Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 234 (2024) 553–560

Procedia Computer Science

www.elsevier.com/locate/procedia

Seventh Information Systems International Conference (ISICO 2023)

Matching Scientific Article Titles using Cosine Similarity and Jaccard Similarity Algorithm

Tri Puspa Rinjeni[a], Ade Indriawan[a], Nur Aini Rakhmawati[a*]

[a]*Departemen of Information Systems, Institut Teknologi Sepuluh Nopember, Sukolilo, Surabaya 60111, Indonesia*

Abstract

This study compared various methods for academic article similarity matching. We employed two similarity algorithms, specifically Cosine and Jaccard Similarity. Moreover, these two similarity algorithms were combined with TF-IDF to increase the similarity results. We performed 16 experimental scenarios, measured based on the processing speed and accuracy, to compare the two algorithms. Highest similarity value in Cosine Similarity with circumstances without vectors, either with or without stemming. The findings showed that Cosine similarity performs better than Jaccard similarity.

Keywords: String Matching, Jaccard Similarity, Cosine Similarity, Research Title

1. Introduction

Finding relevant academic papers is a crucial task in the process of conducting scientific research reports and is one of the ways to enhance the quality of references. Owing to the vast number of papers published by researchers and publishers, this activity has become a time-consuming endeavor. Fortunately, numerous tools (e.g., Google Scholar, Publish or Perish) exist to obtain references from related studies using keywords. The keywords determined the search results using both the title and content of the reference. Although existing tools can help eliminate irrelevant references, the search results often remain excessive and require further examination to obtain research manuscripts. Apart from the selection of appropriate keywords, the search results were influenced by the algorithms employed by the tools. Various algorithms, including string matching, can be used to calculate the similarity values. The complexity of calculating similarity is the varying title structure: titles can be words, phrases, or sentences of varying lengths. Another reason is that titles that describe the same entity may differ syntactically [1].

* Corresponding author.
E-mail address: nur.aini@is.its.ac.id

10.1016/j.procs.2024.03.039

Selecting a suitable string matching algorithm for modern applications is a challenging task [2]. To accomplish this, it is essential to compare the algorithms used to execute the matching process. Based on previous research, some studies have succeeded in applying algorithms to text comparison. Wahyuningsih conducted text mining to compare students' short answers with answer keys on an exam. The study used three algorithm comparisons: Cosine Similarity, Jaccard Similarity and Dice's Coefficient. The results show that Cosine has a higher average correlation than Jaccard. [5]. Rakhmawati compared two algorithms, namely the Euclidean distance and Cosine similarity, to detect halal products [6]. Similarly, it found similarities in foodstuffs based on conceptual and textual similarities [7]. Jaccard Similarity is a *two-set string* matching algorithm that focuses on the ratio between the intersection of a match and the combination of *the strings* contained in the two *sets*. Cosine Similarity is a vector-based method [4]. The similarity algorithm has been used extensively in previous studies, and can be applied to the reference search process. This study aimed to compare Jaccard and Cosine similarity in paper titles to optimize the search process on reference-search process.

2. Methodology

This study was divided into four stages. Fig. 1 summarizes the methodology used in this study. First stage in the data collection, the data are the search results of 100 papers on Google Scholar using the Publish or Perish with the keyword "Gamification Education." Data retrieval results were stored as comma-separated values (CSV). The data used for matching were the titles of the papers. The initial stage before performing similarity calculations is pre-processing the data, such as case folding, removing symbol characters, tokenizing, stop words, and stemming. Case folding aims to convert uppercase letters into lowercase letters in a document and to delete character symbols[8]. Lowercased data are then stripped of symbol characters, commas, colons, and question marks. The deletion of one of them, comma (,), served to tidy the data. [9]. Tokenization is the process of separating text into smaller units called tokens, one of which is from sentence to word[10]. The main function of a tokenizer is to divide textual data into tokens such that they can be applied to morphological analysis [11]. Tokenization is performed using delimiters such as colons, semicolons, spaces, and punctuation marks [12]. Stopwords are the process of filtering out words that have no meaning. The process of stop words produces nouns from the data [13]. In a study using stop words in English with the addition of several keywords, such as gamification and education. A comparison of the results of the titles that were tokenized with titles after the stopword process is shown in Table 1. Stemming is the process of obtaining the basic word for each word/token [14]. Porter stemming was used in this study. Porter is a stemming algorithm based on rule-based Stemmers [15]. The processed data are shown in Table 1.

3. Similarity Measure

Experiments were conducted to calculate similarity values using Jaccard and Cosine similarity for each scenario and algorithm. The experiment was conducted using the Python programming language on Google ColLab. Google Collab is a service for executing cloud-based Python programs aimed at researchers[16]. Pythons can efficiently integrate systems and can be used for big data analysis[17].

3.1. Cosine Similarity (CS)

The cosine similarity (CS) algorithm has evolved from the Euclidean distance technique for comparing two strings. The more words in the two papers that are the same in the Euclidean distance calculation, the more similar are the two documents. This strategy is usually incorrect because two documents that share many terms are not always on the same topic. This is a flaw in the Euclidean distance method [18]. The CS algorithm computes the distance between two objects. Method calculations are often based on vectors [4]. The angular cosine between two vectors projected in a multidimensional field was used to calculate the similarity between the two papers using the CS method. The two vectors in question are collections of words (in the form of arrays) from the two documents under consideration. Mathematically, the CS algorithm is expressed by Equation(1) [4], where *ai* and *bi* are the components of each vector. The Cosine similarity values are in the range of 0-1.

$$cos\theta = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\sqrt{\sum_1^n b_i^2}} \quad (1)$$

In this study, vector weighting was employed to calculate the cosine similarity in vector form using Term Frequency-Inverse Document Frequency (TF-IDF) to calculate the CS. TF in TF-IDF refers to the appearance of specific words in a document; words with a high TF value are relevant in a document [13]. A word is the result of multiplying the TF and IDF[19].

3.2. *Jaccard Similarity* (JS)

The Jaccard similarity (JS) algorithm is an algorithm of the q-gram class [1]. This technique determines the similarity between two things (items). The calculations in this approach, such as the distance cosine and matching coefficient, are generally based on the similarity in the size vector space [4]. The JS formula is shown in Equation (2) [20]. The idea behind this algorithm is that the sequence of characters is as essential as the character itself [20]. In two documents, the JS method computed the sum of two successive items (bigrams). The JS algorithm then divides the number of bigrams by the unique total bigrams in the two documents to generate a similarity value. The JS algorithm is expressed by Equation (2), where *s* and *t* are two string sets measured in terms of similarity. The Jaccard value ranges from 0 to 1. Algorithms that fall into *the q-grams* class, including JS, work well for troubleshooting *typographical errors* but are unable to measure *string* matches if the order changes.

$$Jac(s,t)_{token} = \frac{|s \cap t|}{|s \cup t|} \quad (2)$$

3.3. *Experiment Scenario*

The scenario was separated into two algorithms, each of which was divided into two conditions: non-vector and vector. Each of the two non-vector and vector conditions was subdivided into four subconditions: without stopwords without stemming, with stopwords without stemming, without stopwords with stemming, and with stopwords with stemming. The scenario details are presented in Table 2.

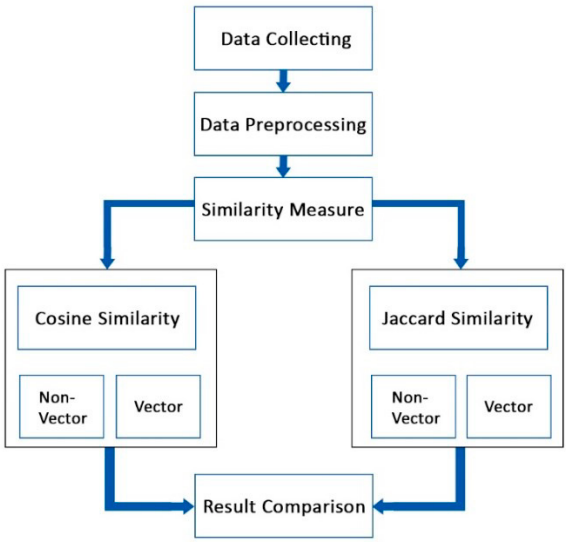

Fig. 1. Methodology

Table 1. Example of Case Folding.

Title	Title after Case Folding	Title after Tokenize	Title after Stopwords	Title after Stemming
Gamification in science education. A systematic review of the literature	gamification in science education a systematic review of the literature	literature in gamification science the systematic reviews of a education	literature science the systematic reviews	scienc systemat literature reviews

Table 2. Experiments Scenario.

No	Experiment	Non-Vector	Vector	Stop words	Stemming	Jaccard	Cosine
1	J	√				√	
2	JStop	√		√		√	
3	JStem	√			√	√	
4	JStopStem	√		√	√	√	
5	JV		√			√	
6	JVStop		√	√			
7	JVStem		√			√	
8	JVStopStem		√		√	√	
9	C	√					√
10	CStop	√		√			√
11	CStem	√			√		√
12	CStopStem	√		√	√		√
13	CV		√				√
14	CVStop		√	√			√
15	CVStem		√		√		√
16	CVStopStem		√	√	√		√

4. Results and Discussions

The accuracy result is calculated based on a similarity value greater than 0.5 (≥0.5) with manual matching. Therefore, we assign 0.5 as a threshold value. Subsequently, the percentage of accurate data compared to the amount of data that matched the threshold value was calculated. The accuracy results are presented in Table 3. Subjective accuracy calculations were performed by comparing the two original titles calculated with their similarity values for each situation. Ratings were given on a scale of 0 to 5, with 0 indicating incorrect and 5 indicating very accurate. Both approaches measure accuracy in non-vector scenarios because it is expected that the match values of the calculation results for the vector and non-vector scenarios do not differ considerably.

Table 3. Similarity Accuracy.

No	Experiment	Number of Data Pairs	Accurate Amount of Data	Percentage (%)
1	J	20	20	81
2	JStop	4	4	95
3	JStem	21	21	82
4	JStopStem	4	4	90
5	C	122	122	79
6	CStop	23	23	92
7	CStem	153	153	89
8	CStopStem	26	26	93

As shown in Table 3, the largest data pair is the CStem experiment, which is a similarity calculation experiment employing Cosine similarity with stemming. More data pairs are generated by vector scenarios utilizing Cosine Similarity than by the other scenarios. In the JStop and JStopStem experiments, the data pair with the least amount. JStop is a stopword-based experiment based on Jaccard similarity. JStopStem, on the other hand, is an experiment that uses Jaccard Similarity with stopwords and stemming. Aside from the amount of data pairings, the percentage of correctness is also crucial. According to the accuracy results, Jaccard Similarity with stopwords has the greatest value of 95%. The experimental accuracy is shown in Fig.2.

The average execution time from 16 trials was in milliseconds. Pre-processing situations involving the two algorithms were compared. For example, in a comparison condition with no vectors, the compared scenario is scenario J, and the average time for Jaccard similarity is 21,680 and 68.052 for cosine similarity. The calculation of the identical scenario takes longer when cosine similarity is used. The minimum and maximum times required to determine similarity using Jaccard Similarity are 17,907 ms and 74,227 ms, respectively, in Table 4. In vector circumstances, the time required for Jaccard Similarity performance is longer. The average of the full Jaccard Similarity time was 44.448 ms when calculated. The shortest and longest times required to calculate similarity using Cosine Similarity were 12.952 and 70.402 ms, respectively. In non-vector settings, the CS algorithm takes longer to calculate than in vector scenarios. The entire calculation time of the CS algorithm was 36.624 ms. The average difference in overall time between the JS algorithms and JS algorithms was 7,824 ms. Table 4 displays the results of the experiment's computation time.

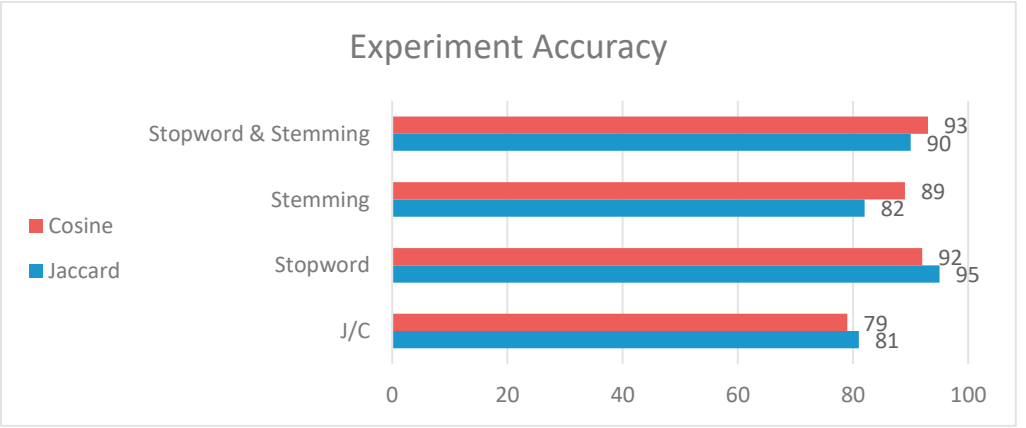

Fig. 2. Experiment Accuracy.

Table 4. 1of each scenario in milliseconds.

No	Experiments (Jaccard, Cosine)	Jaccard Time	Cosine Time
1	No Vectors (J,C)	21,680	68,052
2	No Vectors with Stopwords (JStop, CStop)	18,773	51,774
3	No Vectors with Stemming (JStem, CStem)	21,620	70,402
4	No Vectors with Stopwords and Stemming (JStopStem, CStopStem)	17,907	49,460
5	Vector (JV,CV)	70,193	12,952
6	Vector with Stopwords (JVStop, CVStop)	65,407	13,287
7	Vectors with Stemming (JVStem, CVStem)	65,407	13,755
8	Vectors with Stopwords and Stemming (JVStopStem, CVStopStem)	74,227	13,312
Average		44,448	36,624

All similarity results from each scenario were compiled into an Excel file for descriptive analysis. Mean, standard deviation, and minimum and maximum descriptive analytic calculations were used in this investigation. In addition, quartile values of 1 (25%), 2 (50%), and 3 (75%) were generated to evaluate how the data were distributed. Each scenario yielded 4950 title pairs out of 100 titles. The largest median or average was demonstrated in the CStem experiment, which is a non-vector stemmed cosine similarity experiment. The JStop and JVStop tests produced the lowest mean values. Stopwords were employed in preprocessing in both studies. The experiment with the smallest standard deviation was experiment J, which employed Jaccard similarity calculations on non-vectors without employing stop words or stemming. The CStem experiment had the highest standard deviation. The median value in some scenarios is 0.000, indicating that the results are also supported by the minimal similarity value of the entire study, which is worth 0, indicating that all of the titles studied have nothing in common. The C and CStem trials showed the highest similarity (0.800). The CStem experiment had the highest value for each calculation, according to the tie of descriptive analysis data. The total number of calculations based on a more thorough descriptive study is presented in Table 5.

Table 5. Summary of descriptive statistical analyses of the match values for each scenario.

Experiment	Count	Mean	Std	Min	25%	50%	75%	Max
J	4950	0,159	0,001	0	0,105	0,150	0,200	0,667
JStop	4950	0,023	0,053	0	0	0	0	0,500
JStem	4950	0,164	0,076	0	0,111	0,158	0,200	0,667
JStopStem	4950	0,028	0,057	0	0	0	0,053	0,500
JV	4950	0,157	0,074	0	0,105	0,150	0,200	0,647
JVStop	4950	0,023	0,054	0	0	0	0	0,500
JVStem	4950	0,162	0,076	0	0,107	0,154	0,200	0,647
JVStopStem	4950	0,028	0,058	0	0	0	0,053	0,500
C	4950	0,273	0,107	0	0,199	0,269	0,338	0,800
CStop	4950	0,042	0,091	0	0	0	0	0,671
CStem	4950	0,281	0,109	0	0,202	0,276	0,350	0,800
CStopStem	4950	0,051	0,098	0	0	0	0,102	0,671
CV	4950	0,071	0,065	0	0,027	0,054	0,092	0,676
CVStop	4950	0,024	0,060	0	0	0	0	0,610

Experiment	Count	Mean	Std	Min	25%	50%	75%	Max
CVStem	4950	0,077	0,069	0	0,030	0,059	0,102	0,682
CVStopStem	4950	0,029	0,065	0	0	0	0,037	0,632

A boxplot diagram was used to depict the general distribution. Fig. 3 shows the distribution of the 16 scenarios. Four of the 16 scenarios show the highest similarity values of 0 in the JStop, JVStop, CStop, and CVStop scenarios. All four scenarios used pre-processing stopwords in both nonvector and vector forms. This proves that the title compared contains a lot of words in stopwords as well as gamification and education. Experiments C and CStem show similarity values whose middle values are greater than or equal to those of previous experiments or can be assigned to quartile 2. The top limit values of C and CStem are the same, whereas CStem has a slightly higher median value. The Cosine similarity has the weakest value in vector experiments. Both scenarios had a maximum value in the third quartile. The majority of the tests were in quartile 1, specifically J, JStem, JStopStem, JVStem, JVStopStem, CStopStem, CV, CVStem, and CVStopStem. As can be observed in the experimental situation, all stemmed scenarios are in quartile 1, as shown in Fig. 3.

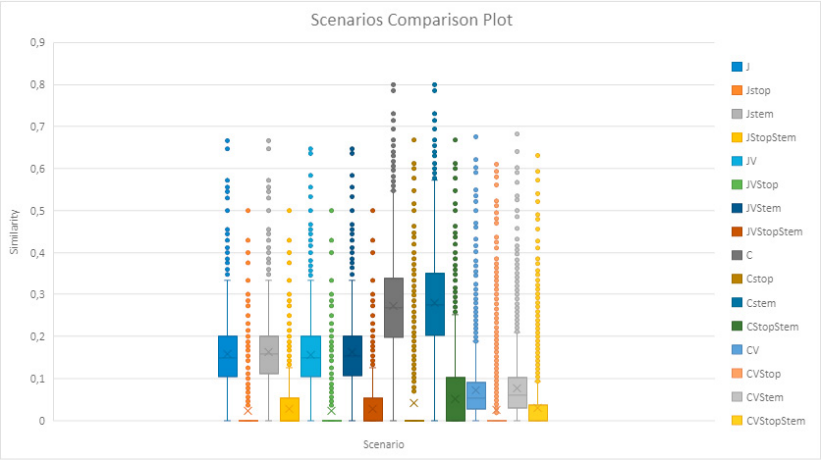

Fig. 3. Boxplot of the match value range for each scenario.

5. Conclusion

Sixteen experimental scenarios were run for the calculation of Cosine and Jaccard Similarity. Based on the experimental results, the Cosine Similarity method shows the highest results by a value of 0.8, with circumstances without vectors, either with or without stemming. This is because Cosine similarity is dependent on the angle of the vector created by the document's term. According to the average time of trials, cosine similarity requires up to 15 times less time than Jaccard similarity for determining the value of similarity.

The execution of the two Cosine and Jaccard algorithms in preset circumstances reveals that Jaccard algorithms perform a more rigorous match calculation than Cosine algorithms. This is consistent with the nature of cosine algorithms, which are not affected by the size of existing data [18]. This was initially the advantage of the cosine algorithm over the Euclidean distance algorithm. However, similar characteristics are not always advantageous. Because of these qualities, two string patterns with differing attribute values can have a high match value. Jaccard similarity requires more computation time than cosine algorithms for the same dataset and circumstances but has simpler calculations. While assessing accuracy, the cosine algorithm produces a more accurate accuracy value than the Jaccard method, but the differences are small. Based on the experimental results, it can be considered that in the reference search environment, an algorithm that suits the publisher's needs is chosen. This study only compared the two algorithms in terms of accuracy and time. For further research, new aspects, such as comparing the content or substance of the paper, can be explored rather than only from the title. Given the substantial amount of content in the manuscript, more algorithms will be explored in experiments.

Acknowledgements

The authors gratefully acknowledge financial support from the Institut Teknologi Sepuluh Nopember for this work, under project scheme of the Publication Writing and IPR Incentive Program (PPHKI) 2023.

References

[1] Gali N, Mariescu-Istodor R, Fränti P. Similarity measures for title matching. Proc - Int Conf Pattern Recognit 2016;0:1548–53. https://doi.org/10.1109/ICPR.2016.7899857.

[2] Hakak SI, Kamsin A, Shivakumara P, Gilkar GA, Khan WZ, Imran M. Exact String Matching Algorithms: Survey, Issues, and Future Research Directions. IEEE Access 2019;7:69614–37. https://doi.org/10.1109/ACCESS.2019.2914071.

[3] Guo S, Jiang X, Thornton T, Saunders D. Approximate String Matching of Power System Substation Names. 2019 IEEE Power Energy Conf Illinois, PECI 2019 2019. https://doi.org/10.1109/PECI.2019.8698905.

[4] Pikies M, Ali J. String similarity algorithms for a ticket classification system. 2019 6th Int Conf Control Decis Inf Technol CoDIT 2019 2019:36–41. https://doi.org/10.1109/CoDIT.2019.8820497.

[5] wahyuningsih T. Text Mining an Automatic Short Answer Grading (ASAG), Comparison of Three Methods of Cosine Similarity, Jaccard Similarity and Dice's Coefficient. J Appl Data Sci 2021;2:45–54. https://doi.org/10.47738/jads.v2i2.31.

[6] Rakhmawati NA, Firmansyah A, Effendi PM, Abdillah R, Cahyono TA. Auto halal detection products based on euclidian distance and cosine similarity. Int J Adv Sci Eng Inf Technol 2018;8:1706–11.

[7] Rakhmawati NA, Jannah M. Food Ingredients Similarity Based on Conceptual and Textual Similarity. Halal Res J 2021;1:87–95. https://doi.org/10.12962/j22759970.v1i2.107.

[8] Hamidah N, Yusliani N, Rodiah D. Spelling Checker using Algorithm Damerau Levenshtein Distance and Cosine Similarity. Sriwij J Informatics Appl 2020;1:22–5.

[9] Pratama RP, Hidayat R, Fano NF, Akbar A, Rakhmawati NA. Implementasi Deep Learning untuk Entity Matching pada Dataset Obat (Studi Kasus K24 dan Farmaku). JISKA (Jurnal Inform Sunan Kalijaga) 2021;6:130–8. https://doi.org/10.14421/jiska.2021.6.3.130-138.

[10] Song X, Salcianu A, Song Y, Dopson D, Zhou D. Fast WordPiece Tokenization 2021:2089–103. https://doi.org/10.18653/v1/2021.emnlp-main.160.

[11] Vasiu MA, Potolea R. Enhancing Tokenization by Embedding Romanian Language Specific Morphology. Proc - 2020 IEEE 16th Int Conf Intell Comput Commun Process ICCP 2020 2020:243–50. https://doi.org/10.1109/ICCP51029.2020.9266140.

[12] Gupta G, Kumar N, Chhabra I. Optimised Transformation Algorithm For Hadoop Data Loading in Web ETL Framework. EAI Endorsed Trans Scalable Inf Syst 2020;7:1–8. https://doi.org/10.4108/eai.13-7-2018.160600.

[13] Kim SW, Gil JM. Research paper classification systems based on TF-IDF and LDA schemes. Human-Centric Comput Inf Sci 2019;9. https://doi.org/10.1186/s13673-019-0192-7.

[14] Rianto, Mutiara AB, Wibowo EP, Santosa PI. Improving the accuracy of text classification using stemming method, a case of non-formal Indonesian conversation. J Big Data 2021;8. https://doi.org/10.1186/s40537-021-00413-1.

[15] Haroon M. Comparative Analysis of Stemming Algorithms for Web Text Mining. Int J Mod Educ Comput Sci 2018;10:20–5. https://doi.org/10.5815/ijmecs.2018.09.03.

[16] Akbar A, Fano NF, Informasi JS, Nopember S. Deep Learning Untuk Entity Matching Produk Kamera Antar Online Store Menggunakan 2021;7:1–5.

[17] Qurashi AW, Holmes V, Johnson AP. Document Processing: Methods for Semantic Text Similarity Analysis. INISTA 2020 - 2020 Int Conf Innov Intell Syst Appl Proc 2020. https://doi.org/10.1109/INISTA49547.2020.9194665.

[18] Xia P, Zhang L, Li F. Learning similarity with cosine similarity ensemble. Inf Sci (Ny) 2015;307:39–52. https://doi.org/10.1016/j.ins.2015.02.024.

[19] Yuan H, Tang Y, Sun W, Liu L. A detection method for android application security based on TF-IDF and machine learning. PLoS One 2020;15:1–19. https://doi.org/10.1371/journal.pone.0238694.

[20] Diana NE, Hanana Ulfa I. Measuring performance of n-gram and jaccard-similarity metrics in document plagiarism application. J Phys Conf Ser 2019;1196. https://doi.org/10.1088/1742-6596/1196/1/012069.