3-1-2003

The Mechanics of Plagiarism Detection Tools

Jackie Shane

The Mechanics of Plagiarism Detection Tools

Published In The Cyberskeptic's Guide, March, 8 (3) 2003, pp. 4-5.
By: Jackie Shane, University of New Mexico.

In the last issue I discussed the increasing ease of both plagiarizing and catching those who attempt it by using the Internet. Not only can students and yes, even famous authors, easily copy and paste from web sources, but there exists, in addition, a plethora of sites that sell or give away entire essays. Likewise sites like Turnintin.com exist to help university professors confirm plagiarism when it strikes. This article explains the mechanics behind plagiarism detection programs.

Using the Service

Using Turnitin.com, a popular detection tool discussed in my last article as an example, the first step is to set up a user name and password as an instructor. You will then need to set up an account number and a password to login. If you have a departmental administrative account, you can use your account number here. Once an account is created, as an instructor you create a new class and an assignment. The class includes the class title, course number, and the grade level. This generates a unique class identification number. This class ID plus a user password that you create, is what you give to students so that they can submit their papers to Turnitin.com themselves. When the students log in, they will look for their professor's name, the class number, and their assignment. Once they log into their assignment, they will see a start and due date for the assignment, any instructions or details regarding the assignment, and a window that explains how their assignments will be checked. The most comprehensive search that Turnitin can accomplish is to check against the Internet and an archive of previous and current student papers. Turnitin recognizes Microsoft Word, WordPerfect, PDF, postscript, RTF, and plain text files. A professor chooses either FastTrack or 24-hour turnaround time to generate an Originality Report. With FastTrack, a report takes three to four hours. The 24 hour option permits students to make last minute revisions as many times as they want during the 24 hour period. The reason the students have an entire day to resubmit their paper is because the actual time that it takes to do the analysis takes only a matter of seconds. The concept is a little like Priceline.com whose analysis also takes only seconds, but the software forces a staged delay.

Whether the student or the professor submits the assignment, the professor ultimately receives an Originality Report. In addition the professor has the option to let the students view their own Originality Reports. One feature only available to students who submit their own reports is the option for peer review from their classmates. The other advantage to having students submit their own papers is that the instructor needs to key in less information. There is a psychological deterrent to violate copyright when students realize that their papers will be run through a plagiarism detection site.

Judgement Call

Logging into a class, the instructor will see a list of assignments displayed, much as an inbox for email. Originality Reports are color-coded, with red showing the highest degree of similarity. This means that at least 80% of the text shows a match somewhere to an online source. In this case, the matching text is underlined in red and a corresponding URL sits above, also in red. Two frames permit the user to compare the original text to the alleged source text. The color-coded Originality Report indicates the proportion of matching material. Professors are warned to not make snap judgement calls based on the Originality Report alone, but to rather examine the student's paper and analyze the student's methodology on a more human scale. The software can easily detect even subtle dishonesty, and it is important to remember that minor copying can easily be a result of ignorance, rather than malicious intent.

Detecting Plagiarized Computer Code

There are several computer programs that detect cheating in computer programming assignments. It makes sense that computer science instructors would devise software solutions for detecting copied computer programs, since writing code is what they do best. Whereas plagiarism detection software for text is usually fee-based, equivalent programs for reading computer code is generally free. The added cost for text-based systems probably comes from the cost of licensing proprietary databases, and the fact that an English teacher is less likely to write their own similarity detection software. Computer science assignments generally require students to automate a task by writing a computer program, which is then graded, for effectiveness and style. Since it is very easy for students to exchange code, they often do. One obvious dichotomy is that if a student is savvy enough to make enough sophisticated modifications to a program that they dupe the detection software, could it not be argued that the student deserves credit for their skill in program editing? The reality however is that students generally cheat because they have weak programming skills, and thus will typically make simple modifications which can be immediately detected by a program such as MOSS, BOSS, or SHERLOCK.

MOSS, which stands for Measure of Software Similarity, is a program developed by a graduate student (also at UC Berkeley) that searches lines of computer code for similarities. MOSS compares a batch of submitted student programming assignments against each other, and highlights passages with similar code. Registration is limited to programming instructors, but an instructor can register within 24 hours, and the system is simple to use and free. The biggest difference it has from the text-based packages is that student's programs are compared only against other programs in the same class, so it is best used by instructors who teach large survey classes.

The Algorithm

So what is happening behind the scenes? Generally plagiarism detection programs divide documents into smaller elements that can be scanned and compared against a large database of other documents as quickly as possible. Some scan documents word by word, while others look for key phrases. The database used for comparison might be a collection of student assignments from one class, or may in addition contain books, manuscripts, web pages, or an archive of previous assignments.

Turnitin.com mines the Internet daily for new documents with web-crawlers. The result is a locally mounted mini-Internet consisting of sites likely to be sources of material for students. These include paper mills, online encyclopedias, and news services. Added to these huge repositories of data is the continual accumulation of papers submitted to the site for review. Computer algorithms developed by computer scientists at UC Berkeley then analyze the text and assign “digital fingerprints” to passages of text. In other words, the character of the text is identified as much as the actual words. This means that a student can radically alter the wording of a concept, and yet the major content can still be identified as stolen.

It is difficult to find definitive information about the actual algorithm that powers a plagiarism detection site, primarily because the creators assume that the more the information is publicized, the greater the likelihood that students or owners of paper mills will attempt to outsmart the software. There are various software programs that detect similarities among documents, some of which can been found within the patent literature. The problem at hand is to compare a paper against an enormous collection of existing documents. Part of the credit is owed to the availability of fast processors, but the big benefit of these software programs that they efficiently package the material to be read by applying fingerprints, or identifiers, to portions of each document.

A document is read as a data object. Features from data objects are extracted as elements consisting for example, of a phrase or a word. A grammatical dictionary might splice out insignificant words within a sentence, such as prepositions or conjunctions to eliminate bulk. In the case of MOSS, the software can eliminate false positives by automatically eliminating code that was intended to be shared. The remaining elements are then characterized as fingerprints. A probability model then is needed to determine whether the overlap of key elements is random. Certainly an instructor could do their own spot checking by simply extracting some random key phrases from the paper and see if an Internet search engine finds a match. The drawback is that depending on the length of the paper, this might be tedious or incomplete. The other advantage to using a detection site is that they may provide access to papers hidden within the assorted paper mills or articles within proprietary research databases. The algorithm therefore makes it fairly easy to detect copied documents, even when phrases are rearranged, or slightly modified.

Comparison detection tools extend beyond the classroom. Law-enforcement agencies use them to draw patterns from crimes, patent agents can detect similarities in applications, and funding organizations can detect duplicate grant proposals. It is no wonder then that the very same grant foundation that funded the development of one similarity detection tool used the resulting software for this very purpose.

Likewise, fingerprinting has applications that extend beyond the task of hunting down cheaters. Clustering similar data into groups is useful for building an efficient search engine or indexing tool. Which brings us full circle to the cheapest cheater's detection tool around. Google.

Related Sites

http://www.turnitin.com/static/index.html
http://www.cs.berkeley.edu/~aiken/moss.html