2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)

2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM) | 978-1-6654-1407-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICSECS52883.2021.00121

Comparison of document similarity algorithms in extracting document keywords from an academic paper

M. Saef Ullah Miah
Faculty of Computing
College of Computing and Applied Sciences
Universiti Malaysia Pahang
26600 Pekan, Malaysia
md.saefullah@gmail.com

Junaida Sulaiman
Center for Data Science and Artificial Intelligence (Data Science Center)
Faculty of Computing
College of Computing and Applied Sciences
Universiti Malaysia Pahang
26600 Pekan, Malaysia
junaida@ump.edu.my

Saiful Azad
Department of CSE
Green University of Bangladesh
Dhaka, Bangladesh
saiful@cse.green.edu.bd

Kamal Z. Zamli
Faculty of Computing
College of Computing and Applied Sciences
Universiti Malaysia Pahang
26600 Pekan, Malaysia
kamalz@ump.edu.my

Rajan Jose
Faculty of Industrial Sciences & Technology
Universiti Malaysia Pahang
Gambang, Malaysia
rjose@ump.edu.my

***Abstract*—The idea of this study is to validate a list of keywords derived from a scientific article by a domain expert from years of knowledge with prominent document similarity algorithms. For this study, a list of handcrafted keywords generated by Electric Double Layer Capacitor (EDLC) experts are chosen, and relevant documents to EDLC are considered for the comparison. Then, different similarity calculation algorithms were employed in different settings on the documents such as using the whole texts of the documents, selecting the positive sentences of the documents, and generating similarity score with automatically extracted keywords from the documents. The experiment's outcome provides us with findings that the machine-generated keywords are mostly similar to the curated list by the domain experts. This study also suggests the preferable algorithms for similarity calculation and automated key-phrase extraction for the EDLC domain.**

***Keywords*—Document similarity calculation, Relevant Document Selection, keyword extraction comparison, keyword validation, Electric Double Layer Capacitor, EDLC, Keyword Based Recommendation System**

I. Introduction

With the rise of Industrial Revolution 4.0 (IR 4.0) [1], material science also experiencing Materials 4.0 [2] where Material Informatics [3] plays a vital role. Materials 4.0 and IR 4.0 both focus on automation of production process and automatic selection of materials for production process [4], [5]. One way to know about the production processes and materials used in different processes is to go through the scientific papers published in those domains. However, there is a catch that many documents on specific domains are published each year. According to the Scopus database, the total number of published documents in the material science domain in the last 20 years (2000-2020) is 357429, which is around 17000 articles per year [6], which is a vast number to process by a human. That is why machine learning techniques are implemented to overcome this barrier. However, another problem arises here: whether all the papers are appropriate or closely related to the topics required for the information people are looking for. If the papers are not closely related to the topic, then it is a complete waste of time and computation power for the machine to work on those papers. So, it is important to find the relevant papers to the domain or processes that are being targeted. Topic modeling can be a solution for this problem [7], [8], research shows that topic modeling is very suitable for scientific literature where the number of topics is clearly mentioned. However, there are too many parameters to be tuned and predetermined input. Moreover, the output should be human intervened unless a concrete document topic theme is provided or established [9], [10]. In this situation, a one- shot human-guided approach can help machines understandingthe relevance of a document to a specific domain based on domain expert-curated keywords [11]. There are many keywords extraction techniques that extract top keywords from text based on different parameters [12], [13]. It could be an interesting experiment to observe how similar or different these techniques extract keywords to those curated by the domain experts. In other words, it can be experienced whether the machine-generated keywords are as feasible as domain expert- curated keywords to find relevant documents for any specified domain.

Therefore, in this paper, the similarity between domain expert-curated keywords and machine-generated keywords from text using different key-phrase extraction techniques are

978-1-6654-1407-4/21/$31.00 ©2021 IEEE
DOI 10.1109/ICSECS52883.2021.00121

measured to validate the domain expert-curated keywords to select relevant documents for any specified domain.

The rest of the subsequent sections of this paper are organized as follows. The employed key-phrase extraction techniques and relevant other works are discussed in Section II with their known limitations and features. The details of the methodology employed in this paper are mentioned in Section III. Afterward, the analysis and results of the experiment is discussed in Section IV. This paper ends with concluding remarks in Section V.

II. Background Study

This study employs some prominent similarity calculation algorithms and key-phrase extraction algorithms. All the algorithms with their best features and limitations are discussed in this section.

A. Text Similarity Calculation

This study is being carried out on scientific papers of any specific domain, which are in textual data format. Therefore, text similarity calculation algorithms are taken into consideration. Text similarity algorithms generally determine how close two pieces of text are both in lexical similarity and semantic similarity. Many works have been taken place on text similarity calculation on different text data and different settings. However, all of them have some algorithmsin common, and they are Jaccard similarity, Cosine similarity, and Cosine similarity with word vector approach [14]–[19]. Hence, in this work, these standard similarity algorithms are employed as their capabilities are already proven.

1) Jaccard Similarity: It measures the similarity for the two sets of data, with a range from 0% to 100%. The higher the value from this range, the more similar the two sets. Jaccard similarity is extremely sensitive to small sample sizes and may give erroneous results, especially for tiny samples. On the other hand, it is computationally costly to calculate a large sample size [20], [21]. Jaccard similarity is calculated using the equation 1, where A and B are two different sets of text or two documents.

$$J(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|} = \frac{|A \bigcap B|}{|A| + |B| - |A \bigcup B|} \qquad (1)$$

2) Cosine Similarity: It is a measure to determine how similar two different documents are, disregarding their size. This method measures the cosine of the angle between two multi-dimensional vectors in a multi-dimensional space. It is a dot product of two vectors or two documents converted into vectors, which are originally arrays of word counts of these two documents representation divided by the product of two vectors' magnitude. Cosine similarity is computationally very expensive for a large number of data [22], [23]. Cosine similarity is calculated using the equation 2, where A and B are the vector representation of two documents.

$$\cos(A,B) = \frac{AB}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2}\sqrt{\sum_{i=1}^{n} (B_i)^2}} \qquad (2)$$

3) Word Vector: Word vectors are a type of word representation which arrange words with similar meaning to posses similar representation. Word vectors are a class of techniques where individual words are represented as real-valued vectors in a predefined vector space, and each word is mapped to one vector [24]. In a different word, words or text data gets areal mathematical value in which all sorts of mathematical calculations can be done. Word2vec [25] is such a model which produces the vectors. It obtains the word association from a large text corpus and store as a pre-trained model. This sort of pre-trained model can detect synonymous words or semantically similar words, increasing the performance on detecting similarity. In the cosine similarity algorithm, it uses arrays of words, but while using cosine similarity with word vector, it can also calculate the semantic similarity. As a result,it increases the performance in detecting similarities between provided documents.

B. Key-phrase Extraction

The key-phrase extraction is another essential part of this study. As this study compares the domain expert-curated key-words for any specific domain, it is obvious to get key-phrases or keywords from the documents to compare with. Many pieces of research have been done on key-phrase extraction techniques and their performance evaluation [26]–[29]. From all those techniques, a mixture of statistical, graph-based, supervised, and unsupervised techniques are employed in this study.

1) Yake: The complete form of yake is Yet Another Key-word Extractor. It is an unsupervised statistical approach for automatic keyword extraction from text data. It does not require training, and it has no dependency on dictionaries, external-corpus, size of the text, language, or any specific domain [30].

2) TopicRank: It is a graph-based unsupervised keyword extraction technique that relies on the topical representation of the document. In this technique, candidate key-phrases are grouped into topics and utilized as vertices in a graph. Each topic is assigned a significance score based on a graph ranking model. Key-phrases are generated by selecting candidate key-phrases from each top-ranked topics [31]. It performs better than TF-IDF, TextRank, and SingleRank for four different data-sets; hence it is employed in this study.

3) MultipartiteRank: MultipartiteRank is also an unsupervised graph-based keyword extraction technique that encodes topical information within a multipartite graph structure. This technique represents candidate key-phrases and topics in asingle graph and utilizes their mutually reinforcing relation- ship to improve candidate ranking. As a result, most of the time, it outperforms different other key-phrase extraction techniques [32].

4) KPMiner: It is an unsupervised statistical method for keyword extraction. This method does not require any pre-training. It has a flexible option of configuring rules and heuristics based on the nature of the document [33].

III. Methodology

The experiment is divided into three major phases: data collection, data processing, and similarity calculation. The similarity calculation phase is divided into several other portions like calculating Jaccard similarity, cosine similarity, and cosine similarity with word vector for the document's whole document text and positive sentences text. For the similarity calculation phase, key-phrase extraction techniques are also used, besides using the whole document. The idea is to calculate the similarity between the extracted keywords from the documents and the provided list of the keywords curated by domain experts, domain experts from the Electric Double Layer Capacitor (EDLC) domain. To start with the experiment, scientific papers related to the EDLC domain are collected and converted into plain text documents using the data processing method. After that, similarity calculation is done with different settings for both positive sentences and whole document and keywords extracted with different key-phrase extraction techniques with the domain expert-curated keyword list. An overview of the employed methodology is depicted in Figure 1.

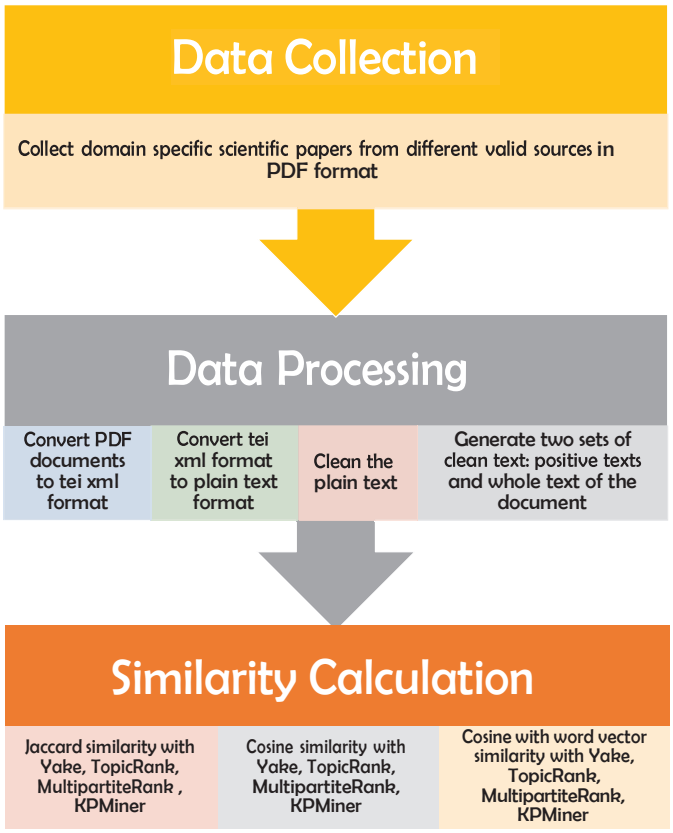

Fig. 1. Overview of the employed methodology

A. Data Collection

As this study considers the EDLC as the domain, a list of keywords related to the technology of EDLC is curated by the domain experts. Two primary types of keywords can be achieved for the EDLC domain. One is the science or fundamental knowledge of EDLC like what it is, how it works, and another one is the technology of EDLC, for example, how an EDLC mechanism can be produced, what are the materials that are being used in an EDLC system. For this study, the technology-based keywords list is chosen as domain expert-curated keywords. Keywords are listed in Table I. After obtaining the curated keywords list, related documents are collected, which are scientific papers published in different renowned journals in portable document format (PDF). Two sets of these documents are collected. One set contains 10 documents which contains the keywords in their text and is provided by the domain experts as a standard set. And another set contains 190 documents where some of them have all the keywords and some of them have some of the keywords, and some do not contain any keywords from the list of curated keywords. The purpose of these two sets of documents is to observe and validate the score difference and similarities between two sets, where one set contains all the keywords from the curated list in their text. And another set contains a different number of keywords or no keywords from the curated list of keywords.

TABLE I. Domain Expert Curated Keywords for EDLC Domain

Sl	Keyword	Sl	Keyword
1	Supercapacitors	17	Graphene
2	(SCs)	18	Graphite oxide
3	Electrochemical capacitors	19	(GO)
4	Energy storage device	20	Reduced graphite oxide
5	Electric double layer capacitor	21	(rGO)
6	(EDLC)	22	Surface charge accumulation
7	Pseudocapacitance	23	High power applications
8	Electrostatic adsorption	24	Charge separation at electrode interface
9	Electrosorption	25	Charge separation at electrolyte interface
10	Faradaic redox reactions	26	Non-faradaic process
11	Stern layer	27	Specific surface area
12	Helmholtz double layer	28	Pore size distribution
13	Double layer formation	29	Electrochemical interface
14	Activated carbon	30	EDLC characteristics
15	Porous carbon	31	Diffuse double layer
16	Carbon nanotubes	32	Polarizable capacitor electrode

B. Data Processing

As the documents are collected in portable document format, they need to be converted into plain text format. To convert these pdf files to plain text format, the pdf files are initially converted to tei xml format using grobid [34]. A custom tei xml parser is developed using Python programming language to convert those tei xml files to plain text format. After converting to plain text format, each document is cleaned using custom data cleaning technique. This includes removing double spaces, removing parentheses, replacing

garbage characters using the appropriate characters, removing extra spaces between digits and letters, and removing square brackets. Within the cleaned text, the negative sentences are separated using negatives and negation–grammar Rules [35]–[37] and discarded from the text. Positive sentences are stored as a set of positive sentences. From the data processing phase, two sets of plain text data are obtained, one contains the cleaned whole text of a document, and another one contains the positive sentences of a document.

C. Similarity Calculation

Two sets of processed data are obtained from the data processing phase for each document. One is cleaned whole text of the document, and another is only positive sentences of the document. For each set of processed data, three prominent similarity calculation techniques, namely, Jaccard, Cosine, and Cosine with word vectors similarity, are calculated for the extracted keywords, extracted by five prominent key-phrase extraction techniques, namely, Yake TopicRank, MultipartiteRank, and Kpminer with the domain expert-curated keywords. All the scores obtained by these algorithms are stored in a tab- separated value format file. In short, the following operations are executed for two sets of plain text data of a document,

- Jaccard similarity calculation of all the keywords obtained by Yake, TopicRank, MultipartiteRank, and Kpminer individually with the domain expert-curated keywords for two sets of text.
- Cosine similarity calculation of all the keywords obtained by Yake, TopicRank, MultipartiteRank, and Kpminer individually with the domain expert-curated keywords for two sets of text.
- Cosine similarity with word vectors calculation of all the keywords obtained by Yake, TopicRank, MultipartiteRank, and Kpminer individually with the domain expert- curated keywords for two sets of text.

D. Experimental Setup

In this experiment, all codes are developed using Python programming language version 3.7.3 [38]. Jaccard and Cosine similarity algorithms are developed following the equation described in their original papers [39], [40]. Cosine similarity with word vector algorithm is implemented using spacy Python library [41]. All key-phrase extraction algorithms are implemented using pke [42] Python library. The experiment is done in a MacBook with macOS Big Sur version 11.3.1 (20E241) with a 1.2 GHz dual-core intel core m5 processor and 8 gigabytes of RAM.

IV. Results and Discussion

For result analysis, standard documents are analysed to find the maximum and minimum score for all the similarity calculation algorithms using all the key-phrase extraction techniques mentioned earlier in this study in section III-C. From this analysis, it can be decided what should be the minimum threshold value to decide whether a document is related to a specific domain or not. This analysis also provides an insight into the selection of whole text or positive text while calculating the document similarity score for any specific domain. Two tables are provided below, Table II represents the analysis result of standard ten papers for whole text, and Table III represents the analysis result of standard ten papers for positive text only. Both tables represent the result for the EDLC domain. For this domain, from the analysis it can be said that cosine with word vector similarity algorithm with MultipartiteRank key-phrase extraction technique performs best. It scored about 94% for whole text and around 93% similarity for positive text. On the other hand, the Jaccard Similarity algorithm performs least in the experiment for all the key-phrase extraction techniques. It gives around 6% to 17% and 4% to 18% similarity score for whole text and positive text of the same document where cosine similarity with word vector algorithm with MultipartiteRank key-phrase extraction technique provides the best scores. Using Table II and Table III, Figure 2 and Figure 3 are generated. From the figures, it is clearly visible that the cosine with word vector similarity algorithm performs better than other two similarity calculation algorithms. The same trends from these ten documents are also found in the rest 190 documents.

From the results, the threshold value for machine-dependent relevant document selection for the EDLC domain can also be determined if domain expert-curated keywords are available. This threshold value is considered for use with the MultipartiteRank key-phrase extraction technique and cosine similarity with word vector algorithm, which is around 88% for minimum value and 100% is the maximum value.

Based on the results analysis and discussion in this section, it can be concluded that, for the EDLC domain, cosine with word vector similarity algorithm and automated unsupervised keyword-extraction, MultipartiteRank algorithm are superior to the other algorithms used in this study. Besides this study can also point to the fact that, these algorithms can also be used to produce recommended keywords from scientific documents.

V. Conclusion

This study aimed to compare the performance between document similarity algorithms with domain expert curated keywords extracted from a scientific paper. Despite excluding author-provided keywords from the papers, having a small sample size, and other limitations, it can be concluded that machine-generated keywords are at least 80% similar to domain expert curated keywords for an EDLC domain. In future, this work can be extended accounting the author provided keywords and a larger data-set for different other domains.

Acknowledgment

This work has been supported by RDU190307, UMP grant. Authors also acknowledge the UMP Flagship grant RDU182201, as this work originated from this grant. Authors would like to express their gratitude to the domain experts for their support and knowledge.

TABLE II. SIMILARITY RESULTS FOR WHOLE TEXT OF STANDARD DOCUMENTS BY DOMAIN EXPERTS

	Jaccard Similarity											
	0.074		0.114	0.079		0.178	0.207		0.831		0.900	0.886
	0.147		0.159	0.145		0.211	0.278		0.895		0.927	0.898

TABLE III. SIMILARITY RESULTS FOR POSITIVE TEXT OF STANDARD DOCUMENTS BY DOMAIN EXPERTS

	Jaccard Similarity											
	0.081		0.086	0.095		0.173	0.160		0.809		0.897	0.884
	0.065		0.095	0.136		0.107	0.178		0.809		0.899	0.890

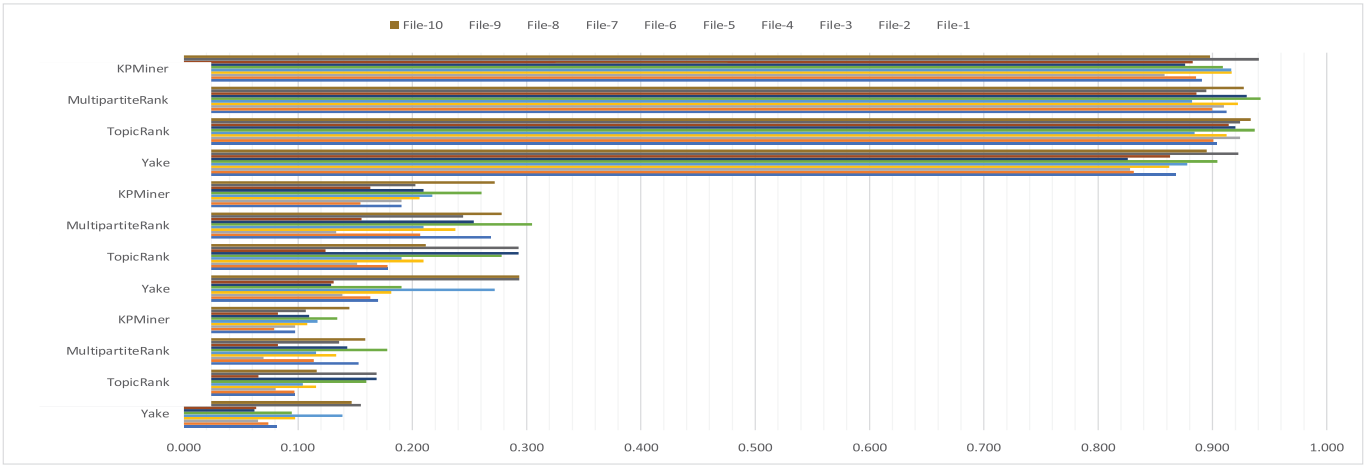

Fig. 2. Score comparison for different similarity calculation algorithms for whole text of documents

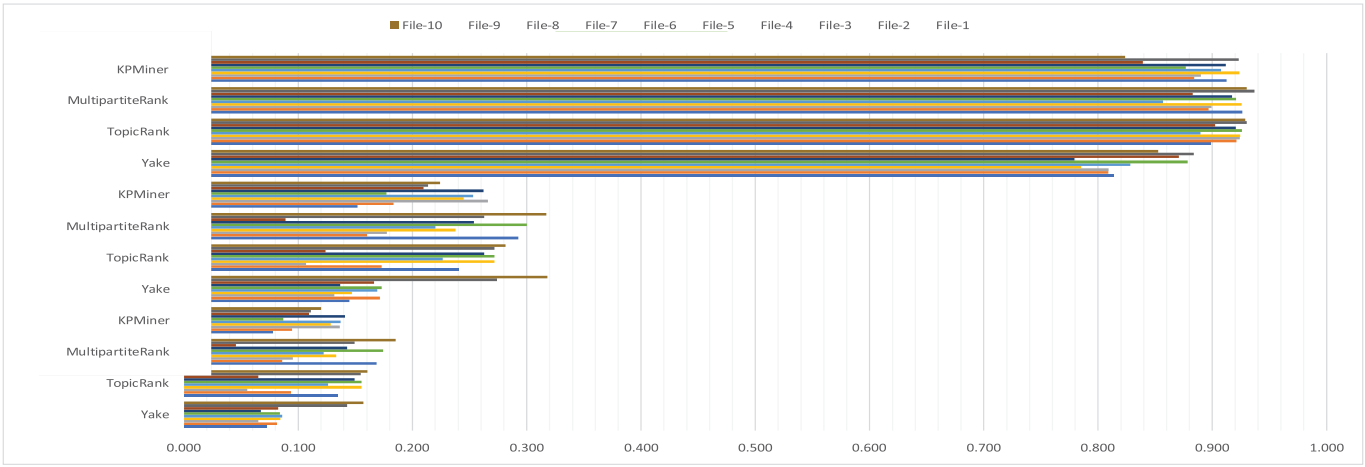

Fig. 3. Score comparison for different similarity calculation algorithms for positive text of documents

REFERENCES

[1] "Industry 4.0: fourth industrial revolution guide to Industrie 4.0." [Online]. Available: https://www.i-scoop.eu/industry-4-0/

[2] R. Jose and S. Ramakrishna, "Materials 4.0: Materials big data enabled materials discovery," *Applied Materials Today*, vol. 10, pp. 127–132, 2018.

[3] G. J. Mulholland and S. P. Paradiso, "Perspective: Materials informatics across the product lifecycle: Selection, manufacturing, and certification," *APL Materials*, vol. 4, no. 5, p. 053207, 2016.

[4] M. Hermann, T. Pentek, and B. Otto, "Design principles for industrie 4.0 scenarios," in *2016 49th Hawaii international conference on system sciences (HICSS)*. IEEE, 2016, pp. 3928–3937.

[5] J. M. Rickman, T. Lookman, and S. V. Kalinin, "Materials informatics: From the atomic-level to the continuum," *Acta Materialia*, 2019.

[6] "Scopus preview - Scopus - Welcome to Scopus." [Online]. Available: https://www.scopus.com/standard/marketing.uri

[7] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent semantic indexing: A probabilistic analysis," *Journal of Computer and System Sciences*, vol. 61, no. 2, pp. 217–235, 2000.

[8] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999, pp. 50–57.

[9] J. Tang, Z. Meng, X. Nguyen, Q. Mei, and M. Zhang, "Understanding the limiting factors of topic modeling via posterior contraction analysis," in *International Conference on Machine Learning*. PMLR, 2014, pp. 190–198.

[10] P. Kherwa and P. Bansal, "Topic modeling: a comprehensive review," *EAI Endorsed transactions on scalable information systems*, vol. 7, no. 24, 2020.

[11] Z. Fu, X. Wu, C. Guan, X. Sun, and K. Ren, "Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2706–2716, 2016.

[12] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, pp. 1262–1273.

[13] M. S. U. Miah, M. S. Tahsin, S. Azad, G. Rabby, M. S. Islam, S. Uddin, and M. Masuduzzaman, "A geofencing-based recent trends identification from twitter data," *IOP Conference Series: Materials Science and Engineering*, vol. 769, p. 012008, jun 2020. [Online]. Available: https://doi.org/10.1088/1757-899x/769/1/012008

[14] G. Maheshwari, P. Trivedi, H. Sahijwani, K. Jha, S. Dasgupta, and J. Lehmann, "SimDoc: Topic Sequence Alignment based Document Similarity Framework," 2017. [Online]. Available: https://doi.org/10.1145/nnnnnnn.nnnnnnn

[15] C. Yang, B. He, and ·. Y. Ran, "Utilizing Embeddings for Ad-hoc Retrieval by Document-to-document Similarity," Tech. Rep.

[16] S. Aryal, K. M. Ting, T. Washio, and G. Haffari, "A new simple and effective measure for bag-of-word inter-document similarity measurement," Tech. Rep., 2019.

[17] P. Sitikhu, K. Pahi, P. Thapa, and S. Shakya, "A Comparison of Semantic Similarity Methods for Maximum Human Interpretability," in *International Conference on Artificial Intelligence for Transforming Business and Society, AITB 2019*. Institute of Electrical and Electronics Engineers Inc., nov 2019.

[18] V. Thada and V. Jaglan, "Comparison of Jaccard, Dice, Cosine Similarity Coefficient To Find Best Fitness Value for Web Retrieved Documents Using Genetic Algorithm," Tech. Rep.

[19] R. Steinberger, B. Pouliquen, and J. Hagman, "Cross-lingual document similarity calculation using the multilingual thesaurus EUROVOC," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2276. Springer Verlag, 2002, pp. 415–424. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-45715-1_44

[20] "9.5.1. The Jaccard Similarity algorithm - 9.5. Similarity algorithms." [Online]. Available: https://neo4j.com/docs/graph-algorithms/current/labs-algorithms/jaccard/

[21] "Jaccard Index / Similarity Coefficient - Statistics How To." [Online]. Available: https://www.statisticshowto.com/jaccard-index/

[22] "Cosine Similarity - Understanding the math and how it works? (with python)." [Online]. Available: https://www.machinelearningplus.com/nlp/cosine-similarity/

[23] "9.5.2. The Cosine Similarity algorithm - 9.5. Similarity algorithms." [Online]. Available: https://neo4j.com/docs/graph-algorithms/current/labs-algorithms/cosine/

[24] "What Are Word Embeddings for Text?" [Online]. Available: https://machinelearningmastery.com/what-are-word-embeddings/

[25] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[26] E. Papagiannopoulou and G. Tsoumakas, "A review of keyphrase extraction," p. e1339, mar 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/widm.1339 https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1339 https://onlinelibrary.wiley.com/doi/10.1002/widm.1339

[27] V. M. H. Lim, S. F. Wong, and T. M. Lim, "Automatic keyphrase extraction techniques: A review," in *IEEE Symposium on Computers and Informatics, ISCI 2013*. IEEE Computer Society, 2013, pp. 196–200.

[28] Z. A. Merrouni, ·. Bouchra Frikh, and ·. B. Ouhbi, "Automatic keyphrase extraction: a survey and trends." [Online]. Available: https://doi.org/10.1007/s10844-019-00558-9

[29] G. Rabby, S. Azad, M. Mahmud, K. Z. Zamli, and M. M. Rahman, "A flexible keyphrase extraction technique for academic literature," *Procedia Computer Science*, vol. 135, pp. 553–563, 2018, the 3rd International Conference on Computer Science and Computational Intelligence (ICCSCI 2018) : Empowering Smart Technology in Digital Era for a Better Life. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050918314984

[30] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "Yake! keyword extraction from single documents using multiple local features," *Information Sciences*, vol. 509, pp. 257–289, 2020.

[31] A. Bougouin, F. Boudin, and B. Daille, "Topicrank: Graph-based topic ranking for keyphrase extraction," in *International joint conference on natural language processing (IJCNLP)*, 2013, pp. 543–551.

[32] F. Boudin, "Unsupervised keyphrase extraction with multipartite graphs," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 667–672. [Online]. Available: https://www.aclweb.org/anthology/N18-2105

[33] S. R. El-Beltagy and A. Rafea, "Kp-miner: A keyphrase extraction system for english and arabic documents," *Information systems*, vol. 34, no. 1, pp. 132–144, 2009.

[34] "Grobid," https://github.com/kermitt2/grobid, 2008–2021.

[35] Grammarly, "Negatives and Negation–Grammar Rules — Grammarly," 2021. [Online]. Available: https://www.grammarly.com/blog/negatives/

[36] J. Col, "Negative Vocabulary Word List - Enchanted Learning," 1998. [Online]. Available: https://www.enchantedlearning.com/wordlist/negativewords.shtml

[37] Y. HaCohen-Kerner and H. Badash, "Positive and Negative Sentiment Words in a Blog Corpus Written in Hebrew," in *Procedia Computer Science*, vol. 96. Elsevier B.V., jan 2016, pp. 733–743.

[38] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[39] P. Jaccard, "THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE." *New Phytologist*, vol. 11, no. 2, pp. 37–50, feb 1912.

[40] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513–523, jan 1988.

[41] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020. [Online]. Available: https://doi.org/10.5281/zenodo.1212303

[42] F. Boudin, "pke: an open source python-based keyphrase extraction toolkit," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, Osaka, Japan, December 2016, pp. 69–73. [Online]. Available: http://aclweb.org/anthology/C16-2015