14th BILETA Conference: "CYBERSPACE 1999: Crime, Criminal Justice and the Internet".

Monday, March 29th & Tuesday, March 30th, 1999.
College of Ripon & York St. John, York, England.

Comparing Student Assignments by Computer

Lia Combrink-Kuiters, Richard V. De Mulder, Henk Elffers, Kees van Noortwijk.
Centre for Computers and Law
Erasmus University Rotterdam

Abstract

The possibilities offered by electronic methods of communication are increasingly being made use of these days in education. Course material is made available to students in electronic form. The students' assignments can be handed in via e-mail or Internet. Discussions between students or between students and their lecturer can now take place over the network.

The Centre for Computers and Law has developed a program for evaluating the work handed in by students if that work has a digital form. Not only can the program assist in assigning a mark to the students' work, it can also check to see whether students have cheated. In this paper, an outline of the program is given as well as a description of the experiments that have been conducted to test the efficiency of the system.

The program described in this paper is a result of the work the Centre has previously carried out on word similarity and conceptual retrieval. The program compares every document with every other document and then sorts all the document pairs on the basis of word similarity. Document pairs with the highest scores are examined by the lecturer and compared. Two experiments were carried out in order to test how effective the program was in detecting fraud. In the first experiment, we gradually changed one document by adding parts of a second document. The second experiment involved trying to fool the system by using synonyms.

The results of these experiments were very satisfactory. It could, therefore, be concluded that the system was reliable.

Contents

1 Similarity and the CODAS program

1.1 An outline of the CODAS program

Since the early 1990's, the Centre for Computers and Law at the Erasmus University, Rotterdam has been involved in research into the conceptual retrieval of documents from a database. Emphasis has been on document ordering and recognition based on certain internal characteristics, in particular the word use in these documents. Several statistical techniques have been used and evaluated for the purpose of this research.

A few years ago, it was decided to investigate the possibility of using these statistical techniques for a somewhat different purpose, that of assessing student assignments. The Centre offers several optional courses as part of the Master of Law programme. These courses attract up to 200 law students every year. Each student must hand in several assignments during the course. Each assignment requires the student to carry out a number of tasks and answer open questions on the subject. This means that some two-hundred assignments, each consisting of several pages of text produced with the help of a text processing program such as Word, are sent to the lecturers in electronic form via the network at regular intervals during the course. The completed assignments have to be checked and assessed within a short time so as to allow the students to monitor their own progress as the course continues. In order to help lecturers evaluate and grade these assignments quickly, we developed two computer programs:

- **a program that would compare all assignments on a particular subject with each other in order to make sure that students have not copied (parts of) each other's assignments, and**
- **a program which would enable the lecturer to sort and assess a set of assignments by marking some of the documents as 'examples' (of a good assignment), and some as 'counter-examples'.**

We have given this set of programs the name CODAS, which stands for 'Conceptual Document Analysis System'. This article will concentrate on the first of the two modules, that of comparing student assignments to check for similarity. Basically, the program calculates a 'similarity score' for every possible pair of assignments, again based on the word use in the assignments. It then presents a list of the pairs which resemble each other the most, ordered by a similarity score. This is very useful as it picks up on assignments which might have failed to attract the lecturer's attention but, when examined more closely, reveal such a striking level of similarity that the resemblance cannot be coincidental. For this reason, we call this program the 'fraud module'.

Several methods can be used to calculate a similarity score. After a number of experiments, we selected two methods for use in the fraud-module. To be able to appreciate how these methods work and what the differences are between them, it is necessary to give a brief outline of the theory of similarity.

1.2 Similarity: a short introduction

A general introduction to the subject of document similarity was given in [Van Noortwijk and De Mulder 1997]. The most important points made in that introduction are set out below.

The characteristic that forms the basis for calculating the similarity of two documents is the presence or absence of word types. The number of word types in a document is the number of different words in the document. For practical reasons, we will not take into account the frequency of a word type within each document (although the method could be extended to include this as well). The only type of frequency that plays a role here is the *number of documents in which a word type appears*. This characteristic, for which we will use the term *document frequency*, has a strong relationship to the dispersal of word types over the documents.

When we want to determine the similarity between two documents X and Y by means of the word types present in these documents, two deductions seem to be possible at first sight:

- **a word type is present in both documents; because this means that the documents have a common characteristic, it should *increase* similarity. For this situation we use the term *'hit'*.**
- **a word type is present in one document, but not in the other; at this point the documents differ from each other and therefore similarity should *decrease*. This situation is called a *'miss'*.**

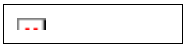
When a restricted number of documents (for example, a database) is considered, there is another characteristic that is perhaps not so obvious. The other documents will probably contain word types which are not present in either document X or document Y. The absence of such a word type in *both* documents can be considered a point of resemblance, which should *increase* the similarity of the documents. Therefore, this is also a kind of 'hit', just as in the situation where a word type is *present* in both documents. This means that two types of hits are possible:

- **a *'type 1 hit'* (in short, 'hit1') if a word type is present in both documents; and**
- **a *'type 2 hit'* (in short, 'hit2') if a word type is absent in both documents, but is used in other parts of the database.**

Not every word type appears the same number of times in a database. This does not necessarily influence the similarity of documents because, as stated earlier, in our definition word frequency *within a document* is not taken into account when calculating similarity. However, when the *number of documents* in which a word type is present (the 'document frequency' of a word type) differs from the number of documents in which other word types appear, this *can* have an influence on similarity. The probability that a word type with a high (document) frequency is present in a certain pair of documents and, therefore, is responsible for a 'hit1', is much higher than the probability that this will happen with a low frequency word. Conversely, the probability of a 'hit2' is higher with word types of a low frequency.

The probability that a word type will cause a hit or miss is therefore directly related to the

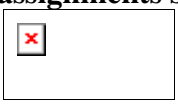
(document) frequency of that word type. That means that not every hit or miss can be considered to be of equal significance. When a word type with a frequency of only 2 (when the number of documents is high, say 1000) is found in a pair of documents, this gives us much more information than when a high frequency word type (for instance 'the', 'it', etc.) is found. Therefore, the similarity of the documents should increase more in the first situation than in the second. This means that we cannot just count the number of hits and misses when calculating similarity. With every hit and miss, we have to correct for the *probability* that the hit or miss occurs in a certain database. The way in which this can be accomplished is described in [Van Noortwijk and De Mulder 1997]. The most important point is that the *significance* of detecting a high frequency word type in both documents is generally low. To compensate for this, it is necessary to multiply every hit or miss with a 'weight factor' *W*, which should have a value opposite to that of the probability for the word. We use a weight factor which is the complement of this probability:

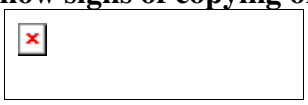


Using these weight factors, which have to be calculated for every word type, we can determine the *adjusted* number of hits and misses. This equals the sum of the weights of all word types involved. These adjusted numbers of hits and misses are suitable for calculating similarity because they take into account *which word types* are 'responsible' for a specific hit or miss.

To make it possible to compare the similarity that is calculated by means of these adjusted numbers of hits and misses for different pairs of documents, the numbers have to be made relative to their respective maximum values. These maximum values are different for every document. The maximum hit1-weight, for instance, is the sum of the hit1-weights of all word types present in the documents. When we divide the adjusted numbers of hits and misses by their respective maximum values and multiply the result by 100%, we get hit and miss percentages. When similarity is calculated from these percentages, we get similarity values that are comparable for all possible pairs of documents from a set.

Using the characteristics described above, a similarity score for every pair of documents in a certain set can be calculated. Several formulae can be used for these calculations (these formulae may be found in the literature on the subject). We are of the opinion, however, that, in principle, the following two formulae are particularly useful in order to reveal whether two assignments show signs of copying or some other form of fraudulent behaviour.





In both cases, the value of *S* will be in the range [0..1]. Using the formulae, it is possible to calculate the similarity between all possible document pairs in a database. The exact procedure to accomplish this is described in [Van Noortwijk and De Mulder 1997]. It involves the program reading the assignments and the creation of a matrix containing all word types, plus the numbers of the assignments in which they are present. From this matrix, every word type's weight can be determined, after which the similarity score (using either of the two formulae) for every pair of documents can be calculated.

1.3 Similarity used for fraud detection

Similarity scores, calculated as described above, can be a very effective tool to check a set of student assignments for fraud – does one assignment contain a copy of (parts of) another assignment? The two methods which can be used to calculate similarity could produce quite different results, however. The method which uses type 2 hits as well as type 1 hits (formula 3) is more sophisticated as it takes into account *all* common points between a certain pair of assignments, namely all word types that are *used* in both, as well as all types that are *'avoided'*

in both (but used in other assignments). However, the influence of type 2 hits can be expected to vary with the size of documents. If two relatively small documents are compared, there could be a lot of type 2 hits, possibly to such an extent that the type 1 hits have almost no role anymore. The simple method which uses only type 1 hits in fact just measures the 'overlap' between two assignments.

To check the validity of the results of the fraud detection program, and to find out which method for calculating similarity works the best in practice, we decided to conduct two experiments. These experiments are described in the next section.

2 Two experiments to test the efficiency of the program

In this section, the procedure and results of the two experiments we conducted will be set out. These two experiments were an attempt to answer the following question:

how sensitive and reliable is the fraud detection program?

In other words, what degree of similarity is necessary between two documents before the program would show that similarity as being more than coincidental? We tried to find an answer to this question in two ways.

- Firstly, we gradually changed one document by adding parts of a second document;
- Secondly, we tried to fool the system by changing the text of a document using synonyms.

2.1 The first experiment: gradually changing a document by adding parts of a second document

This first experiment was fairly extensive. An outline of the research procedure used will be given here, followed by the results and a comparison of both methods.

2.1.1 The procedure

An assignment (from a course on Computers and Law) was selected, containing 12 questions. For this assignment, 3 document pairs were selected.

Table 1: Document pairs from the assignment

	student number	number of word types	Student Number	number of word types
1	100191WB	270	126593JM	612
2	114157EB	184	141106VB	533
3	130520JM	331	147756EK	539

We gradually mixed the two documents in every row by replacing one answer to a question from the document in the left-hand column by an answer to the same question from the document in the right-hand column. By so doing, the resemblance to the original document gradually decreased, whereas the resemblance to the other document gradually increased. The shortest document from each pair was always the one that was altered.

Two methods for calculating a similarity score were mentioned in the former section. The 'sophisticated' method makes use of hit1's as well as hit2's, the 'simple' method just uses hit1's. Using these two methods – first the sophisticated, then the simple - we compared the newly created documents with both the original document as well as with the document from which the answers were copied. This comparison was repeated after each change to the text.

As could be expected, the similarity to the original document was high when only one or two answers were exchanged, but decreased when more answers were copied from the other assignment. At the same time, the similarity to the other assignment increased with every answer that was copied from it. The increasing similarity (with a higher number of common answers) to both the original and the other document is illustrated in the following two graphs. Note that the program only records the 300 highest similarity scores. Therefore, when similarity drops below a certain level, no score is plotted in the graph anymore. The dotted horizontal line in the graph indicates the most similar set of 'unmanipulated' documents, in which we found no signs of cheating.

Using the sophisticated method (hit1's + hit2's)

The graph shows that similarity rises above the 'maximum normal level' when 5 to 6 out of 12 answers to questions from the first document have been replaced by the same number of answers to questions from the second document. Although this was according to our expectations, it had been presumed that this point would be located even further to the left.

Using the simple method (just hit1's)

The results using this method essentially resemble those of the first (sophisticated) method. There are, however, a few differences which could be significant. Firstly, the level of similarity of the 'highest normal pair' is slightly lower using the simple method (6537 instead of 6786). This means that document pairs which involve cheating are somewhat more likely to be recognised by the lecturer. Furthermore, the similarity scores generally tend to increase faster using this method. Again, this could lead to a clearer distinction between the 'normal' document pairs and the fraudulent ones. The actual number of answers that have to be inserted before fraud can be detected positively is more or less identical with both methods (in both cases 5 to 6 answers – about half the assignment).

2.1.2 Results of the first experiment

When a significant part (more than half) of a certain assignment is copied, the similarity score of the manipulated and the original document is significantly higher than that of the highest pair of non-manipulated documents. This means that the probability that the fraud will be detected is very high. In this case, the simple method to calculate similarity scores produces slightly better results than the more sophisticated method. A possible explanation for this result is that, especially when only part of a certain document has been copied to another, the documents still contain quite a lot of the original (own) text. It is possible that this limits the usefulness of hit2's to calculate similarity. Leaving out the hit2's (and thus using the simple method) makes the program concentrate on 'overlap' (the answers which have been copied), which improves results.

2.2 The second experiment: trying to fool the system by using synonyms

In this section, we will describe the second experiment, in which only one document was used apart from the set of original documents. This test was applied to an assignment written in Dutch. However, in order to make this experiment accessible to non-Dutch speakers, a small section of the text has been translated into English. We translated the first answer from the original text and from the fraudulent counter-part.

2.2.1 The procedure

In order to fool the fraud detection system, we tried to construct an ‘adapted copy’ of an answer to the questions in the following way. We started with a student’s answer as a source text and changed many "fill in words"; these are words that have nothing to do with the content of the questions and the subject on which the questions touch. In that way the text would get high marks when marked by the human teacher, but – as would be the hypothesis – would not be recognised by the CODAS fraud detector as almost identical to the source text. For example, if the answer read: *"In that case, you have to minimise the word processor running under the Windows operating system*", we classified words as ‘word processor’, ‘minimise’, ‘Windows’ and ‘operating system’ as content words, not to be changed, and the other ones as ‘fill in words’ that could be altered without changing the content of the answer, while at the same time diminishing the formal likeness with the original sentence. We then produced a sentence such as: "*Therefore, it is possible to minimise the word processor as it runs under the Windows operating system"*. We also used the trick of alternative spelling (minimize instead of minimise; wordprocessor instead of word processor – word merging is a source of disagreement in Dutch orthography –). We contend that it is possible to change the form of a text without substantially changing its meaning. We noticed, however, that this treatment is by no means easy. We would question, therefore, even if it would be possible for a student with little or no knowledge of the content of the subject to change a text in this way, whether it would not be more work than just answering the questions. Nonetheless, as a procedure for testing the quality of the fraud detection algorithm it is a good one.

Original assignment (answer to question 1):

> Er bestaan zoals in de vraag wordt aangegeven twee manieren om een MS-DOS programma te draaien. Via de MS-DOS prompt en via de MS-DOS mode. Via de MS-DOS prompt kun je wisselen tussen fullscreen of in venster. Dit kan door alt/enter in te drukken. In de MS-DOS mode is dit niet mogelijk. Zo kom je er dus achter of men gebruik maakt van de DOS-prompt of van de MS-DOS mode.
>
> De MS-DOS prompt kan op verschillende manieren gevonden worden, bijvoorbeeld door de short-cut op de desktop, via de icon in de office-balk of via het start- menu. MS-mode kan gevonden worden als men bij het opstarten van de computer F8 ingedrukt houdt of wanneer men de computer afsluit kan er gekozen worden voor deze optie. Via F8 kan men overigens ook nog kiezen voor MS-prompt. Bij grote DOS programma’s is het raadzaam om via de MS-dos mode het programma te draaien. Dit omdat nu het hele grafische scherm wordt weggelaten.
>
> Tijdens het werken in bijvoorbeeld Word kan even een uitstapje gemaakt worden naar MS-DOS prompt (om even je opdrachtcijfers te bekijken). De MS-DOS mode is niet bereikbaar zonder dat Word afgesloten wordt. Dit zijn zo de grootste verschillen tussen deze twee applicaties.

English translation:

> There are, as indicated in the question, two ways to operate an MS-DOS program. Via the MS-DOS prompt and via the MS mode. Via the MS-DOS prompt you can change between full-screen and in the window. This can be done by pressing alt/enter. In the MS-DOS mode this is not possible. This way you can find out whether use is being made of the DOS prompt or the MS-DOS mode.
>
> The MS-DOS prompt can be found in various ways, for example by the short-cut on the desktop, via the icon in the office bar, or via the start menu. MS-mode is found if when starting the computer, F8 is kept pressed or when the computer is closed down this option can be chosen. Via F8, a choice can also be made for MS prompt. In the case of large DOS programs, it is advisable to operate the program via de MS-DOS. This is because the whole graphic display is now no longer shown.
>
> While working in Word, for example, an outing can be made to MS-DOS prompt (to have a quick look at your assignment grades). The MS-DOS mode is not accessible without leaving Word. These are the major differences between these two applications.

Adapted copy of the original assignment (answer to question 1):

> Je kunt op twee wijzen een MS-DOS programma draaien, namelijk met behulp van de MS-DOS prompt, maar ook in de zogenaamde MS-DOS mode. Bij de eerste manier kun je switchen tussen full screen en een venster, door alt/enter. Dat werkt niet In de MS-DOS modus. Om er achter te komen in welke van beide mogelijkheden je zit is ALT/ENTER toetesen dus de aangewezen weg.
>
> Je kunt naar de MS-DOS prompt gaan via een short-cut (desktop), door te klikken op de ikoon in de office-balk, maar ook met behulp van

het start- menu. Je komt in de MS-mode terecht door bij het aanzetten van de PC F8 ingedrukt te houden, of bij het afsluiten de betreffende optie te kiezen. Met F8 kan je trouwens ook naar de MS-prompt gaan. Omvangrijke DOS programs kan je beste in de DOS-modus laten uitvoeren, want dan laat je het grafische scherm weg.

Als je, als voorbeeld, in Word werkt is het makkelijk even naar de DOS prompt te gaan om andere zaken op te zoeken, maar de MS-DOS mode kan alleen gebruikt worden als je eerst Word afsluit.

English translation:

You can operate a MS-DOS program in two ways, namely with the help of the MS-DOS prompt but also in the so-called MS-DOS mode. Using the first way, you can switch between full screen and a window using alt/enter. That does not work in the MS-DOS modus. To find out which of these possibilities is being used, pressing ALT/ENTER is the indicated way.

You can go to the MS-DOS prompt via a short-cut (desktop) by clicking on the icon in the office-bar, but also with the help of the start-menu. You come into the MS-mode by pressing down on F8 while starting up the PC, or when closing down by choosing the right option. With the F8 you can also go to the MS-prompt. For extensive DOS programs, it is best to operate them in DOS-modus, because then the graphics screen does not appear.

If you work, for example, in Word, it is easier to go to the DOS prompt to search for other things, but the MS-DOS mode can only be used if you first quit Word.

2.2.2 Results of the second experiment

The results of this second experiment are in some respects comparable to those of the first. Although we tried to make the fake document look as different from the original as possible, it was still possible to detect it using the fraud program. Using the simple method, the fake document and its original counterpart ended right at the top of the list, with a similarity score of 6318. The highest 'normal' pair (2nd on that list) showed a score of 5813. Using method 1, however, the pair only appeared relatively low down on the list namely at position 17 with a similarity score of 6581. The highest 'normal' pair produced a score of 6938. In practice, we would not have checked a document pair at such a low ranking manually which means that the fraud would have gone unnoticed.

3 Conclusion

In both experiments, the simple method (measuring overlap) worked better. In the first experiment, there was only a difference in degree between the methods, but in the second experiment the result was very clearly in favour of the simple method. In both cases, the expectation was that the type 2 hits (words that are absent in both documents, but present in some of the other documents) would also give a positive contribution to measure the similarities between the original and the fraudulent documents. However, in both cases the similarities with respect to absent words seem to confuse the overall similarity rather than to sharpen it. The fact that this effect was even stronger in the second experiment is also surprising. A possible explanation would be that the replacement of the numerous "fill in words" is responsible for a low number or a low average weight of type 2 hits. The type 1 hits (words that are present in both documents), however, continue to score well just as in the first experiment because a number of important content words (which usually have a quite high weight) remain.

References

[Batagelj and Bren 1993]
V. Batagelj and M. Bren1993. *Comparing Similarity Measures*. University of Ljubljana, Ljubljana.

[Combrink-Kuiters *et al.* 1998]

Lia Combrink-Kuiters, Richard V. De Mulder and Kees van Noortwijk, "Analysis of Case Law: Measuring Similarity as an Aid to Coding Factors in Cases". In: *The Law in the Information Society; Legal Documentation Administrative Innovation The Lawyer's Training and Role*, Florence, 2-5 December 1998.

[De Mulder *et al.* 1993]

R.V. De Mulder, M.J. van den Hoven and C. Wildemast, "The concept of concept in

'conceptual legal information retrieval'". Paper voor *The 8th Bileta Conference*, 1st and 2nd April 1993, University of Warwick, Coventry. In: Proceedings of the Conference, pp. 79-91.

[De Mulder and Van Noortwijk 1994]

R.V. De Mulder and C. van Noortwijk, "A system for ranking documents according to their relevance to a (legal) concept". Paper voor The conference *RIAO 94, Intelligent Multimedia Information Retrieval Systems and Management*, 11-13 October 1994, Rockefeller University, New York, N.Y. - U.S.A 1994.

[De Mulder 1995]

R.V. De Mulder, "Probabilistic approaches to legal concepts". In: *Verso un Sistema Esperto Giuridico Integrale, Tomo I.* Ciampi C. *et.al.* (eds.), ed. Milaan, pp. 125-140. Towards a Global Expert System in Law; A Glance at the Conference, 'I.D.G.', Florence 1-3 December 1993.

[Van Noortwijk 1995]
C. van Noortwijk, *Het woordgebruik meester. Een vergelijking van enkele kwantitatieve aspecten van het woordgebruik in juridische en algemeen Nederlandse teksten* (Legal word use, a comparison of some quantitative aspects of the word use in legal and general Dutch texts). With a summary in English. Koninklijke Vermande, Lelystad 1995.

[Van Noortwijk and De Mulder 1995]

C. van Noortwijk and R.V. De Mulder, "Word use in legal texts: Statistical facts and practical applicability". In: Kralingen, R.W. van *et al.* (eds.), *Legal Knowledge Based Systems: Foundations of Legal Knowledge Systems* (Jurix'96), p 91-100. Tilburg University Press, Tilburg 1996, ISBN 90-361-9657-4.

[Van Noortwijk and De Mulder 1997]

Noortwijk, C. van and R.V. De Mulder, "The Similarities of Text Documents". In: *JILT – Journal of Information, Law and Technology*, Issue 2/1997. University of Warwick, Coventry 1997, http://elj.warwick.ac.uk/jilt/artifint/97_2noor/.

[Wildemast and De Mulder 1992]

C.A.M. Wildemast en R.V. De Mulder 1992. "Some considerations for the design of conceptual legal information retrieval systems". In *Legal knowledge based systems, Information Technology and Law,* Jurix '92. vermande, Lelystad, p.81-92.

Notes