information

MDPI

Article

Skeleton to Abstraction: An Attentive Information Extraction Schema for Enhancing the Saliency of Text Summarization

Xiujuan Xiang [1,2,3], Guangluan Xu [1,2,3,*], Xingyu Fu [1,2,3], Yang Wei [2,3], Li Jin [2,3] and Lei Wang [2,3]

[1] University of Chinese Academy of Sciences, No. 19 (A), Yuquan Road, Shijingshan District, Beijing 100049, China; xiangxiujuan16@mails.ucas.ac.cn (X.X.); fuxingyu07@mails.ucas.ac.cn (X.F.)
[2] Institute of Electronics, Chinese Academy of Sciences, No. 19, North Fourth Ring West Road, Haidian District, Beijing 100190, China; weiyang_tj@outlook.com (Y.W.); jinlimails@gmail.com (L.J.); wanglei19@yeah.net (L.W.)
[3] Key Laboratory of Spatial Information Processing and Applied System Technology, Chinese Academy of Sciences, No. 19, North Fourth Ring West Road, Haidian District, Beijing 100190, China
* Correspondence: gluanxu@mail.ie.ac.cn; Tel.: +86-1881-0211-693

Received: 14 July 2018; Accepted: 28 August 2018; Published: 29 August 2018

Abstract: Current popular abstractive summarization is based on an attentional encoder-decoder framework. Based on the architecture, the decoder generates a summary according to the full text that often results in the decoder being interfered by some irrelevant information, thereby causing the generated summaries to suffer from low saliency. Besides, we have observed the process of people writing summaries and find that they write a summary based on the necessary information rather than the full text. Thus, in order to enhance the saliency of the abstractive summarization, we propose an attentive information extraction model. It consists of a multi-layer perceptron (MLP) gated unit that pays more attention to the important information of the source text and a similarity module to encourage high similarity between the reference summary and the important information. Before the summary decoder, the MLP and the similarity module work together to extract the important information for the decoder, thus obtaining the skeleton of the source text. This effectively reduces the interference of irrelevant information to the decoder, therefore improving the saliency of the summary. Our proposed model was tested on CNN/Daily Mail and DUC-2004 datasets, and achieved a 42.01 ROUGE-1 f-score and 33.94 ROUGE-1, recall respectively. The result outperforms the state-of-the-art abstractive model on the same dataset. In addition, by subjective human evaluation, the saliency of the generated summaries was further enhanced.

Keywords: recurrent neural network (RNN); abstractive text summarization; information extraction; attention mechanism; semantic relevance; saliency of summarization

1. Introduction

With the rapid development of Internet technology, people are exposed to vast amounts of text information every day such as news, blogs, reports, papers, etc. When we are faced with a large amount of disorganized information, quickly and accurately locating the required information becomes a problem to be solved. Automatic text summarization provides an efficient solution to this task. Text summarization can create a shorter version containing the main idea of the source text automatically. We can judge whether an article is interesting to us based on the shorter version. This can greatly reduce the time consumed in retrieving information.

Text summarization is generally divided into two branches, namely, extractive and abstractive. Extractive summarization selects some sentences from the source text to compose a summary.

Abstractive summarization is based on the semantics of the source text to generate novel sentences as the summary. Abstractive summarization is thus more difficult than copying sentences from the source text, and most of the work in the past has been focused on extractive summarization [1–4].

However, in recent years, abstractive summarization based on deep learning has also made great progress. The current popular abstractive model is mostly carried out under the framework of encoder and decoder. In order to improve the accuracy of the decoder, Bahdanau et al. [5] added an attention mechanism to the encoder-decoder framework and produced state-of-the-art performance in machine translation (MT). Due to the similarities between MT and text summarization, the subsequent text summarization follows the model of MT. Under the framework, the encoder reads the source text and understands the semantics of the text, the decoder generates summary words, and the attention mechanism is responsible for aligning the input and the output information to make the output more reliable. Despite the similarities, abstractive summarization is a very different problem from MT. The decoder must receive all contents of the source text in MT, however, in text summarization, the decoder only needs the important information from the source text to generate a summary. Humans also write summaries like this. Before the summary is generated, the important information is first extracted, and then during the process of writing a summary, only the important information is considered. A good summary should be concise and have high saliency, namely, containing more key information. However, based on the current abstractive model, the summary generation is based on all contents of the source text. Under this condition, when the source text contains plenty of information irrelevant to the summary, the encoder cannot correctly represent the semantics of the text. This means that the decoder is influenced by this irrelevant information, thereby resulting in the saliency of the summary declining. As shown in Figure 1, the generated summary has poor saliency.

Source Text
An officer , responding to reports of a suspicious person , shot and killed an unarmed man who was running around in a metro atlanta apartment complex naked . the officer fired two shots when the man charged at him , said cedric alexander , the public safety director of dekalb county . but given that the man was not carrying a weapon , the police department immediately turned over the case to the georgia bureau of investigations for an independent probe . `` what i have requested here. A result of what 's going on currently across this country as it relates to police shootings , " alexander told reporters . the officer was white ; the deceased man was african-american , alexander said . the incident took place monday afternoon at an apartment complex in chamblee , a suburb of atlanta . someone called 911 to report a man `` acting deranged , knocking on doors and crawling around naked , " alexander said . when the officer arrived , the man charged at him , alexander said . `` the officer called him to stop while stepping backward , drew his weapon and fired two shots , " he said . the man , struck twice in the upper body , died .
Reference Summary
Police : officer fired two shots when the man charged at him .The case was immediately turned over to the gbi .
Generated Summary
suspicious person , shot and killed unarmed man who was running around in metro apartment complex naked .the incident took place monday afternoon at an apartment complex in chamblee , a suburb of atlanta .someone called 911 to report a man `` acting deranged , knocking on doors and crawling around naked "

Figure 1. An example of abstractive text summarization. Green font is the key information in the source text. Red font is the key information obtained by current abstractive model.

Based on the above discussion, in order to reduce the interference of the irrelevant information for the decoder, thereby improving the saliency of generated summary, this paper proposes an attentive information extraction model. This model is also proposed with reference to the way that humans write summaries. During the process of people writing a summary, they first read and understand the source text; then they will outline the important information and filter the information that is useless to the summary; next, they compare the important information with true semantics to ensure that the outlined information is correct; finally, they will write summaries. The current attentional encoder-decoder model is able to read and understand the source text as well as write a summary. However, preliminarily outlining the important information and ensuring the correctness of important information have not been realized. Thus, we firstly use an extra attention mechanism, namely, a multi-layer perceptron (MLP) network, to obtain the important information after the encoder and before the decoder. The important information is the skeleton of the source text. Furthermore, the

semantic information between the reference summary and the source text is consistent, so we calculate semantic similarity scores between the reference summary and the extracted important information to ensure the correctness of the extracted information. In order to further enhance the ability of the MLP network, we maximize the similarity score to encourage high semantic similarity between the reference summary and the source text. As one of the targets of the abstractive model is to maximize the probability of target words, we think the decoder has good writing ability. We skip the decoder to maximize the score so that the encoder's semantic expression capabilities and the ability of the MLP network to extract information are improved as much as possible without affecting the ability of the decoder writing a summary. Our model extracts the important information before the decoder and the decoder generates summaries according to the important information. It cannot be influenced by the irrelevant information, therefore it can capture the main idea of the source text more completely and accurately, thus the saliency of the summary is higher.

We conduct experiments on the CNN/Daily Mail and DUC-2004 datasets. Our model achieved a 42.01 ROUGE-1 f-score and 33.94 ROUGE-1 recall, respectively, and outperformed the state-of-the-art abstractive model on the same datasets. In addition, by anonymous and subjective human evaluation, the saliency of the summary generated by our model was further enhanced. The readability of the summary generated by our model was stronger than the baseline model.

2. Related Work

The current abstractive model was carried out based on an encoder-decoder model [6]. This model was originally used in the field of MT. In order to improve the accuracy of the decoder, Bahdanau et al. [5] added the attention mechanism to the model and obtained state-of-the-art results in MT. Due to the strong similarity between text summarization and MT tasks, the current popular text summarization models mostly followed this structure.

In the early days of text summarization studies, most of the work was done around extractive summarization [1–4,7–9]. However, in recent years, the study of text summarization mainly focused on abstractive summarization. Rush et al. [10] proposed a data-driven network model to generate summaries. They used the convolutional neural network (CNN) to encode the source text and used a neural language model to decode a summary. State-of-the-art results were obtained on the DUC-2004 and Gigawords datasets. In an extension of this work, Chopra et al. [11] used Recurrent neural network (RNN) instead of the neural language model in the decoder, resulting in further improvement in the datasets. As RNN can better represent serialized data, Nallapati et al. [12] implemented both the encoder and the decoder using a RNN and constructed a multi-sentence summarization of the dataset CNN/Daily Mail.

Under the framework of an attentional encoder and decoder, researchers began to solve the problem of repeatability, poor readability, and out-of-vocabulary (OOV) words. Vinyals et al. [13] used the pointer mechanism in the encoder-decoder network model to solve the OOV problem. Experiments have proved that the mechanism can achieve good results. Gu et al. [14], Gulcehre et al. [15], and Nallapati et al. [12] also adopted the pointer mechanism on abstractive summarization to solve the OOV problem. See et al. [16] used a similar mechanism to generate summaries. In order to solve the problem of repeatability, the coverage mechanism [17] was introduced. Experiments achieved state-of-the-art results on the CNN/Daily Mail datasets. Suzuki et al. [18] mitigated the repeatability of summaries by evaluating the upper bound frequency of each target word in the encoder and controlling the output word in the decoder. Nema et al. [19] dealt with the sentences input into the model so that they were orthogonal to each other, thereby reducing the repeatability of the generated summaries. Li et al. [20] added latent structured information to the decoder and introduced an editing vector [21] to edit the generated summary, thereby enhancing the readability of the summary. Recently, Paulus et al. [22] applied reinforcement learning (RL) to generate a summary and adopted the attention mechanism inside the decoder.

In addition, Xu [23] used a multi-layer perceptron (MLP) model inside the encoder to predict the weight of each sentence in the source text. This model reduced the interference of irrelevant sentences when generating summaries. Zhou et al. [24] also adopted a MLP model after the encoder to weaken the irrelevant information and improve the model performance. Ma et al. [25] added a similarity comparison module between the generated summaries and the original text after the decoder to improve the semantic relevance of the summary. Ma et al. [26] combined text sentiment classification with text summarization tasks and proposed a hierarchical end-to-end model with a highway network, which achieved good experimental results on the Amazon online review dataset. In another experiment by Ma et al. [27], they proposed a supervised learning model to improve the ability of encoder text representation, thereby improving the result of summarization. Hsu et al. [28] combined extractive and abstractive summarization to generate a summary, this improved the informativity and readability of summaries. Lin et al. [29] controlled the information flow from encoder to decoder to improve the semantic relevance of the summary. Li et al. [30] also combined extractive with abstractive models to generate summaries and improve the informativity of the summary. Celikyilmaz et al. [31] presented deep communicating agents in an encoder-decoder architecture to address the challenges of representing a long document for abstractive summarization. Under the conditions of solving the problem of OOV words and repeatability, our model refers to the idea of Zhou et al. [24], adopting an extra attention mechanism to extract the important information. In order to ensure the correctness of the extracted information and enhance the ability of extra attention mechanisms, we calculate semantic similarity between the reference summary and the extracted information, and maximize the similarity score to encourage high similarity between the reference summary and the extracted information. Experiments show that our model outperformed the state-of-the-art abstractive model and the saliency of the summary generated by our model was further enhanced.

3. Proposed Model

In this section, we will introduce our proposed model in detail. In Section 3.1, we introduce the flow diagram of our model. In Section 3.2, we make an overview of the various parts of the model. In Section 3.3, we describe every part of the model in detail.

3.1. Model Flow Diagram

The flow diagram of our model is shown in Figure 2.

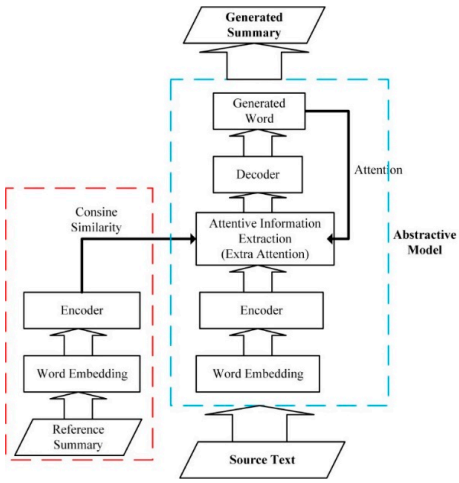

Figure 2. The flow diagram of our model. At the training stage, the Generated Summary in the figure does not exist. Our target is to train an abstractive model, namely, the part drawn by the blue dotted line. At the test stage, our input is only the source text and the part represented by the red dotted line does not exist, the output is the summary generated using the abstractive model.

The input of the model is the source text and the output is the generated summary. Firstly, the source text is embedded into a series of word vectors. Next, these word vectors are encoded to achieve the reading and understanding of the source text. Then, we adopt an extra attention mechanism after the encoder to obtain the important information for generating the summary, thereby reducing the interference of useless information to the decoder. At the training stage, in order to improve the performance of important information extraction, we compare the semantics of the reference summary with the semantics of the important information to obtain the consine similarity score. Note that the reference summary does not exist at the test stage. Finally, the decoder generates summaries according to the important information.

3.2. Model Overview

The concrete architecture of our model is shown in Figure 3. It mainly consists of five parts: source text encoder, extra attention, consine similarity module, reference summary encoder and summary decoder.

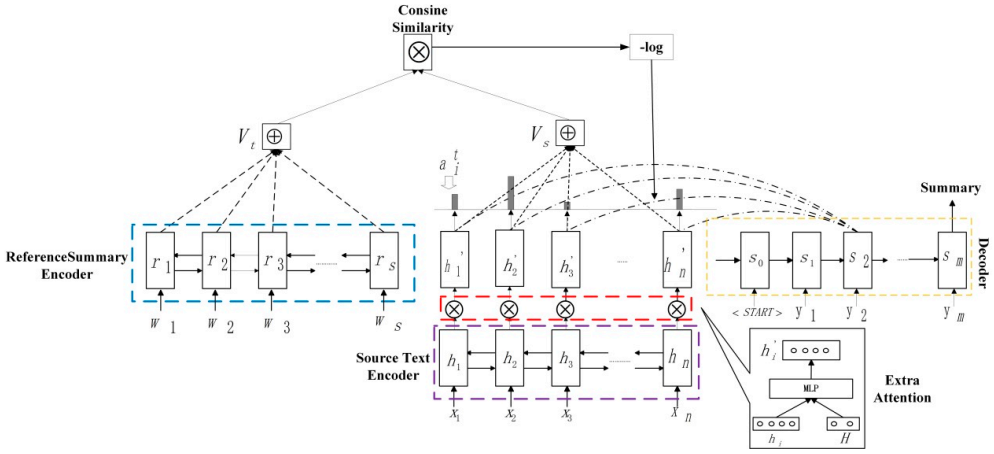

Figure 3. Our proposed model. Before the decoder, an extra attention mechanism is used to extract important information and compare the similarity of the reference summary and the extracted information to ensure the correctness of the necessary information. In order to enhance the ability of the extra attention mechanism in extracting information, the similarity score is fed back to the network, skipping the decoder.

The text encoder uses a bidirectional long short term memory network (Bi-LSTM) to read and represent the source text. It maps the source text to the semantic vector space, forming a series of semantic vectors. After the text encoder, we use the extra attention mechanism to measure the importance of each word in the source text. The extra attention mechanism is a MLP network. At each time step, the output of MLP is a weight vector that represents the importance of the word for the text. Then, we use these weight vectors to weigh the hidden states and form a series of new semantic vectors. These vectors represent the important information of the source text. Next, we also use Bi-LSTM to encode the reference summary. We compare the cosine similarity between the semantic of the reference summary and the semantic of the important information to ensure the correctness of the extracted information input into the decoder. In order to enhance the ability of the extra attention mechanism, we provide a similarity score to the encoder and MLP to maximize it. Lastly, the model adopts unidirectional LSTM to decode the important information and generate summaries. During the generation of summaries, we also use traditional attention mechanisms to provide different attention scores for different parts of the source text at different time steps.

3.3. Model Details

In this section, we will introduce our model in detail. We divided our model into five parts in Section 3.2. Since extra attention and similarity modules work together to extract the useful information and filter the irrelevant information, we can merge them into an attentive information extraction module. Our model has three large blocks, namely, text encoder, attentive information extraction and summary decoder. The text encoder reads and understands the source text. Attentive information extraction is responsible for extracting the important information for summary generation. The summary decoder writes the summary words.

3.3.1. Text Encoder

The text encoder imitates the process of human reading and understanding the source text. This part is responsible for mapping the source text to the semantic vector space and forming a series of semantic vectors. As RNN can better represent the serialized data, the encoder is implemented using RNN. However, the general RNN has the problem of long-short-term dependence and vanishing gradient, thus we adopted the variant LSTM (http://colah.github.io/posts/2015-08-Understanding-LSTMs/).

In order to obtain more complete and accurate vector representations of the source text, we used Bi-LSTM to encode it. Forward LSTM reads word vectors from left to right, resulting in a series of hidden states $(\overrightarrow{h_1}, \overrightarrow{h_2}, \overrightarrow{h_3}, \ldots, \overrightarrow{h_n})$. Backward LSTM reads the word vector in the opposite direction, and obtains a series of hidden states $(\overleftarrow{h_1}, \overleftarrow{h_2}, \overleftarrow{h_3}, \ldots, \overleftarrow{h_n})$. n is the length of the source text.

$$\overrightarrow{h_i} = LSTM(x_i, \overrightarrow{h_{i-1}}) \tag{1}$$

$$\overleftarrow{h_i} = LSTM(x_i, \overleftarrow{h_{i+1}}) \tag{2}$$

We concatenate $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ to represent h_i, i.e., $h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}]$, x_i represents the *i-th* word in the source text. h_i is the semantics of all contents before the *i-th* word in the source text.

3.3.2. Attentive Information Extraction

Although the architecture of text summarization borrows from machine translation (MT), it is a very different problem from MT. In MT, the decoder must fully receive all the information from the source text. In summarization, the effective information of the text is enough for the decoder. However, the current abstractive model generates a summary based on all contents in the source text, which causes the encoder to not correctly represent the text. This makes the information that has been inputted into the decoder imprecise. In this case, the generated summary will not be accurate. This situation is not what we expect. In addition, we observed the process of humans writing a summary. They will outline important information in the text before writing a summary, which can reduce the interference of useless information to the decoder. Thus, we refer to humans writing summaries and propose an attentive information extraction model to solve the problem.

After the text encoder, the network will generate a series of hidden states $(h_1, h_2, h_3, \ldots, h_n)$. In order to completely and accurately represent the entire text H, we concatenate $\overrightarrow{h_n}$ with $\overleftarrow{h_1}$ to obtain it.

$$H = [\overrightarrow{h_n}; \overleftarrow{h_1}] \tag{3}$$

In order to extract the important information, we apply an extra attention mechanism. Here, we introduce a weight vector g_i. It represents the importance of the *i-th* word for the full text. H and h_i are

input into a MLP to obtain g_i. Then, h_i and g_i will carry out dot product operations to get h'_i, which represents the extracted information at time step i.

$$g_i = \sigma(W_s.H + V_i.h_i + b) \tag{4}$$

$$h'_i = h_i \odot g_i \tag{5}$$

where W_s, V_i and b are learnable parameters. $\odot$ is dot product operation.

After information extraction, the important information of the source text is strengthened and the unnecessary or useless information is weakened or ignored. We add these new states (h'_1, h'_2, h'_3, ..., h'_n) to represent the semantics of the source text, namely V_s. It will be inputted into the decoder to generate summaries.

$$V_s = \sum_{i=1}^{n} h'_i \tag{6}$$

We extract important information to the decoder after MLP, but we cannot guarantee the correctness of extracted information. As the semantics of the reference summaries and source texts are consistent, we compare the semantics of reference summaries with the source text to ensure the extracted information's semantic correctness. We refer to the encoder of the source text and also use Bi-LSTM to encode the reference summary, and add all hidden states to represent its semantics V_t.

$$\overrightarrow{r_i} = LSTM(w_i, \overrightarrow{r_{i-1}}) \tag{7}$$

$$\overleftarrow{r_i} = LSTM(w_i, \overleftarrow{r_{i+1}}) \tag{8}$$

$$r_i = [\overrightarrow{r_i}, \overleftarrow{r_i}] \tag{9}$$

$$V_t = \sum_{i=1}^{s} r_i \tag{10}$$

where s is the length of the reference summary and r_i is the semantics of all contents before the *i-th* word in the reference summary.

Here, we adopt cosine similarity to measure the semantic similarity between the reference summary and the source text. This will tell us whether extracted information is correct or not. The similarity score is larger, the extracted information is more accurate.

$$cos(V_s, V_t) = \frac{V_s.V_t}{\|V_s\|.\|V_t\|} \tag{11}$$

In order to improve the information extraction ability of MLP, the similarity score is fed back to the network. In the training process, we maximize the score to encourage the high semantic similarity score between the reference summary and the extracted information. In our model, we minimize the negative log likelihood of the similarity.

$$loss_s = -\log(cos(V_s, V_t)) \tag{12}$$

As the current summarization model's target function is to maximize the possibility of the target word, namely, minimizing its negative log likelihood, we believe that the decoder has good writing skills.

$$loss_t = -logP(W_t^*) \tag{13}$$

where W_t^* is the target word. Therefore, in order not to affect the decoder's writing ability, we feed the similarity score back to the encoder and MLP, skipping the decoder. The final loss function is as follows:

$$loss = \frac{1}{m}\sum_{t=0}^{m} loss_t + \lambda.loss_s \tag{14}$$

where λ is a hyper-parameter and m is the length of the generated summary.

3.3.3. Summary Decoder

We use the unidirectional LSTM to generate summaries after the text encoder and attentive information extraction. We use V_s (in Section 3.3.2) to initialize the LSTM hidden state. It represents all important information in the source text.

$$s_t = LSTM(s_{t-1}, y_t) \tag{15}$$

$$s_0 = V_s \tag{16}$$

where s_t and y_t are the hidden state and the input of LSTM at time step t, respectively.

During the process of decoding, we also use traditional attention mechanisms to pay attention to different parts of the important information at different time steps.

$$e_i^t = V^T.\tanh\left(W_h.h_i' + W_t.s_t + b_i^t\right) \tag{17}$$

$$a^t = softmax\left(e^t\right) \tag{18}$$

$$c_t = \sum_{i=1}^{n} a_i^t.h_i' \tag{19}$$

where W_h, W_t, V and b_i^t are learnable parameters, a_i^t is the attention score at time t to the *i-th* word in the source text, and c_t is the context vector at time t. Finally, in order to solve OOV words and the repeatability of summaries, we also use pointer and coverage mechanisms [16]. Now, the attention score is calculated as follows:

$$c_i^t = \sum_{t'=0}^{t'=t-1} a_i^{t'} \tag{20}$$

$$e_i^t = v^T.tanh\left(W_h.h_i' + W_t.S_t + W_c.c_i^t + b_i\right) \tag{21}$$

where c_i^t is the sum of the attention scores before time t. We will penalize the model when it repeatedly attends to the same location of the source text, namely, minimize the minimum of the sum of the attention scores of the *i-th* word so far and the attention score of the *i-th* word at the current moment. Thus part of the loss function $loss_t$ is changed as follows:

$$loss_t = -logP(W_t^*) + \mu.min\left(a_i^t, c_i^t\right) \tag{22}$$

where μ is a hyper-parameter.

4. Experiment

In this section, we will introduce our experiments in detail, including the datasets we used, the implementation details and the evaluation metrics.

4.1. Datasets

We trained our model on the CNN/Daily Mail dataset [12,32]. This is a news dataset that contains 312,085 documents with multi-sentence summaries. Through statistics, each text contains an average of 766 words spanning 29.74 sentences and the corresponding summary contains 53 words spanning

3.72 sentences. We follow the same pre-processing method described in See et al. [16] to process our datasets. The large size of the dataset makes the process of training very slow. Thus we filtered out texts longer than 500. Eventually, we had 70,065 training pairs, 3806 validation pairs and 3212 test pairs.

In addition, we also tested our model on the DUC-2004 dataset for tasks 1 and 2 [33]. Although DUC 2004 is old, DUC-2004 contains many manual abstracts generated by an expert. It provides a standard dataset for summarization. Thus it is used widely in academic research and industrial applications. Since most past works were evaluated based on DUC 2004, we also used it to evaluate our model. However, it is too small to train neural networks, so we only used it to test our model. The corpus contains 500 documents. Each document has four manual reference summaries and each reference summary contains 75 bytes on average.

4.2. Experiment Details

All our experiments are implemented based on python3 and tensorlow1.2.0. In Table 1, we show our model parameters at the training stage.

Table 1. Our model parameters at the training stage. Max_enc_steps and Max_dec_steps are the allowed maximum length of the source text input encoder and the generated summary, respectively.

Model Parameters	Values
Hidden dimension	256
Embedding dimension	128
Vocabulary size	50K
Max_enc_steps	400
Max_dec_steps	100
Batch size	16
Beam size	4
Learning rate	0.15
μ	1
λ	0.001

We set all hidden state sizes to 256 and word embedding sizes to 128. The vocabulary size was 50,000. Before the model is trained, we did not pre-train the word vectors. We randomly initialized the word vectors at the beginning of training process. We fixed the maximum length of input text at 400. In addition, the length of reference summary was 53 on average, so we set the maximum length of the generated summaries to 100. Besides, when we tested our model on DUC-2004, we changed the maximum length of the generated summaries to 25, because the reference summary contained 75 bytes on average. The batch size was 16. We adopted the beam search algorithm to generate summaries and set the beam size to four; thus the batch size was also changed to four at the test stage. We used AdaGrad [34] to optimize our model and its learning rate, the initial accumulator value was 0.15 and 0.1, respectively. The hyper-parameter λ and μ were set to 0.001 and 1, respectively.

At the end of training, the loss of seq2seq, namely, $loss_t$ (in Section 3.3.2) converged to about 2.6 from an initial value of about 7.0, and the coverage loss converged to 0.2 from an initial value of about 0.5.

4.3. Evaluation Metrics

1. ROUGE

We evaluated our model using ROUGE [35]. ROUGE is a common evaluation metric in text summarization. It measures the overlap of lexical units between reference summaries and generated

summaries, such as unigrams, bigrams and longest common subsequence. The calculation of ROUGE is as follows:

$$ROUGE - N = \frac{\sum_{s \in \{reference\ summaries\}} \sum_{gram_n} Count_{match}(gram_n)}{\sum_{s \in \{reference\ summaries\}} \sum_{gram_n} Count(gram_n)} \quad (23)$$

where N represents the length of $n\text{-}gram$, $\{reference\ summaries\}$ is the reference summary, $Count_{match}(gram_n)$ is the number of n-grams co-occurring in the reference summaries and generated summaries, and $Count(gram_n)$ is the number of n-grams in reference summaries. For the CNN/Daily Mail dataset, we calculated ROUGE F1 (https://blog.csdn.net/u014380165/article/details/77493978). However, for the DUC-2004 dataset, because most works were evaluated in the past based on ROUGE recall and the official DUC metric is also ROUGE recall, we also used it to evaluate our model.

2. Human evaluation

For text summarization, to some extent, ROUGE only evaluates literal similarity between the reference summary and the generated summary. For the saliency of the generated summaries, there is no suitable way to evaluate them automatically. Thus, in order to evaluate the saliency of the summaries generated by our model, we randomly selected some examples for visual evaluation. We compared the summary generated by our model and See et al. [16] in terms of informativity. The summary containing more key information had higher saliency.

In addition, in order to make the evaluation experiment more representative, we randomly picked more examples. Each example contains three parts, namely, the reference summary, the summary generated by the model of See et al. [16] and the summary generated by our model. We assigned them to three different human evaluators to score each summary. The saliency scoring criteria are shown in Table 2. Finally, we collected the results of different human evaluators and calculated the mean value. Note that during the process we did not tell them which summary was generated by our model and which summary was generated by the model of See et al. [16]. We only told them which summary was the reference summary.

Table 2. Saliency scoring criteria. Relevance indicates the informativity of the summary by the model (our model or See et al. [16] model). Score is from 0 to 5 and higher scores are better. Higher score indicates higher saliency.

Relevance	Score
No relevance	0
Little relevance	1
A little relevance	2
Relevance	3
A lot of relevance	4
Great relevance	5

Besides, for the text summarization, readability is also an important indication for evaluation of the quality of the summarization. Thus we also randomly selected some examples and assigned them to three different evaluators to evaluate the readability of the summaries. We mainly evaluated the syntax and the grammar of the summary. The three evaluators scored each summary according to the syntax and the grammar. The scoring details are shown in Table 3. Similarly, we calculated the mean value of the three evaluators as the final readability result. Note, our readability evaluation process was also anonymous.

Table 3. Readability Scoring Criteria. Higher score indicates stronger readability. Score is from 1 to 5 and higher scores are better.

Syntax	Grammar	Score
Very Poor	Very Bad	1
Poor	Bad	2
Barely Acceptable	Barely Acceptable	3
Good	Good	4
Very good	Very Good	5

3. Weigh heat map

We visualized the weight vector obtained by MLP, namely, g_i (in Section 3.3.2), to check whether our model extracted important information before the decoder. However, because g_i is a high dimensional vector, it is difficult to visualize it directly. In order to visualize it, we converted it to a scalar. As we all know, the biggest relevance appears between themselves. Thus, we calculated the weight vector between the source text and the source text as the gold vector. Then, we calculated the Euclidean distance between the gold vector and the weight vector at each time step. With this, we can convert a high dimensional weight vector to a scalar. The concrete calculation is as follows:

$$\mathrm{G} = \sigma(W_s.H + V_i.H + b) \tag{24}$$

$$a_i = \sqrt{\sum_{i=1}^{k}(G - g_i)^2} \tag{25}$$

where G is the gold vector, k is the dimension of the weight vector and a_i represents the scalar corresponding to g_i. H is the source text (in Section 3.3.2). We will visualize the a_i to represent the weight heat map.

5. Results and Discussion

In this section, we report the ROUGE F1, ROUGE recall for the CNN/Daily Mail and DUC-2004 test sets, respectively. We use the pyrouge package (https://pypi.org/project/pyrouge/) and the official ROUGE script (https://github.com/summanlp/evaluation/tree/master/ROUGE-RELEASE-1.5.5) to obtain our ROUGE scores. In addition, we will show the result of the saliency evaluation and readability evaluation. Then, we will provide the weight heat map. Finally, we will discuss our results.

5.1. Results

For CNN/Daily Mail and DUC-2004, their reference summaries have different lengths, so we set different sizes at the test stage. For CNN/Daily Mail, the length of the reference summaries was 53 on average. We set the maximum decoder steps to 100. Table 4 shows the results for CNN/Daily Mail. We can see that our model achieves state-of-the-art results without reinforcement learning. We only used maximum likelihood (ML) to train our model. We did not use RL to train the model, but the experiments of Celikyilmaz et al. [31] show that RL can apparently improve the value of ROUGE. This may become a part of our future work.

Table 4. ROUGE F1 on CNN/Daily Mail. All our ROUGE scores have a 95% confidence interval in the official ROUGE script.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Maximum Likelihood			
Words-lv2k-temp-att	35.46	13.30	32.65
Pointer-Generator + Coverage	39.53	17.28	36.38
ML, with intra-attention	38.30	14.81	35.49
Controlled summarization	39.75	17.29	36.54
End2end w/inconsistency loss	40.68	17.97	37.13
Attentive Information Extraction (Ours)	42.01	20.09	38.78
Reinforcement Learning			
DCA MLE + SEM	41.11	18.21	36.03
DCA MLE + SEM + RL	41.69	19.47	37.92

Words-lv2k-temp-att: Nallapati et al. [12] used a pointer mechanism to solve the problem of out-of-vocabulary and used the feature-rich-encoder to embed the word.

Pointer-Generator + Coverage: See et al. [16] also adopted pointer to handle OOV words and introduced an extended vocabulary. Besides, in order to prevent the repeatability, the model used coverage to solve it. This model was our baseline model.

ML, with intra-attention: Paulus et al. [22] used an attention mechanism inside the decoder to solve the problem of repeatability.

Controlled summarization: Fan et al. [36] presented a neural summarization model to enable users to specify some high level attributes, such as the desired length, style, and the entities, in order to control to the shape of the generated summaries to better suit users' needs.

End2end w/inconsistency loss: Hsu et al. [28] combined an extractive model with an abstractive model to generate summaries.

DCA MLE + SEM + RL: Celikyilmaz et al. [31] presented deep communicating agents in an encoder-decoder architecture to address the challenges of representing a long document for abstractive summarization and trained their model using reinforcement learning to generate summaries.

DCA MLE + SEM: Celikyilmaz et al. [31] did not use RL to train their model.

For DUC-2004, the reference summary contains 75 bytes on average, so we change the maximum decoder steps to 25. In addition, in the past, most of the work on DUC-2004 was evaluated using ROUGE recall, so we also obtained ROUGE recall for DUC-2004. The results are shown in Table 5. The results show that our model outperforms the state-of-the-art baseline model in ROUGE-1 and ROUGE-L recall with least 1.9 points.

Table 5. ROUGE recall on DUC-2004. All our ROUGE scores have a 95% confidence interval in the official ROUGE script.

Model	ROUGE-1	ROUGE-2	ROUGE-L
ABS+	28.18	8.49	23.81
Words-lv5k-1sent	28.61	9.42	25.24
C2R + Atten	28.97	8.26	24.06
SEASS	29.21	9.56	25.51
AC-ABS	32.03	10.99	27.86
Attentive Information Extraction (Ours)	33.94	8.99	28.44

ABS+: Rush et al. [10] used CNN encode the source text and neural language model to decode.

Words-lv5k-1sent: Nallapati et al. [12] trained the model on the first sentence from the source text and adopted the large vocabulary trick based on an attentional encoder-decoder model.

C2R + Atten: Chopra et al. [11] used a CNN to encode and RNN to decode, which outperformed the ABS+ model.

SEASS: Zhou et al. [24] adopted selective encoding to extend the seq2seq model, which reduced the burden of the decoder.

AC-ABS: Li et al. [37] employed an actor-critic framework to enhance the traditional abstractive model to improve the quality of the generated summaries.

We randomly selected three examples to evaluate visually. The result is shown in Figure 4. We can see that the summary generated by our model captures more key information contained in the source text. This indicates that our summaries have a higher saliency than the summary of See et al. [16].

Source Text 1
Former vice president walter mondale was released from the mayo clinic on saturday after being admitted with influenza , hospital spokeswoman kelley luckstein said. " he's doing well . We treated him for flu and cold symptoms and he was released today, " she said . Mondale , 87 , was diagnosed after he went to the hospital for a routine checkup following a fever , former president jimmy carter said friday . "He is in the bed right this moment , but looking forward to come back home , " carter said during a speech at a nobel peace prize forum in minneapolis . "He said tell everybody he is doing well . " mondale underwent treatment at the mayo clinic in rochester , minnesota . the 42nd vice president served under carter between 1977 and 1981 , and later ran for president , but lost to ronald reagan . but not before he made history by naming a woman , u.s. rep. geraldine a. ferraro of new york , as his running mate . before that , the former lawyer was a u.s. senator from minnesota . his wife , joan mondale , died last year .
Gold Summary
Walter mondale was released from the mayo clinic on saturday , hospital spokeswoman said . The former vice president , 87 , was treated for cold and flu symptoms .
Generated Summary(See et al., 2017)
mondale , 87 , was diagnosed after he went to the hospital for a routine checkup following a fever , hospital spokeswoman says . Mondale underwent treatment at the mayo clinic in rochester , minnesota .
Our Model
"We treated him for flu and cold symptoms at the mayo clinic and he was released today," she says. Mondale,87, was diagnosed after he went to the hospital for a routine checkup.
Source Text 2
around in a metro atlanta apartment complex naked . the officer fired two shots when the man charged at him , said cedric alexander , the public safety director of dekalb county . but given that the man was not carrying a weapon , the police department immediately turned over the case to the georgia bureau of investigations for an independent probe . `` what i have requested here. A result of what 's going on currently across this country as it relates to police shootings , " alexander told reporters . the officer was white ; the deceased man was african-american , alexander said . the incident took place monday afternoon at an apartment complex in chamblee , a suburb of atlanta . someone called 911 to report a man `` acting deranged , knocking on doors and crawling around naked , " alexander said . when the officer arrived , the man charged at him , alexander said . `` the officer called him to stop while stepping backward , drew his weapon and fired two shots , " he said . the man , struck twice in the upper body
Gold Summary
Police : officer fired two shots when the man charged at him .The case was immediately turned over to the gbi .
Generated Summary(See et al., 2017)
suspicious person , shot and killed unarmed man who was running around in metro apartment complex naked .the incident took place monday afternoon at an apartment complex in chamblee , a suburb of atlanta .someone called 911 to report a man `` acting deranged , knocking on doors and crawling around naked "
Our Model
the officer fired two shots when the man charged at him , said cedric alexander , the public safety director of dekalb county .but given that the man was not carrying a weapon , the police department immediately turned over the case to the georgia bureau of investigations
Source Text 3
your gift will be multiplied . that may sound like an esoteric adage , but when zully broussard selflessly decided to give one of her kidneys to a stranger , her generosity paired up with big data . it resulted in six patients receiving transplants . that surprised and wowed her . `` i thought i was going to help this one person who i do n't know , but the fact that so many people can have a life extension , that 's pretty big , " broussard told cnn affiliate kgo . she may feel guided in her generosity by a higher power . `` thanks for all the support and prayers , " a comment on a facebook page in her name read . `` i know this entire journey is much bigger than all of us . i also know i 'm just the messenger . " cnn can not verify the authenticity of the page . but the power that multiplied broussard 's gift was data processing of genetic profiles from donor-recipient pairs . it works on a simple swapping principle but takes it to a much higher level , according to california pacific medical center in san francisco . . in late march , the medical center is planning to hold a reception for all 12 patients .such long-chain transplanting is rare . it 's been done before , california pacific medical center said in a statement , but matching up the people in the chain has been laborious and taken a long time . that changed when a computer programmer named david jacobs received a kidney transplant . he had been waiting on a deceased donor list , when a live donor came along -- someone nice enough to give away a kidney to a stranger .
Gold Summary
zully broussard decided to give a kidney to a stranger . a new computer program helped her donation spur transplants for six kidney patients .
Generated Summary(See et al., 2017)
the super swap works on a simple swapping principle but takes it to a much higher level .the chain of surgeries is to be wrapped up friday .in late march , the medical center is planning to hold a reception for all 12 patients .
Our Model
zully broussard selflessly decided to give one of her kidneys to a stranger , her .she may feel guided in her generosity by a higher power .the chain of surgeries is to be wrapped up friday

Figure 4. Examples of abstractive summarization. Green font is the key information of the source text and red font represents the effective information generated by the abstractive model. The source text (1–3) represents original texts, the gold summary is the reference summary, the generated summary is the summary by the model of See et al. [16] and our model represents the summary by our proposed model.

In addition, we picked 100 examples randomly and assigned them to three different people to score anonymously. The result of the saliency evaluation is shown in Table 6. From the result, the summary that is generated by our model has a higher relevance score than the unimproved model, so our proposed model enhances the saliency of text summarization. Besides, we also selected 100 examples randomly and assigned them to three different evaluators to score for readability. The syntax score and grammar scores are presented in Table 7. We found that the summary generated by our model had a higher syntax score and a higher grammar score than the summary generated by See et al. Thus we can say that the summary generated by our model has stronger readability.

Table 6. Saliency evaluation results. See et al. [16] is the summary generated by the model of See et al. [16] and our model is the summary generated by our model.

Summary	Evaluator 1	Evaluator 2	Evaluator 3	Average Score
See et al. [16]	3	2.86	3.22	3.03
Our model	3.5	3.12	3.36	3.33

Table 7. Readability evaluation results. See et al. [16] is the summary generated by the model of See et al. [16] and our model is the summary generated by our model. A/B: A represents the score of the syntax and B indicates the score of the grammar.

Summary	Evaluator 1	Evaluator 2	Evaluator 3	Average Score
See et al. [16]	4.23/2.97	4.09/3.01	4.50/3.22	4.27/3.07
Our model	4.31/3.08	4.28/3.20	4.71/3.53	4.43/3.27

Finally, we randomly selected an example to visualize the weight a_i (in Section 4.2). Figure 5 shows the result, in which we can see that key words in the source text were picked by MLP, such as "Cambodian", "rejected", "demands", "talks", "outside", "political", "Government", "opposition", "asked", "meeting", etc. This shows that our extra attention mechanism and similarity module determined the importance of each word in the source text. They obtained the important information before the decoder. This effectively reduced the interference of irrelevant information to the decoder. Therefore, the generated summaries contain more key information, namely, their saliency is higher.

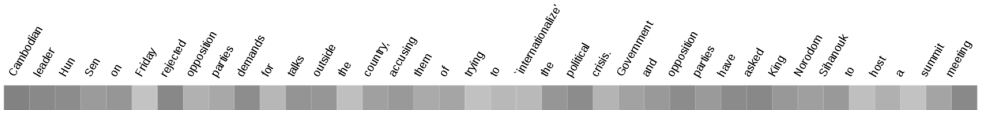

Figure 5. The weight heat map. The word in the picture is the source text. The darker color has greater weight and corresponding word is more important. The reference summary was "Cambodian government rejects opposition's call for talks abroad". The generated summary by our model was "Cambodian leader Hun Sen rejected opposition parties demands for talks outside the country".

5.2. Discussion

Current abstractive models implicitly apply attention mechanisms to extract the key information while the summaries are generating. We think the model benefits from explicitly extracting key information before the decoder. We propose an attentive information extraction model to obtain the important information before the decoder. In Section 5.1, the result showed that our model effectively reduced the interference of the irrelevant information in the source text. This makes the summary more accurate and the saliency of the summary higher than the baseline model [16]. Besides, through readability evaluation, we found that the summary generated by our model had stronger readability.

However, the target of the abstractive model was not only to generate a summary with higher saliency, but also to generate more novel n-grams as in the reference summaries. In order to evaluate the abstractive ability of our model, we conducted detailed statistical analysis about the percentage of new n-grams for DUC 2004 and CNN/Daily Mail. The result is shown in Figures 6 and 7.

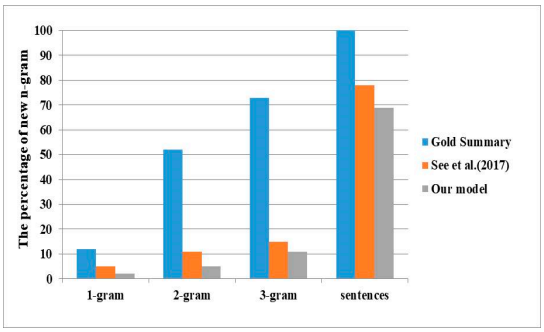

Figure 6. The percentage of new n-grams for CNN/Daily Mail. Larger percentage indicates stronger abstraction.

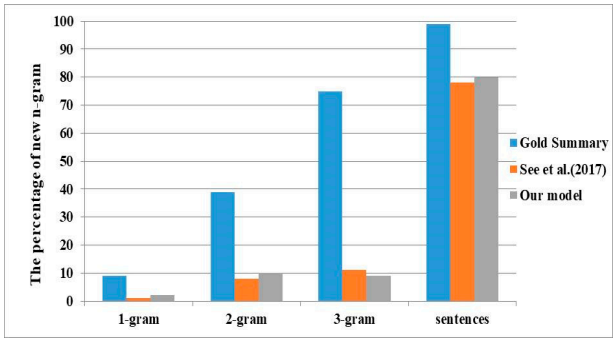

Figure 7. The percentage of new n-grams for DUC 2004. Larger percentage indicates stronger abstraction.

From Figures 6 and 7, we can see that although our model is abstractive, it does not produce new n-grams as often as reference summaries. For CNN/Daily Mail, the model of See et al. [16] produced more novel n-grams than our model. However, we can also see in Figure 4 that although the model of See et al. produced more new n-grams, most of them were erroneous. Although our model produces less novel n-grams, the saliency of the summaries generated by our model was higher and most of the summaries were correct. Thus, our attentive information extraction schema is still useful, apart from the fact that most of the contents was copied from the source text. For DUC 2004, we found the result of our model and See et al. [16] to be less different. The number of novel n-grams for DUC was less than for CNN/Daily Mail on the whole. Maybe the length of the summaries was too short, so the summary generating process was already over when the model massively began to generate new n-grams. Thus the percentage was lower than for CNN/Daily Mail. In this situation, regardless of which model used, the result of our model and the model of See et al. was similar.

Additionally, the probability of generating a novel word also provides a measure of the abstractive ability of the model. Here we used P_{gen} [16] to represent the probability. In order to measure the abstractive ability of our model, we recorded the value of P_{gen} at the beginning of training and at the end of training. We found that P_{gen} started with a value of about 0.26 then increased, converging to about 0.55 by the end of training. This shows that the model first learns to mostly copy, then learns to generate. However, during the test stage, P_{gen} was very low, and only had a mean value of 0.16. The result was similar for See et al. [16]. For this reason, we agree with the opinion of See et al. namely that the model receives word-by-word supervision in the form of the reference summary during the

training stage, but during the test it does not. This is far from the purpose of abstractive summarization. Solving this problem without affecting the performance of the model is a part of our future work. Perhaps we can additionally adopt RL to train our model. We can calculate the rate of abstraction to encourage a higher rate, thereby making the model produce more novel n-grams. Besides, if we want to generate a summary that is more similar to the reference summary, we can also adopt RL to encourage larger ROUGE so as to improve the performance of the model. Maybe this way also can improve the degree of abstraction.

In addition, our model extracts the important information before the decoder, thereby enhancing the saliency of the summary, but we can see that it cannot completely filter all irrelevant information from Figure 5, such as "accusing", "parties", "the", etc. Maybe we can apply the hard attention mechanism [38] to solve the problem in the future, but hard attention may result in the loss of information if the information extraction mechanism is not very good. Besides, the ROUGE-2 recall reduced obviously for DUC-2004 (see Section 5.1). This is a negative outcome. In order to understand the reason for this, we tried to increase the length of Max_dec_steps (in Section 4.2) to 30; in this case the ROUGE-2 recall was 9.75. If we continued increasing the length to 35, the ROUGE-2 recall also continued increasing. The result is shown in Table 8.

Table 8. Results for DUC-2004. +Max_dec_steps (25/30/35) represents setting different lengths for Max_dec_steps.

Our Model	ROUGE-1	ROUGE-2	ROUGE-L
+Max_dec_steps 25	33.94	8.99	28.44
+Max_dec_steps 30	36.96	9.75	30.69
+Max_dec_steps 35	39.14	10.26	32.17

We can infer that the result declines because the CNN/Daily Mail does not match the DUC-2004. The reference summary length for CNN/Daily Mail was 53 words on average, but for DUC-2004 it was only 75 bytes. Even so, the ROUGE-1 and ROUGE-L recall increased. We mainly considered ROUGE-L, which represents the rate of the longest common subsequence between the reference summary and the generated summary.

As we adopted three LSTM models, the training speed of our model was very slow. During the training process, we applied some tricks such as discarding the source text with a length over 500 and setting the Max_enc_steps and Max_dec_steps to 250 and 50 respectively in the early stages of training. As CNN/Daily Mail is a news dataset, we think the key information is shown in the first half of the text. Therefore, we fixed the maximum length of input text at 250. When the model began to converge, we changed this to 400 and 100, which effectively speeded up the training.

In general, our model has the above weaknesses, but using anonymous and subjective human evaluation, the saliency of the generated summary was enhanced and the readability of the generated summary was also better than the baseline model. The result for CNN/Daily Mail and DUC-2004 also outperformed the state-of-the-art baseline model. In the future, we will encourage our model to write a summary more abstractively using RL and try to adopt the hard attention mechanism before the decoder to extract important information.

6. Conclusions

In this work, our target was to enhance the saliency of the summary in abstractive text summarization. In order to achieve this, we proposed an attentive information extraction model to obtain the skeleton of the source text, namely, the important information for the decoder. We conducted our experiments using CNN/Daily Mail and DUC-2004. The experiments showed that our proposed model can effectively extract important information in the source text before the decoder. In addition, we achieved a 42.01 ROUGE-1 f-score and 33.94 ROUGE-1 recall for the CNN/Daily Mail and DUC 2004 datasets, respectively. Our results outperformed the state-of-the-art abstractive

model by at least 1.33 points for the CNN/Daily Mail dataset. For DUC 2004, our model outperformed the state-of-the-art model by at least 1.9 points. Finally, using human evaluation, the saliency of the summaries generated by our model was further enhanced. The readability of the summaries generated by our model was better than the baseline model. As a part of our future work, we plan to apply RL and hard attention mechanisms to the abstractive model to further improve the performance of the model.

Author Contributions: X.X. conceived the idea, performed the experiments and wrote the paper. Y.W. provided the ideas for the experiments. L.J. helped build the experimental environment. G.X., X.F. and L.W. provided writing guidance. Besides, Y.W. and L.J. also helped revise the paper.

Funding: This work was funded by the Pre-research Project (Grant No.: 31510010502), and the Research of Nuclear emergency Application Technology and Application Demonstration (Grant No.: 41-Y30B12-9001-17/18).

Acknowledgments: We are thankful to Key Laboratory of Spatial Information Processing and Applied System Technology, Chinese Academy of Sciences for providing support in the experimental condition. Thanks for Hermann et al. to provide the raw data. And thanks for all authors to their efforts.

Conflicts of Interest: The authors have no relevant financial interests in this article and no potential conflicts of interest to disclose. Research data is some public datasets.

References

1. Cheng, J.; Lapata, M. Neural summaryzation by extracting sentences and words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 484–494.
2. Cao, Z.; Chen, C.; Li, W.; Li, S.; Wei, F.; Zhou, M. TGSum: Build Tweet Guided Multi-Document Summarization Dataset. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 2906–2912.
3. Yang, Y.; Bao, F.; Nenkova, A. Detecting (un) important content for single-document news summarization. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain, 3–7 April 2017; pp. 707–712.
4. Isonuma, M.; Fujino, T.; Mori, J.; Matsuo, Y.; Sakata, I. Extractive summarization using multi-task learning with document classification. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 2101–2110.
5. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**.
6. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In Proceedings of the Advances in Neural Information Processing Systems Conference, Montreal, QC, Canada, 8–13 December 2014; pp. 3104–3112.
7. Cao, Z.; Wei, F.; Dong, L.; Li, S.; Zhou, M. Ranking with Recursive Neural Networks and Its Application to Multi-Document Summarization. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–29 January 2015; pp. 2153–2159.
8. Nallapati, R.; Zhai, F.; Zhou, B. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–10 February 2017; pp. 3075–3081.
9. Cao, Z.; Li, W.; Li, S.; Wei, F.; Li, Y. Attsum: Joint learning of focusing and summarization with neural attention. *arXiv* **2016**.
10. 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; pp. 379–389.
11. Chopra, S.; Auli, M.; Rush, A.M. Abstractive sentence summarization with attentive recurrent neural networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 93–98.

12. Nallapati, R.; Zhou, B.; 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, 11–12 August 2016; pp. 280–290.
13. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer networks. In Proceedings of the Advances in Neural Information Processing Systems Conference, Montreal, QC, Canada, 7–12 December 2015; pp. 2692–2700.
14. Gu, J.; Lu, Z.; Li, H.; Li, V.O. Incorporating copying mechanism in sequence-to-sequence learning. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 1631–1640.
15. Gulcehre, C.; Ahn, S.; Nallapati, R.; Zhou, B.; Bengio, Y. Pointing the unknown words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 140–149.
16. 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, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1073–1083.
17. Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; Li, H. Modeling coverage for neural machine translation. *arXiv* **2016**.
18. Suzuki, J.; Nagata, M. Cutting-off Redundant Repeating Generations for Neural Abstractive Summarization. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain, 3–7 April 2017; pp. 291–297.
19. Nema, P.; Khapra, M.; Laha, A.; Ravindran, B. Diversity driven attention model for query-based abstractive summarization. *arXiv* **2017**.
20. Li, P.; Lam, W.; Bing, L.; Wang, Z. Deep Recurrent Generative Decoder for Abstractive Text Summarization. *arXiv* **2017**.
21. Guu, K.; Hashimoto, T.B.; Oren, Y.; Liang, P. Generating Sentences by Editing Prototypes. *arXiv* **2017**.
22. Paulus, R.; Xiong, C.; Socher, R. A deep reinforced model for abstractive summarization. *arXiv* **2017**.
23. Xu, J. Improving Social Media Text Summarization by Learning Sentence Weight Distribution. *arXiv* **2017**.
24. Zhou, Q.; Yang, N.; Wei, F.; Zhou, M. Selective encoding for abstractive sentence summarization. *arXiv* **2017**.
25. Ma, S.; Sun, X.; Xu, J.; Wang, H.; Li, W.; Su, Q. Improving semantic relevance for sequence-to-sequence learning of Chinese social media text summarization. *arXiv* **2017**.
26. Ma, S.; Sun, X.; Lin, J.; Ren, X. A Hierarchical End-to-End Model for Jointly Improving Text Summarization and Sentiment Classification. *arXiv* **2018**.
27. Ma, S.; Sun, X.; Lin, J.; Wang, H. Autoencoder as Assistant Supervisor: Improving Text Representation for Chinese Social Media Text Summarization. *arXiv* **2018**.
28. Hsu, W.T.; Lin, C.K.; Lee, M.Y.; Min, K.; Tang, J.; Sun, M. A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss. *arXiv* **2018**.
29. Lin, J.; Sun, X.; Ma, S.; Su, Q. Global Encoding for Abstractive Summarization. *arXiv* **2018**.
30. Li, C.; Xu, W.; Li, S.; Gao, S. Guiding Generation for Abstractive Text Summarization Based on Key Information Guide Network. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 55–60.
31. Celikyilmaz, A.; Bosselut, A.; He, X.; Choi, Y. Deep communicating agents for abstractive summarization. *arXiv* **2018**.
32. Hermann, K.M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching machines to read and comprehend. In Proceedings of the Advances in Neural Information Processing Systems Conference, Montreal, QC, Canada, 7–12 December 2015; pp. 1693–1701.
33. Over, P.; Dang, H.; Harman, D. DUC in context. *Inf. Process. Manag.* **2007**, *43*, 1506–1520. [CrossRef]
34. Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* **2011**, *12*, 2121–2159.
35. Lin, C.Y. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004, Barcelona, Spain, 25–26 July 2004.
36. Fan, A.; Grangier, D.; Auli, M. Controllable Abstractive Summarization. *arXiv* **2017**.

37. Li, P.; Bing, L.; Lam, W. Actor-critic based training framework for abstractive summarization. *arXiv* **2018**.
38. Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhutdinov, R.; Zemel, R.S.; Bengio, Y. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 2048–2057.

© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).