DOI: 10.xx/ijdsml.2023.xx

BERT-SIMILARITY-BASED FIREFLY (BSBF) ALGORITHM FOR MULTIPLE DOCUMENT SUMMARIZATION

Rushi Desai, Saloni Patel, Shourya Kothari and Pankaj Sonawane
Department of Computer Science, Dwarkadas Jivanlal Sanghvi College of Engineering, India

Abstract

Extracting knowledge from various sources is a tedious task. Multi-document Text Summarization (MDTS) aims to extrapolate data from various sources and present it in a concise, cohesive form, in a way that is simple for the reader to comprehend and yet ensures that they obtain the important information. A meta-heuristic optimization algorithm called the firefly algorithm is utilized to generate a summary. Topic Relevance Factor, Coherency Factor, and Readability Factor are used to establish the fitness function. We use BERT-based similarity to calculate these factors which are then later input to the fitness function. The experiments are conducted on DUC 2003 and DUC 2004 datasets. The suggested algorithm's performance is compared to that of previous meta-heuristic and graph-based techniques.

Keywords:

Multiple document summarization, Meta-heuristic optimization, Firefly algorithm, ROUGE, BERTscore

1. INTRODUCTION

A concise representation of text that includes a substantial portion of the content from one or multiple source texts, but is limited to no more than half the length of the original material, is commonly known as a Multi-Document Text Summarization (MDTS). A text summary's main advantage is that it can reduce the user's reading time.

Automatic text summarization can be classified into two main categories: single-document summarization and multi-document summarization. Multi-document summarising entails processing a connected collection of documents, whereas single-document summation [1] processes just one input document to produce a summary. In both cases, the summary can take the form of an abstract or an extract. An extract is an overview that is produced by excluding major portions of the input, whereas an abstract is produced by restating the input's key points. Two categories of text summary exist i.e informative and indicative [2]. A suggestive summary conveys to the user the text's main topic. The original content is around 5% shorter in this summary. A brief summary of the main content is provided by the informative summarising mechanism. Around 20% of the provided content is taken up by the useful summary.

An extractive summary technique involves selecting important sentences, paragraphs [3], or other components from the source material and combining them to form a condensed version. This technique heavily relies on statistical and linguistic features to determine the significance of each sentence. In contrast, abstractive summarization involves using advanced natural languages processing tools, such as grammar and lexicons, to rephrase the content of the text in a unique way.

According to the goal [4], the summarizer can be classified as generic, in which case the model treats the input impartially, or domain-specific, in which case the model uses domain data to produce an improved synopsis based on verified knowledge, or query-based, in which case the summary only contains well-known responses to natural language questions [3] about the input text.

As compared to summarising a single document, a multi-document summary presents greater difficulties. It addresses problems like redundant information in various papers, the compression of multiple documents, and the extraction of sentences quickly. These challenges are overcome utilizing statistical tools and optimization strategies. Any automatic summarizer must pay careful attention to the relevancy and redundancy of the document while summarising it.

By manually presenting and recognising the essential concepts within a lengthy publication, text summarising tools can help physicians and researchers save resources and time without having to read the entire text [5]. Text summarising first relies on frequency attributes to identify the text in documents that is most important to summarise. Afterward, a variety of algorithms and attributes have been incorporated by a number of summarization systems into the procedure of content selection.

Text summarization software has a variety of uses, including media monitoring, marketing for search engines, internal paperwork management, analysis of finances, digital marketing, and aiding the disabled.

2. RELATED WORK

The basic steps involved in MDTS are compilation and reduction of multiple documents, sentence extraction, removal of redundant information, sentence ranking or selection, and sentence ordering. Statistical tools have been used to resolve these issues previously [6] [7], but the performance of such tools has been poor. Early attempts in abstract summarization used variables like key phrases, word frequency (tf-idf), and sentence placements to score sentences [7] [8] [9] and then apply the concept of compression-ratio to select the top n redundant sentences.

Redundancy removal is a crucial aspect of Multi-Document Text Summarization (MDTS). Several research papers, including [10] and [11], employ the maximal margin reduction (MMR) technique [12] to select the most important sentences and minimize redundancy while generating summaries. Another approach, presented in [21], involves creating a unified document from multiple sources using a combination of Google-based similarity algorithms and word embedding. The aptness issue is then addressed as an optimization problem, and the Shark Smell Optimization Algorithm [16] is utilized to handle it.

Approaches based on clustering have been implemented to ensure adequate coverage and prevent redundancy. These approaches utilize clustering techniques to group similar

sentences into clusters to discover common informational topics before choosing individual sentences from each category to produce a summary [13] - [15]. Here, the use of the sentence similarity measure has a significant impact on cluster quality. Clustering-based methods are not as accurate as supervised methods that use annotated data since they rely on heuristics and assumptions about similarity. They may not be effective at capturing the overall coherence and meaning of the source documents, since they only consider sentence-level similarity.

The use of graph-based approaches has been demonstrated in numerous papers [22] - [27], where documents are represented in a graph with a weighted layout where every node is a phrase and the weighted edges show how similar the sentences are to one another. They rely on the overall relationships between sentences in the graph, rather than just looking at each sentence individually. Cosine similarity is a common technique used by these methods to determine the relationships between sentences. For instance, [28] combines centrality-based and centroid-based methods to assess the similarity and significance of each sentence. The word graph produced by [29] is based on the alignment data between pairs of related phrases and uses SBERT[30] to transform each sentence into fixed vectors. This method makes it easier to construct sentences that include numerous bits of information. The grammatical correctness and informativeness of the sentences created are assessed using an intensification function and sentence scoring tool. Finally, to guarantee that the final summary only contains the highest-scoring sentences, integer linear programming is used during the sentence selection procedure.

There have been a few metaheuristic optimization algorithms that have been used for single and MDTS tasks like Shark Smell Optimisation (SSO) [16], Genetic Algorithm [17], and Cuckoo Search Optimisation [18]. These techniques prioritized minimized redundancy and extensive coverage.

3. PROPOSED FRAMEWORK

In the literature survey, we saw meta-heuristic optimization algorithms being used to generate multi-document summaries. One such algorithm is the firefly algorithm [31], which draws inspiration from nature, specifically from a swarm of fireflies. The method is based on how fireflies behave, where each one is drawn to the firefly which is the brightest nearby.. A version of this algorithm has been used for MTDS [32]. We propose a similar algorithm with a difference in document representation and calculation of fitness score.

3.1 PREPROCESSING

Preprocessing is an essential step in any NLP task that involves transforming the raw input documents into a format easily processed by the summarization algorithm.

3.1.1 Case Conversion:

All text is converted to lowercase to ensure uniformity during processing.

3.1.2 Text Cleaning:

The raw text may contain noise, irrelevant information, or formatting tags. Text cleaning involves removing these elements from the text, such as HTML tags, special characters, and punctuation.

3.1.3 Sentence Segmentation:

Once the documents are selected and cleaned, the text must be segmented into individual sentences. This is done by identifying sentence boundaries based on punctuation, capitalization, and other linguistic cues.

3.1.4 Sentence Normalization Or Stop Word Filtering:

Normalization involves converting words to their base form and removing common words that do not add meaning or are not important, such as “the” and “a.”

3.1.5 Stemming:

It returns inflected (or occasionally derived) words to their root form. Using singular rather than plural nouns, for instance, or dropping the -ed or -ing from verbs are other examples. Many NLP tools use stemmer algorithms to carry out the stemming process.

3.2 DOCUMENT REPRESENTATION

We represent the document with BERT [33]. BERT is a bi-directional transformer model pre-trained on 3300M words for Masked Language Modelling(MLM) tasks. BERT uses an encoder model to create word embeddings in a latent space. We use 12 transformer layers to get a representation of the document. The final hidden layer’s output serves as the embedding which is later used to calculate similarity in the fitness function.

3.3 FITNESS FUNCTION

A summary is said to be good when it accurately and concisely conveys vital information from the original text in an articulate way. A good summary includes the main points of the text while omitting minor details, examples, and supporting evidence. A well-written summary should have coherence and flow; each sentence should be logically connected to the previous and next sentences. This creates a sense of continuity and makes the summary more readable and understandable. To achieve these features we consider three factors as mentioned in [32] as well: i) Topic Relativity factor ii) Linking Factor and iii) Comprehensibilty factor.

3.3.1 Topic Relativity Factor (TRF):

The Topic Relativity Factor (TRF) is a measure of the degree to which an abstract is connected to its subject matter. It is calculated by considering two factors: (i) the resemblance between the title and summary, referred to as Title Similarity (TS), and (ii) the similarity of the summary to the original text, known as Original Text Similarity (OTS). To calculate the degree of similarity between the summary and the title, the average similarity between each sentence and the title is first calculated. This value is then normalized by dividing it by the maximum similarity.

$$TS = \frac{\sum bertsim\left(S_j, q\right)}{S} \quad (1)$$

S_j stands for the summary’s j^{th} sentence, while q stands for the title. S stands for the summary’s total number of sentences. To determine the similarity between each sentence in the summary and the text, we employ the BERT-similarity metric. This

approach has several advantages, as highlighted in studies such as [30] and [34].

One of the key benefits of using BERT-similarity is its ability to accurately capture the semantic similarity between different texts, even when the texts use distinct words or syntactic structures. This is explained by the fact that the BERT has already been trained on a sizable corpus and has honed its ability to represent word and phrase meanings based on context.

$$OTS = \frac{\sum bertsim\left(S_j, T_i\right)}{S * T} \tag{2}$$

where T denotes the total number of sentences in the original text and T_i denotes the i^{th} sentence in the original text. Finally, we add TS and OTS to determine the TRF

$$TRF = OTS+TS \tag{3}$$

3.3.2 Linking Factor (LF):

Linking or cohesion refers to the degree of connectedness and unity among the elements of a text, such as words, phrases, and sentences. It reflects how well these different parts are integrated into a meaningful and coherent whole. The summary is converted into a weighted graph, where each sentence is represented as a node and the edge weight denotes the degree of similarity between two sentences, in order to calculate the cohesiveness factor. Two factors are required to calculate the cohesion factor: C and M. C is obtained by calculating the mean of the similarities between all sentences in the summary.

$$C = \frac{\sum \forall S_i S_j \in subsummary\ graph W\left(S_i, S_j\right)}{N_S} \tag{4}$$

where *Ns* is the total number of edges in the summary subgraph and $W(S_iS_j)$ indicates the sum of the weights of all the edges on the path from sentence i to sentence j.

$$N_s = (S*(S-1))/2 \tag{5}$$

In the summary subgraph, M is the highest weight.

$$M = maxsim_{i,j} \quad i,j \leq N \tag{6}$$

Finally, LF is given by the logarithm equation as mentioned in [35]. When the maximum is significantly bigger than the mean, the logarithm helps prevent the low magnitude of LF.

$$LF = \frac{\log_{10}\left(C*9+1\right)}{\log_{10}\left(M*9+1\right)} \tag{7}$$

3.3.3 Readability Factor (RF):

A comprehensible summary is where two sentences are correlated and there is a similarity between two consecutive sentences. However, we need to ensure that the sentences are not very similar as they may repeat the same information and increase redundancy. Hence, first, we set an upper limit between the similarity of two sentences, α.

$$\text{sim}_{i,j} \leq \alpha \tag{8}$$

After experimenting with different values of α, the optimal value came out to be 0.88. RF is calculated as follows:

$$\text{Readability}\left(R\right) = \sum W\left(S_i, S_j\right) \tag{9}$$

$$RF = \frac{R}{\max \forall R_i} \tag{10}$$

where R_i is the greatest distance in the weight matrix with a specific number of nodes.

3.3.4 Fitness Function:

The factors are used for the calculation of the fitness function. Each factor is assigned a weight that is user-defined. So for example, if a user wants a more cohesive summary, he/she can increase the weight associated with LF.

$$F = \alpha*TRF + \beta*LF + \gamma*RF \tag{11}$$

3.4 FIREFLY ALGORITHM

The metaheuristic optimisation technique known as the "firefly algorithm" [31] was modelled after the natural behaviour of fireflies. Fireflies are known for their ability to produce light and use it for communication, attraction, and mate selection. The algorithm was first bought to light by Xin-She Yang in 2008. The firefly algorithm is a swarm-based algorithm that simulates the behavior of fireflies.

3.4.1 Representation of Firefly:

Every document (firefly) is represented as a vector, and each sentence is labeled as 1 or 0 depending on whether it should be included in the summary. The method initialises K fireflies at random and determines each firefly's light intensity using a fitness function that integrates the three features stated above. The firefly with the highest fitness value is deemed the brightest firefly and serves as the leader. Other fireflies are attracted to this leader. The primary objective of this algorithm is to locate the summary that maximizes the fitness function.

3.4.2 Firefly Algorithm Usage:

Based on the brightness of other fireflies in the population, the firefly algorithm modifies each firefly's position.

$$x_i = x_i + \beta_0 e^{-\gamma r^2}\left(x_j - x_i\right) + \alpha\left(\in(\ ) - 0.5\right) \tag{12}$$

The Eq.(12) includes several parameters. The firefly's absolute brightness is represented by the value β_0, which is normally set to 1. A firefly's level of random movement is controlled by the parameter α while the distance between them is adjusted by the parameter γ. The symbol for a random vector, which has values between 0 and 1, is denoted by $\epsilon()$. For the brightest firefly, the update equation is simplified (Eq.13) by neglecting the second term. This means that the brightest firefly does not move toward other fireflies, but rather moves randomly.

3.4.3 Firefly Calculations:

To create a summary representation, values greater than 0.5 in the vector are treated as 1, indicating that the corresponding sentence should be included in the summary. Conversely, values less than 0.5 are treated as 0, indicating that the corresponding sentence should not be included in the summary. However, this conversion is solely performed while computing the intermediate summaries, not the final summary.

3.4.4 Generation of Summary:

Once the maximum number of iterations is reached, the final summary is generated. The firefly with the highest fitness value is selected as the best firefly. In order to select the sentences with the greatest values for the final summary, we sort the values in the best firefly. The procedure continues until the maximum amount of phrases or words is not achieved, at which point the final

summary is prepared for all the input documents. The FbTS algorithm proposed for text summarization has a time complexity of $O(n^2t)$.

4. EXPERIMENTS AND RESULTS

The BSBF algorithm is built in this section, and the effectiveness of the method is assessed by contrasting it with algorithms. The datasets from DUC-2003 and DUC-2004 are used to conduct the evaluation.

4.1 DATASET

For our model, we employed the Document Understanding Conferences (DUC) 2003 Task 2 data set and the DUC 2004 Task 2 dataset. News articles on a specific topic made up the DUC 2003 dataset. There were ten subjects in the dataset, and each topic had ten to twenty sources. Similar to this, DUC 2004 has fifty subjects with ten sources each.

4.2 EVALUATION METHODOLOGY

A generated summary or translation's resemblance to a collection of reference summaries or translations is measured using a set of metrics called ROUGE. ROUGE has several variants, including ROUGE-N (which looks at n-gram overlap), ROUGE-L, and ROUGE-W (which looks at weighted n-gram overlap) [19].

Table.1. Dataset Description

Dataset Description	DUC 2003	DUC 2004
Set of Documents	30	100
Documents per set	10	10
Source documents	Associated Press (AP) newswire	English Gigaword corpus
Summary length	100	655 bytes

However, ROUGE has some limitations, It only looks at the surface-level overlap between the generated and reference texts and does not consider semantic similarity or other aspects of text quality. It is sensitive to differences in the length of the generated and reference texts, which can lead to artificially high or low scores.

4.3 RESULTS

On the DUC-2003 dataset, BSBF achienes a Rouge-1 score of 0.4351 and Rouge-2 score of 0.1792 which is better than other algorithms used. We compare the results with Genetic algorithm with JS divergence [36], Particle Swarm Optimisation (PSO) with JS divergence [37], Ant Colony Optimisation (ACO)[38], Firefly based Text Summarization[32] and LexRank[23]. The Table.2 shows the comparison of the various algorithms. We made similar comparisons on the DUC-2004 dataset, where we achieved a ROUGE-1 score of 0.4351 and Rouge-2 score of 0.1792 as mentioned in Table.3.

Table.2. Comparison of BSBF with other methods on DUC-2003 datset

	Evaluation Metric	
Methods	**ROUGE-1**	**ROUGE-2**
GA(JS-Divergence)	0.4397	0.1416
PSO(JS-Divergence)	0.4321	0.1507
ACO (CosineSimilarity)	0.4231	0.1407
FbTS	0.4419	0.1602
LexRank	0.3574	0.0793
BSBF	**0.4612**	**0.1702**

Table.3. Comparison of BSBF with other methods on DUC-2004 dataset

	Evaluation Metric	
Methods	**ROUGE-1**	**ROUGE-2**
GA(JS-Divergence)	0.3546	0.0937
PSO(JS-Divergence)	0.3521	0.0935
ACO (CosineSimilarity)	0.3542	0.0837
FbTS	0.4244	0.1764
LexRank	0.326	0.079
BSBF	**0.4351**	**0.1792**

5. CONCLUSION

This research introduces a new algorithm for MDTS, called BSBF based on the firefly algorithm. The algorithm uses a fitness function that combines TRF, CF, and RF to rank each sentence and select the highest-scoring sentences for the summary. Experimental evaluations on DUC-2003 and DUC-2004 datasets using the ROUGE score show that the proposed algorithm outperforms genetic algorithms in terms of ROUGE-1 and ROUGE-2 scores. In order to further enhance the quality of abstractive summaries, the study recommends experimenting with new feature selection techniques and fitness features with bio-motivated algorithms and merging these extractive techniques with deep neural-based models. A summary generation may perform better when hybrid models are used.

REFERENCES

[1] K. Sarkar, "Automatic Text Summarization using Internal and External Information", *Proceedings of International Conference on Emerging Applications of Information Technology*, pp. 1-4, 2018.

[2] D. Das and A. Martins, "A Survey on Automatic Text Summarization", Available at https://www.cs.cmu.edu/~afm/Home_files/Das_Martins_survey_summarization.pdf, Accessed in 2007.

[3] N. Moratanch and S. Chitrakala, "A Survey on Extractive Text Summarization", *Proceedings of International Conference on Computer, Communication and Signal Processing*, pp. 1-6, 2017.

[4] I. Awasthi, K. Gupta, P.S. Bhogal, S.S. Anand and P.K. Soni, "Natural Language Processing (NLP) based Text

Summarization - A Survey", *Proceedings of International Conference on Inventive Computation Technologies*, pp. 1310-1317, 2021.

[5] A.S. Almasoud, S. Ben Haj Hassine, F.N. Al-Wesabi, M.K. Nour and A. Mustafa Hilal, "Automated Multi-Document Biomedical Text Summarization using Deep Learning Model", *Computers, Materials and Continua*, Vol. 71, No. 3, pp. 5799-5815, 2022.

[6] Vikrant Gupta, Priyamvada Chauhan, Sohan Garg, Anita Borude and Shobha Krishnan, "An Statistical Tool for Multi-Document Summarization", *International Journal of Scientific and Research Publications*, Vol. 2, No. 5, pp. 1-5, 2012.

[7] M. Ponnusamy, P. Bedi and T. Suresh, "Design and Analysis of Text Document Clustering using Salp Swarm Algorithm", *The Journal of Supercomputing*, Vol. 78, No. 14, pp. 16197-16213, 2022.

[8] H.P. Edmundson, "New Methods in Automatic Extracting", *Journal of the ACM*, Vol. 16, No. 2, pp. 264-285, 1969.

[9] H.P. Luhn, "The Automatic Creation of Literature Abstracts", *IBM Journal of Research and Development*, Vol. 2, No. 2, pp. 159-165, 1958.

[10] Dragomir R. Radev, Hongyan Jing and Daniel Tam, "Centroid-Based Summarization of Multiple Documents", *Information Processing and Management*, Vol. 40, No. 6, pp. 919-938, 2004.

[11] Dragomir R. Radev, Hongyan Jing and Malgorzata Budzikowska, "Centroid-Based Summarization of Multiple Documents: Sentence Extraction, Utility-Based Evaluation, and User Studies", *Proceedings of International Workshop on Automatic Summarization*, pp. 1-12, 2000.

[12] Goldstein-Stewart and Jaime G. Carbonell, "Summarization: (1) using MMR for Diversity- Based Reranking and (2) Evaluating Summaries", *Proceedings of International Workshop on Association for Computational Linguistics*, pp. 1-15, 1998.

[13] D. Marcu and L. Gerber, "An Inquiry into the Nature of Multi Document Abstracts, Extracts, and Their Evaluation", *Proceedings of Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 107-116, 2001.

[14] P. Ji, "Multi-document Summarization based on Unsupervised Clustering", Springer, 2006.

[15] Kamal Sarkar, "Sentence Clustering-based Summarization of Multiple Text Documents", *TECHNIA - International Journal of Computing Science and Communication Technologies*, Vol. 2, pp. 325-335, 2009.

[16] O. Abedinia and A. Ghasemi, "A New Metaheuristic Algorithm based on Shark Smell Optimization", *Complexity*, Vol. 21, pp. 97-116, 2016.

[17] A. Kogilavani and P. Balasubramanie, "Clustering based Optimal Summary Generation using Genetic Algorithm", *Proceedings of International Conference on Communication and Computational Intelligence*, pp. 324-329, 2010.

[18] Rasmita Rautray and Rakesh Chandra Balabantaray, "An Evolutionary Framework for Multi Document Summarization using Cuckoo Search Approach: MDSCSA", *Applied Computing and Informatics*, Vol. 14, No. 2, pp. 134-144, 2018.

[19] A. Ganesan and A. Kavita, "ROUGE 2.0: Updated and Improved Measures for Evaluation of Summarization Tasks", *Proceedings of International Conference on Computation and Language*, pp. 1-10, 2019.

[20] M. Ramina and A. Dhruv, "Topic Level Summary Generation using BERT Induced Abstractive Summarization Model", *Proceedings of International Conference on Intelligent Computing and Control Systems*, pp. 1-13, 2020.

[21] Pradeepika Verma and Hari Om, "MCRMR: Maximum Coverage and Relevancy with Minimal Redundancy based Multi-Document Summarization", *Expert Systems with Applications*, Vol. 120, pp. 43-56, 2019.

[22] Minakshi Tomer and Manoj Kumar, "Multi-Document Extractive Text Summarization based on Firefly Algorithm", *Journal of King Saud University - Computer and Information Sciences*, Vol. 34, No. 8, pp. 6057-6065, 2022.

[23] Gunes Erkan and Dragomir R. Radev, "LexRank: Graph-Based Lexical Centrality as Salience in Text Summarization", *Journal of Artificial Intelligence Research*, Vol. 22, No. 1, pp. 457-479, 2004.

[24] Rada Mihalcea and Paul Tarau. "Text Rank: Bringing Order into Text", Proceedings of International Conference on Empirical Methods in Natural Language Processing, pp. 404-411, 2004.

[25] Rada Mihalcea and Paul Tarau, "A Language Independent Algorithm for Single and Multiple Document Summarization", *Proceedings of International Conference on Recent Trends in AI and Machine Learning*, pp. 1-6, 2005.

[26] Erkan Guneş, "LexPageRank: Prestige in Multi-Document Text Summarization", *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, pp. 365-371, 2004.

[27] Xiaojun Wan and Jianwu Yang, "Improved Affinity Graph Based Multi-Document Summarization", *Proceedings of International Conference on Human Language Technology*, pp. 181-184, 2006.

[28] K. Sarkar, K. Saraf and A. Ghosh, "Improving Graph based Multi-Document Text Summarization using an Enhanced Sentence Similarity Measure", *Proceedings of International Conference on Recent Trends in Information Systems*, pp. 359-365, 2015.

[29] Raksha Agarwal and Niladri Chatterjee, "Improvements in Multi-Document Abstractive Summarization using Multi Sentence Compression with Word Graph and Node Alignment", *Expert Systems with Applications*, Vol. 190, pp. 1-17, 2022.

[30] Nils Reimers and Iryna Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks", *Proceedings of International Conference on Computation and Language*, pp. 3973-3983. 2019.

[31] X.S. Yang, "Firefly Algorithms for Multimodal Optimization", Springer, 2009.

[32] Minakshi Tomer and Manoj Kumar, "Multi-Document Extractive Text Summarization based on Firefly Algorithm", *Journal of King Saud University - Computer and Information Sciences*, Vol. 34, No. 8, pp. 6057-6065, 2022.

[33] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding", *Proceedings of International Conference on Computational Linguistics: Human Language Technologies*, pp. 4171-4176, 2019.

[34] Nicole Peinelt, Dong Nguyen and Maria Liakata, "BERT: Topic Models and BERT Joining Forces for Semantic Similarity Detection", *Proceedings of International Conference on Association for Computational Linguistics*, pp. 7047-7055, 2020.

[35] Vahed Qazvinian and Ramin Halavati, "Summarising Text with a Genetic Algorithm-based Sentence Extraction", *International Journal of Knowledge Management Studies*, Vol. 2, pp. 1-10, 2008.

[36] Suhad Kadhem and Zuhair Ali, "Multi-Document Summarization using Fuzzy Logic and Firefly Algorithm", *Journal College of Education*, Vol. 3, pp. 1-14, 2018.

[37] H. Asgari, B. Masoumi and O. S. Sheijani, "Automatic Text Summarization based on Multi-Agent Particle Swarm Optimization", *Proceedings of Iranian Conference on Intelligent Systems*, pp. 1-5, 2014.

[38] Asma Al-Saleh and Mohamed El Bachir Menai, "Ant Colony System for Multi-Document Summarization", *Proceedings of International Conference on Computational Linguistics*, pp. 734-744, 2018.