

Plagiarism Detection by Identifying the Keywords

Sandipan Dutta

Department of Information Technology
Heritage Institute of Technology
Kolkata, India

Debotosh Bhattacharjee

Department of Computer Science and Engineering
Jadavpur University
Kolkata, India

Abstract— Plagiarism occurs when authors copy one's work and presents that as their own work. In an earlier time, Plagiarism means copy-paste work i.e. simply copy ones work or code in its own work without giving credit to the author. Actually definition of plagiarism is not restricted to only copying word by word but also copying ones' thought, idea, and represents it as own idea without giving proper acknowledgement to the original author. In research area, it is a serious crime and still now not so many systems have been developed to detect plagiarism. Our area of focus is on the keywords or index terms present in different research papers. Our system will extract the keywords present in the research papers and compare with each other depending upon certain criteria and will conclude the possibility of plagiarism between two research papers.

Keywords— Plagiarism; Keywords; White Space; Synonym.

I. INTRODUCTION

Plagiarism detection is a mechanism to identify the location of plagiarism within a document or work. Though most of the cases of plagiarism are found in academia like copying essays, reports, research paper but it also available in case of software by copying one's code but cleverly changing the function name, variable name, etc. Here our main focus area is plagiarism in research paper. Plagiarism detection can be manual or software based. Manual detection of plagiarism needs enormous effort and nearly to be impossible where to many documents needs to be compared. On the other hand, software based detection technique mainly focuses on 'word by word' comparison though definition of plagiarism is not restricted only to copy the document but also copying the ideas also. Our current study indicate that [2] has discussed a new taxonomy of plagiarism that highlights difference between literal plagiarism and intelligent plagiarism from plagiarist behavioral point of view. This taxonomy was developed with the help of study on various kind of plagiarism and recent literals on plagiarism concept. This taxonomy is supported by various example of plagiarism. After that different textual features are illustrated to represent the text document. Then all this are co-related to detect plagiarism. [1] Had proposed two major metrics spatial plagiarism similarity and temporal plagiarism similarity mainly for Korean documents. These two metrics are combined to develop evolutionary plagiarism probability model adopting Weibull Distribution. By this method direction of copy is measured. [3] Mainly deals with software plagiarism. More specifically for those where there is type redefinition plagiarism. Here the main focus is to improve CCFinder in the aspect of type

redefinition to increase the accuracy of homologues software. [7] Focuses on that type of plagiarism which related to different equations present in different documents. Here different equations are extracted from different documents and then analyze them to find out the similarity between those equations. In this paper we will present a technique by which we will extract the keywords present in the suspected documents and then compare those keywords with the keywords of multiple documents considering the synonym(s) of each keyword and finally draw a conclusion whether there is a chance of plagiarism in the suspected document or not.

II. PLAGIARISM IN KEYWORD

Keyword is the most common feature present in research document. It is also known as Index Terms. Our idea is to find out the similarity between the keywords in the original and suspected document. As research papers are available in portable document format so at first document needs to be converted in image format. Next set of keywords will be extracted from this image. After extraction of the set of keywords, individual keywords will be separated. Next all the characters are separated from each keyword which is compared with character database to get the actual keyword in text format. Output of the operation may differ from the original text as al the operation was performed on scanned image. To get actual keyword text format of extracted keyword is compared with the database of keywords. After comparison, most similar keyword will be treated as the actual keyword. Now it may happen that to avoid plagiarism synonym of the original keyword may be used as a keyword in the suspected document. We consider this fact also. A database has been created consist of different synonym of the keyword. Once we got the actual keyword we will search the synonym database and extract the synonym(s) of the corresponding keyword. Now compare keywords of the original document and suspected document considering the synonym(s). If similarity found then it could be concluded that suspected document may be plagiarized.

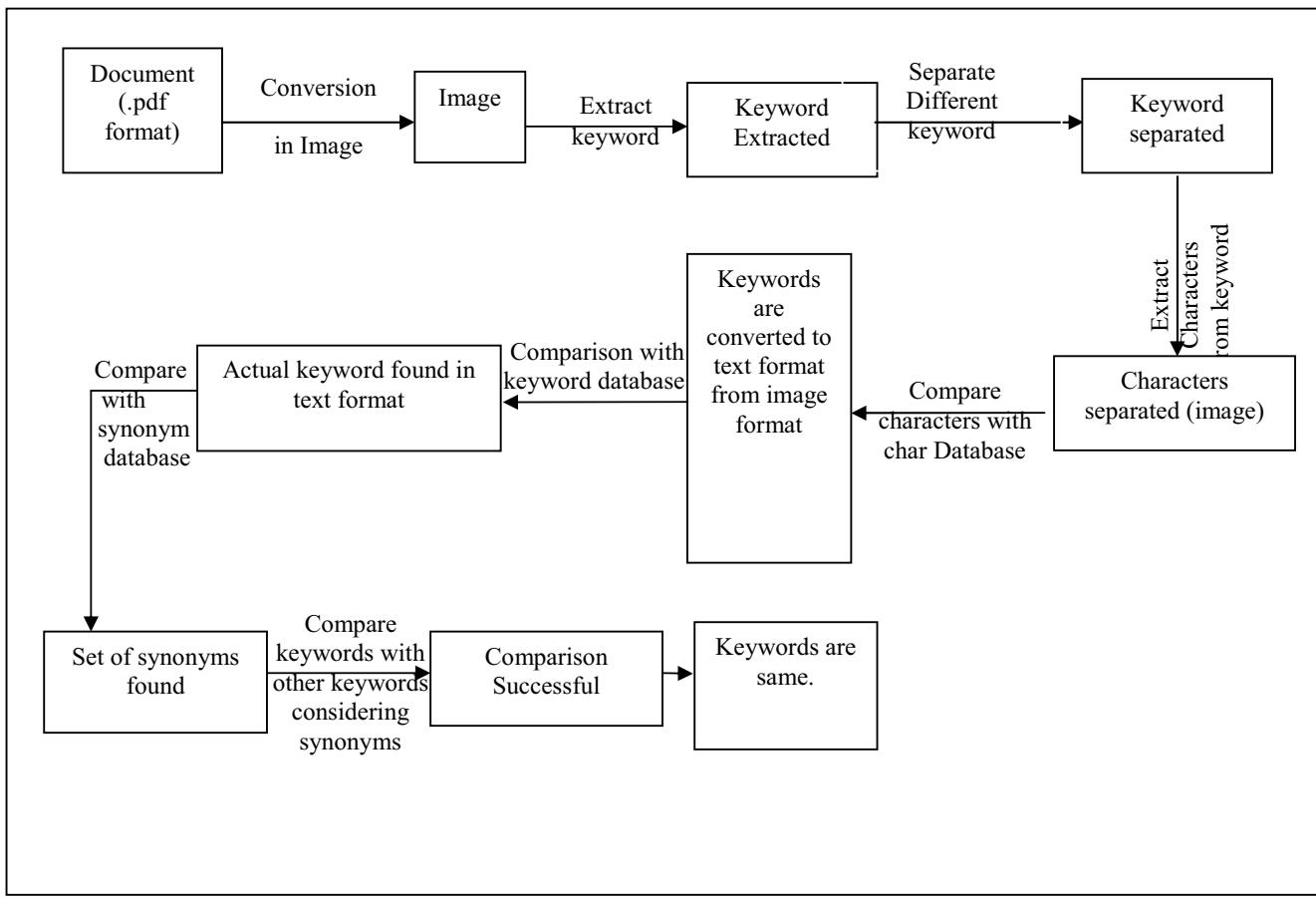


Fig 1: Block Diagram of the system

III. PROPOSED METHODOLOGY

A. Database Creation

As discussed in [7] there are certain fonts like Times New Roman, which are recommended to develop a research paper for conferences and journals. We also follow the same approach to develop a database for the purpose of character matching. That is we will create a database for A-Z, a-z. All the characters of the databases are collected by analyzing different research papers and all the characters in the database are in image format.

A database also has been created manually to store different words focusing mainly on technical words, so that after consulting this database, original keywords can be fetched from the partial keyword.

Another database for synonyms of keywords also created manually where different synonyms of different keywords are stored. It has been done to facilitate the comparison process of different keywords and their synonyms.

B. Scan

As all the documents are available in portable document format, so at the beginning first page of the document is converted in image format. The correctness of the result strongly depends upon the quality of the scanning. Sample Output is shown below.

ABSTRACT

In this effort, we propose a new image fusion technique, utilizing Empirical Mode Decomposition (EMD), for improved face recognition. EMD is a non-parametric data-driven analysis tool that decomposes non-linear non-stationary signals into Intrinsic Mode Functions (IMFs). In this method, we decompose images from different imaging modalities into their IMFs. Fusion is performed at the decomposition level and the fused IMFs are reconstructed to form the fused image. The effect of fusion on face recognition is measured by obtaining the Cumulative Match Characteristics (CMCs) between galleries and probes. Apart from conducting face recognition tests on visible and infrared raw datasets, we use datasets fused by averaging, principal component (PCA) fusion, wavelet based fusion and our method, for comparison. The face recognition rate due to EMD fused images is higher than the face recognition rates due to raw visible, raw infrared and other fused images. Examples of the fused images and illustrative CMC comparison charts are shown.

Index Terms—Face recognition, image analysis

1. INTRODUCTION

Fig 2: Sampled scanned image

C. Extracting the keywords

If we observe any research document there is a significant gap or whitespace between title and abstract of the research

paper. There is also a gap between the heading ‘Abstract’ and the first line of the abstract. There is another significant gap or whitespace between the last line of abstract and the keywords or index terms. We consider these facts. Start horizontal scanning of the entire image. Consider white pixels as separator. If there are consecutive white pixel blocks i.e. various numbers of consecutive rows of white pixels each of size of the columns of the image at a stretch then, there is an immense gap of white spaces. This type of the first gap is between title and abstract, and the second gap is between the heading ‘Abstract’ and the first line of the abstract, and the third is between the last line of abstract and keywords. By this methodology, keywords can be extracted from multiple documents.

Algorithm 1 extracts set of keywords and the sample output is shown below.

Algorithm 1:

Input: Document in image format.

Output: Keywords are separated.

```

1: Begin
2: Read image file.
3: Calculate the size(x y) of the image.
4: Discard the threshold value of pixels from the top.
5: Initialize flag=0;
6: Calculate upper bound and lower bound of a line.
7: Calculate pixel difference between two black lines.
8: Repeat
    8.1: Calculate row difference of two consecutive black line,
    say s=z-z1;
    8.2: IF (s>p+4) and flag=0 then
        8.2.1: a=z1;
        8.2.2: flag=1;
        8.2.3: END IF.
    8.3: If (s>p+4) and flag=1 then
        8.4: b=z1;
        8.5: END IF.
    8.6: Till End of document.
9: Crop image between a and b and store it.
10: END

```

Keywords—Face Recognition, Face Registration, Image Resolution

Fig 3: Output of Algorithm1

D. Separating every keyword

Once the set of keywords in a document has been separated next job is to separate every keyword. It is very clear that there are white spaces among every keyword. There is white space between each character of the word. But the size of the white space among every keyword is much greater

than the white space between each character of the keyword. We consider these facts and separate every keyword. Algorithm 2 has been used to separate all the keywords.

Algorithm 2:

Input: Set of keywords in image format.

Output: All the keywords are separated.

```

1: Begin
2: Read extracted keyword which is in image format.
3: Rotate the image 90 degree clockwise.
4: Calculate the size(x y) of the image.
5: Initialize i=1, j=1, p=0;
6: Repeat
    6.1: Start horizontal scanning.
    6.2: IF p=0 then
        6.2.1: Locate black pixel. Store the row value in a (say).
        6.2.2: Set p=1;
        6.2.3: END IF
    6.3: Locate the number of the white pixel say c provided that
    there is no black pixel in between.
    6.4: IF c>2*y then
        6.4.1: Store row value in b (say)
        6.4.2: Crop image between a and b and store it.
        6.4.3: Set c=0, p=0;
        6.4.4: END IF.
    6.5: Till the End of the Image File.
7: End

```

**network pattern fusion,
training**

Fig 4: Output of Algorithm2

Table I: Result for extracting the keyword

Document Name	Keyword present?	Keyword extracted?
1.jpg	Yes	Yes
2.jpg	Yes	Yes
3.jpg	Yes	Yes
4.jpg	Yes	Yes
5.jpg	Yes	No
6.jpg	Yes	Yes
7.jpg	Yes	Yes
8.jpg	Yes	No
9.jpg	Yes	Yes

E. Extracting every character

Next job is to separate every character from the extracted keywords. For this operation, we follow the same approach taken in [7] i.e. we use white space as a separator between two characters and once the number of whitespaces in a row is equal to the size of the column of the image then it

can be concluded that this white space is between two characters. By this methodology all the characters can be separated. Sample output is shown below.



Fig 5: Sample character after separating

F. White Space Removal

From the previous output, it is very clear that all the separated characters contain extra white space with it. If it is not removed, it will create a problem at the time of comparison with the database. For this reason, extra white space removal is required. Here also we will follow the approach taken in [7] to remove the white space. Sample output is shown below.

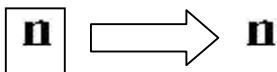


Fig 6: After removal of White Space

G. Comparison with Character Database

After successful completion of all the above processes next job is to convert the extracted character which is in image format to its equivalent text format. For future comparison this conversion is required. Following the methodology described in [7] all the characters are compared with the stored image in the database and after a successful comparison all the characters are converted to its equivalent image format. By this method all the keywords which are in image format are converted wholly or partially to its equivalent image format. Sample output is shown below.

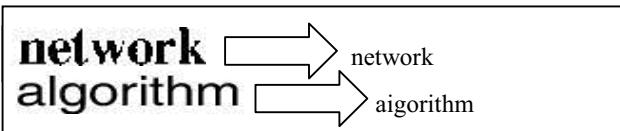


Fig 7: Conversion to Text from Image

H. Matching Methodology

1) Comparison with Keyword database

As all the previous operations are performed on the scanned image so all the characters may not be separated properly and as a result, text form of the keywords differs from the original one. Output from sec E and G confirms these facts. Here we develop a database which is the collection of words emphasizing those words used as keywords. Our next job is to compare this database with the extracted keywords and to find out the word(s) which are most similar. In this way, proper keywords are found and stored.

Algorithm 3 will compare the extracted keyword with the keyword database.

Algorithm 3:

Input: Text form of Extracted keyword.

Output: Corrected keyword.

```

1: Begin
2: Read the extracted keyword, say k1
3: Calculate the length of the keyword say, l1;
4: Repeat
   4.1: Read content of keyword database say, k2
   4.2: Calculate the length of k2, say l2.
   4.3: IF l1>l2 then
         4.3.1: l=l2;
         4.3.2: ELSE
         4.3.3: l=l2;
         4.3.4: END IF
   4.5: Set i=0, match=0, match1=0;
   4.6: Repeat
         4.6.1: IF k1[i]=k2[i] then
         4.6.2: match=match+1;
         4.6.2: i=i+1;
         4.6.3: END IF
         4.6.4: Until i=l;
   4.7: IF match > match1 then
         4.7.1:match1=match;
         4.7.2: word=k2;
         4.7.3: END IF
   4.8: Till the end of document.

```

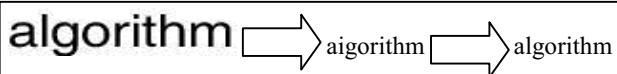


Fig 8: Output of Algorithm3

Table II : Result after comparing partially corrected keywords with database

Document Name	Number of extracted partially corrected keyword	Number of keyword corrected fully
1.jpg	3	3
2.jpg	4	4
3.jpg	4	3
4.jpg	7	5
6.jpg	4	4
7.jpg	4	4
9.jpg	14	12

2) Comparison with synonym database

As discussed earlier to avoid plagiarism synonym(s) of the keyword may be used. For that reason once keyword properly identified next job is to find out the synonym(s) of that particular keyword. A database has been developed with a collection of synonym(s) from where synonym will be searched and stored for further comparison.

Algorithm 4 will compare keyword with synonym database and store synonym(s).

Algorithm 4:

Input: Text from of corrected keywords.

Output: Synonym(s) of the corresponding keyword.

- 1: Begin
- 2: Read the keyword
- 3: Read the synonyms of the corresponding keywords.
- 4: Store the synonyms
- 5: End

Accuracy → correctness, accurateness, exactness, precision, truth

Fig 7: Output of Algorithm4

3) Comparison between different keywords considering Synonym

Now all the keywords with the synonym(s) are available. Compare it with each other to find the similarity between the keywords. If the similarity reaches a threshold value then, it can be concluded that there is a chance of plagiarism in the suspected document.

Algorithm 5 will compare different keywords considering synonym(s).

Algorithm 5:

Input: Keywords

Output: Similarity between keywords considering synonym(s).

- 1: Begin
- 2: Repeat.
 - 2.1: Read keyword (say k1) from document1.
 - 2.2: Repeat
 - 2.2.1: Read keyword (say k2) from document2
 - 2.2.2: IF k1=k2 then
 - 2.2.2.1: Similar word.
 - 2.2.2.2: END IF
 - 2.2.3: Till the end of the second document
 - 2.3: Till the end of the first document.

IV. RESULT

Name of the first document	Number of keyword present	Name of the second document	Number of keyword present	Number of plagiarized keyword present	Properly Identified
1	6	2	8	2	2
3	12	4	22	2	2
6	13	7	14	2	2

We have tested the proposed system with different research paper, and following is the result.

V. CONCLUSION

Keywords or Index terms are indispensable properties of a research paper. In this work, this property has been targeted to detect the plagiarism but this method has some drawbacks. As this process is dependent on the quality of the scan and therefore poor scanning would certainly lead to poor results. To yield excellent performance, the content of the database must be well enriched by the various possible contents. In future work, some online dictionary may be consulted by the system to make its working more robust.

REFERENCES

- [1] Detecting and Tracing Plagiarized Documents by Reconstruction plagiarism-Evolution Tree by Chang-Keon Ryu, Hyong-Jun Kim, Seung-Hyun Ji, Gyun Woo, and Hwan-Gue Cho
- [2] Understanding Plagiarism Linguistic Patterns, Textual Features and Detection Methods by Salha M. Alzahrani, Naomie Salim, and Ajith Abraham in IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, Vol. 42, No. 2, March 2012
- [3] Type Redefinition Plagiarism Detection of Token-Based Comparison by Lifang Han1, Baojiang Cui,Ru Zhang, Zhongxian Li, Jianxin Wang,Yongle Hao in 2010 International Conference on Multimedia Information Networking and Security
- [4] Toshihiro Kamiya,Shinji Kusumoto, and Katsuro Inoue. CCFinder: A multilingual token-based code clone detection system for large scale source code. IEEE Transactions on Software Engineering, 28(7):645–670, July 2002.
- [5] R. M. Howard Plagiarism: Some sources on attitudes, definitions, and detection methods, 2007
- [6] L. Guterman "Copycat articles seem rife in science journals, a digital sleuth finds", Chronicle of Higher Education, 2008 [online] Available: <http://chronicle.com/daily/2008/01/1362n.htm>
- [7] Plagiarism Detection by Identifying the Equations by Debotosh Bhattacharjee and Sandipan Dutta. International Conference on Computational Intelligence: Modeling Technique Techniques And Application, September 2013.