. (2004). Turnitin.com and the scriptural enterprise of plagiarism detection. In Computers and Composition, 21(4), (pp. 427-438).

Maurer, H., & Zaka, B. (2007). Plagiarism– A problem and how to fight it. In Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications, 2007 ED-MEDIA, (pp. 4451-4458).

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. In AAAI-98 Workshop on Learning for Text Categorization (Vol. 752), (pp. 41-48).

Mitchell, T. (1999). Machine Learning and Data Mining Over the past. In Communications of the ACM, 42(11), (pp. 30-36).

Runeson, P., Alexandersson, M., & Nyholm, O. (2007). Detection of Duplicate Defect Reports Using Natural Language Processing. In 29th International Conference on Software Engineering, ICSE'07, (pp. 499-510).

Shivakumar, N., & Garcia-Molina, H. (1996). Building a scalable and accurate copy detection mechanism. In Proceedings of ACM International Conference on Digital Libraries, (pp. 160-168).

Stein, B., Rosso, P., Stamatatos, E., Koppel, M. & Agirre, E. (eds.) (2009). Uncovering Plagiarism, Authorship and Social Software Misuse - 3rd PAN Workshop. In 25th Annual Conference of the Spanish Society for Natural Language Processing, SEPLN 2009.

Stolcke, A. (2002). SRILM- An extensible language modelling toolkit. In Proceedings of the Seventh International Conference on Spoken Language Processing, 3, (pp. 901-904).

Terol, R., Martinez-Barco, P. & Palomar, M. (2006). Applying NLP Techniques and Biomedical Resources to Medical Questions in QA Performance. In Lecture Notes in Computer Science (Vol. 4293), Springer, (pp. 996-1006).

Williams, J. (2009). The plagiarism problem: Are students entirely to blame? In Proceedings of the 19th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education, ASCILITE, (pp. 934-937).

Zini, M., Fabbri, M., Moneglia, M., & Panunzi, A. (2006). Plagiarism Detection through Multilevel Text Comparison. In IEEE Second International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution, AXMEDIS'06, (pp. 181-185).