symmetry

Article

Extractive Summarization Based on Dynamic Memory Network

Ping Li * and Jiong Yu

College of Information Science and Engineering, Xinjiang University, Urumqi 830046, China; yujiong@xju.edu.cn

Abstract: We present an extractive summarization model based on the Bert and dynamic memory network. The model based on Bert uses the transformer to extract text features and uses the pre-trained model to construct the sentence embeddings. The model based on Bert labels the sentences automatically without using any hand-crafted features and the datasets are symmetry labeled. We also present a dynamic memory network method for extractive summarization. Experiments are conducted on several summarization benchmark datasets. Our model shows comparable performance compared with other extractive summarization methods.

Keywords: text summarization; recurrent neural network; embedding; dynamic memory network

check for updates

Citation: Li, P.; Yu, J. Extractive Summarization Based on Dynamic Memory Network. *Symmetry* **2021**, *13*, 600. https://doi.org/10.3390/sym13040600

Academic Editors: Kóczy T. László and István A. Harmati

Received: 6 March 2021
Accepted: 1 April 2021
Published: 3 April 2021

1. Introduction

Summarization is an import problem of natural language understanding and information retrieval. The aim of the summarization is to condense the input text and remain the core meaning of the input text. The methods of the summarization are classified into two categories: extractive summarization method and abstractive summarization method. These two methods are symmetry important. The extractive summarization method selects the salient content from the documents while the abstractive summarization method paraphrases the content of the document. The earlier research mainly concentrated on extractive summarization method and the recent research focuses on neural extractive summarization and neural abstractive summarization. In this paper, we only pay attention to the extractive summarization method.

The early work of the extractive summarization method which was done by Edmundson [1] scores the sentences by considering the title words, clue words, and the sentence positions. Lin [2] uses some regulations to find the topic sentences and trains a model to predict the topic sentences based on the positions.

As the development of deep learning, researchers mainly concentrate on using the neural network method to resolve the extractive summarization problem. In particular, the development of the neural network language model [3] and the text representation methods [4] make the natural language processing take off. Cao [5] applies the neural network to extractive query-focused summarization which is a task of information retrieval. In their model, they employ the CNN(Convolutional Neural Network) to project the sentences of the document and the query to latent space. To get the document representation, they use the weighted-sum pooling over the sentence embedding. Lastly, they rank and select the sentences of document after comparing the similarity between the sentence embedding and document embedding.

Because of the success of the RNN (Recurrent Neural Network) in machine translation [6], Rush [7] first employs the RNN based on an attention mechanism for abstractive summarization.

Nallapati [8] uses the sequence model based on RNN to extract summarization of a single document, which is the problem we are focusing on. In their model, they see the extractive summarization task as a binary classification task and use the RNN model as a sentence classifier. Recently, Zhou [9] integrates the MMR (Maximal Marginal Relevance) selection strategy proposed by Carbonell and Goldstein [10] into the scoring model. In their model, they employ the BiGRU (Bidirectional Gated Recurrent Unit) [4] as their encoder to get the sentence representation and document representation and they construct their labeled training data by maximizing the ROUGE-2 F1 score [2] . These neural extractive summarization methods mentioned above are all using the RNN as their encoder and the labeled data construction method is computationally expensive. Narayan [11] employs the CNNs [12] as the sentence encoder and employs the RNN as the document encoder. Because of the strong ability to extract text features, we will use the transformer [13] as the encoder just like Bert [14] does.

Our main contributions are as follows:

1. We propose an extractive summarization model that achieves the comparable result against other baselines.
2. We propose a simple and effective sentence label method used in the extractive summarization problem.
3. We incorporate the positional encoding to a dynamic memory network.
4. We propose to use a dynamic memory network method for extractive summarization.

2. Materials and Methods

2.1. Problem Formulation

A document is represented as D and S represents the sentence of the document. D is defined as Equation (1):

$$D = \{S_1, S_2, \ldots, S_n\} \tag{1}$$

The n in Equation (1) is represented as the max number of the document sentences. The target of the extractive summarization task is to extract r sentences of the document ($r < n$) and the r sentences maintain the core information of the document. We make the extractive summarization task as a binary classification task just like Cheng [15] and Nallapati [8] do. Given a sentence s_i $(0 \leq i \leq n)$, which consists of tokens represented as w_j $(0 \leq j \leq m)$, we predict the probability of $P(Y|S)$ $(Y \in \{0,1\})$. The last hidden output of the Bert is denoted as h_k $(0 \leq k \leq m)$ and the output of the sentence encoder is denoted as $s_i^{'}$, which is the embedding representation of sentence i. The target sentence in training set is denoted as t_l $(0 \leq l \leq p)$ and p is the length of the target summaries. The embedding of the t_l is denoted as $t_l^{'}$. The b_l is denoted as the sentence labeled with 1.

2.2. Extractive Summarization Based on Bert

2.2.1. Sentence Encoder

Differentiated with Narayan [9,11], we don't use the hierarchical encoder and only use the sentence encoder. The architecture of the sentence encoder is shown in Figure 1. Because of the strong ability to extract the text features, we use the Bert to get the word embeddings of the sentences. We refer you to the detailed description of the pre-training Bert model in the paper written by Jacob [14]. Using the Bi-Transformer and the masked language model makes the Bert better than other pre-training models.

The input of the sentence encoder is the one-hot representations of the words in a sentence and the model parameters of pre-training are loaded in the sentence encoder model. Given sentence I, through the pre-training Bert model, we will get a list of embeddings. The embedding of sentence I is calculated as Figure 2. We just use this sentence encoder in the process of building a labeled training set and the process of the prediction and extractor will use the fine-tuned Bert directly:

$$S_i^{'} = (h_1 + h_2 + \cdots + h_m)/m \tag{2}$$

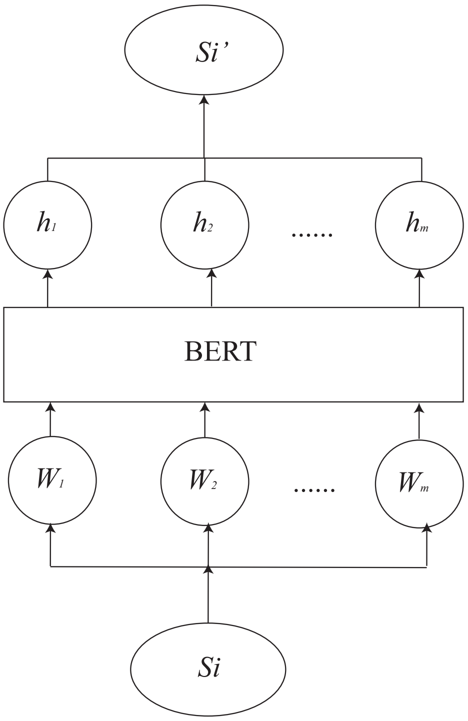

Figure 1. Overview of the sentence encoder based on Bert.

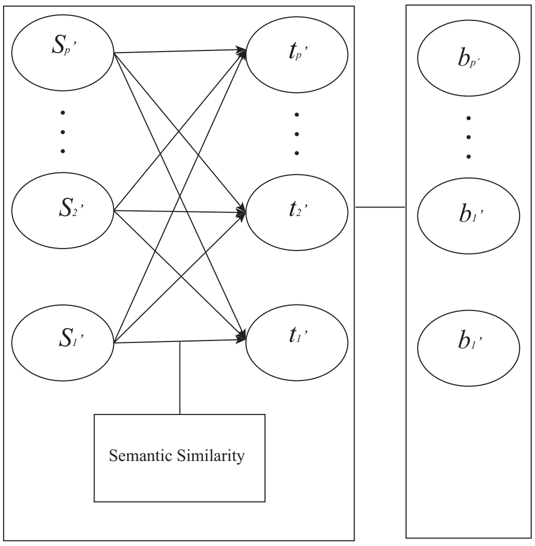

Figure 2. Overview of the labeling process.

2.2.2. Label Training Set

We conduct our experiments on the CNN/Dailymail dataset [16] and several other datasets. We will train a binary classifier on these datasets. However, there is no labeled training dataset. In order to construct this labeled dataset, we need to label the sentence of the document with 0 and 1. The label 0 shows that this sentence should not be included in the summaries, and label 1 shows that this sentence should include the summaries. Differentiated with Zhou [9,11], we don't use the rouge method to build the supervised datasets, and we will use the semantic similarity method to label the training set. The overview of the labeling process is shown in Figure 2.

Given the $s_i{}'$ and the $t_l{}'$, the equations of calculating the semantic similarity are shown as Equations (3) and (4). For every sentence in the target summaries, we find the maximum similarity sentence from the source document. These selected sentences are labeled with 1 and the others are labeled with 0:

$$similarity = \cos\left(s_i{}', t_l{}'\right) \tag{3}$$

$$\cos\left(s_i{}', t_l{}'\right) = \frac{s_i{}' \cdot t_l{}'}{\|s_i{}'\| \times \|t_l{}'\|} \tag{4}$$

2.2.3. Sentence Extractor

We directly use the fine-tuned Bert model to train our binary classifier which will predict the probability of the sentence to be extracted. The classify model that loaded the pre-training parameters reads a single sentence, and the classified probability is denoted as Equation (5). In Equation (5), the C is the final hidden state of the input representation in the Bert model, and the W are the new parameters to be learned. The loss function is a cross entropy loss function that is denoted as Equation (5). In the test phase, we predict the probability of the label when we input the test dataset. We assume that the probability of label 1, which is greater than the probability of label 0, should be the extractive summary. The max sequence length of the input sequence is set to 128:

$$P(Y|S) = Softmax(CW^{\top}) \tag{5}$$

$$Loss = -logP(Y|S) \tag{6}$$

2.3. Extractive Summarization Based on the Dynamic Memory Network

Kumar et al. first introduce the DMN (dynamic memory network) for the QA (question answering) problem [17]. Caiming et al. [18] propose several improvements over the base dynamic memory network. The DMN consists of input module, question module, episodic memory module, and answer module. The episodic memory module will produce the focusing parts of the inputs through the input module and question module. The answer module will generate the answers based on the outputs of the memory module. Because the similarity of the QA problem and the summarization problem, we employ the DMN as our base model to classify the sentences.

Dynamic Memory Network

In order to extract the summarizations from the document, we propose a dynamic memory network that is composed of input module, memory module, summarization module, and linear module. What we are focusing on is to memorize the salient content of the sentence and then classify the facts. The modules of our model are showed in Figure 3.

Input Module: the inputs of our model are the all sentences of our document sets. We need to encode the sentences by using an encoder. The encoder of [17] is a GRU (gated recurrent network) [4,19,20]. The input of the GRU is the word embeddings and then the encoder calculates the hidden states of the words, which are concatenated as the sentence representations. The hidden state of the token i can be defined as $h_i = GRU(x_i, h_{i-1})$, where x_i is the embedding of token i and h_{i-1} is the hidden state of previous token. The GRU is defined as:

$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1} + b^{(z)}\right) \tag{7}$$

$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1} + b^{(r)}\right) \tag{8}$$

$$\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1} + b^{(h)}\right) \tag{9}$$

$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t \tag{10}$$

where z_t is the update gate and r_t is the reset gate, the W and U are the weights, and the b is the bias. The σ and the *tanh* are the activation functions.

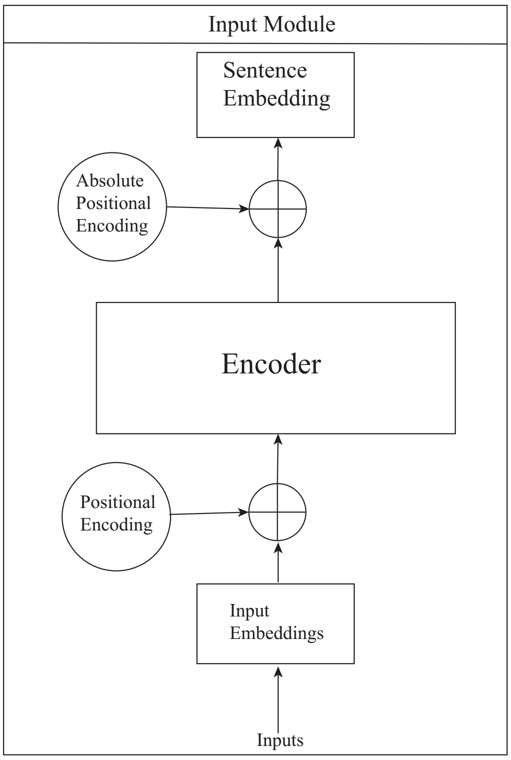

Figure 3. Input module.

The input of our model is one sentence, and we encode one sentence by using a bi-GRU. Thus, we can use not only the information of previous words but also use the information of the afterwards words. When we encode the sentences, we also consider the positions of the words and the positions of the sentences. Differentiated with [17], we combine the word positional encoder and sentence positional encoder with the sentence encoder. The word positional encoder [13] is defined as:

$$WPE_{(wordPos,2i)} = \sin\left(wordPos/10000^{2i/d}\right) \tag{11}$$

$$WPE_{(wordPos,2i+1)} = \cos\left(wordPos/10000^{2i/d}\right) \tag{12}$$

where the $wordPos$ is the position of the word, i is the dimension, and d is the dimension of the word embedding. From previous research, we can find that the position of the sentence is very important when extracting the sentences from the document. We define the sentence positional encoder as:

$$SPE_{(i)} = p_i \tag{13}$$

where the p_i is the absolute position of the sentence. In our experiments, we also consider using the Bi-RNN [21] and the transformer [13] as the encoder. In the BiRNN model, the forward RNN encodes the input sequence as

$$\overrightarrow{f} = \left(\overrightarrow{h}_1, \cdots, \overrightarrow{h}_{T_x}\right) \tag{14}$$

and the backward RNN encodes the input sequence as

$$\overleftarrow{f} = \left(\overleftarrow{h}_1, \cdots, \overleftarrow{h}_{T_x}\right). \tag{15}$$

The final representation of word j is denoted as

$$\vec{h}_j = \left[\overrightarrow{h}_j^{\top} ; \overleftarrow{h}_j^{\top} \right]^{\top}. \tag{16}$$

Through the $\vec{h}_j$, we will get the sentence embedding.

Summarization Module: in the problem of question and answer, the dynamic memory model should encode the question by using the question module. However, in our model, there is no question. In order to memory the salient content in the source sentence, we need to encode the salient content, which is the abstracts of the document in the training phase, while, in the testing phase, it is the first three sentences of the document. We call this encoder module as the summarization module. The summarization module is very similar to the input module. The difference between the summarization module and input module is that we use the RNN as encoder when we encode the content .

Memory Module: the memory module mainly gets the similarity representation of the sentence to be classified and the salient content of the document. The input of the memory module is the sentence representation that is the output of the input module and the summarization representation, which is the output of the summarization module. In order to get the similarity representation, we employ the same mechanism as [17] to compute the representation of similarity. In our module, we only compute three episodes. In order to capture the similarities of the input module and summarization module, we define the similarity content as

$$z(I,S) = Concat(I * S, S * memory, |S - memory|, |I - S|,) \tag{17}$$

where I is the output of the input module, and S is the output of the summarization module. The memory is initialized with I. The gate function in our model is defined as

$$g = \sigma\Big(W^{(2)} \tanh\Big(W^{(1)} z(I,S) + b^{(1)}\Big) + b^{(2)}\Big). \tag{18}$$

With the gate function g and the similarity content representation z, we compute the episode as:

$$h_t = g_t RNN\Big(\vec{I}, h_{t-1}\Big) + (1 - g_t) h_{t-1}. \tag{19}$$

The memory is updated as

$$memory = GRU(h_t). \tag{20}$$

Linear Module: in the linear module, we need to map the similarity representation to a two-dimensional output by using a linear layer and then get a probability by using a softmax layer. Through this probability, we can train our model with the labeled training set:

$$out_i = L\Big(RNN(h_t, \vec{S})\Big) \tag{21}$$

$$P_i = softmax(out_i) \tag{22}$$

3. Experimental Setup

We will present our experimental setup for assessing the performance of our model which we call ESBOB and SDMN in this section. We will discuss the datasets used for training and evaluation. The implementation details and the evaluation method are described for comparison.

3.1. Datasets

We train and evaluate our model on the non-anonymized CNN and DailyMail datasets [16], which are developed for the question-answering system. We split the dataset for 287, 226 training pairs, 13,368 validation pairs, and 11,490 testing pairs followed by [22–24]. Because the CNN/Dailymail datasets don't include the reference extractive summarization, we will use the abstractive summarizations as the reference summarizations.

The second dataset we use is the WikiHow developed by Koupaee and Wang [25] . This dataset is a diverse dataset that is extracted from an online knowledge base. The dataset has 168,128 training pairs and 6000 testing pairs

The third dataset we use is the XSum developed by Narayan [26] . This dataset is a one-sentence summary dataset. The summaries in this dataset are written professionally.

3.2. Baselines

In order to compare, we choose some approaches as our baselines. These baselines are listed below:

(1) Leading three sentences (Lead-3). This method constructs the summary by extracting the first three sentences of the document. We have given our lead-3 result and the Lead3 result of [23] .
(2) Cheng and Lapata [15] . Cheng and Lapata propose an extractive model which consist of the hierarchical document encoder and an attention-based extractor.
(3) SummaRuNNer [8] . This model is based on a recurrent neural network.
(4) REFRESH [11] . Narayan makes the extractive summarization task as a sentence ranking task and optimizes the rouge metrics through the reinforcement learning object.
(5) NEUSUM [27] . This model makes the sentence scoring and sentence selection to an end-to-end neural network framework.
(6) BANDITSUM [28] . In the field of text summarization, many methods have been proposed with reinforcement learning. In this model, a policy gradient reinforcement learning algorithm is used to train to select the summarization sentences.

3.3. Implementation Details

The pre-trained Bert we use in the process of labeling, training, and testing is the uncased Bert-base model. In the pre-trained model, there are 12 transformer blocks. The model employs 768 hidden sizes and 12 self-attention heads. The optimizer used in our model is Adam optimizer [29] with initial learning rate 0.001. We use a batch size of 128 and an epoch of 3 at training time. We train our sentence classifier on 4 Tesla K80 GPU. At test time, we extract the first three sentences as the baseline because of the LEAD3 is a commonly used baseline. We also extract the sentences of the training set as the reference summarizations based on the semantic similarity and the abstractive summarizations.

When we train our dynamic memory network, we also use the Adam optimizer [29] with an initial learning rate of 0.001. The batch size we use is 32, and we train our model in two epochs. The training set we use is the data labeled by Bert. The summarizations we use in the summarization module are the first three sentences of the document. We train an extractive system based on the dynamic memory network on 1 Tesla K80 GPU.

3.4. Evaluation

The F1 ROUGE value [30] is used to evaluate our summarization model. We will report the f scores of ROUGE-1, ROUGE-2, and ROUGE-L. We will compare our model against the lead-3 baseline which just selects the first three sentences in the document as the summary. On the CNN/Daily mail dataset, the result of the approach put forward by Cheng and Lapata [15] is reported. We compare our model against the approach called BanditSum [28] and the approach called Refresh [11], which trains the extractive summarization with reinforcement learning. We also compare our model against the approach called SummaRuNNer [8], which treats the extractive summarization as a binary classify task. The result of approach called Neusum [9] is also reported.

4. Results

4.1. Results on CNN/Daily Mail

Table 1 shows the experiment results by using the dynamic memory network called SDMN. We can find that the SDMN is effective in processing the summarization problem. Incorporation the pre-training method into the SDMN model, the experiment result performs very well. There are three encoders that we have used in our experiments. The Bi-LSTM (Bi directional Long Short Term Memory) encoder with the dynamic memory network shows the best result. Thus, in some extent, the Bi-LSTM has not been replaced by transformers.

Table 1. ROUGE evaluation (%) on the CNN/DailyMail test set using dynamic memory network (SDMN) with different encoders.

Method	ROUGE-1	ROUGE-2	ROUGE-L
SDMN + BiGRU + pe	36.69	15.53	33.14
SDMN + BiLSTM + pe	40.24	17.53	36.49
SDMN + trans + pe	40.2	17.5	36.48

Table 2 shows the experiment results using automatic metrics. We report the lead3 result which is supplied by [23], and the second method result is supplied by [11]. From the table, we can find that the neusum [9] achieves the state-of-the-art result. The method trained by reinforcement learning depends on the rouge score when labeling the dataset and training the model, which leads to the hard improvement on the experiment result. The semantic experiment is conducted by us, which extracts the summarizations by semantic similarity. However, this semantic similarity approach can not be applied into the inference phase. From the result, we can find out that the training set built on semantic similarity is effective to a certain extent. In order to improve the experiment result, we need to find some more effective method to label the training set or use some methods to extract the summarizations without labeling the training set such as reinforcement learning method. Our method does not beat the method neusum and method banditsum. However, our method named SDMNTransPe achieves a comparable result as other methods. From this, we can find that our method is effective.

Table 2. ROUGE evaluation (%) on the CNN/DailyMail test set. Models marked with * are trained and evaluated on the anonymized dataset, and so are not comparable to our results.

Method	ROUGE-1	ROUGE-2	ROUGE-L
LEAD3 (ours)	39.89	17.24	36.12
LEAD3 (See et al.)	40.3	17.7	36.6
SemanticSim	50.54	27.63	46.77
Cheng and Lapata	35.5	14.7	32.2
SummaRuNNer *	39.6	16.2	35.3
REFRESH	40.0	18.2	36.6
NEUSUM	41.59	19.01	37.98
BANDITSUM	41.5	18.7	37.6
SDMNTransPe	**40.2**	**17.5**	**36.48**

4.2. Results on WikiHow and XSum

WikiHow is a summarization dataset that has short summaries. The XSum dataset has symmetry long summaries. We will evaluate the short summarization dataset to find the result by using the extractive summarization method. In Table 3, the first section contains the lead-1 result and the second section contains the groundtruth result. From Table 3, we can reconfirm that the first line in the document contains the important information.

The last section in Table 3 shows the result of our extractive summarization method. The last result is improved on the lead-1 result and is not comparable to oracle results. Thus, we can use the extractive summarization to improve the result on short summarization datasets.

The summary in the WikiHow contains one sentence. The average number of the sentences in the XSum summary is 8.4 and the average number of the sentences in the CNN/DailyMail dataset is 4.8. The XSum dataset is the long summaries dataset. Our model is effective in processing short summaries dataset and one sentence summary dataset. In order to prove that our model is effective on the long summaries dataset, we conduct the experiments on the XSum dataset. When we train our model, we select the one sentence, two sentences, and all sentences from the summaries as the candidate summary. From Table 4, we can find that the more sentences we select, the better performance the extractive summarization model gets.

Table 3. ROUGE evaluation (%) on the WikiHow test set. Lead indicates selecting the first sentence as the summary.

Method	ROUGE-1	ROUGE-2	ROUGE-L
Lead	24.97	5.83	23.24
oracle	35.59	12.98	32.68
SDMN	30.23	7.58	27.34

Table 4. ROUGE evaluation (%) on the XSum test set. Num indicates the number of sentences we choose to form a candidate summary. All indicates the sentences the model selected from the candidate summary.

Method	ROUGE-1	ROUGE-2	ROUGE-L
SDMN (1)	21.15	4.11	15.23
SDMN (2)	22.82	4.21	16.54
SDMN (all)	23.51	4.35	17.43

5. Analysis

Our analysis is driven by the following two questions:

(1) Where are the salient sentences in the document?
(2) Why is our method effective compared with other methods?

In order to find the discipline about the positions of the salient sentences in the document. We choose the CNN/DM, WikiHow, and XSum datasets. There are relations between the extractive summarization method and the abstractive summarization method. The abstractive summarization model is to find the salient sentences or salient words in the document and then paraphrase the selected contents. What the extractive summarization model does is just what the abstractive method needs. The discipline of the positions of the salient sentences in the document is very important to the abstractive summarization problem and the extractive summarization problem.

Figures 4–6 show the salient sentences' positions inspected by the Bert in the three datasets. We can find that the XSum dataset contains one sentence summary, the CNN/Dailymail dataset contains the medium summary sentences, and the WikiHow dataset contains the large summary sentences. The salient sentences are not always in the first line of the document. The salient sentences are always located in the whole article and the two salient sentences have nearly equal distance. Table 5 shows the title, first sentence and the labeled sentence in the XSum dataset. We can find that the sentence labeled by the Bert Model is more adherent to the title than the first sentence of the document. When the authors are editing the article, they always highlight the gist in some paragraphs. Thus, when we write the summarization automatically, we should not just concentrate on the first lines of the document but on the whole paper.

The document we are processing is always the long text. In order to select the salient sentences from the document, we need to classify the sentences of the document. Relatively speaking, the sentences of the document are short text. The transformer encoder is good at processing the long text with the attention mechanism while the LSTM encoder is good at processing the short text. That makes our model get equal performance when we choose the transformer and the LSTM as the encoder. The model based on the reinforcement learning depends on the rouge score when they label the dataset and train their model while we label our dataset using the semantic matching method which makes our method comparable with other methods. Our method makes full use of the reference summaries when training our model by using the dynamic memory network. The dynamic memory network mechanism can help our model find the most important sentence in the document. Our method is slightly weaker than the methods based on the reinforcement learning.

In the encoder phase, we employ the GRU, LSTM, and transformer as our encoder. The difference between our model and someone else's model is that the sentences labeled for the classification model are identified by pre-training model Bert. While other models only use the sentences' information, we incorporate the sentence position information into the encoder model. Our model makes full use of the referenced summarizations. First, we use the referenced summarizations and the Bert model to spot the salient sentences in the document. Second, we use the dynamic memory network to compute the similarity representations of the document sentences and the reference summarizations for the classification model. The similarity feature is computed dynamically, which is good for the classification model. The characteristics of our model described above make our model comparable with other extractive models.

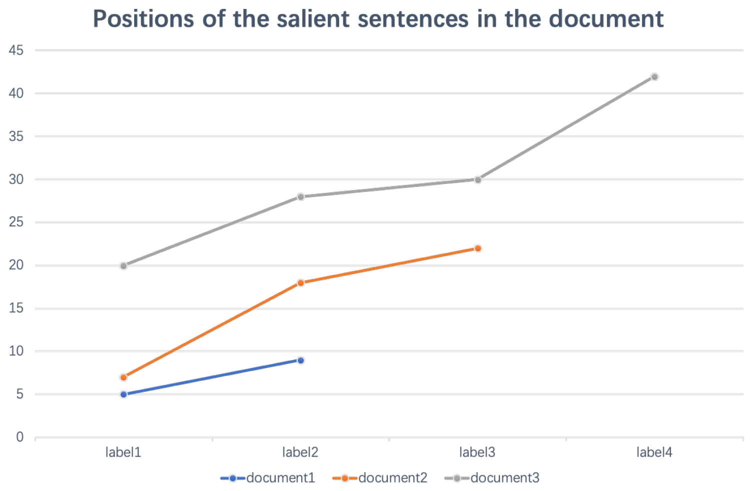

Figure 4. Positions of the salient sentences in the document of the cnndm dataset.

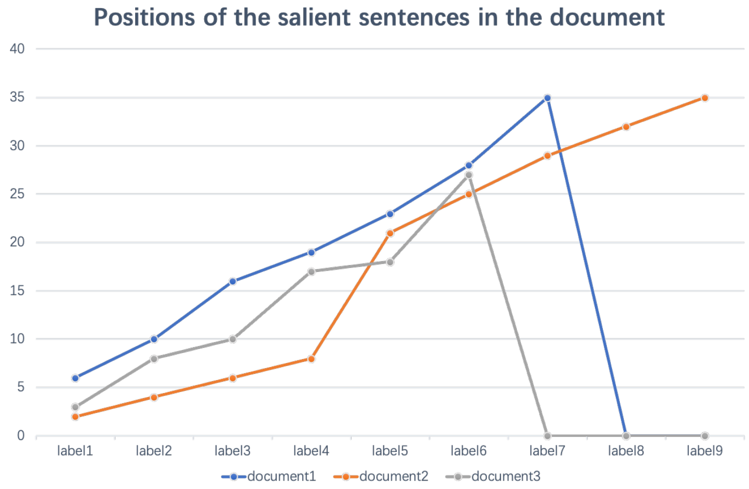

Figure 5. Positions of the salient sentences in the document of the wikihow dataset.

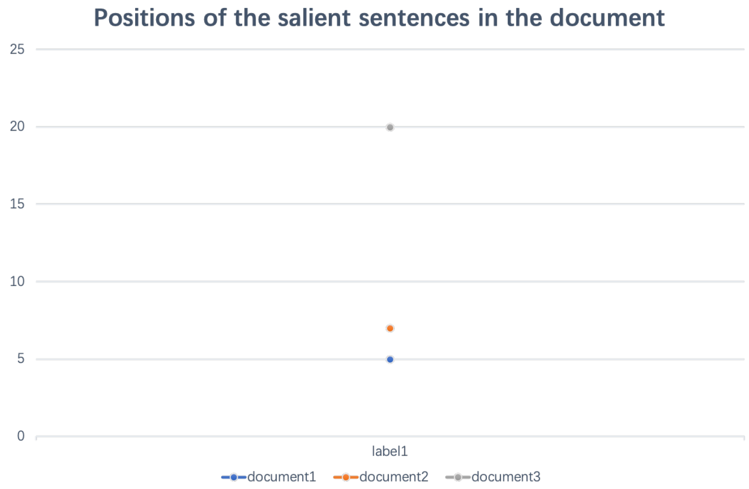

Figure 6. Positions of the salient sentences in the document of the xsum dataset.

Table 5. The sentences in the xsum document.

Doc	Title | First, Sentence | Labeled Sentence
doc1	Spend £3.3 m fund on Wales-based stars, says Gareth Davies
doc2	Alliance Party east Belfast alert was a hoax, PSNI say
doc3	UK energy policy ‘deters investors’
doc1	New Welsh Rugby Union chairman Gareth Davies believes a joint £3.3 m WRU-regions fund should be used to retain home-based talent such as Liam Williams, not . . .
doc2	A suspicious package left outside an Alliance Party office in east Belfast has been declared a hoax
doc3	The UK’s international reputation for a strong and well-balanced energy policy has taken another knock
doc1	3 m should be spent on ensuring current Wales-based stars remain there
doc2	Condemning the latest hoax, Alliance MLA Chris Lyttle said: “It is a serious incident for the local area, it causes serious disruption, it puts people’s lives at risk, . . . ”
doc3	A spokesman for her department, commenting on the WEC report, said: “We’ve made record investments in renewables and are committed to lower-carbon . . . ”

6. Related Work

Recently, with the springing up of the deep learning method, all kinds of neural network methods are applied in extractive summarization [5,9,11,15,28].

Cheng [15] sees the task of sentence extractive summarization and word extractive summarization as a binary classifier task. In the sentence encoder, they employ a hierarchical encoder which uses a single-layer CNN for obtaining the sentence-level representations and use a recurrent neural network for obtaining document representations. The sentence extractor in [15] is an MLP(Multi-layer Perceptron) with an attention mechanism. When building the training dataset, Cheng uses the rule-based method including taking into account the position of the sentence in the document. Cao [5] presents a query-focused extractive summarization system which is used in information retrieval. This extractive summarization system consists of the CNN Layer, Pooling Layer, and Ranking Layer. The sentence ranking process only compares the semantic similarity between the sentence embedding and the document embedding. Nallapati [8] presents an RNN based sequence model for extractive summarization of documents. In their work, they also treat extractive summarization as a sequence classification problem. They use a two-layer bi-GRU for obtaining the word-level representation and the sentence-level representation. The sentence extractor uses a logistic layer. They build the labeled training dataset by maximizing the rouge score with respect to gold summaries. Zhou [9] obtains the sentence representations by using a hierarchical encoder which consists of BiGRU. They couple the sentence scoring step and sentence selection step. The sentence extractor consists of a BiGRU and a MLP(Multi-Layer Perceptron). Narayan [11] conceptualizes extractive summarization as a sentence ranking task and proposes a novel training algorithm with a reinforcement learning objective [31] which optimizes the rouge metric. They use a convolutional sentence encoder and a LSTM document encoder. The sentence extractor consists of LSTM cells and a softmax layer. When they rank the sentences, they train their model in a reinforcement learning framework. This reinforcement learning avoids labeling the training set and makes the model better at discriminating among sentences. Dong [28] treats the extractive summarization as a contextual bandit problem. They also use a policy gradient reinforcement learning algorithm to select sentences and maximize rouge score. The difference between [11] and [28] is the action space. In [11], they approximate the action space while Ref. [28] uses the true action space.

There are some extractive summarization methods that used the pretrained model [32–35].

7. Conclusions

In this work, we put forward an extractive summarization model that is based on Bert and dynamic memory network. In our model, we use a simple semantic matching method to label the training set and train our model using the pre-trained Bert model. A strong ability to extract the text features makes the model effective. Experimental results show that the model based on Bert and dynamic memory network achieves the comparable result against other extractive systems on the datasets. The dynamic memory network with the bi-LSTM encoder we use for the extractive summarization problem achieves good results. In the future, we will incorporate this extractive summarization method to the abstractive method.

Author Contributions: Conceptualization, P.L.; methodology, J.Y.; software, P.L.; validation, J.Y.; formal analysis, J.Y.; data curation, P.L.; writing—original draft preparation, P.L.; writing—review and editing, J.Y.; visualization, P.L.; supervision, J.Y.; project administration, J.Y.; funding acquisition, J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 61862060, No. 61562086, No. 61562087).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We would like to thank the anonymous reviewers for their valuable feedback.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Edmundson, H.P. New Methods in Automatic Extracting. *J. ACM* **1969**, *16*, 264–285. [CrossRef]
2. Lin, C.Y.; Hovy, E. Identifying Topics by Position. In Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington, DC, USA, 31 March–3 April 1997; Association for Computational Linguistics: Stroudsburg, PA, USA, 1997; pp. 283–290. [CrossRef]
3. Bengio, Y.; Ducharme, R.; Vincent, P.; Janvin, C. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* **2003**, *3*, 1137–1155.
4. Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
5. Cao, Z.; Li, W.; Li, S.; Wei, F.; Li, Y. AttSum: Joint Learning of Focusing and Summarization with Neural Attention. In Proceedings of the COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; Association for Computational Linguistics:Stroudsburg, PA, USA, 2016; pp. 547–556.
6. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv* **2014**, arXiv:1409.0473.
7. Rush, A.M.; Chopra, S.; Weston, J. A Neural Attention Model for Abstractive Sentence Summarization. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015. [CrossRef]
8. Nallapati, R.; Zhai, F.; Zhou, B. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 3075–3081.
9. Zhou, Q.; Yang, N.; Wei, F.; Huang, S.; Zhou, M.; Zhao, T. Neural Document Summarization by Jointly Learning to Score and Select Sentences. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15–20 July 2018; Association for Computational Linguistics: Stroudsburg, PA, USA,2018; pp. 654–663.
10. Carbonell, J.; Goldstein, J. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 24–28 August 1998; ACM: New York, NY, USA, 1998; pp. 335–336. [CrossRef]
11. Narayan, S.; Cohen, S.B.; Lapata, M. Ranking Sentences for Extractive Summarization with Reinforcement Learning. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, Louisiana, 1–6 June2018; Association for Computational Linguistics: Stroudsburg, PA, USA, 2018; pp. 1747–1759. [CrossRef]
12. ColloBert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
13. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
14. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2018**, arXiv:1810.04805.
15. Cheng, J.; Lapata, M. Neural Summarization by Extracting Sentences and Words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; Association for Computational Linguistics: Stroudsburg, PA, USA, 2016; pp. 484–494. [CrossRef]
16. Hermann, K.M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching Machines to Read and Comprehend. In *Advances in Neural Information Processing Systems 28*; Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2015; pp. 1693–1701.
17. Kumar, A.; Irsoy, O.; Ondruska, P.; Iyyer, M.; Bradbury, J.; Gulrajani, I.; Zhong, V.; Paulus, R.; Socher, R. Ask me anything: Dynamic memory networks for natural language processing. In Proceedings of the 33rd International Conference on Machine Learning, New York, USA, 19–24 June 2016; pp. 1378–1387.
18. Xiong, C.; Merity, S.; Socher, R. Dynamic memory networks for visual and textual question answering. In Proceedings of the 33rd International Conference on Machine Learning, New York, USA, 19–24 June 2016; pp. 2397–2406.
19. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv* **2014**, arXiv:1412.3555.
20. Cho, K.; van Merrienboer, B.; Bahdanau, D.; Bengio, Y. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In Proceedings of the SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014; pp. 103–111.
21. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
22. Nallapati, R.; Zhou, B.; dos Santos, C.; Gulcehre, C.; Xiang, B. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany, 7–12 August 2016; Association for Computational Linguistics: Stroudsburg, PA, USA,2016; pp. 280–290. [CrossRef]

23. See, A.; Liu, P.J.; Manning, C.D. Get To The Point: Summarization with Pointer-Generator Networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada, July 30–August 4 2017; Association for Computational Linguistics: Stroudsburg, PA, USA, 2017; pp. 1073–1083. [CrossRef]
24. Gehrmann, S.; Deng, Y.; Rush, A. Bottom-Up Abstractive Summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; Association for Computational Linguistics: Stroudsburg, PA, USA, 2018; pp. 4098–4109.
25. Koupaee, M.; Wang, W.Y. WikiHow: A Large Scale Text Summarization Dataset. *arXiv* **2018**, arXiv:1810.09305.
26. Narayan, S.; Cohen, S.B.; Lapata, M. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1797–1807.
27. Zhou, Q.; Yang, N.; Wei, F.; Huang, S.; Zhou, M.; Zhao, T. Neural Document Summarization by Jointly Learning to Score and Select Sentences. *arXiv* **2018**, arXiv:1807.02305.
28. Dong, Y.; Shen, Y.; Crawford, E.; van Hoof, H.; Cheung, J.C.K. BanditSum: Extractive Summarization as a Contextual Bandit. In Proceedings of the EMNLP 2018: 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 3739–3748.
29. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
30. Lin, C.Y. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Text Summarization Branches Out, Barcelona, Spain, 25–26 July 2004.
31. Montague, P. Reinforcement Learning: An Introduction, by Sutton, R.S. and Barto, A.G. *Trends Cogn. Sci.* **1999**, *3*, 360. [CrossRef]
32. Zhang, X.; Wei, F.; Zhou, M. HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Brussels, Belgium, 31 October–4 November 2018; Association for Computational Linguistics: Florence, Italy, 2019; pp. 5059–5069. [CrossRef]
33. Zhong, M.; Liu, P.; Wang, D.; Qiu, X.; Huang, X. Searching for Effective Neural Extractive Summarization: What Works and What's Next,. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; Association for Computational Linguistics: Florence, Italy, 2019; pp. 1049–1058. [CrossRef]
34. Bae, S.; Kim, T.; Kim, J.; Lee, S.G. Summary Level Training of Sentence Rewriting for Abstractive Summarization. In Proceedings of the 2nd Workshop on New Frontiers in Summarization, Hong Kong, China, 4 November 2019; Association for Computational Linguistics: Hong Kong, China, 2019; pp. 10–20. [CrossRef]
35. Liu, Y.; Lapata, M. Text Summarization with Pretrained Encoders. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; Association for Computational Linguistics: Hong Kong, China, 2019; pp. 3730–3740. [CrossRef]