

Document Plagiarism Detection Tool using Edit Distance Text Similarity Measure

Asra Masrat¹, Hardika Gawde², Mohammed Ammar Makki³, Utsav Parekh⁴

¹Asra Masrat, Dept. of Information Technology, K.J. Somaiya College of Engineering, Maharashtra, India

²Hardika Gawde, Dept. of Information Technology, K.J. Somaiya College of Engineering, Maharashtra, India

³Mohammed Ammar Makki, Dept. of Information Technology, K.J. Somaiya College of Engineering, Maharashtra, India

⁴Utsav Parekh, Dept. of Information Technology, K.J. Somaiya College of Engineering, Maharashtra, India

Abstract -Plagiarism is a compelling issue that is faced inresearch. However, there are different views on how to defineplagiarism and the criteria that makes plagiarism as culpable.In this paper, we have explored plagiarism, its concept anddeveloped a tool to check plagiarism between two-word documents using the edit distance text similarity measure. Plagiarismshould be realized as using someone’s intellectual property suchas ideas or text, and implying it as one’s own. We displayour implementation for the plagiarism check and display theplagiarism as a percentage between the two documents. Weanalyze the two documents and use the edit distance values thatwe calculate, to calculate the mentioned percentage. This can beused to realize the plagiarism and work on the severity of thesame.

Key Words:Edit distance, plagiarism, docx2txt, Levenshteindistance

1.INTRODUCTION

In this era, information is an abundant resource. It is widely

available in the palm of our hands through our interconnected devices. The Internet is the most widely used medium for information exchange across the globe. Every day, variant userstransfer documents and work on-line each for public and personal uses. With the help of different search engines like Google, Bing, DuckDuckGo, Yandex, etc. it is easier to find large amounts of information within a fraction of a second. This has led to frequent exploitation of intellectual property in the form of plagiarism.

Plagiarism is intellectual larceny, it is filching other’s thoughts, ideas, literature without consent or original source’s citation. Plagiarism occurs in different ways, it is done on all types of data on the internet but it is observed frequently on textual data like documents, essays, articles and research papers. A method used to calculate the similarity and differences between the data objects is called data comparison.

The most common field where plagiarism is seen is in academics, this happens mostly due to the similar work assigned to students as the work already present online. That being said, there is an urgency for a serviceable method to detect similarities between two documents.

We have developed a plagiarism tool for Examining the similarity between two documents i.e, to detect the plagiarism of documents. We have used edit distance text similarity measures to design this tool.

Edit distance algorithm was invented by Vladimir Levenshtein. In computer science, the Edit distance algorithm is used to quantify how similar or dissimilar two strings are from each other by calculating the least number of functions necessary to convert the first string to the other.

The following are the three basic edit operations to replace one string to another or to modify one document to another by the Edit distance algorithm:

- Substitution
- Insertion
- Deletion

$$\text{Dist}[i][j] = \begin{cases} \text{when } X[i-1] == Y[j-1] \\ \text{dist}[i-1][j-1] \\ \text{when } X[i-1] != Y[j-1] \\ 1 + \text{Min} \left\{ \begin{matrix} \text{dist}[i-1][j] \\ \text{dist}[i][j-1] \\ \text{dist}[i-1][j-1] \end{matrix} \right\} \end{cases}$$

Fig -1: Edit Distance formula

We expect the implementation of our plagiarism detection tool using edit distance algorithm will be efficacious in inspecting the similitude between two documents.

2. APPLICATIONS OF SIMILARITY MEASURE

2.1 File Revision

Edit distance algorithm is used in file revision commands such as diff in UNIX which finds the difference between two files, producing an edit script to convert file1 to file2. This was very important in earlier days when someone edits a large file on a remote computer and wants to transfer it to his/her local machine, instead of transferring the entire file he/she can just transfer the edit script generated by diff command and then make the changes in the local machine thus saving large bandwidth.

2.2 Remote Screen Update Problem

If 2 machines are connected via remote login, here files from machine1 are being accessed by machine-2. Then machine-1 may need to update the screen on machine-2 as well. Here edit distance can be used to find the differences between two pictures of the current screen of machine-1 on machine-2 and the second picture of what the screen should look like. Here the difference between machine1 and machine-2 are transmitted which leads to less transmission bandwidth.

2.3 Spelling correction

Edit distance algorithm is also used in spell correction where transposition errors are present, i.e. it can be treated as simple deletion and insertion. When a text contains a word which is not in the dictionary but close to another word in the dictionary which has small edit distance, that word may be suggested as a correction.

2.4 Molecular Biology

As edit distance finds how close two strings are in the same way it can also be used to calculate distance between DNA sequences like strings (A,C,G,T) protein sequence, Genes.

2.5 Speech Recognition

Edit distance is used in some speech recognition systems where it is impossible to find the exact match in audio signals, edit distance can find a close match between a new voice and the stored old voice.

2.6 Plagiarism Detection

In this the edit distance helps in identifying copied content. When there is an instance of utilizing another person's words or intellectual property, without giving credit, in such a case, plagiarism occurs. They are contemporary tools that are used to trace fragments of plagiarized/copied or identical content in research papers/journals, etc. These are possible in numerous languages. The AI-powered tools were multi-layered search mechanisms come into use.

These tools explore through a large number of documents, papers, blogs and journals.

3. METHODOLOGY

Application Algorithm:

1. Start
2. Application window opens.
3. Use the 'open document 1' button to select the first document.
4. Use the 'open document 2' button to select the second document.
5. Click on the 'plagiarism check' button:
 - a. The document is split into sentences using the '.' as the delimiter, into two lists.
 - b. The lists are converted to a set to remove duplicates and reconverted to lists for further processing.
 - c. The documents are compared using the edit distance on sentences and we check for the minimum distance pair for a sentence, then if it matches with our criteria, the plagiarism counter increases.
 - d. The plagiarism counter stores the total addition of the document similarity between the two.
 - e. Similarity (plagiarism) percentage is calculated in accordance with the plagiarism counter and the document size.

The similarity/plagiarism percentage is then displayed to the user using Tkinter GUI.

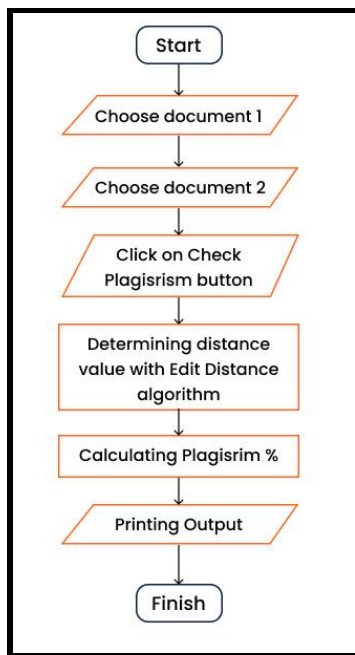


Fig -2: Flowchart of application

4. IMPLEMENTATION

Tkinter package uses the Tk interface which is the most common Python interface to the Tk GUI toolkit. It is used to create GUI in python. Both Tk and Tkinter are available on Windows as well as unix platforms for usage. Tkinter mainly comprises ‘Tk’ which is the interface module containing widgets and also contains modules which can be used to open files, and dialogs.

Filedialog is used for opening/saving a file of user’s choice. We have used this module to open the files that we used to checking the similarity. Button is used for creating buttons in a Python application. These buttons can describe their purpose by using their ability to exhibit text or images. The buttons have methods attached to them, when a click event occurs on a button, the function attached to that button gets called automatically. We have used buttons to call the filedialog as well as call the function to calculate similarity.

Text Widget is used where a user wants to place multi-row text fields. Text widget has multi-purposes uses like exhibiting or dispatching information, etc. We have used the text widget to display plagiarism percentage.

Docx2txt A pure python-based utility that is used to extract text from docx files. We have used the function ‘Process’ from the library. This function is specifically used for converting the docx file into a string that contains only text from the docx file.

5. Results & Discussions

Here in **Case 1**, we have considered two similar documents where document-1 is related to good health and its constituents, and document-2 which is the exact same document. Then we have calculated the plagiarism percentage between these two documents using edit distance. As we can observe in the GUI of the program, the output that is calculated is 97.6744% which is the amount of similarity between the two documents owing to the elimination of duplicates. The program has calculated the plagiarism or similarity between the two input documents and displayed its result in the text box on the GUI. We also show the plagiarized lines that are detected in the program output. These are the lines that are similar and have been considered in the calculation of the plagiarism percentage. In this case, we have the maximum number of plagiarized lines considering the removal of duplicates.

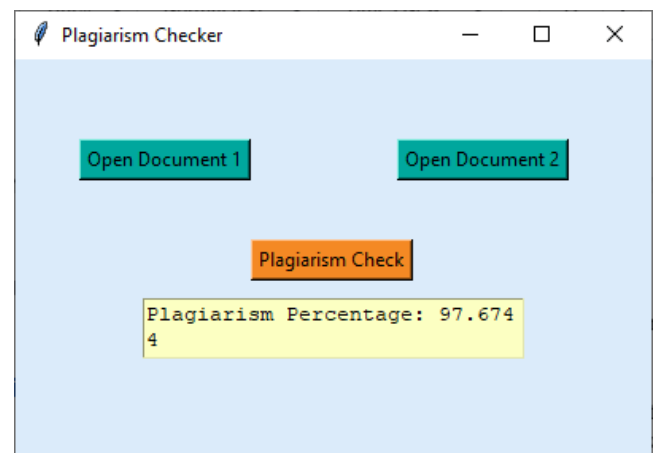


Fig -3: Case 1: Highest similarity between documents

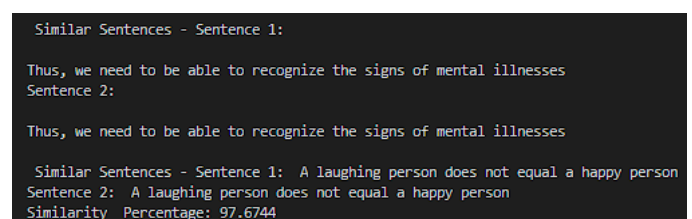


Fig -4: Case 1: Highest Plagiarized Lines

Here in **Case 2**, we have considered two similar documents where document-1 is related to good health and its constituents, and document-2 which is a paraphrased version of the former document. Then we have calculated the plagiarism percentage between these two documents using edit distance. As we can observe in the GUI of the program, the output that is calculated is 86.0465% which is the amount of similarity between the two documents.

The program has calculated the plagiarism or similarity between the two input documents and displayed its result in the textbox on the GUI. We also show the plagiarised lines that are detected in the program output. These are the lines that are similar and have been considered in the calculation of the plagiarism percentage. In this case, we have several plagiarised lines.

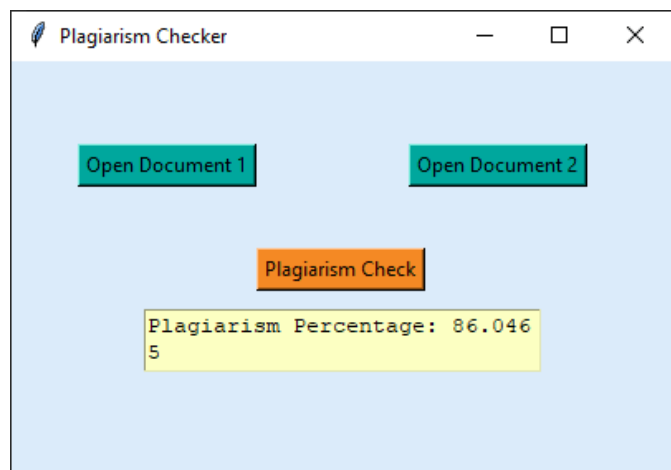


Fig -5: Case 2: High similarity between documents

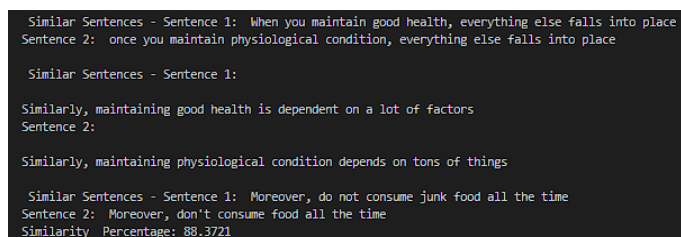


Fig -6: Case 2: Many Plagiarized Lines

In **Case 3**, we have considered two documents where document-1 as well as document-2 are essays on global warming but from different sources. Then we have calculated the plagiarism percentage between these two documents using edit distance. As we can observe in the GUI of the program, the output that is calculated is 3.8462% which is the amount of similarity between the two documents. The program has calculated the plagiarism or similarity between the two input documents and displayed its result in the textbox on the GUI. We also show the plagiarized lines that are detected in the program output. These are the lines that are similar and have been considered in the calculation of the plagiarism percentage. In this case, we have fewer plagiarized lines since the documents are different.

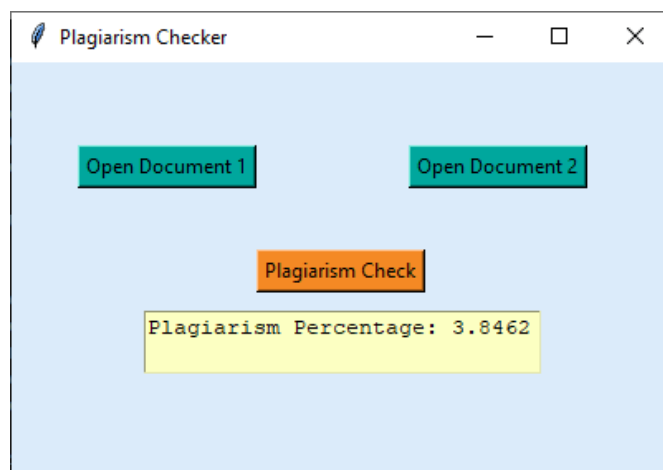


Fig -7: Case 3: Low similarity between documents

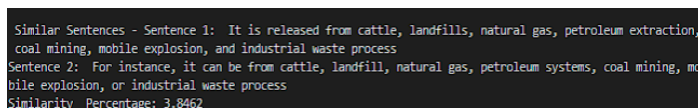


Fig -8: Case 3: Some Plagiarized Lines

Here in **Case 4**, we have considered two similar documents where document-1 is related to global warming and its constituents, and document-2 related to health is a completely different document. Then we have calculated the plagiarism percentage between these two documents using edit distance. As we can observe in the GUI of the program, the output that is calculated is 0.0% which is the amount of similarity between the two documents. The program has calculated the plagiarism or similarity between the two input documents and displayed its result in the textbox on the GUI. We also show the plagiarized lines that are detected in the program output. These are the lines that are similar and have been considered in the calculation of the plagiarism percentage. In this case, we have no plagiarized lines since the documents are completely different.

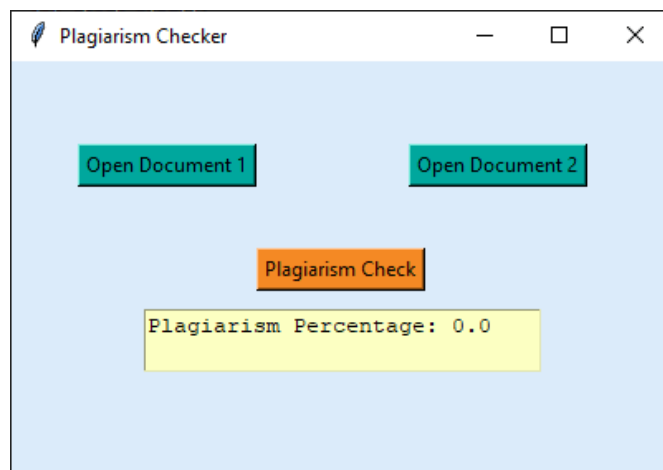
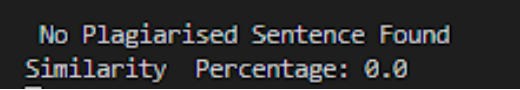


Fig -9: Case 4: No similarity between documents



No Plagiarised Sentence Found
Similarity Percentage: 0.0

Fig -10: Case 4: No plagiarized lines

- 6) <http://users.monash.edu/~lloyd/tildeAlgDS/Dynamic/Edit/>

6. Conclusions

The Edit distance is an effective tool that can be used to detect the similarity between the documents because it is able to represent the amount of editing one sentence needs to transform into its comparative counterpart. It is used to check plagiarism in several aspects and fields where the document is relevant and significant. These plagiarism detection tools come into the picture when there is a high availability of resources via the internet and it is very easy to get information online. In case of plagiarism, it can be via the internet or even the assignments and works of colleagues. Thus, it is necessary to work on this and detect plagiarism. We perform plagiarism checks via our application between documents and show the plagiarism percentage. This value varies with the similarity between the documents. We present our vision for plagiarism detection via this research.

REFERENCES

- 1) Makalah, "Plagiarism Detection Using Levenshtein Distance With Dynamic Programming", Indonesia M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.
- 2) Rani S., Singh J. (2018) Enhancing Levenshtein's Edit Distance Algorithm for Evaluating Document Similarity. In: Sharma R., Mantri A., Dua S. (eds) Computing, Analytics and Networks. ICAN 2017. Communications in Computer and Information Science, vol 805. Springer, Singapore.
- 3) Anohina-Naumeca, Alla Mislevics, Antons Grundspenkis, Janis. (2008). Development of the Plagiarism Detection Tool for Processing Template-Based Documents. 187. 67-78. 10.3233/978-1-58603-939-4-67.
- 4) B. Berger, M. S. Waterman and Y. W. Yu, "Levenshtein Distance, Sequence Comparison and Biological Database Search," in IEEE Transactions on Information Theory, doi: 10.1109/TIT.2020.2996543.
- 5) Vibhakti V. Bhaire, Ashiki A. Jadhav, Pradnya A. Pashte, Mr. Magdum P.G - SPELL CHECKER - published at: "International Journal of Scientific and Research Publications (IJSRP), Volume 5, Issue 4, April 2015 Edition".