Summary++: Summarizing Chinese News Articles with Attention

Juan Zhao[1], Tong Lee Chung[2], Bin Xu[2(✉)], and Minghu Jiang[1]

[1] Lab of Computational Linguistics, School of Humanities, Tsinghua University, Beijing, China
solomagix@gmail.com, jiang.mh@tsinghua.edu.cn
[2] Computer Science Department, Tsinghua University, Beijing, China
tongleechung86@gmail.com, xubin@tsinghua.edu.cn

Abstract. We present Summary++, the model that competed in NLPCC2018's Summary task. In this paper, we describe in detail of the task, our model, the results and other aspects during our experiments. The task is News article summarization in Chinese, where one sentence is generated per article. We use a neural encoder decoder attention model with pointer generator network, and modify it to focus on words attented to rather than words predicted. Our model archive second place in the task with a score of 0.285. The highlights of our model is that it run at character level, no extra features (e.g. part of speech, dependency structure) were used and very little preprocessing were done.

Keywords: Text summarization · Sequence-to-sequence · Pointer Coverage

1 Introduction

Text summarization is the task of producing a short piece of text from a long one while preserving main information [1]. Summarization is one of the eight in NLPCC2018's evaluation task. As one of the more traditional task in NLP, it is still attracting a lot of attention and with advancements in *deep learning techniques*, there are more opportunities for improvement. As information grows rapidly, the time needed for a person to consume all these data is insufficient, summarization is helpful by providing a short text helping reader decide if they want to read the whole article. NLPCC's summarization task focuses on Chinese News articles, where research and support still lacks. The data is provided by Toutiao.com which consist of News articles. Train and evaluation datasets are provided to train the model and a test set is used to compare different models.

We participate in the task with our model Summary++, which is based of the Pointer Generator Network [2] with modifications to obtain our results. In the paper will focus on pre-processing of data, the model + modification and hyperparameter tuning. We do not use any feature extraction tools and human engineered features during pre-processing. We base our solution on character

M. Zhang et al. (Eds.): NLPCC 2018, LNAI 11109, pp. 27–37, 2018.
https://doi.org/10.1007/978-3-319-99501-4_3

level so that we do not need to perform word segmentation. Our model is end-to-end to keep reduce human involvement during training and testing. Experiment results and other details will be followed in the paper, including improvement at each stage and training time. Models are compared with Character-based ROUGE-F metric [3].

The main contribution of this paper can be summarized as follows:

1. We present our Summary++ which obtained the second highest score of 0.285 in NLPCC2018's summary task.
2. We present a end-to-end solution for Chinese News article summarization.
3. Our model does not require word segmentation or any feature engineering.

1.1 The Summarization Task for Chinese News Article

The task description and data, provided by Toutiao.com, is a single sentence summarization task for News articles. The summary is usually a general description removing all details and comes in one sentence. Figure 1 shows one training point which contains one News article and its corresponding summary. The task can be regarded as an sequence to sequence problem with the article as input and summary as output. The standard model in deep learning for such task is encoder decoder model. Challenges when using such model include accuracy of word generated, out-of-vocabulary problem and repetition. Pointer generator network [4], which we base our solution on, is a outstanding model for such problems which incorporates both copying and generation mechanisms.

```
summarization: 网曝河南商丘肯德基砍死人，警方称系两伙人冲突引发斗殴，双方当事人
均为本地人。
article: 来源：
人民网(北京)微博截图
人民网北京6月17日电
今日，网络论坛流传“商丘肯德基砍死人”的消息引发网民对其是否涉“暴恐”和“邪教”的猜测
议论，中午，商丘市公安局通报此事为当事双方在KTV唱歌消费时引发冲突，致一死一伤。
警方通报表示，6月17日凌晨，曹某、张某在商丘市乐购KTV量贩唱歌消费时，与在此消费的
另一方当事人发生冲突，2时50分左右，双方在该量贩一楼电梯出口处发生打斗，曹某、张
某被砍伤。曹某受伤后跑至该量贩附近的肯德基大厅内倒地，后经抢救无效死亡。现张某正
在医院接受治疗，暂无生命危险。目前，公安机关正在进一步调查此案。警方表示，经调查
，已明确双方当事人均为商丘市人。望广大网民不要信谣传谣，对传播谣言制造恐慌者公安
机关将依法予以严厉打击。(原标题：网曝"商丘肯德基砍死人"
警方:冲突致双方斗殴)
```

Fig. 1. Example of training data point. A article and its summary is provided. In the figure, the News article is about a murder incident that involves one man hacking another man to death at KFC. The article goes into deep detail of the incident, including time, place and aftermath. The summary however is very general and goes as follows, “Internet reports Shangqiu Henan hacking incident at KFC, police says two mans conflict escalated into fight, two man are locals.”

When revising the data, we find some unusual datapoints, where the content consist of only html tags or punctuation. We use regular expression to remove

tags and continuous punctuations for both training and testing set. Another dataset without summary was also provided but was discarded because such data was not needed in our model.

The total number of datapoints is 50000 for training, 2000 for evaluation and 2000 for testing. A only Chinese vocabulary list was provided with 2987 characters but we generated our own which includes digits, punctuation and English words. The total number of unique tokens was 213789 and ones with top 10000 frequency were used.

2 Summary++

In this section, we present our model in detail. We briefly describe the model we use and focus on the modifications we did to improve performance.

2.1 Our Model

Our model is based of Pointer Generator Summarization Network, which has a network that tell the decoder to use a generated word or a word that was attended to. In our solution, we try our model the same way, but when we perform summarization on the test set, we remove the generation process, because we find that the pointer network easily overfit the data and provided false facts in the test results. Figure 2 shows decoding process during training.

The model is encoder decoder model with attention mechanism, pointer network and coverage mechanism [5]. The encoder is a Bi-directional LSTM (BiLSTM) which takes in a sequence of characters and outputs two final states and one hidden states for each input character. BiLSTM is used in many state-of-the-art models and shows promising potentials.

During decoding, A LSTM cell takes in the previous hidden states and context vector as input, and outputs a current hidden state (also known as the decoder state). Attention is a mechanism which shows the decoder which part of the sequence to focus on while decoding. We can think of it as giving a weight to each encoder hidden states. The attention weight is calculated as follows: 1. for each token in the sequence, concatenate its corresponding encoder hidden state with decoder hidden state to form a vector; 2. run the vector through a linear transformation followed by a $tanh$; 3. run the output from the previous step through another linear transformation followed by a $softmax$. We are able to get a probability distribution for each word after the three steps.

The context vector is a state after considering all encoder hidden states and attention. We can see it as a vector that is used to predict the next word. It is calculated by multiplying the encoder hidden state of each token with it attention weight, then adding together all these weighted state. The context vector is project through a linear transformation to the size of our vocabulary.

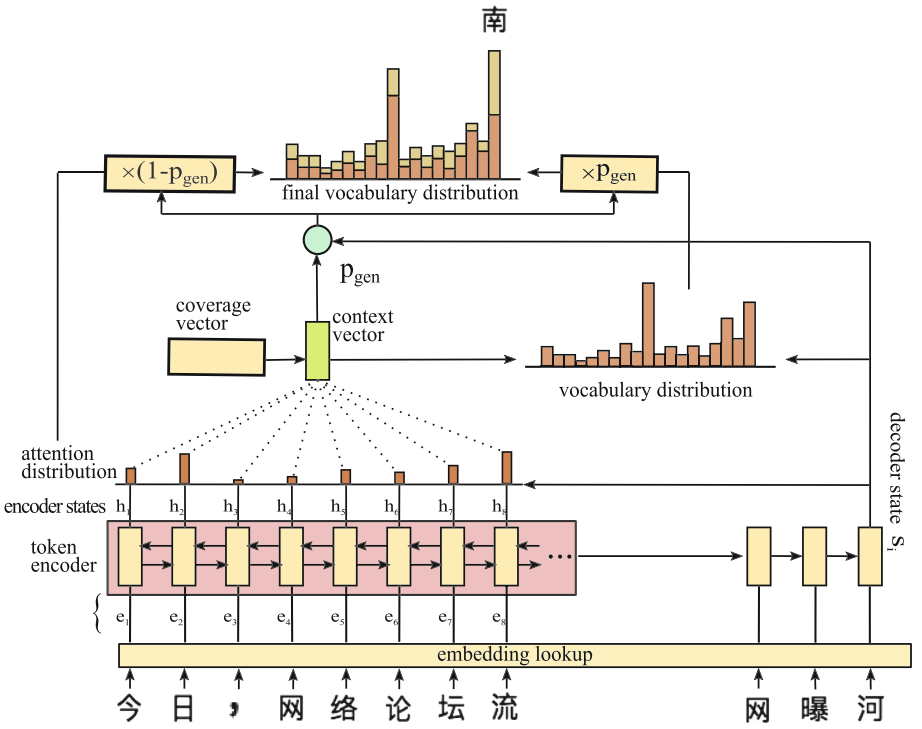

Fig. 2. Pointer Generator Network Model + Coverage that is used in our Summary++ system. During each decoding step, attention, coverage vector and generation probability is calculated to predict the next word.

Finally, a *softmax* is used to obtain the distribution of the next token. The following equations show calculation of attention and context vector:

$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + b_{attn}) \tag{1}$$

$$a^t = \text{softmax}(e^t) \tag{2}$$

$$h_t^* = \sum_i a_i^t h_i \tag{3}$$

where h_i is the hidden encoder state and s_t is the hidden decoder state. v^T, W_h, W_s and b_{attn} are all trainable parameters. h_t^* is the context vector.

The coverage vector is a summation of attention during the decoding process. For each token in the input sequence, its initial coverage weight is 0, that means the token is not attended to. As the output sequence get decoded, coverage increases for each token as it gets assigned attention weight. As a token get more attention, it get more difficult to have higher attention scores. Such mechanism

is used for preventing repetition in the output sequence. The coverage vector is included in the calculation of attention. Coverage is the new attention is calculated as following:

$$c^t = \sum_{t'=0}^{t-1} a^{t'} \tag{4}$$

$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + W_c c_i^t + b_{attn}) \tag{5}$$

c^t is initialized to 0. The attention and context calculation is the same as Eqs. 2 and 3.

The pointer network outputs a scalar the tells the model to look at the predicted token to look at a token that was attened to most. It can be seen as a soft switch between two choices. The network takes in the context vector and the decoder states and outputs a scalar for each token in the vocabulary. The final vocabulary distribution is the weighted sum of predicted distribution and attention of the input token. The switching scalar also known as generation probability is calculated as:

$$p_{gen} = \sigma(w_{h^*} h^* + w_s s_t + w_x x_t + w_d d + b_{ptr}) \tag{6}$$

where w_{h^*}, w_s, w_x, and w_d are trainable parameters. p_{gen} is our generation probability. Predicted distribution and final distribution is calculated as:

$$P_{vocab} = \text{softmax}_{vocab}(V'(V[s_t, h_t^*] + b) + b') \tag{7}$$

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \tag{8}$$

V, V' and b' are trainable parameters.

The loss we use is negative log likelihood of each token in the output sequence. We hope that by training, the probability of a suitable next token will have the highest. Coverage does not converge unless a coverage loss is included. This is the minimum between attention and coverage value, which slows down the gradient descent process. Loss is show as below:

$$loss_t = -\log P(w_t^*) \tag{9}$$

$$loss = \frac{1}{T} \sum_{t=0}^{T} loss_t \tag{10}$$

and loss with coverage:

$$loss_t = -\log P(w_t^*) + \lambda \sum_i \min(a_i^t, c_i^t) \tag{11}$$

2.2 Character Embedding

For Chinese text we choose to operate at character level, thus leaving out the need for segmentation. Each character is represented by a vector which is fed

into the model. We find that segmentation introduces a large number of words (token), such token is difficult to train due to the fact that data is not sufficient. A lot of words appear on a few times even though they are everyday words. Correlation of words become too sparse. Another reason is that we hope that our model is a end to end model and not a pipeline type of model. Error in pipeline can propagate along and a better solution would be minimize all errors within one model. We also do not introduce any human engineered feature of any kind (POS, DEP, etc). We believe that such feature are not a hundred percent correct, thus causing the error to be enlarged in the model.

In the dataset, we do find some pieces of data that requires pre-processing. Due to the fact that our model takes in the first 400 characters as input for an article, some articles contains only tags and punctuation in the beginning, thus causing the model to incorrectly identify crucial tokens in the model. We use a regular expression to remove some text that does not look like news content.

2.3 Attention only Generation

When evaluating our model, we find that some summarizes contain some incorrect information. For example, we find that given a article about CBA players as

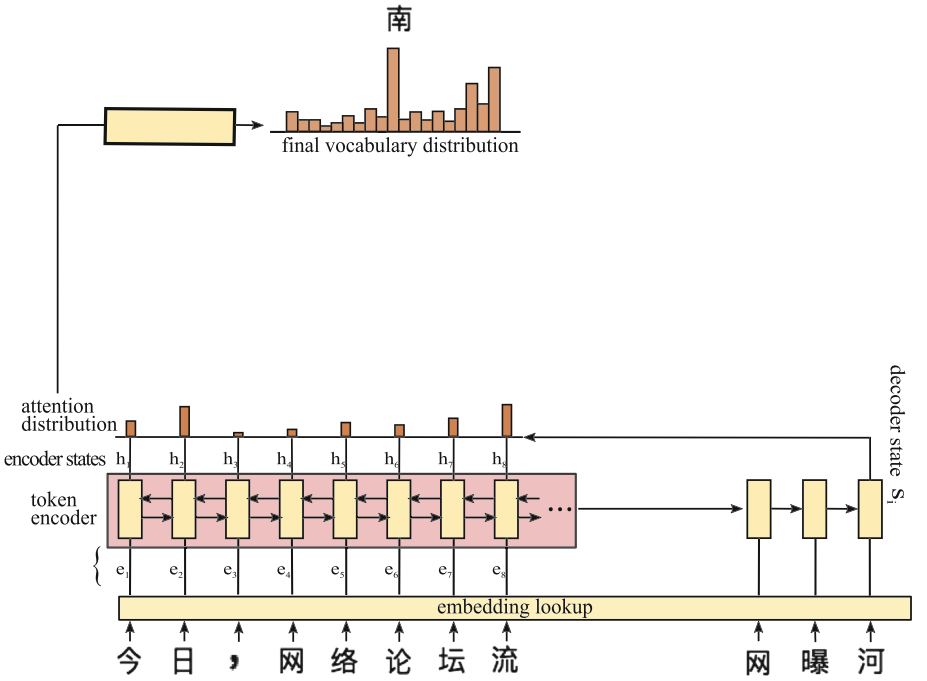

Fig. 3. Modified model, the whole vocabulary prediction part has been removed, the whole generation process relies only on attention weights. The parameters are trained using the full model but only these parts are used for summary generation.

input, the output summary begins with *NBA*. After carefully analyzing the matter, we find that the problem is that predicted value is over confident. We find that by removing the predicted vocabulary distribution, we can achieve optimal results. Figure 3 is the model when in testing phrase. We directly use the word that has the highest attention weight. This is similar to using neural network as a copying mechanism, at each time step, we copy one word from the original article. Different from directly copying, attention distribution was trained on the full model hoping that correct words can be predicted.

3 Experiments

In this section, we describe in detail our experiment. We first discuss hyperparameter settings and results.

3.1 Hyperparameters and Training

Setting hyperparameters is one of the bigger challenges in our experiment. We report ROUGE score on the evaluation set to compare different settings and choose accordingly. We also compare the difference we using the original model and our simplified model.

On Training Steps. We find that the model easily overfits, which means the more training steps the more likely the model will overfit. We report F score on ROUGE-1, ROUGE-2 and ROUGE-L after some number of training to find the optimal step size. Table 2 shows the relation between performance and training step size (Table 1).

Table 1. ROUGE result on training step, we find that after about 100,000 training step, performance seems to stagnate.

num training steps	ROUGE-1	ROUGE-2	ROUGE-L
11353	0.43	0.27	0.37
25791	0.44	0.28	0.38
109937	0.46	0.29	0.40
233769	0.46	0.30	0.40
392774	0.46	0.30	0.40
570743	0.46	0.29	0.40

ROUGE result on evaluation set does not tell us everything about the model. High training step trigger repetition even when coverage is added. In order to have a high ROUGE score and more readable summary, we choose the training step to be around 110,000.

3.2 On Encoder Decoder Max Size Min Size

We find another important factor in competing was setting encoder input size. Due to hard-ware limitations, we were only able to increase encoder input size to 400, that is only the first 400 tokens were taken. We find that smaller size decreases the performance of the model. While sizes too large result in out-of-memory error.

We also set the minimum decoder size to 15. After some observation, we estimate the minimum length of summary is about 15 characters. We also try other sizes such as 20 and 35, we find that performance is slightly worse than 15, so we choose 15 as minimum decoder size. We set the maximum decoder size to 35. Increasing this size does not have any improvement as well.

3.3 On Attention only Summary Generation

We finally test the results of using pointer generator probability to switch between generating the next word and using the attended character. We test the vanilla approach and also the two alternate, where one uses only the attended character and the other uses only the generated word. We operate our experiment on the final test data and report the official ROUGE score

Table 2. ROUGE score when using different generation methods, Vanilla uses both generated character and attention with a soft switch. Attention and Generation uses either only the attended sum or predicted vocabulary.

num training steps	ROUGE score
Vanilla	0.243
Generation only	0.270
Attention only	0.285

We also evaluate some example and find that even with the switch method, we find some obvious mistake in the summary with generated summaries. Figure 4 shows some examples when using a switch between generation and attending. When using attention guided summary, such problem do not appear.

NBA：幼儿园呼得木林大街8#街坊运输公司大院停水时间，停水时间因呼得木林大街8#街坊运输公司大院。
NBA：中国男排3-2胜卡塔尔队获得第四名，卡塔尔获第四名，卡塔尔接应袁志，自由人童嘉骅。
NBA：谌龙领衔国羽逆转印尼晋级决赛，李雪芮、谌龙和唐渊渟/于洋之后各贡献胜利

Fig. 4. Examples of incorrect summary, the model is over confident starting with "NBA" is a correct way to begin a summary.

3.4 Final Results Compared in the Evaluation Task

We finally compare results with other team in the evaluation task. Our team made second place on the score board. All methods are not disclose at the time of writing, so this is reference to where our model stand in the task. Table 3 shows the top five on the leader board. Difference is rather small.

Table 3. Evaluation result on the final score board. Our mode Summary++ made second place.

num training steps	ROUGE score
WILWAL	0.2938
Summary++	0.285
CCNU_NLP	0.282
freefolk	0.281
kakami	0.278

4 Related Works

Summarization has been around for many decades [1,6], this paper focuses on multi-sentence summarization where the task is to generate summaries with multiple sentences. The task has attracted attention when a large multi-sentence summary corpus was introduced [7].

Neural Extractive Summarization. The extractive approach is based on a hypothesis that the main idea of a document can be summarized in a few phrases or words in the document. Then the task of the summarization turns to find the most important words in the document. The neural extractive approaches [8–10] are mainly based on the CNN model and some of its deformations. The greatest problem of this kind of approach is that the generated summarization may be incoherent and inconsistent.

Neural Abstractive Summarization. The abstractive approach needs to understand the meaning of the document, and then briefly summarize it by a highly readable human language. For this target, the RNN/LSTM models and some of their deformations are adopted to complete the neural abstractive task [11,12]. Recently, some researchers have used the latest neural networks model to summarize, such as the sequence-to-sequence model and attention model.

Sequence-to-Sequence Models. Most of current state-of-the-art models are based of sequence-to-sequence models which have gained many successes in machine translation [13,14]. Attention [15], pointer network [4], coverage [5] and controllable summarization [16,17] are some techniques adapted to the task of summarization.

5 Conclusion

This paper present our model in the NLPCC2018 summarization task. We modify the pointer generator network model to achieve SOTA results on the test set. We show that the model can sometimes be over confident with its prediction and we simplify the model to only using attended tokens. We also show that character level summary in Chinese language is not only possible but also practical. Our model requires very little pre-processing and no human-engineered feature. We believe that there is more potentials in the model.

Acknowledgement. This work was supported by the National Key Research and Development Program of China 2017YFB1401903, National Social Science Major Fund (14ZDB154; 15ZDB017) of China, and the National Natural Science Key Foundation of China (61433015).

References

1. Nenkova, A., McKeown, K.: Automatic summarization. Found. Trends Inf. Retr. **5**(2–3), 103–233 (2011)
2. 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. Association for Computational Linguistics (2017)
3. Lin, C.-Y.: Rouge: a package for automatic evaluation of summaries. In: Text Summarization Branches Out: Proceedings of the ACL-04 Workshop
4. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 2692–2700. Curran Associates Inc. (2015)
5. Mi, H., Sankaran, B., Wang, Z., Ittycheriah, A.: Coverage embedding models for neural machine translation. In: EMNLP (2016)
6. Saggion, H., Poibeau, T.: Automatic text summarization: past, present and future. In: Poibeau, T., Saggion, H., Piskorski, J., Yangarber, R. (eds.) Multi-source. Theory and Applications of Natural Language Processing, pp. 3–21. Multilingual Information Extraction and Summarization. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-28569-1_1
7. Nallapati, R., Zhou, B., dos Santos, C.N., Gülçehre, Ç., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond. In: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, 11–12 August 2016 (2016)
8. Cheng, J., Lapata, M.: Neural summarization by extracting sentences and words. In: Meeting of the Association for Computational Linguistics, pp. 484–494 (2016)
9. Nallapati, R., Zhou, B., Ma, M.: Classify or select: Neural architectures for extractive document summarization. CoRR, abs/1611.04244 (2016)
10. Cao, Z., Li, W., Li, S., Wei, F., Li, Y.: Attsum: joint learning of focusing and summarization with neural attention. In: International Conference on Computational Linguistics, pp. 547–556 (2016)
11. Lopyrev, K.: Generating news headlines with recurrent neural networks. arXiv:1512.01712 Computation and Language (2015)
12. Ayana, S.S., Zhao, Y., Liu, Z., Sun, M.: Neural headline generation with sentence-wise optimization. arXiv:1604.01904v2 Computation and Language (2016)

13. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: International Conference on Learning Representations (ICLR) (2015)
14. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, vol. 2, NIPS 2014 (2014)
15. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, Q.V., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 6000–6010. Curran Associates Inc. (2017)
16. Fan, A., Grangier, D., Auli, M.: Controllable abstractive summarization. CoRR, abs/1711.05217 (2017)
17. Kikuchi, Y., Neubig, G., Sasano, R., Takamura, Y., Okumura, M.: Controlling output length in neural encoder-decoders. CoRR, abs/1609.09552 (2016)