Document Summarization using Word and Part-of-speech based on Attention Mechanism

To cite this article: Fu Zhao et al 2019 J. Phys.: Conf. Ser. 1168 032008

View the article online for updates and enhancements.
Document Summarization using Word and Part-of-speech based on Attention Mechanism

Fu Zhao¹, Bing Quan², Jing Yang², Jing Chen², Yubo Zhang² and Xu Wang¹,a
¹College of Computer Science, Sichuan University, 610064 Chengdu Sichuan, China
²China Mobile (Suzhou) Software Technology Co., LTD, 215000 Suzhou Jiangsu, China
a Corresponding author: shawnwongmilab@gmail.com

Abstract. Automatic text summarization is one of the research hotspots in natural language processing (NLP). Recently, there has been a considerable research interest in summarization. While most of the current researches about automatic text summarization are only based on word embedding features, which has the problem of too few text features and can’t effectively utilize some other features of the text. In this paper, we propose a novel deep learning model called Document Summarization using Word and Part-of-speech based on Attention Mechanism (WPABS). We employ the word embedding and part-of-speech embedding to make full use of the features of the text and evaluate our model on English datasets Gigaword and DUC-2004. The experimental results show that our model is better than most methods. Also, compared with WABS(Word Attention-Based Summarization), our proposed model performs better. It is proved that it is useful to combine the word and part-of-speech.

1. Introduction
With the rapid development of the Internet, network information has shown an explosive growth, including a variety of videos, pictures, and text, of which text information occupies the most important part. However, because of the huge scale of network data and the complexity of data, people need to spend a lot of time finding the information they care about from the text. Therefore, it is urgent to find a method that can compress the text information and filter out the useless information, which can quickly and accurately obtain useful information from the Internet.

As one of the hot topics in Natural Language Processing, automatic text summarization aims to capture the central idea of the original text and describe the full text with the shortest information as far as possible. It can help users browse and digest the text information on the Internet better. In 1958, Luhn [1] published the first paper about the summarization. He proposed a method based on word frequency to generate text abstracts by calculating the frequency of keywords in the text to find the central sentence of the text. Since then, after many scholars’ research, automatic text summarization has made great progress.

At present, automatic text summarization can be divided into two categories, extractive and abstractive. The former extracts the existing sentences from the documents and combines them together to ensure the readability of the sentences. For example, the LexRank algorithm proposed by Erkan et al. [2] which introduced a stochastic graph-based method for computing the relative importance of textual units for Natural Language Processing, which classified text and vocabulary mainly by calculating the similarity between sentences. Beaux et al. [3] proposed the phrase
reinforcement algorithm, which constructed a word graph with the topic as the root node, each sentence was converted into a word chain and added to the graph. Then, it updated the words which appeared many times to generate a weighted directed acyclic graph. At last, it selected the path with the maximum weight as the summary. Generally speaking, the extractive summary is relatively simple and mature, while the quality of extraction is not so good. Abstract text summarization is mainly based on the comprehension of the document and reorganization of the text content, which is much more like human to summarize the document. In recent years, deep learning has made a breakthrough in many fields of Natural Language Processing, such as part-of-speech tagging [4], text segmentation [5], intelligent question and answering [6] and named entity recognition [7]. More and more researchers also use the deep neural network for automatic text summarization. In 2014, Google proposed the Sequence-to-Sequence model [8, 9], which opened the upsurge of end-to-end network research in NLP.

In this paper, we propose an automatic text summarization method which integrates word and part-of-speech. Because most of the current researches about automatic text summarization are only based on word embedding features, which has the problem of too few text features and can’t effectively utilize some other features of the text. For this reason, we add the part-of-speech features which can effectively use the features of each word to learn the part-of-speech combination of words by deep semantic features and better generate text summarization.

2. Related work

2.1 The methods of document summarization

Recently, deep learning has made breakthroughs. In the field of automatic text summarization, Rush et al. [10] proposed a data-driven understanding summarization model, which used seq2seq structure with a convolutional neural network as encoder, feed-forward neural network as decoder. Hu et al. [11] constructed a large-scale and high-quality Chinese text summary dataset LCSTS and experimented on the dataset using seq2seq model, which provided a basic learning method for the study of Chinese automatic text summarization. Gehring et al. [12] proposed an architecture based entirely on convolutional neural networks(ConvS2S). The model was mainly used for machine translation which achieved the state-of-the-art performance in some translation tasks. The authors also tried to use the model for automatic text summarization, the experimental results showed the ConvS2S could achieve the performance of the state-of-the-art. Based on ConvS2S, Wang et al. [13] proposed a topic-aware topic ConvS2S model with reinforcement learning for automatic text summarization which can get some high-level contextual information for the summarization. It advanced state-of-the-art methods on various benchmark dataset. Also, it combined the reinforcement learning and deep learning, which was a good idea to combine them in the automatic text summarization task.

2.2 Attention mechanism

Attention mechanism can concentrate on its input subsets, which can find more critical information about the current target from many inputs. It borrows from human visual attention when human scans an object. People usually focus on a specific area of the object while ignoring some secondary information. In recent years, neural network based on attention mechanism has gradually become a hot topic in deep learning. In 2014, Mnih et al. [14] used attention mechanism to classify images on RNN models, focusing on important parts of an image and reducing the complexity of tasks. After that, the research upsurge based on attention mechanism neural network is opened up. Subsequently, Bahdanau et al. [15] applied the attention mechanism to the field of NLP for the first time. They applied the attention mechanism to the task of machine translation, enabling translation and alignment simultaneously. Since then, various deep learning networks incorporating attention mechanisms have been gradually extended to various NLP tasks.

2.3 Encoder-Decoder
Encoder-Decoder [16] is widely used in the field of NLP, such as machine translation, text summarization and machine Q&A. The prototype of the architecture was proposed by Cho et al. in 2014. The model encodes the sentence or article into a certain semantic and then decodes the semantic into another expressed sentences or articles. Its form is as follows, <Source, Target>, our goal is to give the input sentence, expecting to generate the target sentence through the Encoder-Decoder framework. The encoder encodes the input sentences and converts the input sentences into an intermediate semantic representation through a nonlinear transformation. The decoder uses the intermediate semantic and which is generated by the previous history to generate .

\[
y_{i} = g(c, y_{1}, y_{2}, y_{3}, \ldots, y_{i-1})
\]

3. Our proposed model

In this section, we describe the network structure of the proposed model. The architecture of the proposed model is shown in Figure 1, the model mainly includes three layers: Embedding layer, Encoder layer and Decoder layer.

3.1 Embedding layer

3.1.1 Word Embedding

Word embedding makes words into dense vectors that the computer can understand. The traditional representation of word embedding, one-hot representation [17] makes each word as a vector, where only one dimension represents number one while the others represent number zero. This representation can’t express the relationship between different words. Also, there is a problem of large storage space. Luckily, the distribution representation [18] can effectively solve these problems. It can express the similarity between different words and contain more information with a little memory.

In this paper, we use NLTK\(^1\) which is a set of Natural Language Processing tools based on python to get each English word. For example, \(W = \{w_1, w_2, w_3, \ldots, w_l\}\), \(w_i\) to \(w_j\) are the words of the sentence. We use a matrix \(E^* \in \mathbb{R}^{N_{\text{word}} \times D_{\text{word}}}\) to represent the representation of the sentence. Where \(N_{\text{word}}\) is the number of the word of the sentence, \(D_{\text{word}}\) is embedding dimensionality.

\(^1\) http://www.nltk.org/
3.1.2 Part-of-speech Embedding

In linguistics, part-of-speech is the basic grammatical attribute of vocabulary. It can classify the categories of the word and also contain semantic information. But most Natural Language Processing tasks, they just focus on the surface semantic of words. Sometimes it is difficult to make full use of the information in the text. Therefore, we can use other features. For example, part-of-speech, for a summarization task, it is useful to the result if we also know the part-of-speech of a word. As we train the model, the network can gradually learn the characteristic of the combination of part-of-speech.

In this paper, we also use NLTK to get part-of-speech of each English word. For example, \( P = \{p_1, p_2, p_3, \ldots, p_L\} \), \( p_i \) to \( p_L \) are the part-of-speech of the words of the sentence. We use matrix \( E_p \in \mathbb{R}^{N_p \times D_p} \) to represent the representation of the part-of-speech of every word. Where \( N_p \) is the number of the part-of-speech of the sentence, \( D_p \) is embedding dimensionality.

![Figure 2. An example of part-of-speech Tagging.](image)

3.2 Encoder layer

3.2.1 Bi-LSTM

Long Short-Term Memory (LSTM) [19] is a special Recurrent Neural Network (RNN) [20], first proposed by Sepp Hochreiter et al. in 1997, which solved the problem of the disappearance of RNN gradient or gradient explosion [21] and can learn long-term dependencies.

![Figure 3. The structure chart of Cell unit.](image)

Compared with the traditional RNN, the LSTM has a controller cell that can determine whether the information is useful. As shown in Figure 3, three gate structures are set in the Cell: an input gate, a forgotten gate and an output gate. They can recode and update information from the memory unit. The state updates status of each door at time \( t \) as follows:

\[
f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \]

(1)

\[
i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \]

(2)

\[
o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \]

(3)

\[
C_t = f_t \times C_{t-1} + i_t \times \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \]

(4)

\[
h_t = o_t \times \tanh(C_t) \]

(5)

Bidirectional Long Short-Term Memory (Bi-LSTM) [22] is an improvement of LSTM. Due to the serialization of information in LSTM, there is a sequential order in information processing, and it is
impossible to access the contextual information. Bi-LSTM uses two LSTM networks to train together, one training sequence starts from the beginning and another starts from the back. Finally, both training sequences are connected to the same output layer. It will synthesize the past and future information of each point. Figure 4 is the network structure of the Bi-LSTM.

![Network Structure of Bi-LSTM](image)

**Figure 4.** The network structure of Bi-LSTM.

In this model, two bidirectional LSTM layers are used to train word features and part-of-speech features to obtain deep semantic expression features respectively.

After we get the deep semantic expression of words the part-of-speech, we take the following method to compromise the deep semantic of the word and part-of-speech.

\[
H_{wp} = H_w \odot H_p
\]

In the formula (6), we multiply the deep semantic expression of words and part-of-speech to compromise two features.

### 3.2.2 Attention mechanism

In an automatic text summarization task, there must be some words which are more important to the text summarization. The attention mechanism can distinguish the importance of the target unit and other units. In this model, we use the attention mechanism to focus on the valuable parts of the outputs of the Bi-LSTM layer to allocate more weights.

Assuming \(h_1, h_2, ..., h_n\) are the outputs of the Bi-LSTM layer at different time. We can get \(c_i\) like this:

\[
c_i = \sum_{j=1}^{n} a_{ij} h_j
\]

Where \(a_{ij}\) represents the weight of allocation, it assigns different weights when we output the result of number \(i\) according to the input \(h_j\). We calculate \(a_{ij}\) as follows:

\[
Y_j = g(s_{i-1}, h_j)
\]

\[
a_{ij} = \text{softmax}(Y_j) = \exp(y_{ij}) / \sum_{j=1}^{n} \exp(y_{ij})
\]

Where \(s_{i-1}\) is the output of hidden layer of decoder layer when we output the result of number \(i-1\). Then we adopt \(\text{softmax}\) act on \(Y_j\) to get the probability \(a_{ij}\) called attention matrix, which represents the importance of the different hidden state.

Attention mechanism, according to the \(a_{ij}\), which can treat the outputs of the hidden layer differently and pay more attention to the larger weight of the hidden layer.

### 3.3 Decoder layer

In this layer, we use beam search \([23]\) of size 10 to generate abstracts. Also, according to the length of the text, we use three buckets for sampling. In our model, we take buckets = [(30, 10), (100, 20), (200,
30). For example, (30, 10) in the array buckets, ‘30’ represents the max size of the text, while ‘10’ represents the max size of the summary.

4. Experiment

4.1 Corpus
In the paper, we use the Gigaword\(^2\) dataset as training set and verification set. We use about 3.8 million samples for training and 189 thousand samples for verifying.

For the test set, we use Gigaword and DUC-2004\(^3\) dataset for evaluation. We follow by [10] to use the same randomly held-out test set of 2000 sentence-summary pairs. The DUC-2004 dataset has 50 topics, each topic contains 10 documents.

4.2 Evaluation Metrics
In the paper, we adopt the current mainstream automatic text summarization evaluation method called ROUGE [24] to compare the model results with the reference summaries, including ROUGE-N (N=1, 2), ROUGE-L.

Rouge-N is defined as follows:

\[
\text{Rouge-N} = \frac{\sum \sum \text{Count}_{\text{match}}(\text{gram}_n)}{\sum \sum \text{Count}(\text{gram}_n)}
\] (10)

Where \(RS\) is the set of Reference Summaries which represents the standard summary, \(N\) is the length of n-gram and \(\text{Count}_{\text{match}}(\text{gram}_n)\) is the number of n-grams that appear at the same time in the candidate summary and reference summary. \(\text{Count}(\text{gram}_n)\) represents the number of the n-gram in the reference summary.

Rouge-L is defined as follows:

\[
\text{Rouge-L} = \frac{\text{LCS}(X,Y)}{m}
\] (11)

Where \(X\) is the reference summary, \(Y\) is the candidate summary and \(m\) is the length of reference summary. LCS (longest common subsequence): Given two sequences \(X\) and \(Y\), the sequence that maximizes the length of the common subsequence is the longest common subsequence of \(X\) and \(Y\).

4.3 Experimental configuration
We use 300 hidden units for both encoders and decoders. All embedding, including word embedding and part-of-speech embedding have dimensionality 200. We use the learning rate of 0.0001 with Adam [25]. In the experiments, we set 200 samples per batch with a total of 300,000 training steps.

4.4 Baselines
We compare WPABS model with the following baselines:

- **ABS** (Attention-Based Summarization) [10]: A data-driven understanding abstract model, which uses seq2seq structure. It takes convolutional neural network as the encoder and feedforward neural network as the decoder. Also, it integrates the attention mechanism into the encoder.
- **ABS+** (Attention-Based Summarization+) [10]: Based on the ABS, which adds the feature function.
- **RAS** (Recurrent Attention Summarizer) [26]: A conditional recurrent neural network which generates a summary of an input sentence.
- **SEASS** (Selective Encoding for Abstractive Sentence Summarization) [27]: A selected encoding model to extend the sequence-to-sequence framework for abstractive sentence summarization.
- **ConvS2S** (Convolutional Sequence to Sequence Learning) [12]: Its encoder and decoder are based on convolutional neural work which achieves the performance of state-of-the-art.

\(^2\) https://catalog.ldc.upenn.edu/LDC2003T05
\(^3\) https://duc.nist.gov/duc2004/tasks.html
Reinforced-Topic-ConvS2S [13]: A topic-aware topic ConvS2S model with reinforcement learning.

WABS (Word Attention-Based Summarization): We only use the word embedding as an input feature and Bi-LSTM as the encoder. At the same time, we also use the attention mechanism.

4.5 Results and discussion

Table 1. The experimental results on Gigaword.

| Methods            | RG-1(F) | RG-2(F) | RG-L(F) |
|--------------------|---------|---------|---------|
| ABS                | 29.55   | 11.32   | 26.42   |
| ABS+               | 29.76   | 11.88   | 26.96   |
| RAS                | 33.78   | 15.97   | 31.15   |
| SEASS              | 36.15   | 17.54   | 33.63   |
| ConvS2S            | 35.88   | 17.48   | 33.29   |
| Reinforced-Topic-ConvS2S | 36.92   | 18.29   | 34.58   |
| WPABS              | 37.67   | 17.83   | 35.92   |

In table 1, compared with all baselines on Gigaword dataset, our WPABS outperforms better and achieves the 37.67, 17.83, 35.92 for ROUGE 1,2 and L F1-score. Compared to the best model Reinforced-Topic-ConvS2S on ROUGE-2(F), our result is similar to its. But our model outperforms better on ROUGE-1(F) and ROUGE-L(F).

Table 2. The experimental results on DUC-2004.

| Methods            | RG-1(R) | RG-2(R) | RG-L(R) |
|--------------------|---------|---------|---------|
| ABS                | 26.55   | 7.06    | 22.05   |
| ABS+               | 28.18   | 8.49    | 23.81   |
| RAS                | 28.97   | 8.26    | 24.06   |
| SEASS              | 29.12   | 9.56    | 25.51   |
| ConvS2S            | 30.44   | 10.84   | 26.90   |
| Reinforced-Topic-ConvS2S | 31.15   | 10.85   | 27.68   |
| WPABS              | 30.85   | 11.96   | 27.73   |

In table 2, from the comparative results on DUC-2004 between WPABS with all the baselines, our model WPABS also makes good results and achieves the 30.85, 11.96, 27.73 for ROUGE 1,2 and L recall. Also, compared to the best model Reinforced-Topic-ConvS2S, our result is similar to its on ROUGE-1(R). But our model outperforms better on ROUGE-2(R) and ROUGE-L(R).

What’s more, as we can see from the results of the experiments on the Gigaword and DUC-2004, the results of the Gigaword is better because the model is training on the Gigaword dataset.
Table 3. The experimental results between WABS and WPABS on Gigaword.

| Methods | RG-1(F) | RG-2(F) | RG-L(F) |
|---------|---------|---------|---------|
| WABS    | 36.80   | 16.56   | 34.20   |
| WPABS   | 37.67   | 17.83   | 35.92   |

Table 4. The experimental results between WABS and WPABS on DUC-2004.

| Methods | RG-1(R) | RG-2(R) | RG-L(R) |
|---------|---------|---------|---------|
| WABS    | 29.73   | 10.42   | 26.78   |
| WPABS   | 30.85   | 11.96   | 27.73   |

Further, in table 3 and table 4, we contrast the results between the WABS and WPABS on Gigaword and DUC-2004. Compared with WABS which doesn’t merge the part-of-speech, our model WPABS has been improved a little. It is proved that the proposed idea to merge the part-of-speech is useful. Especially, we notice that ROUGE-2 of WPABS improves much than others.

5.Conclusions
In this work, we explore the advantage of combining the word and part-of-speech to improve the performance of automatic text summarization. In linguistics, part-of-speech is the basic grammatical attribute of vocabulary, which also contains some useful features. By taking advantage of this, we present a novel WPABS model that combines the word and part-of-speech. Our experiments conducted on the Gigaword and DUC-2004 datasets show that our model performs better than most baselines, so it is useful to combine word and part-of-speech.

Acknowledgement
This research was supported by the Chinese Education Department-China Mobile Research Foundation (Grant Number: MCM20170309).

References
[1] H.P. Luhn. The automatic creation of literature abstracts. IBM Corp(1958).
[2] Erkan, Radev, R. Dragomir, et al. LexRank: graph-based lexical centrality as salience in text summarization. JQJTC, 22(2004).
[3] H.P. Edmundson. New Methods in Automatic Extracting. JACM, 16(2):264-285(1969).
[4] W. Ling, T. Luis, L. Marujo, et al. Finding function in form: compositional character models for open vocabulary word representation. CS, 1899-1907(2015).
[5] X. Chen, X. Qiu, C. Zhu, et al. Long short-term memory neural networks for Chinese word segmentation. EMNLP, 1197-1206(2015).
[6] D. Wang, E. NYBERG. A long short-term memory model for answer sentence selection in question answering. ACL, 707-712(2015).
[7] G. Lample, M. Ballesteros, S. Subramanian, et al. Neural Architectures for Named Entity Recognition. NAACL, 260-270(2016).
[8] I. Sutskever, O. Vinyals, Q.V. Le, et al. Sequence to Sequence Learning with Neural Networks. 4:3104-3112(2014).
[9] K. Cho, V.B. Merrienboer, C. Gulcehre, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. CS(2014).
[10] A.M Rush, S. Chopra, J. Weston, et al. A Neural Attention Model for Abstractive Sentence Summarization. CS(2015).
[11] B.T. Hu, Q. Chen, F. Zhu, et al. LCSTS: A Large Scale Chinese Short Text Summarization Dataset. CS, 2667-2671(2015).
[12] J. Gehring, M. Aul, D. Grangier, et al. Convolutional Sequence to Sequence Learning. (2017).
[13] L. Wang, J. Yao, Y. Tao, et al. A Reinforced Topic-Aware Convolutional Sequence-to-Sequence Model for Abstractive Text Summarization. (2018).
[14] J.F Kolen, S.C. Kremer. Gradient Flow in Recurrent Nets: The Difficulty of Learning Long Term Dependencies. Wiley-IEEE Press(2009).
[15] V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. 3, 2204-2212(2014).
[16] D. Bahdana, K. Cho, Y. Bengio, et al. Neural Machine Translation by Jointly Learning to Align and Translate. CS(2014).
[17] K. Cho, B.V Merrienboer, C. Culcehre, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. CS(2014).
[18] J. Turian, L. Ratino, Y. Bengio. Word representations: a simple and general method for semi-supervised learning. ACL, 384-394(2010).
[19] S. Hochreiter, J. Schmidhuber. Long Short-Term Memory. NC, 9(8):1735-1780(1997).
[20] J.L. Elman. Finding structure in time. CS, 14:179-211(1990).
[21] G.E. Hinton, J.L Mcclelland, D.E Rumelhart. Parallel distributed processing: explorations in the microstructure of cognition. MIT Press Cambridge, 77-109(1986).
[22] M. Schuster, K.K Paliwal. Bidirectional recurrent neural networks. IEEE Press(1997).
[23] Reddy, D. Raj. Speech Understanding Systems: A Summary of Results of the Five-Year Research Effort. DCS(1977).
[24] C.Y Lin. Looking for a Good Metrics: ROUGE and its Evaluation. NTCIR Workshops(2004).
[25] D.P. Kingma, J.L. Ba. Adam: A Method for Stochastic Optimization. CS(2014).
[26] S. Chopra, M. Auli, A.M. Rush, et al. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. NAACL, 93-98(2016).
[27] Q. Zhou, N. Yang, F. Wei, et al. Selective Encoding for Abstractive Sentence Summarization. 1095-1104(2017).