

# GaSUM: A Genetic Algorithm Wrapped BERT for Text Summarization

Imen Tanfouri<sup>1</sup>  and Fethi Jarray<sup>1,2</sup> 

<sup>1</sup>LIMITIC Laboratory, UTM University, Tunisia

<sup>2</sup>Higher Institute of Computer Science of Medenine, Tunisia

**Keywords:** Single Document Summarization, Natural Language Processing, BERT Model, Genetic Algorithm, Triplet Loss Function.

**Abstract:** Automatic Text Summarization (ATS) is a fundamental problem in natural language processing (NLP), it aims to reduce text size by removing irrelevant data and preserving the semantic structure of the original text. Recently, transformer-based models have shown great success in ATS and have been considered the state-of-the-art model for many NLP tasks. In this research, we are concerned with extractive summarization for a single document, where the goal is to extract a subset of sentences that best represents a summary of the document. We propose a combination of Bidirectional Encoder Representations from Transformers (BERT) and a Genetic Algorithm (GA) for automatic text summarization (named GaSUM) where GA is used as a search space method and BERT is used as a fitness measure. We validated these methods using the CNN/Daily Mail available dataset. Our results showed that GaSUM achieves a ROUGE-1 score of 55.75% and outperforms the state-of-the-art methods by a significant margin in terms of the rouge score.

## 1 INTRODUCTION

Nowadays, the big amount of data available on the internet and its high increase daily causes a big problem for the user to pick out pertinent information. Therefore, it is important to develop automated text summarizing systems that can transform longer texts into small paragraphs to reduce reading time and speed up the research process for pertinent information. There are important applications for text summarization in different NLP-related tasks like question answering, headline generation, news summarization, and text classification.

Machine learning-based models have been proposed for different NLP tasks like word sense disambiguation (Saidi et al., 2022; Saidi and Jarray, 2022), sentiment analysis, and text summarization. Numerous machine-learning-based models are used in text summarization task, most of them model the summarization problem as a classification problem in which the output is whether to include each sentence of the original text in the summary or not. Other approaches have used Latent Semantic Analysis (LSA) (Tanfouri and Jarray, 2022), Reinforcement Learning (Keneshloo et al., 2019), and Sequence to Sequence

models (Nallapati et al., 2017).

Automatic text summarization (ATS) can be divided into extractive text summarization and abstractive text summarization according to the output type of the summarization system. The former aims at selecting the most relevant sentences from a given document as its summary. The latter aims to generate a short text summary by paraphrasing the content of the original document.

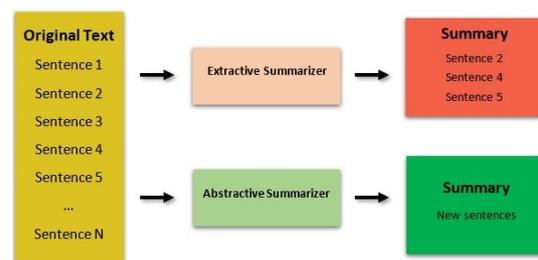


Figure 1: Extractive and Abstractive text summarizer.

According to the approach used, an extractive summarization procedure can be either a sentence-level (Nallapati et al., 2017; Liu and Lapata, 2019; Jia et al., 2021; Gong et al., ) or a summary-level. An extractive sentence level consists in scoring each sentence of the document, then selecting sentences

<sup>a</sup>  <https://orcid.org/0000-0003-2504-5267>

<sup>b</sup>  <https://orcid.org/0000-0003-2007-2645>

with the highest score to create the summary. The main drawback of this method is that it independently selects the sentences rather than taking into account the semantics of the full summary. In order to overcome this challenge, Zhong et al. (Zhong et al., 2020) proposed a summary-level approach, MatchSUM that jointly learns salience and redundancy in the extractive summarization. Globally, it consists in generating summaries and returning the highest score summary. Zhong et al. (Zhong et al., 2020) cast the scoring step as a semantic textual matching problem and designed a BERT-based framework for it. However, MatchSUM becomes impractical when the search space is huge, as in the case of long documents.

In this contribution, we focus on an extractive summary-level approach. The contribution of this work can be summarized as follows:

1. Propose GaSUM a hybrid approach of GA and BERT for extractive summary level approach for single document summarization.
2. Wrap the MatchSUM approach and run it inside a GA structure.
3. Validate our method on the CNN/Daily Mail available dataset and achieve state-of-the-art performance.

The remainder of this paper is structured as follows. Section 2 reviews the related work on ATS. Section 3 explains our GaSUM method for ATS. Section 4 details the experiments and the results. Section 5 concludes this paper and mentions some future extensions.

## 2 RELATED WORK

The approaches developed for extractive text summarization can be divided into classical machine learning (ML) models that require handcrafted feature extraction and deep learning models where feature extraction and prediction are embedded in a single network. Traditional ML approaches include statistically based systems (Al-Hashemi, 2010; Haboush et al., 2012), support vector machine (SVM) (Boudabous et al., 2010), Latent Semantic Analysis(LSA) and Latent Dirichlet Allocation (LDA) (Froud et al., 2013; AL-Khawaldeh and Samawi, 2015; Tanfour and Jaray, 2022; Mohamed and Oussalah, 2019), boosting (Belkebir and Guessoum, 2015) and meta-heuristic approaches (Al-Abdallah and Al-Taani, 2017; Jara-dat, 2015; Tanfour et al., 2021). Miller(Miller, 2019).

Nada et al.(Abu Nada et al., 2020) proposed a BERT-based approach by first embedding the sentences by BERT and then clustering the embedded

data vectors. Finally, sentences closer to the centroids are selected for the summary. The main drawback of this approach is that the sentences are encoded independently into fixed-sized embeddings. To address this issue, Yang Liu (Liu and Lapata, 2019) proposed BERTSUM an extension of BERT for extractive summarization. It can be considered the first BERT-based extractive sentence-level summarization model. It has the ability to deal with multiple sentences by inserting a [CLS] token before each sentence, contrary to the original BERT release, which cannot handle more than two sentences. It achieves a 30.01 ROUGE-1 score on CNN/DM dataset (Nallapati et al., 2016; Hermann et al., 2015).

LEAD (Xu and Durrett, 2019) presents a neural model for single-document summarization based on joint extraction and syntactic compression. This model is a neural network model that encodes a source document, chooses sentences from that document, and selects discrete compression options to apply. They choose sentences from the document, identify possible compressions based on constituency parse, and score those compressions with a neural model to produce the final summary. It achieves a 40.43 ROUGE-1 score on CNN/DM dataset.

MatchSUM (Zhong et al., 2020) was the first summary-level method proposed for extractive summarization. Zhong et al. (Zhong et al., 2020) cast the summarization task as a semantic text-matching problem rather than the commonly used sequence labeling model. They designed a Siamese-based neural network to measure the semantic similarity between the original document and a candidate summary. They used triplet loss, where a good summary should be semantically similar to the source document. Furthermore, they achieved a 44.41 ROUGE-1 score on CNN/DM dataset. Although we have built our proposed approach upon this, the main weakness of MatchSUM is the lack of a search strategy, which determines how to explore the search space in order to find a good summary. One of our contributions is to overcome the exploration issue by designing a genetic algorithm.

## 3 PROPOSED SYSTEM GaSUM

We address the extractive summarization task as an optimization problem, where the goal is to select the best subset of sentences that maximize an evaluation score. Our approach has two components: (1) a local search-based genetic algorithm (GA) to explore the search space and construct an optimal summary, and (2) a BERT-based semantic textual similarity model

(Fitness-BERT) to measure the similarity between the original document and a guess summary that is considered as its fitness score. Unlike the sentence-level summary approach, we compute a global score of the selected summary.

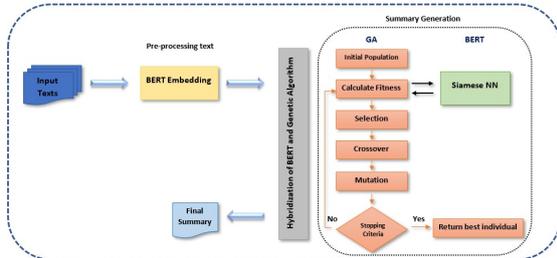


Figure 2: GaSUM flowchart composed by a local search-based genetic algorithm to find the best summary and a BERT-based fitness function to compute the fitness score of a summary.

We introduce the following notations:  $D$  a document consisting of  $n$  sentences,  $C$  a candidate summary made from  $k$  sentences extracted from  $D$ ,  $C$  is the search space, i.e. the space of all feasible solutions (summaries),  $C^*$  is the human-annotated summary of  $D$ ,  $f(D, C)$  the similarity score of  $C$  according to  $D$  and  $\hat{C}$  is the candidate summary with the highest score, i.e. the best summary according to the score.

$$\hat{C} = \arg \max_{C \in C} score(C, D) \quad (1)$$

The major issue in text summarization is the scoring of summaries according to the original document. In this contribution, we use a BERT-based similarity measurement.

### 3.1 Fitness-BERT Component

We cast the similarity between  $C$  and  $D$  as a Semantic Textual Similarity (STS) task. STS seeks to quantify the degree of similarity between the input texts  $C$  and  $D$ . A Siamese neural network (SNN) is a neural network designed to learn the similarity between two texts (see Figure 3).

SNN consists of two identical subnetworks, a.k.a. twin networks, that output the embedding of the given two input sentences. Each sub-network implements the BERT embedding. We feed the document  $D$  and the candidate summary  $C$  to SNN and take  $r_D$  and  $r_C$  as their token representations [CLS], respectively. We use cosine between  $r_D$  and  $r_C$  to measure the similarity between  $D$  and  $C$ ,

$$f(D, C) = cosine(r_D, r_C) \quad (2)$$

SNN is commonly trained with the triplet loss function (Schroff et al., 2015), which ensures that the gold

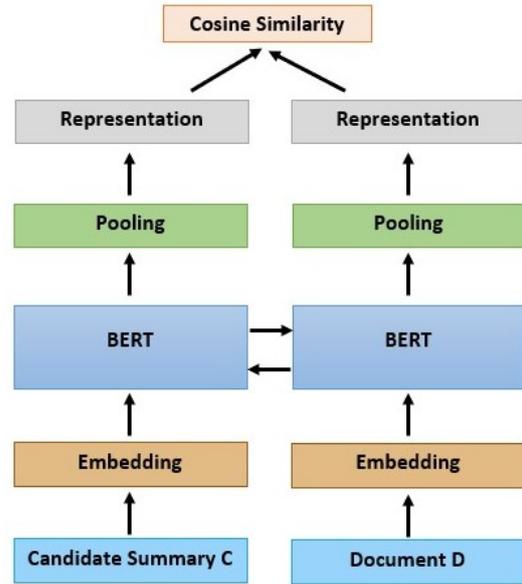


Figure 3: A Siamese neural network (SNN) architecture that outputs the semantic similarity between the candidate summary and the original document.

summary is closer to the document than any other candidate summary. In its terminology,  $D$  is the anchor,  $C^*$  is the positive sample, and  $C$  is the negative sample (Zhong et al., 2020). Mathematically, it is stated as follows:

$$L_1 = \max(0, f(D, C) - f(D, C^*) + \gamma_1) \quad (3)$$

where a  $\gamma_1$  is a margin value (see Figure 4).

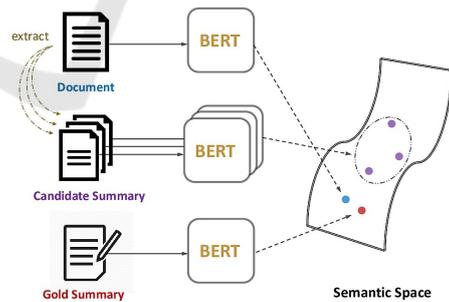


Figure 4: Triplet loss function for training Siamese neural network (Zhong et al., 2020). The golden summary  $C^*$  should be semantically closer to the source document than any other candidate summary.

We suppose that the candidate summaries are sorted in descending order of ROUGE scores, with the gold summary Zhong et al. (Zhong et al., 2020) added a pairwise margin loss for all the candidate summaries as a regularization term. The idea behind this is that the candidate pair with a larger rank gap

should have a larger score margin. Mathematically,

$$L_2 = \max(0, f(D, C_j) - f(D, C_i) + (j - i)\gamma_2) \quad i < j \quad (4)$$

where  $c_i$  denotes the candidate summary number  $i$  and  $\gamma_2$  is a hyper parameter. Finally, the margin-based triplet loss can be expressed as equation (5)

$$L = L_1 + L_2 \quad (5)$$

The major challenge of applying triplet loss is how to generate the negative examples, i.e. the set of all summaries. Here, we apply the proposed pruning approach (Zhong et al., 2020) where we first apply a filtering operation that eliminates irrelevant sentences and then apply a brute-force method to generate all combinations of summaries.

## 3.2 GA Component

The inference phase consists of selecting the highest score summary of document  $D$ .

$$\hat{C} = \arg \max_{C \in \mathcal{C}} f(D, C) \quad (6)$$

Since  $\mathcal{C}$  is a very huge space, we propose GA to explore it. A genetic algorithm (GA) is a search-based algorithm for solving combinatorial optimization problems in machine learning, computer vision (Jarray et al., 2008; Brunetti et al., 2007) and natural language processing. It initially creates a set of possible solutions known as population and then iteratively generates better and better individuals with regard to a fitness function or criterion by selecting, crossing, and mutating them (see Figure 5). One of the main advantages of GA is that it does not need derivative computation.

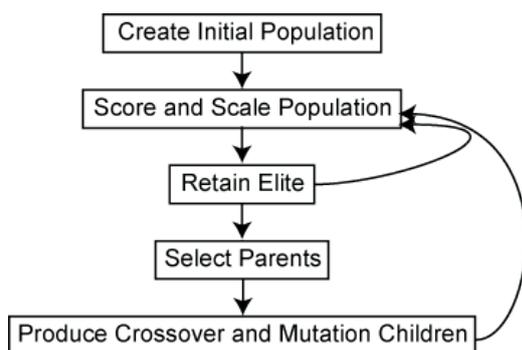


Figure 5: Flowchart of genetic algorithm. The main components are initial population generation, fitness scoring, parent selection and diversification operators such as crossover and mutation.

In the following, the terms chromosome and candidate summary are used interchangeably. The main

phases of the genetic algorithm are population Fitness function, initialization, selection, crossover, mutation, and summary generation. As mentioned above, the score of a population  $C$  according to a document  $D$  is computed as the cosine between their vector representation according to Eq.(2).

### 3.2.1 Initial Population

This step starts by randomly creating the initial population  $P$ . This population contains  $P_s$  chromosomes (candidate summaries), and each chromosome has  $n$  genes, where  $n$  is the number of sentences in the document. Each gene represents a sentence, and it can take the value 1 if the sentence will participate in the summary or 0 otherwise. For example, for a document with four sentences  $n = 4$  and 3 chromosomes,  $p_s = 3$  the initial population may be as follows  $P = ([0110], [1011], [0101])$ . The choice of the size of the population size  $P_s$  is a relevant issue: if the population is too small, the GA may converge too quickly to a bad solution, if it is too large, the algorithm may take a long time to converge, especially when the fitness function is time-consuming.

### 3.2.2 Selection

The selection phase consists in selecting the highest score individuals and letting them pass their genes to the next generation. There are several techniques to select an individual from a population like roulette wheel selection, rank selection, and tournament selection. The last strategy consists of randomly picking  $k$  individuals from the population and keeping the best one, where  $k$  is the tournament size parameter. This process is repeated until we reach the desired number of individuals to reproduce. The tournament candidates are picked in two variants: with or without replacement. In our experimental setting, we applied the tournament selection strategy with replacement. Once the selection of individuals to form the parents is performed, the next phase is the application of diversification operators such as crossover and mutation.

### 3.2.3 Crossover

Crossover is the process that combines two individuals (parents) to produce a new individual (offspring). There are different ways of swapping parents to get offspring such as one-point, two-point, and uniform crossover strategy (see Figure 6). Uniform crossover can be seen as a more general version of the multipoint crossover, where each bit (gene) is randomly swapped between the parents. In this paper, we adopt a one-point crossover strategy.

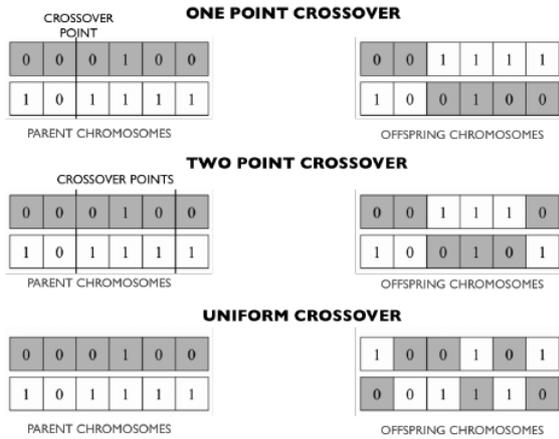


Figure 6: Crossover strategies. From top to bottom: select a pivot point and swap the tails of the two parents to get two new offsprings, select two pivots and swap the middle substring of both parents, and uniformly (probability 0.5) select each bit from parent 1 or parent 2.

### 3.2.4 Mutation

Mutation creates offspring by randomly changing the bits (genes) of individual parents. There are different variants of mutation such as swap, insertion, inversion, and displacement mutation<sup>7</sup>. In this manuscript, we adopt the inversion strategy.

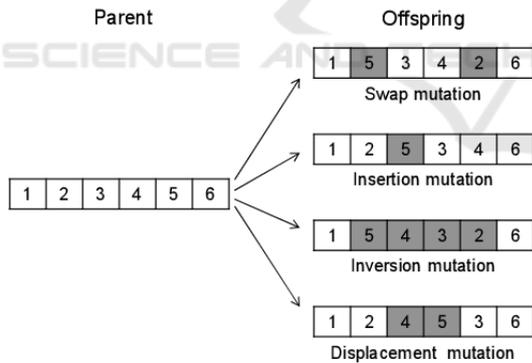


Figure 7: Mutation strategies. From top to bottom; exchange the values of two bits, randomly select a bit and randomly insert it, invert the values of a sequence of bits, and displace a sequence of bits.

### 3.2.5 Summary Generation

In the summarization problem, after the GA execution, we get the best individual with the highest fitness score. The final step is to decode this individual to get the final summary. We take the positions of genes with value 1, and we concatenate sentences of these positions in the same order as the original text.

### 3.2.6 Toy Example

Figure 8 shows a toy example of a document summarization. Table 1 presents the scores of similarity between the original and the generated summary.

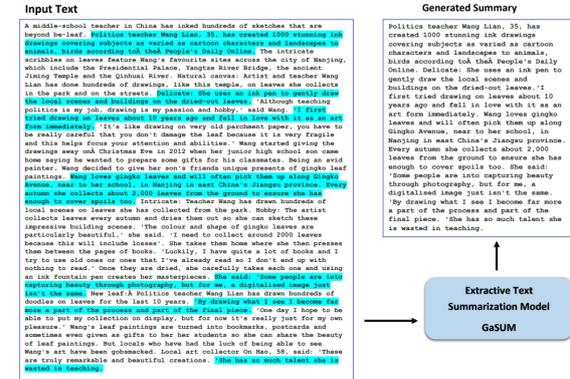


Figure 8: Example of running GaSUM algorithm over a toy example. The extracted sentences are highlighted in the original document.

The final summary returned by GaSum is evaluated using the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metric which considers both Recall and Precision between candidate (model-generated) summary and reference (golden-annotated) summary. It is branched into ROUGE-1, ROUGE-2, and ROUGE-L scores.

In ROUGE-1 precision and recall compare the similarity of uni-grams (each token of comparison is a single word) between reference and candidate summaries.

In ROUGE-2 precision and recall compare the similarity of bi-grams (each token of comparison is 2 consecutive words) between reference and candidate summaries.

In ROUGE-L precision and recall measures the Longest Common Sub-sequence (LCS) words between reference and candidate summaries.

Table 1 presents ROUGE-score for the toy example in terms of recall, precision, and F-score.

Table 1: ROUGE score for the toy example.

	Precision	Recall	F-score
ROUGE_1	0.70	0.51	0.59
ROUGE_2	0.56	0.41	0.48
ROUGE_L	0.66	0.47	0.56

## 4 EXPERIMENTS AND RESULTS

In this section, we describe the dataset used for text summarization and present our implementation of the GaSUM strategy.

### 4.1 Summarization Dataset

We carried out experiments on the CNN / DailyMail dataset. It is an English-language dataset containing over 300k news articles as written by journalists at CNN and the Daily Mail. Each document is paired with 3-4 highlights that summarize its content.

### 4.2 Implementation Details

We implement the genetic algorithm under the parameters presented in Table 2. For the fitness-BERT

Table 2: Parameters of genetic algorithm.

Description	Value
Population size	50
Tournament size	32
Selection strategy	tournament
Tournament variant	by replacement
Crossover strategy	one point
Iteration number	1000
Crossover Probability	0.6
Mutation Probability	0.2

component, we used the BERT-biased Siamese network fine-tuned in (Zhong et al., 2020). Concerning the GA component, we run a genetic algorithm with the following parameters: 1000 generations, population size 50, tournament selection as a selection strategy, crossover probability 0.6, and mutation probability 0.2.

The performance of our proposed method is evaluated using the ROUGE metric. ROUGE stands for Recall-Oriented Understudy for Gisting and is a well-known summary evaluation method. It counts the number of overlapping n-grams between the golden summary and the machine summary. Table 3 shows a performance comparison between our GaSUM model and the state-of-the-art.

Table 3: Comparison of numerical results on CNN/Daily Mail dataset through R\_1, R\_2 and R\_L ROUGE metrics.

Model	R-1	R-2	R-L
MatchSUM (BERT-base)	44.22	20.62	40.38
MatchSUM (RoBERTa-base)	44.41	20.86	40.55
LEAD	40.43	17.62	36.67
<b>GaSUM</b>	<b>55.75</b>	<b>44.98</b>	<b>54.32</b>

The experimental result shows that GaSUM outperforms the state-of-the-art approaches including LEAD and MatchSUM models according to R-1, R-2, and R-L metrics. Moreover, GaSUM is faster and more intelligent in the search for the optimal summary. In addition, GaSUM can be embedded in other metaheuristics, such as particle swarm optimization (PSO). This outperformance is due to the use of GA approach that efficiently explores the search space in order to find near-optimal solutions. In fact, the search space is exponential in the document size, and we have to choose the search strategy carefully to avoid discarding some regions or exhaustively searching for the best summary.

## 5 CONCLUSION

In this paper, we have proposed a hybrid approach of BERT representation genetic algorithm exploration (GaSUM) for summary-level extractive summarization. We prove how BERT based Siamese networks can be efficiently used for summaries scoring and how genetic algorithm can be used as an exploration strategy to find the best summary. GaSUM outperformed the state-of-the-art performance on CNN-Daily Mail dataset. As a future perspective of this work, we plan to further improve the performance of our approach through the introduction of an attention mechanism in the Siamese network.

## REFERENCES

- Abu Nada, A. M., Alajrami, E., Al-Saqqa, A. A., and Abu-Naser, S. S. (2020). Arabic text summarization using arabert model using extractive text summarization approach.
- Al-Abdallah, R. Z. and Al-Taani, A. T. (2017). Arabic single-document text summarization using particle swarm optimization algorithm. *Procedia Computer Science*, 117:30–37.
- Al-Hashemi, R. (2010). Text summarization extraction system (tses) using extracted keywords. *Int. Arab. J. e Technol.*, 1(4):164–168.
- AL-Khawaldeh, F. T. and Samawi, V. W. (2015). Lexical cohesion and entailment based segmentation for arabic text summarization (Iceas). *World of Computer Science & Information Technology Journal*, 5(3).
- Belkebir, R. and Guessoum, A. (2015). A supervised approach to arabic text summarization using adaboost. In *New contributions in information systems and technologies*, pages 227–236. Springer.
- Boudabous, M. M., Maaloul, M. H., and Belguith, L. H. (2010). Digital learning for summarizing arabic doc-

- uments. In *International Conference on Natural Language Processing*, pages 79–84. Springer.
- Brunetti, S., Costa, M.-C., Frosini, A., Jarray, F., and Picouleau, C. (2007). Reconstruction of binary matrices under adjacency constraints. In *Advances in Discrete Tomography and Its Applications*, pages 125–150. Birkhäuser Boston.
- Froud, H., Lachkar, A., and Ouatik, S. A. (2013). Arabic text summarization based on latent semantic analysis to enhance arabic documents clustering. *arXiv preprint arXiv:1302.1612*.
- Gong, S., Zhu, Z., Wu, W., Zhao, Z., and Zhang, D. (2015). Csss: A novel candidate summary selection strategy for summary-level extractive summarization.
- Haboush, A., Al-Zoubi, M., Momani, A., and Tarazi, M. (2012). Arabic text summarization model using clustering techniques. *World of Computer Science and Information Technology Journal (WCSIT) ISSN*, pages 2221–0741.
- Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.
- Jaradat, Y. A. (2015). *Arabic Single-Document Text Summarization Based on Harmony Search*. PhD thesis, Yarmouk University.
- Jarray, F., Costa, M.-C., and Picouleau, C. (2008). Approximating hv-convex binary matrices and images from discrete projections. In *International Conference on Discrete Geometry for Computer Imagery*, pages 413–422. Springer, Berlin, Heidelberg.
- Jia, R., Cao, Y., Fang, F., Zhou, Y., Fang, Z., Liu, Y., and Wang, S. (2021). Deep differential amplifier for extractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 366–376.
- Keneshloo, Y., Ramakrishnan, N., and Reddy, C. K. (2019). Deep transfer reinforcement learning for text summarization. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 675–683. SIAM.
- Liu, Y. and Lapata, M. (2019). Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.
- Miller, D. (2019). Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*.
- Mohamed, M. and Oussalah, M. (2019). Srl-esa-textsum: A text summarization approach based on semantic role labeling and explicit semantic analysis. *Information Processing & Management*, 56(4):1356–1372.
- Nallapati, R., Zhai, F., and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-first AAAI conference on artificial intelligence*.
- Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B., et al. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.
- Saidi, R. and Jarray, F. (2022). Combining bert representation and pos tagger for arabic word sense disambiguation. In *International Conference on Intelligent Systems Design and Applications*, pages 676–685. Springer.
- Saidi, R., Jarray, F., Kang, J., and Schwab, D. (2022). Gpt-2 contextual data augmentation for word sense disambiguation. In *PACIFIC ASIA CONFERENCE ON LANGUAGE, INFORMATION AND COMPUTATION*.
- Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.
- Tanfouri, I. and Jarray, F. (2022). Genetic algorithm and latent semantic analysis based documents summarization technique. In Aveiro, D., Dietz, J. L. G., and Filipe, J., editors, *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2022, Volume 2: KEOD, Valletta, Malta, October 24-26, 2022*, pages 223–227. SCITEPRESS.
- Tanfouri, I., Tlik, G., and Jarray, F. (2021). An automatic arabic text summarization system based on genetic algorithms. *Procedia Computer Science*, 189:195–202.
- Xu, J. and Durrett, G. (2019). Neural extractive text summarization with syntactic compression. *arXiv preprint arXiv:1902.00863*.
- Zhong, M., Liu, P., Chen, Y., Wang, D., Qiu, X., and Huang, X. (2020). Extractive summarization as text matching. *arXiv preprint arXiv:2004.08795*.