2022 27th International Computer Conference, Computer Society of Iran (CSICC), Tehran, Iran

HTS-DL: Hybrid Text Summarization System using Deep Learning

Majid Abolghasemi
Computer Engineering Faculty
K.N.Toosi University of Technology
Tehran, Iran
Abolghasemi@email.kntu.ac.ir

Chitra Dadkhah
Computer Engineering Faculty
K.N.Toosi University of Technology
Tehran, Iran
dadkhah@kntu.ac.ir

Nasim Tohidi
Computer Engineering Faculty
K.N.Toosi University of Technology
Tehran, Iran
n.tohidi@email.kntu.ac.ir

2022 27th International Computer Conference, Computer Society of Iran (CSICC) | 978-1-6654-8027-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/CSICC55295.2022.9780395

***Abstract*— Abstractive text summarization is the task of creating a summary from a document by merging facts from different sources and make a short description of them. In this procedure, the meaning and the content information should be kept. In this paper, a hybrid summarization system using deep recurrent neural network is proposed, which can create new sentences by information extracted from the text. The proposed model is the combination of extractive and abstractive summarization and has the encoder-decoder structure. The encoder extracts information from the source document and encodes this information in a compressed representation. The decoder takes the encoder's output as input and generates a summary, which has an acceptable semantic and syntactic structure. Experimental results show that the proposed model could improve both the performance of abstractive summarization and the time of training. This model does the single-document multi-sentence summarization and does not have any dependency on language. Therefore, it can be used for other languages without any modification in future.**

Keywords*— *hybrid system, abstractive summarization, text, deep learning, decoder-encoder, language processing

I. Introduction

Today, with the everyday expansion of the internet, a huge amount of data is available for anyone and individuals need to extract the main idea from this information in short time. Text summarization is the task of creating a short summary from a document, which captures the main ideas and information of the source document.

Text summarization can be classified into two categories: 1. Extractive Text Summarization (ETS) 2. Abstractive Text Summarization (ATS). ETS is the task of extracting important segment of the entered text and combining the extracted segments in order to generate the summary. In ETS, segments are not modified and are copied from the source document to the final summary. On the other hand, ATS is the task of creating a summary that contains the main idea of the source document and is very similar to the way that human summarization [1]. ATS can be seen as mapping the input document's words to the target sequence of words. In deep learning field, this method is called sequence to sequence (Seq2seq), which has been applied in many applications successfully. Using Seq2seq method, ATS becomes very similar to machine translation. However, it would be much harder, because in machine translation the input and output have mostly the same length but in ATS the output's length should be much shorter.

In ETS, the output sentences have good semantic and syntactic structure but sentences are isolated and only can be in the form of the source document, thus, the variation of them is very low. Besides, the summary may not be coherent and may contain all the information from source document. Unlike ETS, in ATS usually summaries contain the main information because the system has learnt the collocation between all words for constructing the model and then will generate the summary. As the model generates a sequence of words in the summary that may not be seen before, the syntactic structure could be weaker. Another problem in ATS is the rare words. In many Natural Language Processing (NLP) methods, the importance of a word is determined by co-occurrence of it with other words and the rare word sometimes is ignored. However, in ATS, rare words are kept.

Seq2seq methods are deep neural network models. Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called Artificial Neural Net (ANN) [2] that uses multiple layers to extract higher level features from input. As mentioned, ATS is the task of constructing a summary by merging facts from different source sentences and combining them into a shorter representation, while preserving the main purpose and meaning.

It is very difficult and time consuming for people to manually summarize large documents. Therefore, in this paper, a Hybrid Text Summarization model based on Deep Learning framework (HTS-DL) is introduced that can construct new sentences by exploring more fine-grained fragments than sentences, namely, semantic phrases.

HTS-DL combines ETS and ATS and uses LSTM-CNN to reduce syntactic and semantic errors in the text summarization process. It includes two main stages: 1. It extracts phrases from the source sentences using TextRank summarizer. 2. It generates the summaries using deep learning. TextRank is an ETS that extracts some sentences from the source document. This algorithm aims to extract the most informative sentences and its idea is similar to the PageRank algorithm [3].

Experimental results on CNN and DailyMail datasets show that HTS-DL system outperforms the state-of-the-art models in terms of both semantic and syntactic structure, and achieves competitive results on manual linguistic quality evaluation.

The remaining parts of the paper is organized as follows: In section 2, some recent related works are explained. In section 3, the whole process of the HTS-DL system is described. In section 4, the results of our experiments are explained. Finally, in section 5, the conclusion and future works are discussed.

II. Recent Related Works

Lots of improvements in the text summarization method has been experienced after the emergence of deep neural network (DNN) and recurrent neural network (RNN). One of the most popular architectures in this fields is Seq2seq. In

this regard, in the following some related works in this field are discussed.

Nallapati et al. in [1] added attention mechanism to the text summarization task as shown in Fig. 1. They used RNNs with attention mechanism and applied it to the machine translation task [4]. In their system, the attention mechanism was used in word level and a method for keeping rare word's information is defined.

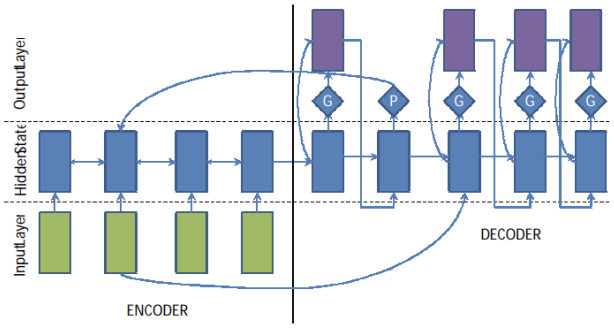

Fig. 1. Text summarization method used in [1]

After developing Seq2seq models, researches tried to build a better and faster encoder to decoder. One of the methods in this domain is Convolutional Neural Networks (CNN) in encoder side. Song et al in [5] by inspiring from B. Hu et al. [4], proposed a LSTM Encoder-Decoder which could successfully be applied in machine translation task. Fig .2 shows the structure of their proposed system. It used encoder-decoder with Seq2seq in order to generate a qualified summary in less time and could increase the quality of the decoder process using CNN method. In this regard, they first analyzed the generated summary by human and understood that we can extract specific structures from summaries in response of questions like "who?", "what's happening?" and etc. Their proposed system used variation auto-encoders as base generator to resolve inference problem in relation with complex generative modeling [6].

Additionally, historical dependencies were added to hidden parameters of variation auto-encoders and a standard discriminative deterministic decoder were used to the framework.

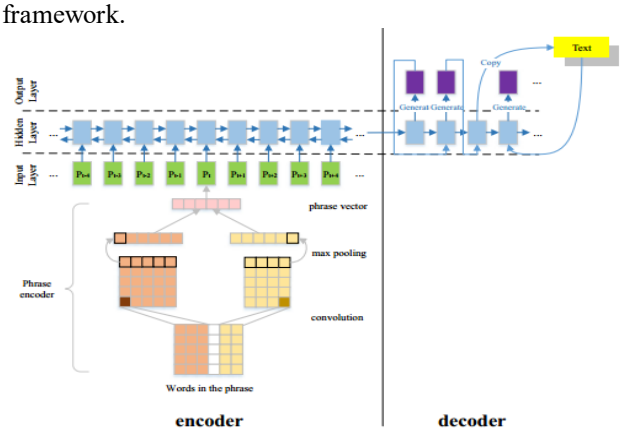

Fig. 2. Text summarization method in [5]

Zhang et al. in [7] used CNN for encoding the input sequence to low dimensional space. Before that, CNN only could encode a fix length of input. They resolved this problem using stacking convolutional layer, also by using this method every part of the input sequence could be encoded in parallel. AS summaries should be generated not only generated from keyword, but also from key sentences, in their work, which is shown in Fig. 3, two networks were applied in input, one in word level and another in sentence level. The first attention layer in the word level was optimized to generate qualified summaries and then according to sentence level attention, weights were optimized too.

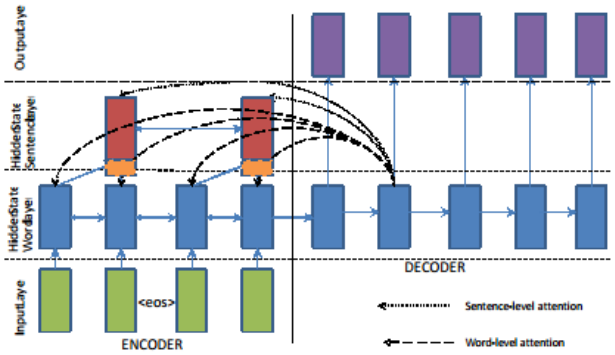

Fig. 3. Proposed system in [7]

Devlin et al. in [8] proposed a system which summarized the entered text using pre-trained BERT models in decoder side. The encoder section of their system contains two stages as shown in Fig. 4 and their system did the same process for rare words.

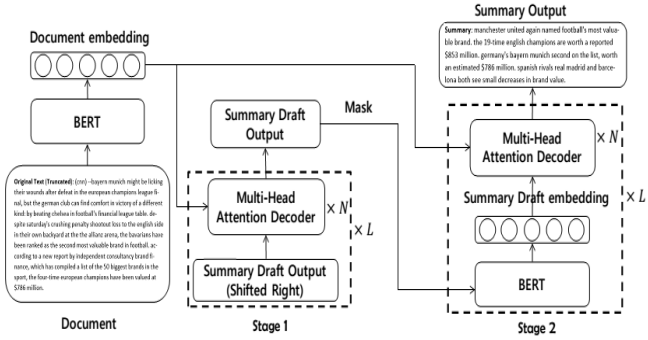

Fig. 4. Structure of proposed method in [8]

It should be noted that the mentioned methods are not the only ways for enhancing the text summarization quality. For example, several papers have worked on optimizing the performance of their systems using evolutionary algorithms, which nowadays are used widely in different fields like the method used in [9] and also for NLP applications such as approaches applied in [10-12].

III. The Proposed System

Fig. 5 shows the overall structure of our proposed system, HTS-DL, which contains 4 steps as follows:

1. Summarizing the input news with an extractive summarizer (Text Rank).
2. Preprocessing the source and the target sequence.
3. Applying the LSTM method that creates a compressed representation of the input text in the encoder side.
4. Passing the encoder's output to a decoder system and the decoder generates the output.

It is noteworthy that for resolving the rare word's problem, we use the bag-of-words representation that encodes all words in the source in a vector.

Here, we have considered three different approaches:

1. Simple RNN encoder-decoder
2. RNN with attention network
3. Combination of ATS and ETS

A. *Preprocessing*

Firstly, we remove stop-words [13] in the source document, because these words are highly frequent in every sequence and embedding them have not a big influence in the encoded embedding vector. We use bag-of-words representation for the source document. As the size of the vocabulary's list may be very large, we use only the most common words in the corpus. It makes our training process very fast. For the target documents, we keep all words and do not remove anything, because we want to generate words by the vocabularies used in the target sequence.

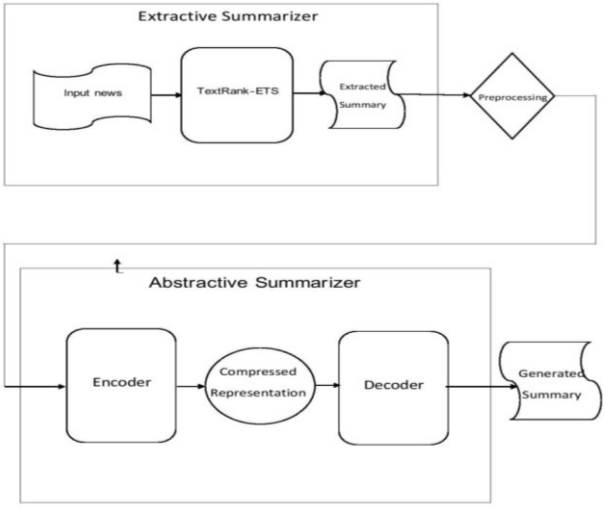

Fig. 5. Structure of HTS-DL.

At the beginning and the ending parts of the generated words, we have considered the symbols "__START__" and "__END__" in the target documents, which will be used in the input of decoder as the beginning and the ending conditions. Beside, we have replaced ',' and '.' with special words "COMMA" and "__POINT__" in both target and source documents, in order to consider the important punctuations in the vector representation and final summary.

B. *Simple RNN*

At first the simplest Seq2seq method which have encoder-decoder structure was implemented as shown in Fig. 6. Encoder reads the input sequence (x1, x2, …, xn) and summarizes the information in a vector representation called the internal state vectors (in LSTM these are called as the hidden state and cell state vectors). Decoder discards the outputs of the encoder and only keeps the internal states. Initial states of LSTM decoder initializes with the final states of the Encoder and it generates the output sequence (y1, y2, …, yn). In this case, the decoder method in the training phase is different from it in the inference time. Also, during the training, teacher forcing is used to train the decoder faster. In the sequence prediction method, normal networks consider the output of the last timestep Y(t-1) as input for the current timestep X(t). Evaluation of this approach shows slow convergence, model instability and poor skills.

In the teacher forcing method, as shown in Fig. 7, the actual or expected output from the training dataset at the current timestep Y(t) is considered as an input that was generated from the previous timestep.

In general, the encoder summarizes the input sequence into state vectors (sometimes also called as thought vectors), and then this vector feed to the decoder as the initial state and generates the output sequence. In other words, decoder is just in the next timestep X(t+1), and it did not use the output generated by the network in previous timestep. During inference time, the decoder's input at each timestep is the output as a language model conditioned on the initial states.

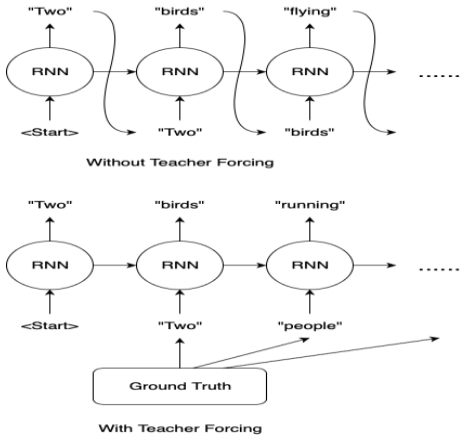

Fig. 6. Teacher forcing method[1]

C. *RNN with Attention*

One of the Simple RNN method's problem is the length of the input sequence. It may be very long; thus it is more difficult for the hidden vector to capture the context. In this case, effects of the beginning words, shown in Fig. 8, has been forgotten in the encoding process. In order to tackle this problem, we have used the attention mechanism that uses all hidden states during the encoding process of a sentence to create thought vectors. In this mechanism, these hidden states combine together with some weights, which can be obtained and optimized during training process. In attention mechanism, each generated output not only is a function of final hidden state, but is a function of all hidden states. It is not a simple concatenation or dot product of all hidden states, it is an attention operation. It generates a distinct vector for every decoder steps that combines all hidden states of encoder by giving different weights to every hidden state. The distinct context vector for an output step is a sum-product of attention weights and all input hidden states. The attention weights for every single output will be different and therefore the sum of the weighted hidden vectors is distinct for each output step.

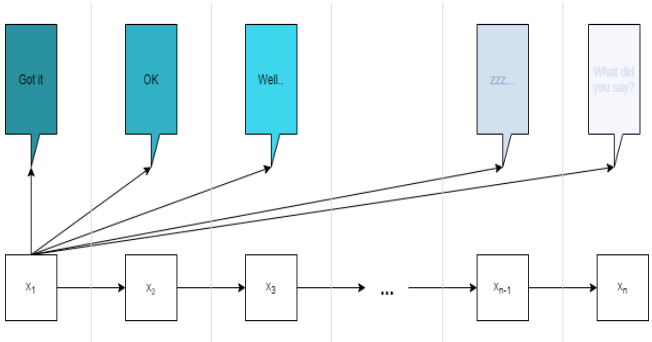

Fig. 8. Forget problem in LSTM

[1] https://towardsdatascience.com/what-is-teacher-forcing-3da6217fed1c

D. HTS-DL: the proposed model

In Fig. 9, another step of the proposed model is shown. Here, one of the most important problems of the previous method is solved by adding attention mechanism between encoder and decoder. Therefore, the proposed model has rich and content-aware encoding of the source document in encoder side and it keeps the effect of all previous words for generating the next word in decoder side. It is noteworthy that by adding it, the learning process becomes 2-3 times slower.

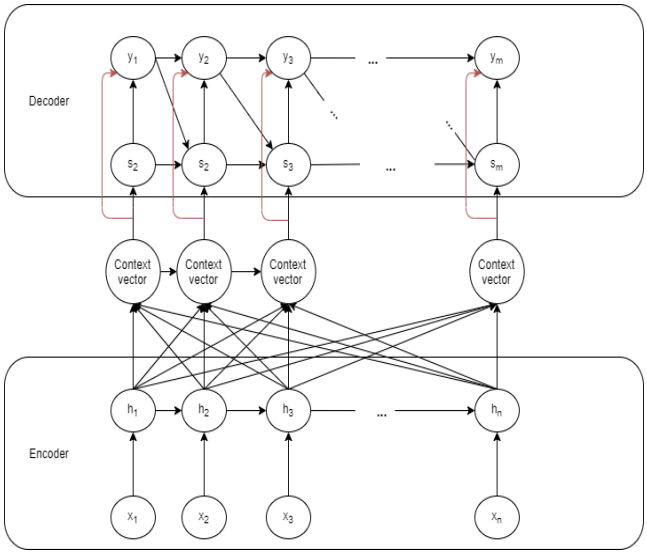

Fig. 9. Seq2seq model with attention mechanism

In previous sections, the structure of an ATS with Seq2seq method was described, then the attention mechanism was added to Seq2seq model to solve the problem of the first method. By adding attention, we can generate better summaries, however, it needs too long training time and if the length of input is very long, the attention mechanism needs a large dataset to train too. Thus, a new hybrid model called HTS-DL is proposed to resolve the mentioned problems for summarization in NLP.

In the proposed method, we add an ETS to Seq2seq method with attention mechanism to decrease the input length and to solve other mentioned problems. In Fig. 10, the general schema of the proposed system is shown, in which, firstly HTS-DL summarizes the input with an ETS and the output of this system is fed to ATS system that described in the previous sections. HTS-DL uses an unsupervised algorithm, TextRank, to extract the most important sentence of the input. TextRank is very similar to PageRank algorithm that search engines use for ranking web pages [3].

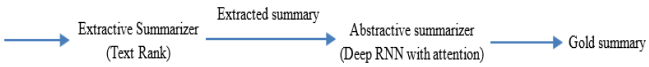

Fig. 10. The structure of HTS-DL

It is noticeable that summarization ratio in TextRank has a huge effect on the results' performance. If we use small summarization rate in TextRank, algorithm, it may remove useful information of the source document and our final summary will not include this useful information. Also, if we use a high summarization rate, we will not reach a great improvement in comparison with previous models.

IV. Experiment Results

In this paper the CNN news and summaries dataset[1] is used for evaluating the proposed model. In this dataset multiple one-sentence summaries exists for every news. First of all, for every summary, we create a row of data that contains duplicate input news associated with these summaries. This dataset has over 280000 training sample data. In this research, we create a row in our dataset for every summary that is related to every sample so dataset contains 800000 training data. We choose 100k rows, 2k rows, 1k rows of the dataset randomly as training, tuning and testing data respectively.

The size of vocabulary in raw news is very big, so for reducing the training time, we only keep the most important word in vocabulary based on TF-IDF (Term Frequency – Inverse Document Frequency) method, which is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus [13].

As mentioned before, we add special notations "__START__" and "__END__" as the start and end point of the summaries, respectively. At the test time, we use a slightly different procedure to generate summaries; For testing, the model uses last timestep output as an input for generating the next word, but in the training procedure, it uses the summary in the training set and get the true word for generating the current timestep word. Moreover, in the test time, the model generates words until it reaches the "__END__" symbol.

At each step, ATS gets the context vector and the previously generated words' sequence from decoder as the input of the system to generate the next word.

We have run HTS-DL on Google Colab GPU system, and it takes 2.5 hours per epoch and the training process stops after 15 epochs according to the validation loss. The training procedure of HTS-DL is done with different hyper parameters. One of them is size of the latent dimension which specifies the size of the encoded embedding of the encoder. It is noteworthy that the encoder embedding cannot encode all information with small value of these parameters. Training process takes much more time and the model do not capture useful and general information of the source documents for big value of these parameters. Training process starts with 150 of latent dimensions and in different runs, increase it to 800. After testing different values in this range and evaluating the results, we choose 350 as our final context vector size.

Another important hyper parameter which we have used is summarization ratio in ETS. As mentioned previously, if we use small summarization rate, this algorithm may remove useful information of the source document and our final summary will not have this useful information. Also, if we use high summarization rate, we will not reach a great improvement comparing the previous models. After testing different values in 0.3 to 0.8 range, we chose 0.5 as our final choice for extractive summarization ratio.

Two popular metrics that has been used for evaluating Seq2seq methods are BLEU [14] and ROUGE score [15]. BLEU score is a modified form of precision as in (1) and

[1] https://edition.cnn.com/

ROUGE score is similar to recall metric as in (2).

$$BLUE - score = \frac{number\ of\ words\ in\ candidate\ that\ are\ found\ in\ the\ reference}{total\ number\ of\ words\ in\ the\ candidate} \quad (1)$$

$$ROUGE - score = \frac{number\ of\ ovelapping\ words}{total word\ in\ reference\ summary} \quad (2)$$

We have used the full length ROUGE to evaluate HTS-DL and to compare the summary's result with gold-standard summaries (typically human-produced).

Table 1 shows the evaluation result comparing to the mentioned related works.

TABLE I. COMPARISION OF RELATED METHODS

Methods	Rouge-1	Rouge-2	Rouge-L
words-lvt2k-temp-att [1]	35.46	13.03	32.65
ATSDL [5]	34.09	17.80	-
CNN-2sent-hieco-RBM [7]	42.04	19.77	39.42
Simple RNN	35.21	8.24	29.02
Attention RNN	39.42	12.01	33.12
HTS-DL	43.71	16.83	38.58

The experimental result shows that our first system, Simple RNN, has worse results than the best previous method. Because of the high length of the source documents, attention mechanism in encoder layer of Attention RNN method encodes input better and works better than previous methods. Attention RNN uses two attention layers in encoder and decoder and it can capture states of LSTM better and generate better summaries based on evaluation metrics than previous methods, However, it has a little weaker performance than recent works in this field. As it can be seen, Table 1 shows that HTS-DL performs better than our previous methods, beside, it outperforms other recent methods in this field.

V. CONCLUSION AND FUTURE WORKS

In this paper, three text summarization models were proposed. The first one used a Simple RNN to create a Seq2seq method. Additionally, we used some techniques such as teacher forcing and large vocabulary trick too. This system could not outperform recent models. Therefore, in the second model, we added the attention mechanism to it. Experimental results show that this model can perform better than some related models, not all of them. Moreover, this model takes much more time to train than previous systems. Hence, the third system named HTS-DL was proposed to cover problems of the Attention RNN method by combining extractive and abstractive summarizers. The experimental result shows that HTS-DL can outperform other models in this field.

In recent years, pre-trained methods such as BERT and GPT-2 and GPT-3 are widely used in NLP [8, 16]. In this regard, we can use these methods to build a better encoder and use their pre-trained information power to have better extraction of information from the input sequence in future works. In addition we can apply this method in various natural languages.

REFERENCES

[1] Nallapati, Ramesh; Zhou, Bowen; Dos Santos, Cicero Nogueira; Gulcehre, Caglar; Xiang, Bing, "Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany, 2016.

[2] LeCun, Yann; Bengio, Yoshua; Hinton, Geoffrey, Deep learning, vol. 521, Nature, 2015, p. 436–444.

[3] Barrios, Federico; López, Federico; Argerich, Luis; Wachenchauzer, Rosa, "Variations of the Similarity Function of TextRank for Automated Summarization," *arXiv:1602.03606,* pp. 65-72, 2015.

[4] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate", ArXiv14090473 Cs Stat, May 2016, Accessed: Mar. 03, 2020. [Online]. Available: http://arxiv.org/abs/1409.0473.

[5] S. Song, H. Huang, and T. Ruan, "Abstractive text summarization using LSTM-CNN based deep learning," Multimed. Tools Appl., vol. 78, no. 1, pp. 857–875, Jan. 2019, doi: 10.1007/s11042-018-5749-

[6] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," ArXiv13126114 Cs Stat, May 2014, Accessed: Mar. 03, 2020. [Online]. Available: http://arxiv.org/abs/1312.6114.

[7] Y. Zhang, D. Li, Y. Wang, Y. Fang, and W. Xiao, "Abstract Text Summarization with a Convolutional Seq2seq Model," Appl. Sci., vol. 9, no. 8, Art. no. 8, Jan. 2019, doi: 10.3390/app9081665.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," ArXiv181004805 Cs, May 2019, Accessed: Feb. 28, 2020. [Online]. Available: http://arxiv.org/abs/1810.04805.

[9] Tohidi, Nasim; Dadkhah, Chitra. "Improving the performance of video Collaborative Filtering Recommender Systems using Optimization Algorithm", International Journal of Nonlinear Analysis and Applications (IJNAA), 2020, Vol:11, issue:1, pp: 483-495. DOI: 10.22075/IJNAA.2020.19127.2058

[10] Tohidi, Nasim; Dadkhah, Chitra; Rustamov, Rustam B., "Optimizing the Performance of Persian Multi-objective question answering system," in The 16th International Conference on Technical and Physical Problems of Engineering, Istanbul, Turkey, 2020, PP:110-116.

[11] Meena, Yogesh Kumar; Gopalani, Dinesh. "Evolutionary Algorithms for Extractive Automatic Text Summarization", Procedia Computer Science, 2015, Vol:48, pp: 244-249.
link: https://doi.org/10.1016/j.procs.2015.04.177

[12] Tohidi, Nasim; Hasheminejad, Seyed Mohammad Hossein, "MOQAS: Multi-objective question answering system," Journal of Intelligent & Fuzzy Systems, vol. 36, no. 4, pp. 3495-3512, 2019.

[13] Jurafsky, Daniel; Martin, James H. "Speech and Language Processing". Upper Saddle River, NJUnited States: Prentice Hall, 2019.

[14] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation," in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, Jul. 2002, pp. 311–318, doi: 10.3115/1073083.1073135.

[15] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in Text Summarization Branches Out, Barcelona, Spain, Jul. 2004, pp. 74–81, Accessed: Feb. 28, 2020.
link: https://www.aclweb.org/anthology/W04-1013

[16] Radford, Alec; Narasimhan, Karthik; Salimans, Tim; Sutskever, Ilya. "Improving Language Understanding by Generative Pre-Training," openai,cs.ubc.ca, 2018, p. 12.