

# Detecting student copying in a corpus of science laboratory reports: simple and smart approaches

Eric Atwell, Paul Gent, Julia Medori, Clive Souter,  
University of Leeds, Leeds LS2 9JT, England

## 0. Introduction

This case study is an evaluation of generic, general-purpose plagiarism detection systems applied to a specific domain and task: detecting intra-class student copying in a corpus of Biomedical Science laboratory reports. From the outset, our project had the practical, pragmatic aim to find a workable solution to a specific problem. Biomedical Science undergraduates learn experimental methods by working through a series of laboratory experiments and reporting on their results. These laboratory reports are “peer-reviewed” in large classes, following a prescribed marking scheme; as the reports are effectively marked by other students rather than by a single lecturer, there is an opportunity for an unscrupulous student to avoid having to carry out and report on an experiment, by simply copying another student’s report. To reduce this temptation, the Biomedical Science director of teaching, Paul Gent, approached Eric Atwell of the School of Computing and Clive Souter of the Centre for Joint Honours in Science, to look at ways to compare laboratory reports automatically, and flag candidates with signs of copying. We were joined by Julia Medori, forensic linguist from Trinity College Dublin, who developed and evaluated a range of possible solutions.

## 1. Detailed requirements analysis

We examined example student reports in the specific genre, to identify measurable characteristic features (e.g. use of identical diagrams and graphs may be indicative of copying but generic checkers like Turnitin assume text-only essays). We also interviewed Biomedical Science teaching staff to elicit/confirm significant diagnostic features which can identify copying, and identify the level of copying considered unacceptable (as some overlap in lab reports is expected).

## 2. Survey of what is available and how well this matches the requirements specification

Unlike other surveys of generic plagiarism detection systems, we aimed to evaluate systems against the detailed specific requirements identified above. A number of candidate systems are available, for example:

- (CheatChecker 2002) - an n-gram based tool;
- (SIM 2002) and (YAP 2002) based on longest common sub-sequence approach;
- detecting ‘unusual’ similarities between a closed set of essays (or programs) is called collusion and is the basis for (Copycatch 2002);
- (Copyfind 2002) is an example of a system aimed at scientific rather than humanities documents;
- (Turnitin 2002) is probably the most widely-known system, but the website gives scant details of the underlying methods;
- (Clough 2002) uses more sophisticated techniques from Natural Language Processing.

We decided to develop two ‘simple’ copying-detection solutions to assess; and compared these against ‘smarter’ commercial-strength systems.

### **3. Collating a test corpus of Biomedical Science student reports**

Evaluation using a test corpus is a well-established methodology in Corpus Linguistics research, and is appropriate to this task: in our case, the aim is to test candidate systems and compare detection (and 'red-herring') rates.

#### **3.1 sources: first and second year coursework exercises**

The initial test corpus used in our study was composed of 103 Biomedical Science first-year student reports. At the outset we did not know whether this included real cases of copying, or if so how many; none had been detected by teaching staff. We found 2 pairs of reports were almost identical, and called these Albert1, Albert2 and Barbara1, Barbara2 (all filenames are randomly generated and are not the real names of the students).

As there turned out to be few examples of 'real' plagiarism in our initial corpus, we extended it by adding 94 reports written by second year students, covering 3 more challenging, less constrained tasks; during subsequent tests, we discovered 4 pairs which appeared to involve copying: Geraldine2, Fred2; Jane2, Naomi2; Colin2, David2; and Jane3, Naomi3. There appeared to be no other clear cases of copying.

To ensure our corpus included samples which we knew were definitely copied, we also added an example of first-year plagiarism that had been detected previously by Biomedical Science staff, which we called Scanfile1, Scanfile2; we were given the hardcopy of two reports to scan using an Optical Character Recognition tool.

#### **3.2 adding artificial TestFiles by merging parts of existing reports**

To check the sensitivity of each plagiarism detection system tested, we created 19 new TestFiles: each composed of one file from which 5% (then 10%, ..., 95%) was removed and replaced by a portion of another file. We also created 2 extra TestFiles: TestFileAB made up of half SourceA and half SourceB; and TestFileABC, made up of TestFileAB with added text from SourceC from a different genre (an email message).

This gave a final corpus of 220 student laboratory reports including artificial TestFiles (which were still "authentic corpus material" as they were composed of parts of real reports).

#### **3.3 Format and content of the laboratory reports**

The first year assignment described the experiment and gave instructions on how to use a computer simulation of the laboratory apparatus to generate results; then the students had to produce tables and graphs and answer a few questions to analyse the results.

It seemed likely that any plagiarism we would find would be in the text of the answers. Unlike the first year student reports, the second year students did not have to answer specific questions; so the text should be more 'free' and show more originality. However, they still follow the same generic structure of all science laboratory reports: Introduction, Methods, Results, Discussion.

A characteristic of this genre is that a lot of the vocabulary from the specific science domain will be shared between all the reports.

The answers were quite short (generally about 25 words, occasionally up to 100 words or more), so there is not much scope for originality from each student. As most of the plagiarism checkers available at the moment appear to be developed to detect plagiarism either in humanities essays (high level of originality) or in programming (with limited vocabulary), there is no existing copying checker for the specific problem of scientific reports: low level of originality but not a limited vocabulary.

#### **4. Testing and Evaluation of a range of candidate copying-detection systems**

We used our Corpus to test contrasting approaches to detection. We developed two "simple" programs: Zipping, based on a standard file-compression tool, and Bigrams, a basic comparison of frequent bigrams; and we tested "smart" methods using commercial-strength plagiarism-checking systems Turnitin, Copycatch, and Copyfind. Our findings are reported in detail in (Medori et al 2002).

##### **4.1 Zipping, a simple compression-based approach**

As a baseline of our study, we tested the idea that using file compressors or zippers, we would be able to measure a 'distance' between files according to their similarities.

The algorithm of common zippers (like gzip) finds duplicated strings and marks up the matching strings with a pointer; which means that, by appending a questioned text to different corpora (e.g. of different authors or languages) and measuring the difficulty of zipping these files, we can measure the level of similarity between two files. The smaller the zipped file of the concatenation, the more common elements the questioned text shares with the corpus.

##### **4.2 Another 'simple' approach: Bigrams**

This method is another 'home-grown' program based on the idea that when 2 files are similar, they should share a high number of their bigrams. The programs were all implemented in Perl. We tested this method first with character bigrams and then, word bigrams. In the first place, we created a model for each file in our corpus by counting all the occurrences of each character bigrams in the file. To compare two files, we simply checked whether the most frequent bigrams were the same in both files. We compared the results for a model containing 10, 20, or 30 most frequent bigrams. Note that this comparison did not take account of the frequencies of the top 10/20/30 bigrams, it just compared members of the two lists.

The results even for "top 30" character bigrams were disappointing: this method will flag copying of an entire document by highlighting documents sharing the same most frequent 30 letter-pairs, but below a total match, most files share a large proportion of character bigrams. This may be at least in part because most laboratory reports will have similar style and content, so need a somewhat more sophisticated discriminator to go beyond finding near-identical documents.

We tried the same experiment with word-bigrams, and got better results: word-bigrams are better discriminators.

##### **4.3 Turnitin: a commercial WWW-based system**

The first commercial system we used was Turnitin.com, a web-based plagiarism detection service first designed to detect material cut and pasted from the Internet but also detects similarities between submitted papers; Leeds University has subscribed to the service.

The submission of papers is done by means of laborious cut-and-paste of the text from each document. After about 24 hours, an "Originality Report" is sent onto the user's account for each submitted paper.

We found that TurnItIn gives quite good results. However, the fact that it gives an index of similarity between the file and all papers found on the web and in the database leads it to miss a few cases of plagiarism, especially in the case of the pair 'Colin-David'.

Turnitin gives as a result the level of similarity between the submitted paper and all the papers in the database and the data found on the web; it seems a waste of time checking on the web and other papers than the ones in the class.

A more significant shortcoming is the laborious input method.

On the positive side, Turnitin underlines the similarities, making them easier to visualise; and it makes it easier to detect partial plagiarism.

#### **4.4 CopyCatch: a commercial PC-based system**

These conclusions led to the next step of our study, to test CopyCatch, a software developed by Woolls from CFL Software Development. It seemed that it might be more appropriate to our specific needs as it gives instant results, sorting them by percentage of similarity between 2 files.

The main distinguishing characteristic of this software was that it was much easier to use, especially the submission process: it allows the user to browse and select the files to check. The results are almost immediate. It outputs a list of pairs of files sorted by percentage of match between them.

It is then possible to have a list of the vocabulary or phrases shared between the 2 files and it can also mark up the files highlighting the similarities between them. It allows the user to check Word documents as well as text, rtf and HTML files: no previous conversion is needed.

CopyCatch found all our 'real' cases of plagiarism. It even found the pair 'Colin-David', which was not an obvious case of plagiarism and which we missed using TurnItIn.

The selection of our files in a few clicks is a very important practical advantage.

#### **4.5 CopyFind: a university-developed freeware alternative**

As another alternative, we decided to test another CopyFind, available for Linux as well as for Windows. This software requires that the Copyfind executable and all the files to be checked are in the same folder. The user has to add a list of all the files that have to be checked. These can be Word documents. The user can define a threshold of similarity. The system will then output a file containing all the pairs of files and the number of matched words for each of them.

An HTML report will then be created for every pair which are above the threshold, with underlined similarities.

We ran the program with standard parameter values; the results were quite good, but it didn't detect the pair 'TestFile-TestFile2'. To find this pair in the results, we had to lower the threshold by changing the minimum number of matching words to report from 500 (value suggested) to 300. The modification of this threshold didn't change any of our other results.

CopyFind is a program that works quite well and gives a clear output underlining the similarities between the files. The results are immediate, but its submission method is cumbersome.

### **5. Discussion**

We found little difference in detection rates between "simple" and "smart" approaches, and in practice the main performance discriminator was ease of use rather than accuracy. We found this surprising, given the naivety of the "simple" models: Zipping found copying as a side-effect of file compression; and Bigrams did not compare frequencies, each document is simply characterised by a vector of its 50 most-used word-pairs. These minimal models were enough to flag all known real cases of plagiarism in our corpus.

Turnitin gave interesting results too, as it was the only system to flag the overlap between our two artificially-constructed test files (TestFileAB, TestFileABC) and their sources (SourceA, SourceB). However, it didn't successfully detect the pair Colin, David. As this plagiarism is more representative of how students really plagiarise, this failure outweighs the success with artificial TestFiles.

In the tests, all approaches successfully flagged examples of "blatant" copying, but results were less clear-cut when reports had small amounts of overlapping text. The other top results were mainly pairs of small but unrelated files. This could be because of the nature of the assignment: the students often rewrite the objectives of the experiment, copy the questions, or answer them rephrasing the question in a negative or affirmative sentence. This could explain why the reports where the students did not develop the answers are found first.

This also suggests that we could try to take account of knowledge specific to our domain. In Biomedical Science laboratory reports, students must always write four sections: Introduction, Method, Results, Discussion. In fact this is standard practice even for research journal papers; and this format is a mandatory part of the course work specification. Biomedical Science lecturers know that certain parts of the text can legitimately be copied (the given specification of Method, and specific questions set by the lecturer), whereas originality is more important in other parts (notably the Discussion section). Perhaps a copying-checker should include some "intelligence", to make it ignore overlap in unimportant sections, and focus on the Discussion section; overlap in Discussion is a much surer indicator of copying.

On the other hand, perhaps we do not need such "intelligence". Research into students' reasons for copying suggests often the reason is either last-minute panic to meet a submission deadline, or else laziness or unwillingness to devote the time needed to do the work. In these circumstances, students generally make blatant copies rather than intelligently disguised copying through time-consuming detailed modifications. Our "simple" approaches suffice for these cases.

Anecdotal evidence from Biomedical Science teaching staff suggests that if cheaters modify the copied document to try to avoid detection, they tend to change just the first page (on the grounds that this is all that will be seen in a cursory skim through 200+ reports for blatant copying). A "simple" answer could be to check for blatant copying but ignore the first page; this would incidentally upgrade the significance of the Discussion section as it comes last.

## 6. Conclusions

We have analysed the requirements of our specific problem, through discussion with Biomedical Science teaching staff. We have surveyed a range of candidate systems, and evaluated five systems (Turnitin, Copycatch, Copyfind, and two "home-grown" systems, Zipping and Bigrams) in experiments with a Test Corpus of Biomedical Science student laboratory reports. We concluded that none of the systems stood clearly above the rest in their basic detection ability, but that Copycatch seemed marginally more successful, and more significantly it had the most appropriate user interface for our needs as it ran on a local PC and simplified the analysis of a whole directory of lab reports.

Following personal contact with Copycatch developers, we agreed to be the first beta-test site for the enhanced Java version. We have installed Copycatch on Biomedical Science computing network for future use by staff.

A side-effect of availability and use copy-checking software is the need to migrate from paper-based report submission and processing to electronic submission, probably via student pigeon-holes in the "Nathan Bodington" virtual building, an e-learning environment set up by the Flexible Learning Development Unit for general teaching use at Leeds University. The Flexible Learning Development Unit has accepted our recommendations (Atwell et al 2002). The School of Biomedical Sciences has decided to require electronic submission of all laboratory reports in future, to facilitate use of copying-detection software.

During tests, we found evidence to support suspected cases of copying, and discovered at least one new (previously unknown) case of student copying, which was reported to the appropriate authority.

A growing array of tools have appeared for detection of plagiarism and copying; some authorities such as JISC in Britain advocate standardised adoption of a generic centralised web-based system. We conclude that our task is an exception meriting a specialised corpus-based study and tailored solution; and that such language analysis tools can be a catalyst for reform in assessment practices.

## 7. References

Atwell, Eric; Gent, Paul; Souter, Clive; Medori, Julia. 2002. Final Report of the C&IT in the Curriculum project: Customising a copying-identifier for Biomedical Science student reports, School of Computing, University of Leeds.

<http://www.comp.leeds.ac.uk/eric/citFinalReport.txt>,  
or <http://www.comp.leeds.ac.uk/eric/citFinalReport.doc>

CheatChecker. 2002. <http://www.cse.ucsc.edu/~elm/Software/CheatChecker/>

Clough, P. 2002. <http://www.dcs.shef.ac.uk/~clough>

CopyCatch. 2002. <http://www.copycatch.freemove.co.uk>

CopyFind. 2002. <http://plagiarism.phys.virginia.edu/software.html>

Medori, Julia; Atwell, Eric; Gent, Paul; Souter, Clive. 2002. *Customising a copying-identifier for biomedical science student reports: comparing simple and smart analyses* in: O'Neill, M, Sutcliffe, R, Ryan, C, Eaton, M, & Griffith, N (editors) **Artificial Intelligence and Cognitive Science, Proceedings of AICS02**, pp. 228-233 Springer-Verlag.

SIM. 2002. <http://www.few.vu.nl/~dick/sim.html>

Turnitin. 2002. <http://www.turnitin.com/>

YAP. 2002. <http://www.cs.su.oz.au/~michaelw/YAP.html>