Renaissance and Reformation
Renaissance et Réforme

Bloomfield, Lou, project creator. WCopyfind. Software

Dennis McCarthy

Volume 44, Number 4, Fall 2021

URI: https://id.erudit.org/iderudit/1089353ar
DOI: https://doi.org/10.33137/rr.v44i4.38646

See table of contents

Publisher(s)

Iter Press

ISSN

0034-429X (print)
2293-7374 (digital)

Explore this journal

Cite this review

McCarthy, D. (2021). Review of [Bloomfield, Lou, project creator. WCopyfind. Software]. *Renaissance and Reformation / Renaissance et Réforme, 44*(4), 196–200. https://doi.org/10.33137/rr.v44i4.38646

Bloomfield, Lou, project creator.
WCopyfind. Software.
Virginia: The Plagiarism Resource Site, 2001. Accessed 30 June 2021.
plagiarism.bloomfieldmedia.com/software/wcopyfind.

Lou Bloomfield, a physics professor at the University of Virginia, describes his WCopyfind as "an open source windows-based program that compares documents and reports similarities in their words and phrases" ("WCopyfind"). You download it from his site and then run it on your computer—with both 32-bit and 64-bit versions available. The program is especially useful for finding matching word-strings among groups of documents or the canons of various authors. It even offers the possibility of searching for matching strings with "imperfections"—i.e., word-strings that also include a certain small number of unmatched words. You can alter the program to search for word-strings that match, say, six of seven words in a row or six of eight, etc. This provides a quick and effective way of tabulating linguistic similarities, which in turn can be used for uncovering source texts and source passages, or perhaps even unravelling knotty authorship questions. WCopyfind can also provide an instant vocabulary list, showing the number of times each word appears in a particular text.

Bloomfield first created this software in 2001 to find examples of plagiarism in the term papers of his physics students. Then he offered it to others, hosting it on his own computer. In 2011, he moved his site to WordPress, and he continues to produce updated versions of the software regularly. According to the website, WCopyfind is "licensed under the Gnu Public License, which basically means that you can do whatever you like with it except to try to sell it to someone else" ("WCopyfind").

Significantly, the program does not search the Internet. Instead, you must download the documents you want to compare into your computer—or download Internet shortcuts to webpages you may want to analyze. Acceptable file formats include docx, doc, pdf, txt, htm, and html. Naturally, you can always download or copy-and-paste files from the Internet. This includes texts from Early English Books Online (EEBO) or books on Google Books that allow full view. One can then convert these texts into either PDF or DOCX format. Importantly, the PDF files cannot comprise pictures of pages but must be in a text format.

WCopyfind has two input windows: "Old Document Files (compare only with new documents, not with one another)" and "New Document Files (compare with old files and with one another)." Right click on the input windows and select the files (or "Internet shortcuts" for webpages) on your computer that you would like to compare. To compare all of the files with one another, you can just drop them all into the "New Documents Files." But perhaps you want to compare a group of anonymous plays, not with each other, but with, say, the plays of the Shakespeare canon. In this case, you would place the anonymous files in the "Old Document Files" and all of the Shakespeare plays in a single file in the "New Document Files."

By checking certain boxes, the program allows you to ignore "All Punctuation," "Outer Punctuation," "Numbers," and "Letter Case." These are all important to check (and so ignore) as these irrelevant variations would cause the program to miss many matching word-strings.

The program also permits you to "Skip Non-Words"; "Skip Words Longer Than x Characters," where you get to choose the number, but twenty is suggested; and search "Basic Characters Only (in DOC files)." These last three let the program ignore "non-textual items, including filenames, URL, image data, and other word-processor junk" ("WCopyfind Instructions"). These are typically unnecessary to check.

WCopyfind allows you to adjust the size of the matching word-string that the program identifies. Although this parameter is described as "Shortest Phrase to Match," it does not regularly find word-strings that are longer than the chosen number. If you choose four for the length of "Shortest Phrase," the resulting word-strings are mostly of that length, with only a few that are five-words or longer.

As noted, you also have the option to choose "Most Imperfections to Allow"—which refers to the number of unmatched words *in a row* that the program will overlook. The default is zero. This must be adjusted with another parameter—"Minimum % of Matching Words," in which the default is 100. This refers to the percentage of non-matching words in a string. If you keep the default values for either of these parameters, the program will only list exact word-for-word matches for the size of the string you have chosen. If you also want the program to list results that include an unmatched word or two, you have to adjust both values. The following examples may help clarify:

Search parameters: Shortest phrase (8); Imperfections (0); Minimum % (100)
Possible result:
Doc1: Something is rotten in the state of Denmark
Doc2: Something is rotten in the state of Denmark

Search parameters: Shortest phrase (8); Imperfections (1); Minimum % (80)
Possible result:
Doc1: Something is rotten in the state of Denmark
Doc2: Something is *very* rotten in the *dreary* state of Denmark

Search parameters: Shortest phrase (8); Imperfections (2); Minimum % (80)
Possible result:
Doc1: Something is rotten in the state of Denmark
Doc2: Something is rotten *and stinking* in the state of Denmark

The second example above helps show that the number of imperfections refers to words in a row. Even though "Imperfections" was set to "1," a word-string with both *very* and *dreary* appears because each of the words is isolated. They do not appear in a row. Since the "Minimum %" is set to 80 and eight of the ten words do match, this word-string meets the parameters. I suggest keeping the number of "Imperfections" low—three or fewer even on word-strings of length seven or higher—and keep "Minimum %" above 70.

In the final steps, you must decide whether you want a "brief report," in which only the matches are shown in side-by-side windows, or leave "brief report" unchecked, which will result in the matching strings highlighted within the full texts, again displayed in side-by-side windows. Conveniently, every matching word-string appears as a red, underlined hypertext, and if you click a word-string in one panel, it will immediately take you to the location of its sister word-string in the text in the other panel. This especially helps in the search for source passages.

For early modern researchers there is, however, a rub. Unsurprisingly, WCopyfind cannot match word-strings that include spelling variations. So, unless you are interested in orthography, this presents a problem for comparisons of early modern texts and does require access to modernized versions or, gulp, forces one to spell-correct the documents you want to compare. Early modern software spell-correction is still in its infancy, so I have

personally created macros in Word to normalize many of the most common spelling variations in early modern texts. I have made these available for Word-users here (sirthomasnorth.com/2021/06/24/ye-olde-spellinge-corrector) and listed the instructions. The macros will help spell-correct many of the most common variants in sixteenth- and seventeenth-century documents. Plenty of spelling errors will be left, but they will at least be rare enough to ensure WCopyfind remains useful. And by using "imperfections," you can check the matching word-strings to see if one of the unmatched words is actually just a spelling variant of an identical word.

As always, some knowledge of basic forensic linguistics, probabilities, and/or source study is helpful in distinguishing between parallels that are compelling and informative, and those verbal matches that are just as likely to be the product of happenstance. This is why context is important in efforts to establish literary obligation. Indeed, while it is true that any two works may share a peculiar word-string just by chance, many underestimate how wildly unlikely it is for two writers, both working independently, to use the same rare language when describing the same specific idea, image, or story. The crucial difference is context—a fact that everyone seems to understand when they are already familiar with the borrowed line.

For example, if a late seventeenth-century text includes the line "lend me your ears," this may or may not have anything to do with Shakespeare's *Julius Caesar*. EEBO confirms that "lend me your ears" was not a unique expression and that it occurs in thirty-seven works other than Shakespeare's tragedy, some preceding the play, and many other instances obviously unrelated. However, if this later text were a satirical play and includes a character named Mark Antony who enters the stage and says, "Hey everyone, lend me your ears," we now know, of course, this is unlikely to be a coincidence and that the satirist is spoofing Shakespeare's famous scene. The difference is that in our hypothetical satire, "lend me your ears" does not just appear *anywhere*; it is uttered in the identical situation as Shakespeare's tragedy. And EEBO confirms that its usage was extremely rare. Yes, it is true that the EEBO database is incomplete—many thousands of early modern texts have been lost or are still unsearchable—but EEBO is more than large enough to provide a statistically significant sample-size, comprising over sixty thousand searchable texts and more than one billion words. Thus, the thirty-seven occurrences of "lend me your ears" on EEBO equates to a usage of once every twenty-seven million words or so. So, it is

extremely improbable that the later writer, just by chance, opted to put "lend me your ears" in the mouth of a "Mark Antony" as he first addresses a group. A similar argument holds even if you aren't already familiar with the line. If two passages are both discussing the same distinctive event, person, image, idea, etc., and they share the same rare language, then the passages are likely related.

In WCopyfind, the best way to search for source passages is to examine the "brief report" of word-strings of various sizes, especially with lengths of five to eight words long, and with zero, one, and two imperfections. If you find any distinctive word-strings, then uncheck "brief report" and examine the matching word-strings in their respective contexts. If indeed the context is the same, then you have likely found two passages that are linked. This is necessarily informative but does not guarantee that the later author has borrowed from the earlier text. It is always possible that the two parallel passages are related through some other work(s)—that the passages are, in effect, siblings or even cousins rather than parent and child. That possibility remains even if no other possible source appears on EEBO. But if we find that other source passages connect the two texts, then it becomes increasingly clear that some version of this earlier text was indeed the seminal source for the related passages in the later text.

Once you have found a source passage, it is then a good idea to copy and paste a few paragraphs of surrounding material from both texts into two new small files and run it through WCopyfind again, narrowing the "Shortest Phrase" to three, two, and one. This will allow you to find matching content words and smaller phrases that the later writer may have echoed.

While the ways to establish a source passage are straightforward, identifying anonymous authors is far trickier and more controversial. The bar is also much higher. The problem is distinguishing between some anonymous work that has greatly influenced some writer and a work that was likely written by that writer. Exactly where we should put that line is not yet clear, but here, too, WCopyfind can provide service, offering scholars a chance to uncover all the matching phrases and shared linguistic peculiarities among texts. WCopyfind, with the right preparation, is an extremely useful tool and can help early modern researchers make important discoveries.

DENNIS MCCARTHY
New Hampshire
https://doi.org/10.33137/rr.v44i4.38646