Radboud Repository

Radboud University Nijmegen

PDF hosted at the Radboud Repository of the Radboud University Nijmegen

The following full text is a publisher's version.

For additional information about this publication click this link.
http://hdl.handle.net/2066/60920

Please be advised that this information was generated on 2018-07-17 and may be subject to change.

Detection of Plagiarism in Student Essays

Hans van Halteren

University of Nijmegen, Language and Speech

Abstract

This paper presents two methods for automatic detection of plagiarism in student essays, using Dutch text corpora to show their effectiveness. The first method is based on measuring the overlap in word trigrams between two essays, excluding all trigrams from the assignment text. This method proves efficient and robust, but relies on the availability of the plagiarized source. The second method compares an essay's linguistic profile to the profile of the claimed author's previous written texts. This approach proves promising, but as yet not completely able to detect all cases of plagiarism.

1 Introduction

In a quite popular teaching method, students are not supposed just to read text on the subject material, but rather to work with it. This brings them in closer contact with the subject and makes them think, thus enhancing both their knowledge and their insight. In the humanities, the method often takes the form of the students writing essays, e.g. weekly. Any teacher working with this method quickly finds that his workload increases with regard to 'traditional' teaching methods, as a) the right exercises have to be developed (just once) and b) all the essays have to be judged and feedback has to be given (every year and for every student). And there is yet another drawback. The method only works if the students actually do the exercises, write the essays. However, the students also notice that their workload is high. Some of them may decide that 'reusing' an existing essay is much more efficient than writing their own. If the number of students in a course is high enough, the chance of being caught is perceived to be low and the student can get the credits for the course without the work and hence without the experience.

If we want to make this teaching method work, therefore, we need to develop techniques to automatically detect that an essay was not in fact written by the student submitting it. There are various computer programs, some of them available freely on the internet, which can help detecting plagiarism. In this paper, I will not discuss these in any detail.[1]

Instead, I present two of techniques I developed myself. The first, discussed in Section 2, assumes that the source of the plagiarism can be found in the set of submitted essays as well, so that it is only necessary to detect suspiciously high levels of overlap. The second, discussed in Section 3, does not refer to the source at all, but rather checks the new essay against the writing patterns of the submitting student.

[1] A survey of systems in 2003, along with various pointers to other internet resources, can be found at `http://www.fdewb.unimaas.nl/eleum/plagiarism/plagiarism.htm`.

2 Overlap Detection

In many courses, the exercises have been developed especially for the course. They are therefore very specific and it is unlikely that anyone outside the course would have produced an essay like the desired ones. This means that the search for plagiarized essays can be limited to the essays submitted for that course, in the same or in previous years. In other words, there is a finite set of texts and we have to spot pairwise overlaps which are large enough to cause suspicion of plagiarism.

2.1 Existing techniques

There are several computer systems which provide the needed overlap measurement. An example is Wcopyfind, which can be found at `http://www.plagiarism.phys.virginia.edu`. It is a well-wrought system, able to read various document formats and allowing adjustment of its parameters. Basically, it looks for phrases (i.e. sequences of words rather than syntactic constructs) which occur in both documents being compared. The user can specify what the minimum and maximum sizes are for phrases to be checked, and at which level of overlap the document pair should be flagged as suspicious.

The system also allows for attempts by the students to hide the fact that the text is copied. At the simplest level, non-textual differences, such as those in punctuation, upper/lower case or numbers can be ignored. Next, some students try to confuse automatic checks by inserting 'spelling errors' in the text.[2] Wcopyfind is able to perform fuzzy matches so that such adjusted words can still be matched. The user can set the desired level of fuzziness. Finally, the system even allows for the use of a 'word map' to cater for cases where words have been replaced by synonyms.

It is clear that this system provides something for everyone. An experienced user can choose the parameter settings that provide the most effective and/or comfortable output. This strength, however, is also a drawback. An inexperienced user is at a loss with so many choices. I myself prefer a system where a minimum of choice, preferably none, can do the job. What this has led to will be the subject of the rest of the following sections. Before that, I wish to point out another disadvantage of Wcopyfind and most, if not all, other existing systems: they compare two texts and only those two texts. In my experience, students tend to copy parts of the instructions into their essay, which of course leads to an immediate overlap and confuses the detection of student text overlap. I will discuss this in more detail in Section 2.3.

2.2 Trigram overlap

Some years back I taught a course to some 150 students. Part of the score for that course was determined by a case study by the students. After spotting a largely

[2] Actually, the simulated errors are usually so different from natural spelling errors that they themselves can be used as a markers for plagiarism.

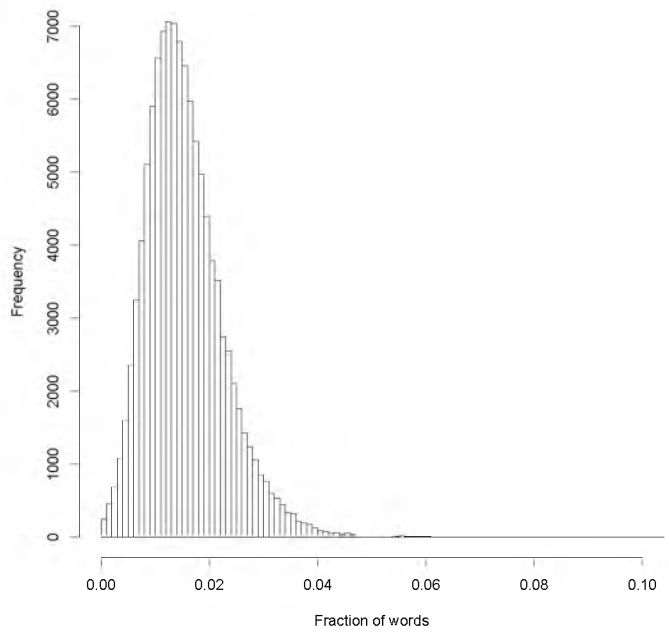

Figure 1: Trigram overlap distribution, with instruction text correction. Only the low end of the distribution is shown, but this covers practically all cases.

copied case study, I decided that an automatic check was needed. Because of the shortcomings of existing systems, of which there were fewer at that time, I built my own system. I followed a traditional approach in language technology and modeled word sequence overlap with word trigram overlap. After conversion to ASCII and tokenization, the set of all trigrams present in each text is taken. The number of times each trigram occurs and the position in the text are simply ignored. Then, all trigrams occurring in the assignment instructions are removed (for the importance of this, see below). Finally, the overlap between two texts is measured as the percentage of trigrams which occur in both texts. Apart from simplifying calculations, the approximation by trigrams automatically corrects for local rephrasings, spelling error introduction and reordering of parts of the text. In my opinion, making enough changes to fool this system is more work than just doing the assignment.

Figure 1 shows the distribution of overlap scores in pairs taken from the 474 texts collected over three years of teaching the mentioned course. If we assume that the normal situation is that students write their own essay, the distribution

indicates that we can expect an overlap between two independent texts of zero to seven or eight percent. This number will vary with each assignment, but can be easily determined from the distribution, probably even automatically. In this case, the number is chosen on the basis of a visual inspection of the histogram, and the fact that there is a clear gap with no text pairs between eight and ten percent.

So any two texts with an overlap higher than, say, ten percent can be flagged as suspicious. This leads to 125 text pairs to be examined. 72 of these are simply pairs of different versions of case studies by the same student, either because students had to improve their case study before it was deemed adequate or because students reused their own case study after failing the final exam in the previous year. Of the remaining 53, 15 more can be put aside because they form repeated reports of suspicious overlaps between one case study and different versions of the same second case study. This leaves 38 overlapping pairs where suspicion is really needed. At the high score range, say 70% and higher, we find outright copies. In the most blatant case, only the name of the student was changed and even the formatting was still exactly the same. Most often, some attempts were made to hide the plagiarism, e.g. by changing some numbers, introducing some 'spelling errors' and such.[3] At the lower ranges of overlap, down to about 11%, we find milder cases of copying, e.g. a single section. This was often the first part, where the planning was done, after which the two students went off and finished the work by themselves. One interesting case in this range needs special mention. A very inventive student made a collage of 20% from the instructions and 30% and 25% respectively from two other case studies. Because a low threshold is usable with the trigram overlap method, even this trick does not work.

2.3 Excluding the exercise instructions

As stated above, some students copy stretches of text from the instructions in order to structure their essay, and these copied stretches can confuse the overlap measurements. Figure 2 shows how much of each essay consists of instruction text. In most of the essays, less than ten percent of the text consists of parts from the instruction, but there are texts with percentages up into the twenties. One even shows a staggering 47% instruction text. This was a rather minimalist and overly optimistic attempt, which was returned to the student for a second try. Still, if no special measures had been taken, it would have seriously confused the plagiarism detection.

The effects on an overlap calculation without special measures becomes clear in Figure 3. This is a repeat of Figure 1, but this time without removing the instruction text trigrams from the set first (and note the different range for the horizontal axis). In this figure, it is much harder to determine a threshold for flagging suspicious texts. With a strict threshold of 15%, we would need to check 588 text pairs rather than the 125 mentioned above. And still we would have missed two cases of copied text. At 20%, 215 text pairs need to be checked and 7 cases of copying are

[3] After the checking program was in place, I warned the students that I would check, but still an alarming number of students reused their colleagues' work.

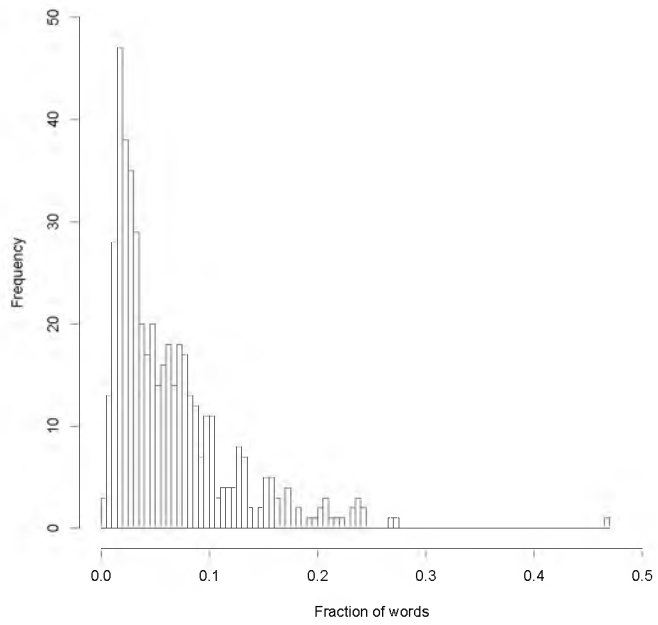

Figure 2: Amount of text copies from the instructions

missed. Only a threshold of 25% would have brought us back down to checking 113 text pairs, but then we would have missed 10 of the 38 cases of copying, a false accept rate of 26%.

3 Author Verification

As we have seen in the previous section, detection of plagiarism on the basis of overlap measurement is possible, but only if we are in the possession of the source text. With very specifi c assignments, this will probably be enough. As soon as the assignments are more variable, however, we open up the opportunity of copying texts from the internet or beyond. We could search on the internet for texts similar to the submitted essays, but this is both computationally costly and not guaranteed to be suffi cient. In other words, we have no source text to compare to. Fortunately, there is an alternative. Plagiarized texts will not only tend to be like the source text, but also unlike the normal writings of the student in question.[4] In this section,

[4] As an anonymous teacher (not of the English department) stated: "Sometimes I spot plagiarism because the English in the essays is much too good for our students."

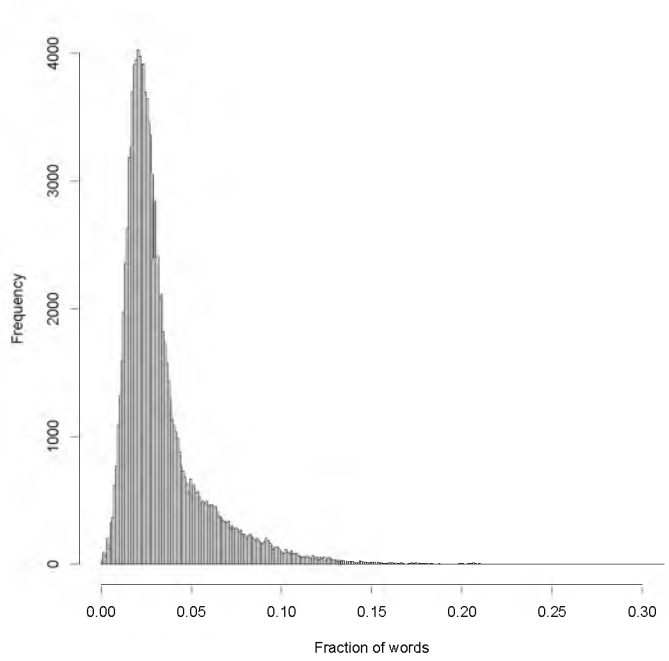

Figure 3: Trigram overlap distribution, without instruction text correction

I attempt to verify the authorship of an essay on the basis of a linguistic profi le of the author's other texts.

3.1 Linguistic profiling

In linguistic profi ling, the occurrences in a text are counted of a large number of linguistic features, either individual items or combinations of items. These counts are then normalized for text length and it is determined how much (i.e. how many standard deviations) they differ from the mean observed in a profi le reference corpus. For the authorship verifi cation task, the profi le reference corpus consists of the collection of all previously available essays by the students and all new essays. For each text, the deviation scores are combined into a text profi le vector. The featurewise average of the profi le vectors of all known texts by an author (the positive examples) is used as the author profi le. The system then determines the distance between the text profi les for new texts and the various author profi les. Rather than using the normal distance measure, I opted for a non-symmetric measure which is a weighted combination of two factors: a) the difference between sample score

and author score for each feature and b) the sample score by itself. This makes it possible to assign more importance to features whose count deviates signifi cantly from the norm. The following distance formula is used:

$$\Delta_T = (\Sigma|T_i - A_i|^D |T_i|^S)^{\frac{1}{(D+S)}}$$

In this formula, T_i and A_i are the values for the i^{th} feature for the text sample profi le and the author profi le respectively, and D and S are the weighting factors that can be used to assign more or less importance to the two factors described. The distance measure is then transformed into a similarity score by the formula

$$Score_T = (\Sigma|T_i|^{(D+S)})^{\frac{1}{(D+S)}} - \Delta_T$$

In this way, the score will grow with the similarity between text sample profi le and author profi le. Also, the fi rst component serves as a correction factor for the length of the text sample profi le vector.

The order of magnitude of the score values varies with the setting of D and S. Furthermore, the values can fluctuate signifi cantly with the sample collection. To bring the values into a range which is suitable for subsequent calculations, I also determine scores for all texts known not to be by an author (the negative examples). I then express all scores as the number of standard deviations they differ from the mean of the scores of the negative examples.

3.2 Profiled features

The fi rst type of features used for profi ling are simply lexical features. Suffi ciently frequent tokens, i.e. those that were observed at least a certain amount of times (in this case 5) in some language reference corpus (in this case the Eindhoven corpus; Uit den Boogaart 1975) are used as features by themselves. For less frequent tokens I determine a token pattern consisting of the sequence of character types, e.g., the token *Uefa-cup* is represented by the pattern *#L#6+/CL-L*, where the fi rst *L* indicates low frequency, *6+* the size bracket, and the sequence *CL-L* a capital letter followed by one or more lower case letters followed by a hyphen and again one or more lower case letters. For lower case words, the fi nal three letters of the word are included too, e.g. *waarmaken* leads to *#L#6+/L/ken*.[5]

In addition to the form of the token, I also use the potential syntactic usage of the token as a feature. I apply the fi rst few modules of a morphosyntactic tagger (in this case Wotan-Lite; van Halteren et al. 2001) to the text, which determine which word class tags could apply to each token. For known words, the tags are taken from a lexicon; for unknown words, they are estimated on the basis of the word patterns described above. The three (if present) most likely tags are combined into a feature, e.g. *niet* leads to *#H#Adv(stell,onverv)-N(ev,neut)* and *waarmaken* to *#L#V(inf)-N(mv,neut)-V(verldw, onverv)*. Note that the most likely

[5] These patterns have been originally designed for English and Dutch and will probably have to be extended when other languages are being handled.

tags are determined on the basis of the token itself and that the context is not consulted. The modules of the tagger which do context dependent disambiguation are not applied.

Op top of the individual token and tag features I use all possible bi- and trigrams which can be built with them, e.g. the token combination *kon niet waarmaken* leads to features such as *wcw=#H#kon#H#Adv(stell,onverv)-N(ev,neut)#L#6+/L/ken*. The lexical features currently also include features for utterance length. Each utterance leads to two such features, viz. the exact length (e.g. *len=15*) and the length bracket (e.g. *len=10-19*).

Since the number of features quickly grows too high for efficient processing, I filter the set of features by demanding that a feature occurs in a set minimum number of texts in the profile reference corpus (in this case two). A feature which is filtered out instead contributes to a rest category feature, e.g. the feature above would contribute to *wcw=OTHER*. For the current corpus, this filtering leads to a feature set of about 100K lexical features.

A second type of features represents the syntactic structure of the utterances in the texts. I used the Amazon parser to derive syntactic constituent analyses of each utterance (Coppen 2003). I did not use the full rewrites, but rather constituent N-grams. The N-grams used were:

- left hand side label, examining constituent occurrence
- left hand side label plus one label from the right hand side, examining dominance
- left hand side plus label two labels from the right hand side, in their actual order, examining dominance and linear precedence

For each label, two representations are used. The first is only the syntactic constituent label, the second is the constituent label plus the head word. This is done for each part of the N-grams independently, leading to 2, 4 and 8 features respectively for the three types of N-gram. Furthermore, each feature is used once by itself, once with an additional marking for the depth of the rewrite in the analysis tree, once with an additional marking for the length of the rewrite, and once with both these markings. This means another multiplication factor of four for a total of 8, 16 and 32 features respectively. After filtering for minimum number of observations, again at least an observation in two different texts, there are about 900K active syntactic features, nine times as many as for the lexical features.

3.3 Test Corpus

At the time of writing of this paper, no suitable corpus of student essays produced in an actual course was available. However, a good alternative is provided by the first component of the Dutch Authorship Benchmark Corpus (ABC-NL1; Baayen et al. 2002). It contains widely divergent written texts produced by first-year and fourth-year students of Dutch at the University of Nijmegen. Each of the eight

authors was asked to write nine texts of about a page and a half. In the end, it turned out that some authors were more productive than others, and that the text lengths varied from 628 to 1342 words. The authors did not know that the texts were to be used for authorship attribution studies, but instead assumed that their writing skill was measured. The topics for the nine texts were fi xed, so that each author produced three argumentative non-fi ction texts, on the television program Big Brother, the unifi cation of Europe and smoking, three descriptive non-fi ction texts, about soccer, the (then) upcoming new millennium and the most recent book they read, and three fi ction texts, namely a fairy tale about Little Red Riding Hood, a murder story at the university and a chivalry romance.

For the initial test of linguistic profi le for the purpose of authorship verifi cation, ABC-NL1 is actually better than 'real' course essays. After all, for the ABC-NL1 texts, we know who wrote what text. For 'real' data, our benchmark might itself be polluted by plagiarism. However, it is clear that, if the method works for the benchmark corpus, later tests with course essays will follow.

3.4 Results

The verifi cation quality at the best parameter settings found so far is shown in Figure 4.[6] The fi gure is a so-called Receiver operating Characteristic (ROC-curve). The horizontal axis represents the False Accept Rate (FAR) at various thresholds, the percentage of cases where somebody who has not written the test text is accepted as having written the text. The vertical axis shows the False Reject Rate (FRR), the percentage of cases in which the correct author is judged not to have written the text. The shape of the curve is as expected as, with increasing threshold settings, FAR will go down, while FRR goes up. The quality is visible in the curvature, the closer it reaches to the bottom left corner (0.0, 0.0), the better. The curves in Figure 4 are the best for lexical features, the best for syntactic features and the best combination.[7]

In some other applications, the overall quality is summed up in a single number, the Equal Error Rate (EER), i.e. the rate at where FAR and FRR are the same. In our curves, the EER appears to be slightly lower for the lexical features than for the combination, with the syntactic features far behind. This would imply that the syntactic features have no useful contribution at all. However, the EER may be misleading, since it does not take into account the consequences of the two types of errors. Given the application, plagiarism detection, there are in fact two interesting thresholds. The fi rst is the threshold below which it is clear that the text is not by the student in question. Any student producing a text scoring below this threshold can be accused of plagiarism without fear of false accusations. The quality of the system is higher if the percentage of plagiarizing texts below the threshold is higher. In terms of the ROC curve, this is the FAR at the point where FRR is zero. The second threshold is the one above which we are sure that the

[6] Because of computation time restraints, only part of the parameter space has as yet been explored.
[7] Interestingly, the parameters used for the best combination model are not those which are best for the single type models.

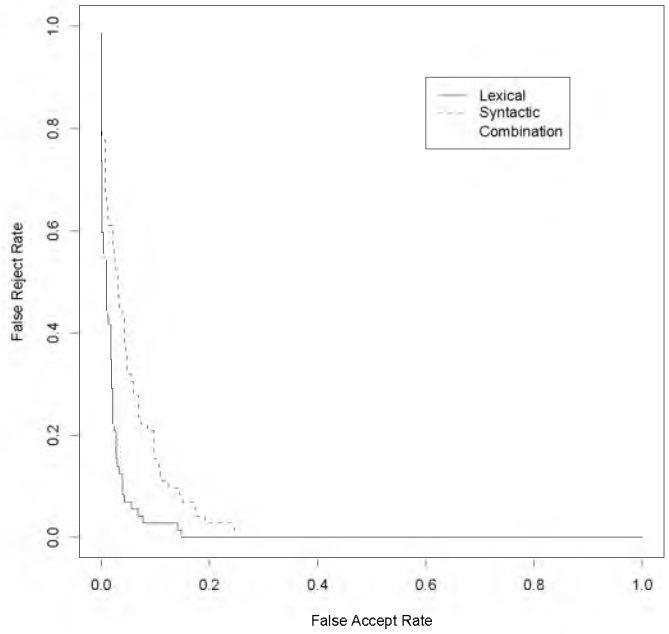

Figure 4: Receiver Operating Curves (ROC) for linguistic profiling with various types of features

student in question did write the essay. Any text scoring over this threshold can be safely accepted. The quality of the system is higher if the percentage of bona fide texts below the threshold is lower. In terms of the ROC curve, this is the FRR at the point where FAR is zero. The "Single threshold" partition of Table 1 shows these percentages for various systems, and demonstrates that the combination of syntactic and lexical features outperforms the lexical features after all.

It must be said that judging the quality of the models by the characteristics of the curves ignores the fact that in an actual application, an actual threshold will have to be selected. The results in Table 1 are based on the idea that a single threshold will have to serve for all runs of the system. As the range of scores for the measured texts rather varies with each run, this seriously hampers the system. It would be much better if a mechanism could be found which automatically chooses the threshold for each specific run. What might become possible then can be estimated by supposing an oracle threshold: for each run of the system, an oracle provides the optimal threshold. Basically, since the oracle threshold will be at the score for the text by the author, we are examining how many texts by other

Table 1: Remaining errors at optimal low and high thresholds

	FAR at FRR=0	FRR at FAR=0
Single threshold		
Lexical	15%	74%
Syntactic	25%	81%
Combination	8%	53%
Oracle threshold		
Lexical	0.8%	4.2%
Syntactic	1.4%	9.7%
Combination	0.2%	1.4%

authors score better than the text by the actual author. The "Oracle threshold" partition of Table 1 shows that in this situation the verifi cation quality can become very high indeed.

3.5 Comparison with Other Methods

The authorship verifi cation task described here is very similar to a task called authorship attribution, best known from humanities computing. There, the most likely author for a given text has to be chosen from a (usually small) set of potential authors who have been selected on the basis of extra-linguistic information (cf. e.g. Mosteller and Wallace 1984; Holmes 1998). Research into authorship attribution is becoming more popular and initiatives are underway to bring more structure to the fi eld. One of these initiatives was the so-called Ad-hoc Authorship Attribution Contest organized by Patrick Juola in 2004, the results of which he presented at the ALLC/ACH conference in Göteborg. The contest used several authorship attribution problems, most focusing on complete books from established authors from various languages and time periods, but also three problems using student essays.

The ABC-NL1 corpus was used for Problem M, but in a slightly different form, viz. in a lower-case-only format. Six topics, two from each general type, were used as training material and the remaining three topics, i.e. 24 texts, served as test samples. Linguistic profi ling appeared to be the best method for this problem, with 21 out of 24 samples correctly attributed. Several other methods were close behind, with 19 (Coburn; cf. Ceglowski et al. 2003), 17 (Keselj and Cercone; cf. Keselj et al. 2003) and 14 (Stamatatos) correct attributions respectively.

Problems A and B also used student essays. The training data consisted of three fi xed-topic 400–1200 word essays by each of thirteen students. Test samples for problem A consisted of one further similar essay from each student. Test samples for problem B consisted of a term paper from each student, which was longer (c. 1800 words) and had no fi xed topic.

Here, Keselj and Cercone's method performed best, with 11 out of 13 for prob-

lem A and 7/13 for problem B. Linguistic profi ling followed closely for problem A, with 9/13, but was far behind for problem B, with only 3/13 correct, just like Stamatatos' method with 9/13 and 2/13. Coburn's method, however, failed for both A and B, with 5/13 and 2/13. Since the texts in problems A and B fall in the same general category, argumentative essays, the question is why there is such a marked difference in recognition quality. A possible explanation is that the fact that the term paper is deemed especially important makes the students pay more attention to it, leading to better structuring and better language. If this really influences the recognizability of each student to such a degree, care has to be taken that enough training material of the correct type is present before decisions on plagiarism are made on the basis of author profi les. Still, there might well be a completely different explanation, and a careful examination of the training and test texts is therefore needed.

Another interesting observation is that the methods which did best on the scarce data student essay problems were outperformed on the problems involving complete books, most notably by Koppel and Schler's method (Koppel and Schler 2003; Koppel et al. 2003). A closer look at the various methods, and the particulars of the individual problems, will have to show an explanation for these differences.

4 Conclusion

The two methods described above support the detection of plagiarism very effectively. However, one should not think that the teacher has no further role in the process. The trigram overlap method detects overlap, not plagiarism. It would seem that accidental overlap is fi ltered out adequately, so that only 38 of the about 160K essay pairs need to be examined. However, there are still other possible reasons for the overlap than outright plagiarism. The teacher will have to check each essay pair with a suspiciously high overlap to determine whether or not there are grounds for accusations. In some cases, there has only been mild cooperation. In others, too much text has been copied and it has to be determined which of the students has to be punished. This may become clear from the rest of the essay, or from inept attempts at hiding the plagiarism, but often the students involved have to be confronted. In my experience they deny their actions only rarely and admit the copying, usually providing excuses at the same time. In the rare case of denial, the overlap method provides suffi cient proof of plagiarism that punishment can be justifi ed.

In the case where the source text is not available, matters are not so simple. The linguistic profi ling method is promising, but only provides a rough classifi cation. The teacher will have to choose a threshold for flagging suspicious essays. The choice lies somewhere between a) deciding that no false accusations can be made and accepting that 8 percent of the plagiarism attempts will get through and b) deciding that every perpetrator has to be caught and manually checking all texts in the doubted 8 percent of the plagiarism attempts plus 53 percent of the bona fi de texts. Furthermore, if linguistic profi ling, or our manual check, shows that it is

very unlikely that the student in question did indeed write the essay in question, there is still the burden of proof. We can, of course, start a search for potential sources, which can be more intensive because fewer essays need to be checked in this way. But if our search fails, and we cannot present the source text which was plagiarized, it will be very hard to convince non-experts that the student has to be punished. We can only hope that confrontation leads to a confession by the student.

What is needed now is first a clear stance on by the universities, not only stating what the penalties for plagiarism are, but also stating the level and type of proof needed, and obliging the students to submit all essays electronically so that they can be checked. It should also be made clear to the students that their work will be checked, but preferably not in which way. Once these prerequisites are in place, the two methods presented in this paper can provide the support the teachers need to then also catch the attempts which are still made.

References

Baayen, H., Halteren, H. van, Neijt, A., Tweedie, F. (2002). An Experiment in Authorship Attribution, in: *Proceedings JADT 2002*, pp. 69–75.

Uit den Boogaart, P.C (1975). *Woordfrequenties in geschreven en gesproken Nederlands*, Oosthoek, Scheltema & Holkema, Utrecht.

Ceglowski, M., Coburn, A. and Cuadrado, J. (2003). An Automated Management Tool for Unstructured Data, in: *Proc. 2003 IEEE/WIC International Conference on Web Intelligence*.

Halteren, H. van, Zavrel, J., and Daelemans, W. (2001). Improving accuracy in word class tagging through the combination of machine learning systems, *Computational Linguistics*, 27(2), pp 199–230.

Holmes, D. (1998). Authorship Attribution, *Literary and Linguistic Computing*, 13(3), pp 111–117.

Keselj, V., Peng, F., Cercone, N. and Thomas, C. (2003). N-gram-based Author Profiles for Authorship Attribution, in: *Proceedings PACLING'03*, pp. 255–264.

Koppel, M., Akiva, N. and Dagan, I. (2003). A Corpus-Independent Feature Set for Style Based Text Categorization, in: *Proc. IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis*.

Koppel, M. and Schler, J. (2003). Exploiting Stylistic Idiosyncrasies for Authorship Attribution, in: *Proc. IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis*.

Mosteller, F. and Wallace, D.L. (1984). *Applied Bayesian and Classical Inference in the Case of the Federalist Papers (2nd edition)*, Springer Verlag, New York.

Schelfhout, C. and Coppen, P.-A. (2004). *Interrupting Constructions in a Rejuvenated Amazon Grammar*, in: *Computational Linguistics in the Netherlands 2003*.