Journal of Electrical and Computer Engineering Innovations (JECEI)
Journal homepage: http://www.jecei.sru.ac.ir
SRTTU
JECEI

Research paper

Efficient GAN-based Method for Extractive Summarization

S.V. Moravvej[1,*], M.J. Maleki Kahaki[2], M. Salimi Sartkahti[3], M. Joodaki[1]

[1]Department of Computer Engineering, Isfahan University of Technology, Isfahan, Iran.

[2]Department of Computer Engineering, University of Kashan, Kashan, Iran.

[3]Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran.

Article Info

Article History:
Received 15 July 2021
Reviewed 27 August 2021
Revised 01 November 2021
Accepted 08 November 2021

Keywords:
Text summarization
Non-greedily
Generative adversarial network
Attention mechanism
Extractive summarization

[*]Corresponding Author's Email Address:
sa.moravvej@ec.iut.ac.ir

Abstract

Background and Objectives: Text summarization plays an essential role in reducing time and cost in many domains such as medicine, engineering, etc. On the other hand, manual summarization requires much time. So, we need an automated system for summarizing. How to select sentences is critical in summarizing. Summarization techniques that have been introduced in recent years are usually greedy in the choice of sentences, which leads to a decrease in the quality of the summary. In this paper, a non-greedily method for selecting essential sentences from a text is presented.
Methods: The present paper presents a method based on a generative adversarial network and attention mechanism called GAN-AM for extractive summarization. Generative adversarial networks have two generator and discriminator networks whose parameters are independent of each other. First, the features of the sentences are extracted by two traditional and embedded methods. We extract 12 traditional features. Some of these features are extracted from sentence words and others from the sentence. In addition, we use the well-known Skip-Gram model for embedding. Then, the features are entered into the generator as a condition, and the generator calculates the probability of each sentence in summary. A discriminator is used to check the generated summary of the generator and to strengthen its performance. We introduce a new loss function for discriminator training that includes generator output, real and fake summaries of each document. During training and testing, each document enters the generator with different noises. It allows the generator to see many combinations of sentences that are suitable for quality summaries.
Results: We evaluate our results on CNN/Daily Mail and Medical datasets. Summaries produced by the generator show that our model performs better than other methods compared based on the ROUGE metric. We apply different sizes of noise to the generator to check the effect of noise on our model. The results indicate that the noise-free model has poor results.
Conclusion: Unlike recent works, in our method, the generator selects sentences non-greedily. Experimental results show that the generator with noise can produce summaries that are related to the main subject.

Introduction

Nowadays, the amount of information in different areas of the World Wide Web in various formats such as web pages, articles, emails, log files, and linked data are increasing sharply [1].

However, much of the information is less important.

On the other hand, manually identifying important information in a large text is time-consuming and practically impossible.

Therefore, we need a system that extracts vital information from the text in the shortest time and with the highest accuracy. This system is called automatic text summarization (ATS) [2]. Primary hypotheses are presented for summarization. A summary is a text produced from one or more texts and is not more than half of the whole text [3]. According to research in [4], summarization is the extraction of the most important information from a source or sources to produce a shorter version for a specific user or task. In other words, this process is one of the crucial steps in diverse cases such as extracting knowledge from text with huge information about the similarity of diseases in the OMIM dataset [5].

The text summarization methods can be categorized in a different view of points. Output-based summarization can be done as extractive or abstractive. Extractive summarization is done by selecting important sentences from different text parts [6], [7]. In abstractive summarization, the system tries to identify the text key concepts and then convert it to another form that is shorter than the original text [8], [9]. Extractive summarization ensures a semantic and grammatical rule between sentences [10]. Input-based summarization may be single-document or multi-document. In multi-document summarization, several documents with the same subject are considered input, and the final summary is the text produced from all the documents [11], [12]. Based on the content, the summary can be general or query-based. In query-based summarization, a summary is generated according to the user's query [13]. This type of summary focuses on the user query and does not consider the general index of concepts in the text. However, in a general summary, the text is produced regardless of the subject and domain. This article presents an extractive, single-document, and general approach.

Since 1950, many solutions for extractive summarization have been offered. Some methods are based on machine learning concepts such as clustering [14], [15] support vector machines [16]. Recently, optimization algorithms [17], [18] like Cuckoo [19], [20] and fuzzy methods [21] have been used for summarization. Graph-based methods are other techniques that have been selected for this purpose [22]. In these methods, each sentence is usually considered as a graph so that the nodes represent the words, and the graph edges express the interval between the words. With recent advances in deep learning, other previous machine learning methods are less commonly utilized in natural language processing tasks [23], [24], [25]. With its relatively complex structure, deep learning can learn the features of words, sentences, or documents automatically [26].

Despite the deep learning-based methods provided for summarization, many of them have not been able to solve the challenges of extractive summarization. The two main components of summarizing are scoring sentences and selecting them [27]. Most previous works suffer from a greedy choice of sentences [10], [28], [29], [30].

In other words, after choosing a high-ranking sentence, they put it aside and do not consider it in choosing the next sentences, which leads to a decrease in the quality of the summary.

Today, Generative Adversarial Networks have been used in many applications of natural language processing, including text generation [31] and question answering [32].

In this research, an extractive summarizer using generative adversarial networks and attention mechanism is presented.

According to our information, this paper presents the first method based on generative adversarial networks for extractive summarization. These networks are made up of two generator and discriminator components that compete in a process.

In this context, the generator goal is to rate each sentence of the document, while the goal of the discriminator is to distinguish the real from the fake summary, which enhances the performance of the generator.

In a non-greedy way, the generator determines the possibility of the presence of sentences in summary at once.

Other contributions to this article are as follows:

- The generator is trained with the feedback it receives from the discriminator. Therefore, if fake summaries are introduced to the discriminator, it can prevent the generator from producing poor-quality summaries. We extract some fake and real summaries from each document and utilize them for discriminator training, which leading to a new loss function for it.
- Another important characteristic of the proposed model is producing multiple summaries for each document during training and testing. During the training, each document enters the generator with different noises, which the model identifies different and appropriate combinations of sentences that produce quality summaries. During the test, we enter each document into the generator with different noises and use the voting system for the final summary.

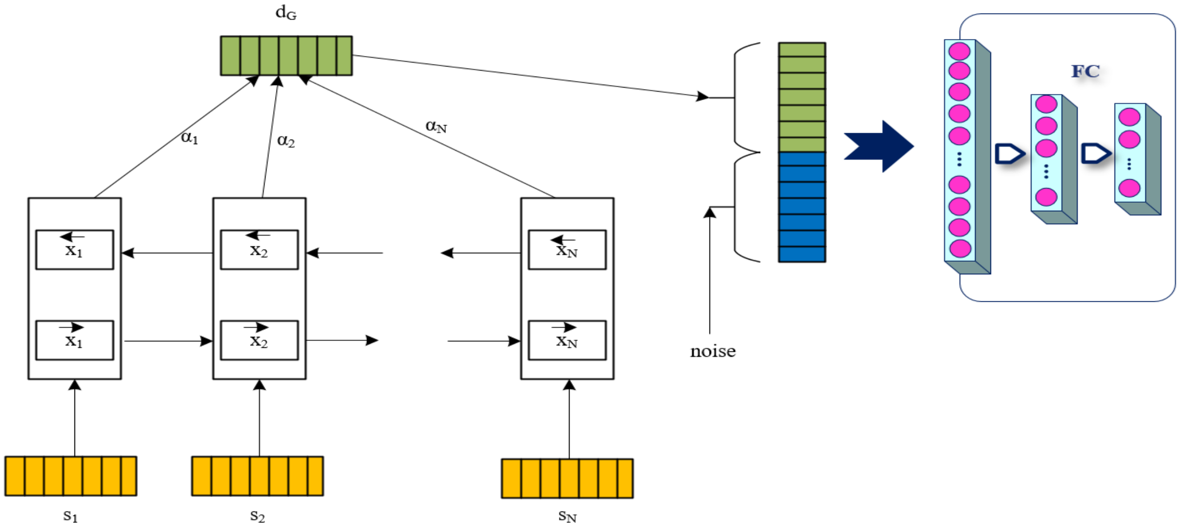

Fig. 1: Our generator architecture.

We evaluate our proposed model on two datasets. In the first evaluation, we use the CNN/Daily Mail dataset[1]. This dataset is a benchmark for evaluating many of the works. In the second application, we utilize the medical dataset available in PubMed Central[2]. The evaluation results show that the proposed model can provide high-quality summaries compared to other compared methods.

Related Work

The two main categories of a summary are abstractive and extractive. In most abstractive methods, an Auto-Encoder is used, in which important features of the text are extracted in the Encoder. In the Decoder section, a summary is generated using the features extracted in the encoder section [33], [34]. So far, many methods for extractive summarization, including graph-based [35], [36], [37], [38] and deep learning, have been proposed. In the following, several extractive summarizations work based on deep learning methods are described.

The works presented in [39], [40] use an auto-encoder to learn sentence features. The authors in [39] make the document word features and then calculate the scores of the sentences using the scores of its words. [40] uses cosine similarity between sentence and subject to score sentences. In [27], the authors used recurrent neural networks to rank sentences. The authors considered each sentence as a tree where the words are on the leaves, and the sentence is at the root. The score of each sentence is obtained from the leaves based on a non-linear process.

Another work in [10] uses reinforcement learning for summarizing.

[1] https://cs.nyu.edu/~kcho/DMQA/

[2] https://www.ncbi.nlm.nih.gov/pubmed/

In this work, the coherence between sentences is considered as a reward. The policy is implemented as a multilayer perceptron that assigns a score to each sentence. The authors in [28] introduced a model called Summarunner for extraction summarization based on RNN networks. They used two RNN layers to embed words and sentences. Then, they employed logistic regression to classify sentences.

Recently, the attention mechanism [41] has gained much attention in many areas, including machine translation [42], question answering [43], and text summarization. In [29], the authors proposed a method based on the Siamese neural network (SNN) and the attention mechanism. They utilized the attention mechanism for words and sentences to score. They estimated the features of sentences using the obtained word features. Finally, they calculated the features of the document according to its sentences and used a classifier for the similarity of the summary and the document. [30] uses the hierarchical structure self-attention method to embed sentences and documents. The proposed method model's summarization as a classification problem in which it calculates sentence-summary probability.

Recently, Bidirectional Encoder Representations from Transformers (BERT) [44] has revolutionized the processing of natural languages. BERT is a pre-trained language model [24] for textual data. In [45], a simple version of BERT called BERTSUM is provided for extractive summarization.

In another example in [46], a method is presented for abstractive and extractive summarization. The proposed method can obtain document semantics using the BERT model.

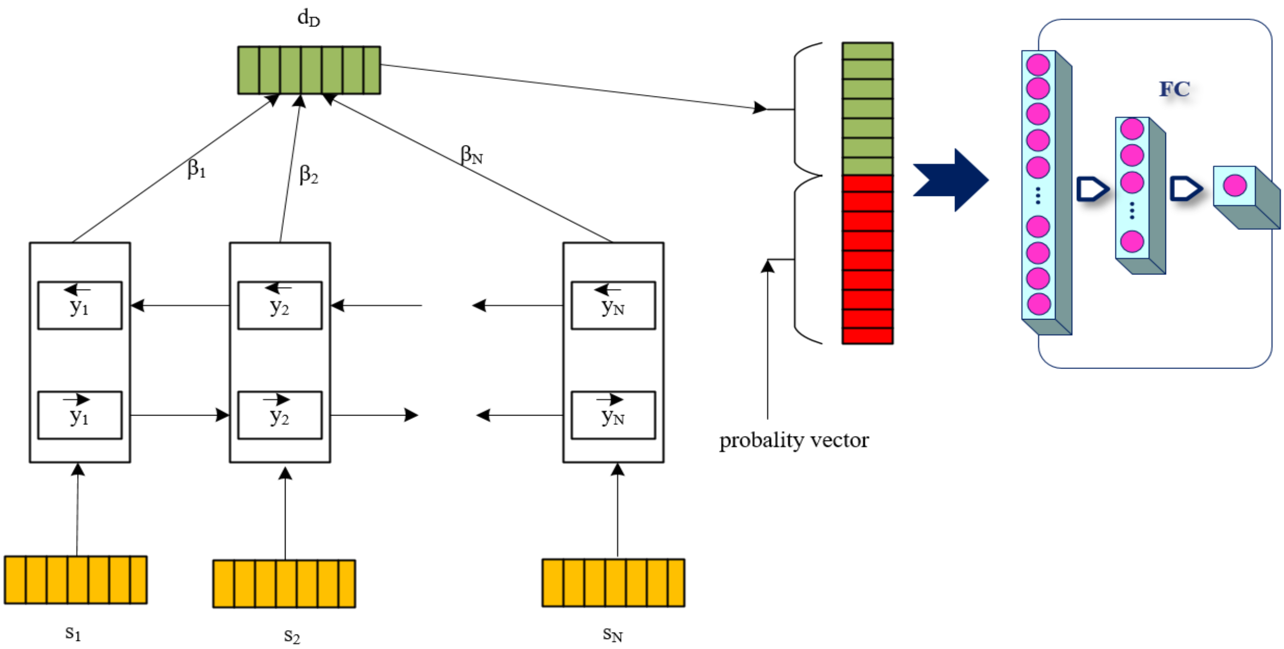

Fig. 2: Our discriminator architecture.

The Proposed Model

In this research, we use adversarial generating networks for extractive summarization. We employ this network to improve the problems of previous methods, including greed. We will first have a description of this network, and then the proposed model is presented.

Generative adversarial networks (GANs) were first proposed by Goodfellow et al. [47]. These networks consist of two separate networks that are similarly trained: the generator and discriminator networks. The purpose of the generator is to produce data such as images, text, etc., which are structurally similar to real data but are fake.On the other hand, the task of the discriminator network is to strengthen the generator.

These two networks play a two-player min-max game with a value function $V(D.G)$ as follows [47]:

$$\min_G \max_D V(D.G) = E_{x \sim p_{data}(x)}[log(D(x))] + E_{z \sim p_z(z)}[log(1 - D(G(z)))] \quad (1)$$

where x and z are input data and noise, respectively. G and D mean the generator and discriminator, respectively. $p_{data}(x)$ and $p_z(z)$ represent the input data distribution and the noise distribution, respectively. E is mathematical expectation. Generative adversarial networks can be extended to a conditional model If condition y is added to the generator and discriminator input. The value function, in this case, changes as follows [48]:

$$\min_G \max_D V(D.G) = E_{x \sim p_{data}(x)}[log(D(x|\mathbf{y}))] + E_{z \sim p_z(z)}[log(1 - D(G(z|\mathbf{y})))] \quad (2)$$

The proposed generator and discriminator model are shown in Fig. 1 and Fig. 2, respectively. We use sentence features as a condition in the generator and discriminator. Let $D = \{s_1.s_2 \dots . s_N\}$ represents the document, where the $s_i \in \mathbb{R}^d$ is the extracted features of the $i-th$ sentence. N is the length of document D, which is equal to the number of restricted sentences in each document.

The attention mechanism calculates the representation vector of the document in the generator and the discriminator according to the following equations:

$$d_G = \sum_{i=1}^{N} \alpha_i \, [\overleftarrow{x}_i . \vec{x}_i] \quad (3)$$

$$d_D = \sum_{i=1}^{N} \beta_i \, [\overleftarrow{y}_i . \vec{y}_i] \quad (4)$$

where $\overleftarrow{x}_i \in \mathbb{R}^{d_1}$, $\vec{x}_i \in \mathbb{R}^{d_1}$, $\overleftarrow{y}_i \in \mathbb{R}^{d_2}$, $\vec{y}_i \in \mathbb{R}^{d_2}$ are the output of step i in BLSTM. α_i and β_i are the coefficients of attention for the i-th sentence in the generator and the discriminator, respectively, which are formulated as follows:

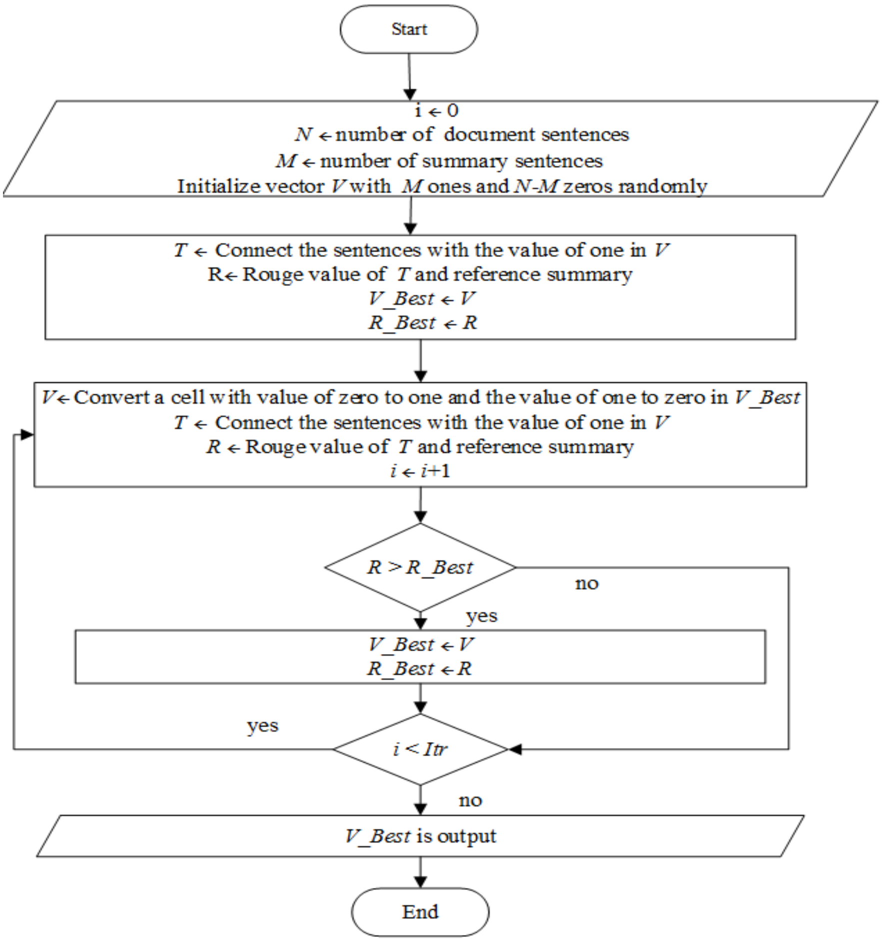

Fig. 3: Generate a real summary for the document.

$$\alpha_i = \frac{e^{u_i}}{\sum_{i=1}^{N} e^{u_i}} \tag{5}$$

$$\beta_i = \frac{e^{v_i}}{\sum_{i=1}^{N} e^{v_i}} \tag{6}$$

$$u_i = tanh(W_u[\overleftarrow{x}_i . \vec{x}_i] + b_u) \tag{7}$$

$$v_i = tanh(W_v[\overleftarrow{y}_i . \vec{y}_i] + b_v) \tag{8}$$

where $W_u \in \mathbb{R}^{2.d_1}$. $b_v \in \mathbb{R}$, $W_v \in \mathbb{R}^{2.d_2}$. and $b_v \in \mathbb{R}$ are the parameters of the attention mechanism for documents.

In Fig. 1, the representation vector of the document is connected to the noise vector and enters a feed-forward neural network. The last layer of this network calculates the probability of the presence of each sentence. Noise causes the generator to produce different outputs. Each document enters the generator to be summarized in different iterations with different noises. The generator tries to produce different summaries of almost the same quality for each document. It allows the generator to identify different combinations of sentences that are appropriate for the summary. Therefore, sentences that may not be useful for the summary alone can lead to a quality summary by being placed next to other sentences.

In the discriminator network, the probability vector of sentences is contacted with the representation vector of the document (see Fig. 2). In this context, the probability vector of sentences is the vector of the number of sentences in a document, and each element is zero or one.

A. The Target Vector

In the general generative adversarial network, the generator output is used as fake data for discriminator training. In addition, a real target is extracted for each sample. In this research, in order to introduce quality summaries to the discriminator, more than one summary is extracted from each document, which is similar in terms of quality. On the other hand, we produce several poor-quality summaries for the document. The real summaries of each document are text and cannot be employed as a target. Therefore, we need a method to express the presence or absence of each sentence in summary as a number. For this purpose, a vector is defined with N element for each document, where N is the number of sentences. Each element of this vector has a value of zero or one. The value of one indicates the presence of a sentence in summary.

We employ a greedy method according to Fig. 3 to produce this vector. First, we get a vector with length N and M number one, where M is the number of sentences in summary. The values of one are randomly arranged in the vector. The sentences corresponding to the value of one are put together in this vector to produce a summary, and the ROUGE metric measures their quality.

After that, a randomly chosen one is converted to zero, and a randomly chosen zero to one, and the Rouge value is recalculated. If its value is better than the previous one, it will be replaced. This process is repeated for Itr times, and the best vector during the process is considered the output. Note that in order to produce any real target, the algorithm must be executed from the beginning. The process of making a fake target is similar to a real target, except that the vector will replace the previous one if the Rogue score is lower. The length of all documents is limited to N sentences. Documents longer than N sentences are cut, and smaller documents are zero-padding.

B. Loss Function

The Loss function is calculated based on the discriminator output for the generator as follows [47]:

$$Loss_G = E_{i\sim Dataset}\left[E_{z\sim p_{z(z)}}\left[log\left(1 - D\left(G(z|y_i)\right)\right)\right]\right] \tag{9}$$

where $Dataset$ is a set of documents, y_i is features of sentences in document i, and E is the mathematical expectation.

The Loss function for the discriminator is computed based on the generator output, real and fake summaries as follows:

$$\begin{aligned} Loss_D &= E_{i\sim Dataset}[\, E_{z\sim p_{z(z)}}[log(1 - D(G(z|y_i)|\, y_i))] \\ &+ E_{k\sim p_{Fake_i}}\left[log\left(1 - D(k|y_i)\right)\right] \\ &+ E_{k\sim p_{Real_i}}[log(D(l|y_i))]] \end{aligned} \tag{10}$$

where p_{Real_i} and p_{Fake_i} show the distribution of real and fake summaries for the document i. Equation (10) forces the discriminator to learn a set of high-quality and low-quality summaries.

On the other hand, be sensitive to the summaries produced by the generator and force the generator to produce a high-quality summary.

C. Summarization

At the time of testing, only the generator is used to generate the summary. By applying different noises in the generator, multiple summaries can be generated for each document, and the quality of these summaries or ROUGE summaries is very close to each other. We consider the voting system to generate a single summary for the document. To do this, the probability of the presence of sentences in summary is calculated for different noises. After that, the sentences are ranked based on their number of selections, and finally, the sentences with the highest rank are selected.

Results and Discussion

A. Feature Extraction

One of the important components of deep learning is feature extraction. There are many ways to do this. In this research, we use two methods to select features. In the first method, we use 12 traditional features. The list of these features is given in Table 1.

Some of these features are at the word level, and some of them at the sentence level. All features are scaled to [0,1].

An important feature of deep models is the automatic learning of features. Sentence features are used as a condition in the model and cannot be changed during training. We use Skip-Gram [49] to embed words. Finally, by averaging word embedding, we extract sentence embedding.

Table 1: Traditional features

Feature	Description
Common Word	The number of occurrences N common words in the dataset, divided by the sentence length.
Position	The position of the sentence. Supposing there are N sentences in the document, for j the sentence, the position is computed as $1-(j-1)/(N-1)$.
Length	The number of words in the sentence, divided by the length of the largest sentence.
Number Raito	The number of digits, divided by the sentence length.
Named entity ratio	The number of named entities, divided by the sentence length.
Tf/Isf	Term frequency over the sentence, divided by the largest term frequency.
Similarity Sentence	The number of occurrences of words in the sentence with the highest Tf/Isf in the sentence divided by the length of the sentence.
None phrase	The number of None phrases, divided by the sentence length.
Pos Ratio	A four-dimensional vector containing the number of nouns, verbs, adjectives, and adverbs. Each vector cell is divided by the sentence length.

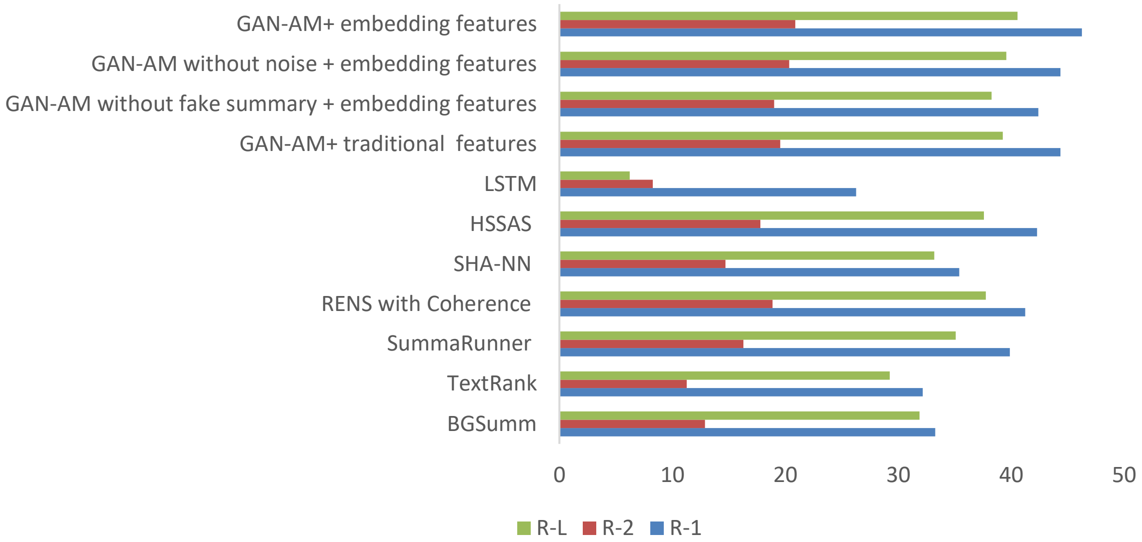

Fig. 4: Graphical comparison of the proposed model and other methods on the CNN/Daily Mail dataset.

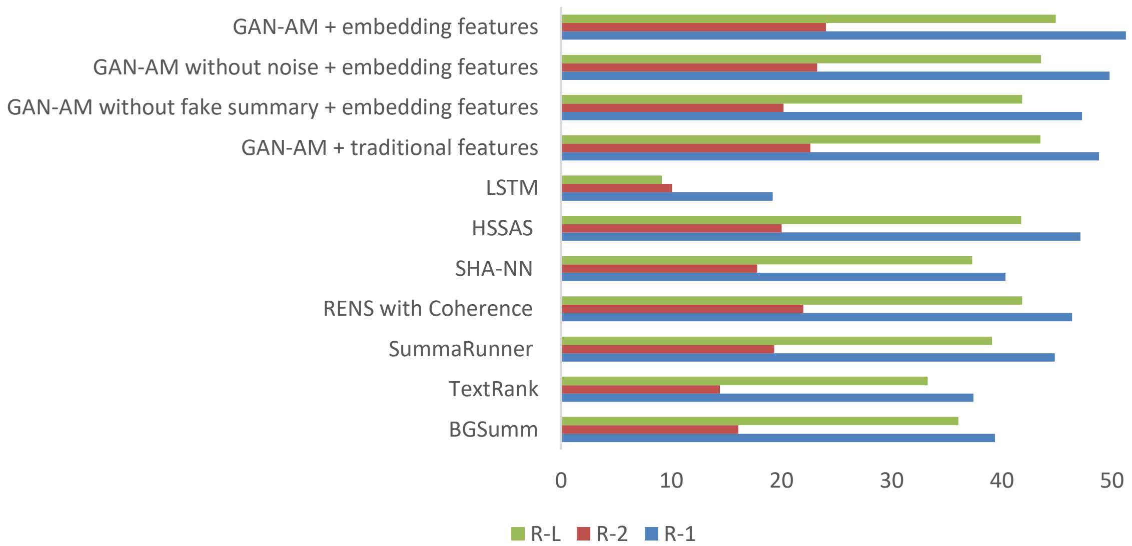

Fig. 5: Graphical comparison of the proposed model and other methods on the Medical dataset.

B. Dataset

We use two known datasets for our evaluations: CNN/Daily Mail and PubMed. The first dataset [50] combines two datasets designed for comprehension, extractive, and abstractive tasks. The datasets have come to the attention of researchers in recent years for automated summarization. This dataset contains 287,226 documents for training, 13,368 for validation and 11,490 for testing. The average number of sentences per document in training data is 28 sentences. The average reference summary of each document is 3-4 sentences, and the average number of words per document in training data is 802 words [28]. More details are available in Table 2.

This dataset consists of two versions. In the first version, all entities are replaced with specific words, while the second version is the original data. We adopt the second version for our model.

Table 2: Statistics of the CNN/Daily Mail dataset

	Train	Validation	Test
Pairs of data	287,113	13,368	11,490
Article length	749	769	778
Summary Length	55	61	58

The second collection is PubMed, which contains many articles in the field of medicine.

The number of these articles is increasing every day. For this purpose, we have randomly downloaded 1000 articles. The dataset was divided into 1334 documents for training, 288 documents for validation, and 378 documents for testing.

C. Detail of Model

In this research, Python language and PyTorch library have been used for implementation. Jupyter has been employed to implement project codes. Another library used in this research is the NLTK library. This library provides classes and methods for processing natural languages in Python. This library can perform a wide range of natural language processing operations.

We employ a two-layer bidirectional LSTM. In generative adversarial networks, the discriminator converges to the optimum point sooner, which causes the generator cannot converge. For this reason, we train the discriminator once for every 15 generator training. In addition, due to the connection of vectors in the two networks, we use batch normalization before the data enters the feed-forward neural network. Table 3 shows the values of the other parameters.

Table 3: The parameters of the model

Parameter	CNN/Daily Mail	Medical
batch size	128	64
embedding dim	60	60
max sentence length	100	50
real summary per document	40	15
fake summary per document	40	15
activation fun(lstm & dense)	relu	relu
dense hidden layer	8	5

D. Metrics

We employ the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) package [51] as an evaluation metric in our experiments.

This metric calculates the similarity between the generated summary and the reference summary by counting the number of common units. Rouge-n recall between an extracted summary and a reference summary is calculated as follows:

$$\text{Rouge-}n = \frac{\sum_{s\,\in\{ref\ sum\}}\sum_{gram_n\,\in\,s} Count_{match}\,(gram_n))}{\sum_{s\,\in\{ref\ sum\}}\sum_{gram_n\,\in\,s} Count\,(gram_n))} \tag{11}$$

where n stands for the length of n-gram, $Count_{match}\ (gram_n)$ is the maximum number of n- gram co-occurring in the extracted summary and the reference summary. Rouge-1 and Rouge-2 are special cases of Rouge-n in which $n = 1$ or $n = 2$. R-L calculates the length of the longest common subsequence between the reference summary and the extracted summary. Based on previous works, Rouge-1(R-1), Rouge-2(R-2) and, Rouge-L(R-L) are most widely used in summarization. For this reason, we use these three metrics in all our experiments.

Table 4: Numerical comparison of the proposed method and other methods on the CNN/Daily Mail dataset

Model	R-1	R-2	R-L
BGSumm [35]	33.27	12.90	31.89
TextRank [38]	32.18	11.26	29.26
SummaRunner [28]	39.60	16.20	35.30
RENS with Coherence [10]	41.25	18.87	37.75
SHA-NN [29]	35.40	14.7	33.2
HSSAS [30]	42.30	17.80	37.60
LSTM	26.28	8.27	6.23
GAN-AM + traditional features	44.36	19.56	39.26
GAN-AM without fake summary + embedding features	42.42	19.02	38.26
GAN-AM without noise + embedding features	44.38	20.34	39.58
GAN-AM + embedding features	**46.26**	**20.89**	**40.56**

E. Experimental Results and Analysis

We consider two techniques for preprocessing in all our experiments: 1- Stop word removal 2- Stemming.

Our project uses a 64-bit Windows operating system with 64 GB of RAM and GPU. The best model was obtained for the CNN/Daily Mail dataset after 50 epochs, while [29] obtained the best model after 70 epochs. The whole process of our training took 5 hours. The best model for the Medical dataset was obtained after 30 epochs. This process took 1.5 hours.

Table 5: Numerical comparison of the proposed method and other methods on the medical dataset

Model	R-1	R-2	R-L
BGSumm [35]	39.37	16.09	36.06
TextRank [38]	37.44	14.40	33.28
SummaRunner [28]	44.82	19.36	39.12
RENS with Coherence [10]	46.39	22.01	41.85
SHA-NN [29]	40.33	17.80	37.32
HSSAS [30]	47.13	20.03	41.76
LSTM	19.21	10.08	9.15
GAN-AM + traditional features	48.84	22.64	43.52
GAN-AM without fake summary + embedding features	47.29	20.19	41.86
GAN-AM without noise + embedding features	49.78	23.24	43.58
GAN-AM + embedding features	**51.26**	**24.04**	**44.91**

Table 6: Execution time of algorithms (in milliseconds)

Length	BGSumm [35]	TextRank [38]	SummaRunner [28]	RENS with Coherence [10]	SHA-NN [29]	HSSAS [30]	GAN-AM
3209	285	250	319	369	332	402	315
3590	319	285	364	413	363	459	346
4006	343	304	401	468	418	509	391
4592	399	369	464	537	482	589	450
5630	496	449	582	751	603	736	563
6972	603	539	710	809	742	860	680
7460	642	572	770	882	793	952	756
8905	801	709	897	1020	942	759	882
9790	873	751	991	1142	1032	1221	958
10196	1006	829	1148	1193	1081	1386	1100

The proposed method is compared with two graph-based methods BGSumm [35], TextRank [38], four deep learning methods SummaRunner [28], RENS with Coherence [10], SHA-NN [29], HSSAS [30] and one basic method LSTM. The LSTM model uses only our generator part. The evaluation results of the proposed system for the two datasets are shown in Table 4 and Table 5. For the CNN/Daily Mail dataset, the results reported for the SummaRunner, RENS with Coherence, SHA-NN methods in [28], [10], and [29] are given in Table 4. BGSumm, TextRank, and HSSAS methods were obtained in the experimentation we did in our laboratory. The total results of Table 5 have been obtained in our laboratory.

As expected, for both datasets, deep learning methods are superior to graph-based methods. Although BGSumm has been tested on a medical dataset, it has failed deep learning methods even in the medical dataset. In general, deep learning-based models are weaker than our model in the two datasets. The RENS with Coherence method is less accurate than our model, although it considers coherence between sentences.

Table 7: Effect of noise on the proposed model for the CNN/Daily Mail dataset

Noise Size	R-1	R-2	R-L
0	44.38	20.34	39.58
10	44.80	20.47	39.62
20	45.34	20.53	39.68
30	45.76	20.6	39.7
40	45.86	20.72	39.92
50	46.13	20.8	40.45
60	46.26	20.89	40.56
70	46.14	20.7	40.42
80	46.07	20.55	40.32
90	45.89	19.92	39.9
100	44.74	19.76	39.72

In addition, the embedding features perform better than the traditional features on our model. By comparing the LSTM model with the GAN-AM method, the importance of the discriminator is seen in our model. As we can see, our method has a relatively strong weakness compared to other models when it does not use a discriminator. In addition, in another experiment, we examined the importance of fake summaries for discriminator training. GAN-AM without fake summary + embedding features shows this model. The difference between the results of this model and our best model for both datasets is noticeable. The superiority of the presented model can be considered in the way the generator scoring of sentences. The generator assigns scores to each sentence, taking into account the rest of the sentences. Another reason that can be mentioned is the voting system used. However, the discriminator has not been ineffective in producing a summary by the generator. To better understanding of results, the evaluation results are shown schematically in Fig. 4 and Fig. 5. To check the time of the algorithms, we selected 10 test documents from the CNN/Daily Mail dataset. The production time of the summary by each method is shown in Table 6. As we can see, graph-based methods take less time than deep learning methods due to less computation. The GAN-AM method consists of an LSTM network and a feed-forward network, so it is expected to take some time to calculate. However, other methods take time to calculate embedding.

Table 8: Effect of noise on the proposed model for medical dataset

Noise Size	R-1	R-2	R-L
0	49.78	23.24	43.58
10	50.69	23.6	43.62
20	50.8	23.72	43.86
30	51.05	23.95	44.78
40	51.26	24.04	44.91
50	51.03	23.86	44.8
60	49.8	23.7	44.62
70	49.52	23.51	44.15
80	49.02	23.42	43.82
90	48.61	22.86	43.42
100	48.2	22.62	42.62

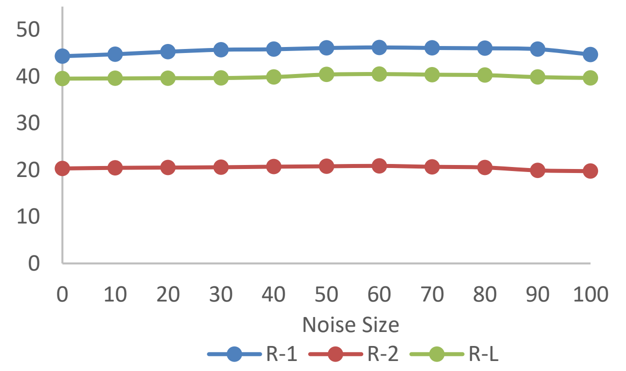

Fig. 6: Results of the proposed model for different noises on the CNN / Daily Mail dataset.

We performed other experiments to determine the effect of noise on the generator. For this purpose, we apply noise of different sizes to the generator. The results R-1, R-2, and R-L for the two datasets are shown in Table 7 and Table 8. For the CNN / Daily Mail dataset, increasing the noise size to 60 raises the R-1, R-2, and R-L criteria, but we have a downtrend from 60 to 100. In this database, the best size for noise is 60. For the Medical dataset, we have an uptrend from 0 to 40 and then a downtrend. For this data set, the noise size is set to 40. As we can see, the proposed model has better performance with noise. For a better understanding, the results for the two datasets are shown in Fig. 6 and Fig. 7.

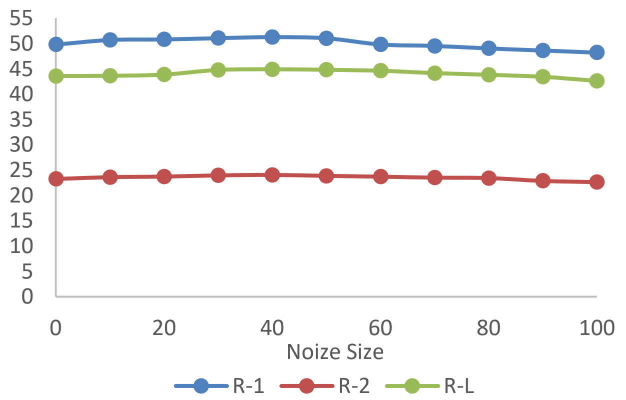

Fig. 8: Results of the proposed model for different noises on the Medical dataset.

As an example of the generator output, the three sentences extracted from a document in the Medical dataset by the generator along with its reference summary are shown in Fig. 8. Common words between each sentence and reference are highlighted. As we can see, the sentences that have more in common with the reference summary are given higher scores.

Conclusion and Future Work

In this study, a method based on the generative adversarial network for extractive summarization was proposed in which sentence features were considered a condition in this network. Sentence features were extracted based on traditional and embedding methods. The generator utilized sentence features to calculate the probabilities of their presence. In addition, due to the use of noise in the generator, several summaries were generated for each document. We set up a new loss function for the discriminator. Experiments have shown that this function can be effective in generating summaries.

In future work, we will consider the coherence between sentences in our model. As a solution, we can consider coherence when constructing the target or as a loss in the generator.

Sentence	Score
Stancheva said she and other doctors including a psychiatrist diagnosed Burkhart with **"autism, severe anxiety, post-traumatic stress disorder and depression."**	0.92
Burkhart, a 24-year-old **German national**, has been charged with 37 counts of arson following a **string** of 52 **fires in Los Angeles**	0.82
A medical **doctor** in Vancouver, British Columbia, said Thursday that California arson suspect Harry **Burkhart** suffered from severe mental illness **in 2010**, when she **examined** him as **part of a team** of doctors.	0.71

Real Summary:
A Canadian **doctor** says she was **part of a team examining** Harry **Burkhart in 2010**,Diagnosis: **"autism, severe anxiety, post-traumatic stress disorder and depression"**,Burkhart is also suspected in a German arson probe, officials say,Prosecutors believe the **German national** set **a string** of **fires in Los Angeles**.

Fig. 7: Three extracted sentences by the generator of the Medical dataset.

Author Contributions

S.V. Moravvej suggested the model and innovation of the problem and wrote the manuscript. M.J. Maleki Kahaki wrote the code of this article. M. Salim designed the experiments. M. Joodaki collected the dataset and edited the text of the article. All authors discussed the results.

Conflict of Interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Abbreviations

ATS	Automatic Text Summarization
GANs	Generative Adversarial Networks
ROUGE	Recall-Oriented Understudy for Gisting Evaluation

References

[1] S. Samiei, M. Joodaki, N. Ghadiri, "A scalable pattern mining method using apache spark platform," in Proc. 7th International Conference on Web Research (ICWR): 114-118, 2021.

[2] S.V. Moravvej, A. Mirzaei, M. Safayani, "Biomedical text summarization using Conditional Generative Adversarial Network (CGAN)," arXiv preprint arXiv:2110.11870, 2021.

[3] R. Mitkov, The Oxford handbook of computational linguistics. Oxford University Press, 2004.

[4] I. Mani, M. Maybury, "Automatic summarization John Benjamin's publishing Co," 2001.

[5] M. Joodaki, N. Ghadiri, Z. Maleki, M.L. Shahreza, "A scalable random walk with restart on heterogeneous networks with Apache Spark for ranking disease-related genes through type-II fuzzy data fusion," J. Biomed. Inf., 115: 103688, 2021.

[6] M. Kågebäck, O. Mogren, N. Tahmasebi, D. Dubhashi, "Extractive summarization using continuous vector space models," in Proc. 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC): 31-39, 2014.

[7] M. Jang, P. Kang, "Learning-free unsupervised extractive summarization model," IEEE Access, 9: 14358-14368, 2021.

[8] Y. Liu, P. Liu, "SimCLS: A simple framework for contrastive learning of abstractive summarization," arXiv preprint arXiv:2106.01890, 2021.

[9] Z. Cao, F. Wei, W. Li, S. Li, "Faithful to the original: Fact aware neural abstractive summarization," in Proc. the AAAI Conference on Artificial Intelligence, 32(1), 2018.

[10] Y. Wu, B. Hu, "Learning to extract coherent summary via deep reinforcement learning," in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[11] A. Abdi, S. Hasan, S.M. Shamsuddin, N. Idris, J. Piran, "A hybrid deep learning architecture for opinion-oriented multi-document summarization based on multi-feature fusion," Knowledge-Based Syst., 213: 106658, 2021.

[12] S. Lamsiyah, A. El Mahdaouy, B. Espinasse, S.E.A. Ouatik, "An unsupervised method for extractive multi-document summarization based on centroid approach and sentence embeddings," Expert Syst. Appl., 167: 114152, 2021.

[13] S. Murarka, A. Singhal, "Query-based single document summarization using hybrid semantic and graph-based approach," in Proc. 2020 International Conference on Advances in Computing, Communication & Materials (ICACCM): 330-335, 2020.

[14] S. Akter, A.S. Asa, M.P. Uddin, M.D. Hossain, S.K. Roy, M.I. Afjal, "An extractive text summarization technique for Bengali document (s) using K-means clustering algorithm," in Proc. 2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR): 1-6, 2017.

[15] M. Joodaki, N. Ghadiri, A.H. Atashkar, "Protein complex detection from PPI networks on Apache Spark," in Proc. 2017 9th International Conference on Information and Knowledge Technology (IKT): 111-115, 2017.

[16] T. Hirao, H. Isozaki, E. Maeda, Y. Matsumoto, "Extracting important sentences with support vector machines," in Proc. COLING 2002: The 19th International Conference on Computational Linguistics, 2002.

[17] S.Y. Ashkoofaraz, S.N.H. Izadi, M. Tajmirriahi, M. Roshanzamir, M.A. Soureshjani, S.V. Moravvej, M. Palhang, AIUT3D 2018 Soccer Simulation 3D League Team Description Paper.

[18] S. Vakilian, S.V. Moravvej, A. Fanian, "Using the Artificial Bee Colony (ABC) algorithm in collaboration with the fog nodes in the internet of things three-layer architecture," in Proc. 29th Iranian Conference on Electrical Engineering (ICEE): 509-513, 2021.

[19] S.H. Mirshojaei, B. Masoomi, "Text summarization using cuckoo search optimization algorithm," J. Comp. Rob., 8(2): 19-24, 2015.

[20] S. Vakilian, S. V. Moravvej, A. Fanian, "Using the cuckoo algorithm to optimizing the response time and energy consumption cost of fog nodes by considering collaboration in the fog layer," in Proc. 2021 5th International Conference on Internet of Things and Applications (IoT): 1-5, 2021.

[21] F.B. Goularte, S.M. Nassar, R. Fileto, H. Saggion, "A text summarization method based on fuzzy rules and applicable to automated assessment," Expert Syst. Appl., 115: 264-275, 2019.

[22] W.S. El-Kassas, C.R. Salama, A.A. Rafea, H.K. Mohamed, "EdgeSumm: Graph-based framework for automatic text summarization," Inf. Proc. Manage., 57(6): 102264, 2020.

[23] S.V. Moravvej, M. Joodaki, M.J.M. Kahaki, M.S. Sartakhti, "A method based on an attention mechanism to measure the similarity of two sentences," in Proc. 2021 7th International Conference on Web Research (ICWR): 238-242, 2021.

[24] M.S. Sartakhti, M.J.M. Kahaki, S.V. Moravvej, M. javadi Joortani, A. Bagheri, "Persian language model based on BiLSTM model on COVID-19 corpus," in Proc. 2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA): 1-5, 2021.

[25] S.V. Moravvej, M.J.M. Kahaki, M.S. Sartakhti, A. Mirzaei, "A method based on attention mechanism using Bidirectional Long-Short Term Memory (BLSTM) for Question Answering," in 2021 29th Iranian Conference on Electrical Engineering (ICEE): 460-464, 2021.

[26] H. Liang, X. Sun, Y. Sun, Y. Gao, "Text feature extraction based on deep learning: a review," EURASIP J. Wireless Commun. Networking, 2017(1): 1-12, 2017.

[27] Z. Cao, F. Wei, L. Dong, S. Li, M. Zhou, "Ranking with recursive neural networks and its application to multi-document summarization," in Proc. the AAAI Conference on Artificial Intelligence, 29(1): 2015.

[28] R. Nallapati, F. Zhai, B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in Proc. Thirty-First AAAI Conference on Artificial Intelligence: 3075-3081, 2017.

[29] J.Á. González, E. Segarra, F. García-Granada, E. Sanchis, L.ı.F. Hurtado, "Siamese hierarchical attention networks for extractive summarization," J. Intell. Fuzzy Syst., 36(5): 4599-4607, 2019.

[30] K. Al-Sabahi, Z. Zuping, M. Nadher, "A hierarchical structured self-attentive model for extractive document summarization (HSSAS)," IEEE Access, 6: 24205-24212, 2018.

[31] W. Nie, N. Narodytska, A. Patel, "Relgan: Relational generative adversarial networks for text generation," in Proc. International conference on learning representations, 2018.

[32] M. Lewis, A. Fan, "Generative question answering: Learning to answer the whole question," in Proc. International Conference on Learning Representations, 2018.

[33] T. Vo, "SGAN4AbSum: A Semantic-Enhanced Generative Adversarial Network for Abstractive Text Summarization," 2021.

[34] R. Paulus, C. Xiong, R. Socher, "A deep reinforced model for abstractive summarization," arXiv preprint arXiv:1705.04304, 2017.

[35] M. Moradi, "Frequent itemsets as meaningful events in graphs for summarizing biomedical texts," in Proc. 2018 8th International Conference on Computer and Knowledge Engineering (ICCKE): 135-140, 2018.

[36] Z. Wu, R. Koncel-Kedziorski, M. Ostendorf, H. Hajishirzi, "Extracting summary knowledge graphs from long documents," arXiv preprint arXiv:2009.09162, 2020.

[37] T. Safavi, C. Belth, L. Faber, D. Mottin, E. Müller, D. Koutra, "Personalized knowledge graph summarization: From the cloud to your pocket," in Proc. 2019 IEEE International Conference on Data Mining (ICDM): 528-537, 2019.

[38] R. Mihalcea, P. Tarau, "Textrank: Bringing order into text," in Proc. 2004 Conference on Empirical Methods in Natural Language Processing: 404-411, 2004.

[39] S.H. Zhong, Y. Liu, B. Li, J. Long, "Query-oriented unsupervised multi-document summarization via deep learning model," Expert Syst. Appl., 42(1): 8146-8155, 2015.

[40] M. Yousefi-Azar, L. Hamey, "Text summarization using unsupervised deep learning," Expert Syst.Appl., 68: 93-105, 2017.

[41] S.V. Moravvej, S.J. Mousavirad, M.H. Moghadam, M. Saadatmand, "An LSTM-based plagiarism detection via attention mechanism and a population-based approach for pre-training parameters with imbalanced classes," arXiv preprint arXiv:2110.08771, 2021.

[42] Q. Dou, Y. Lu, P. Manakul, X. Wu, M.J. Gales, "Attention forcing for machine translation," arXiv preprint arXiv:2104.01264, 2021.

[43] Y. Zhang, Y. Peng, "Research on answer selection based on LSTM," in Proc. 2018 International Conference on Asian Language Processing (IALP): 357-361, 2018.

[44] J. Devlin, M.W. Chang, K. Lee, K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[45] Y. Liu, "Fine-tune BERT for extractive summarization," arXiv preprint arXiv:1903.10318, 2019.

[46] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," arXiv preprint arXiv:1908.08345, 2019.

[47] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, "Generative adversarial nets," Advances in Neural Information Processing Systems, 27, 2014.

[48] M. Mirza, S. Osindero, "Conditional generative adversarial nets," arXiv preprint arXiv:1411.1784, 2014.

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

[50] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, P. Blunsom, "Teaching machines to read and comprehend," Advances in Neural Information Processing Systems, 28: 1693-1701, 2015.

[51] C.Y. Lin, "Rouge: A package for automatic evaluation of summaries," in Text summarization branches out: 74-81, 2004.

Biographies

Seyed Vahid Moravvej received B.Sc. in software engineering from University of Kashan and MSc in artificial intelligence from Isfahan University of Technology. His research interests are Data Mining, Deep Learning, and Optimization Algorithms. He also has experience in Python.

- Email: sa.moravvej@ec.iut.ac.ir
- ORCID: 0000-0002-8286-9521
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: https://scholar.google.com/citations?user=cMK6FN4AAAAJ&hl=en

Mohammad Javad Maleki Kahaki graduated in B.Sc. of software engineering from University of Kashan. His research interests are Data Mining and deep learning. He also has experience in Python, Matlab.

- Email: mjmaleki@std.kashanu.ac.ir
- ORCID: NA
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: NA

Moein Salimi Sartakhti has B.Sc. in software engineering from University of Kashan. He is Master student in software engineering from Amirkabir University of Technology. His research interests are NLP and Graph theory. He also has experience in Python.

- Email: moein.salimi@aut.ac.ir
- ORCID: 0000-0003-3945-6040
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: NA

Mehdi Joodaki received his B.S. degree in Computer Engineering (software) from Lorestan University in 2016 and his M.Sc. degree in Computer Engineering (software) from Isfahan University of Technology in 2019. His research interests include Bioinformatics, Parallel Computing, Graph mining, Deep Learning and Data mining.

- Email: mehdi.joodaki@ec.iut.ac.ir
- ORCID: NA
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: NA

How to cite this paper:

S.V. Moravvej, M.J. Maleki Kahaki, M. Salimi Sartkahti, M. Joodaki, “Efficient GAN-based method for extractive summarization,” J. Electr. Comput. Eng. Innovations, 10(2): 287-298, 2022.

DOI: 10.22061/JECEI.2021.8051.475

URL: https://jecei.sru.ac.ir/article_1629.html