Abstractive Text Summarization using LSTMs with Rich Features

Viet Nguyen Quoc[1], Huong Le Thanh[1], Tuan Luu Minh[1,2]

[1] Hanoi University of Science and Technology, Hanoi, Vietnam
[2] National Economics University, Hanoi, Vietnam
nguyenquocviet1306@gmail.com ,huonglt@soict.hust.edu.vn,
tuanlm@neu.edu.vn

Abstract. Abstractive text summarization using sequence-to-sequence networks have been successful for short text. However, these models have shown their limitations in summarizing long text as they forget sentences in long distance. We propose an abstractive summarization model using rich features to overcome this weakness. The proposed system has been tested with two datasets: an English dataset (CNN/Daily Mail) and a Vietnamese dataset (Baomoi). Experimental results show that our model significantly outperforms recently proposed models on both datasets.

Keywords: Abstractive Text Summarization, Sequence to Sequence, LSTM, Rich Features.

1 Introduction

Abstractive text summarization is the task of generating a condensed text that captures the main content of the original one. It is not done by selecting the most important sentences from the input document as in extractive summarization. Instead, it rewrites and compresses the original text, similar to how human does when summarizing a document.

Most recent research on abstractive summarization is based on a sequence to sequence (seq2seq) network, as it can generate new text from the original one. The input of the original version of seq2seq is often short since the character of seq2seq is "short term memory". That means, it often processes the recent sentences, but forgets sentences in a longer distance. As a result, most summarization models using seq2seq networks tend to ignore the first part of the long input text. This is a challenge for summarization systems whose processing target is news articles because, in such type of text, the important information often situates at the beginning of the text.

In this paper, we propose a model for abstractive summarization that can take into account the whole input article and generate a multi-sentence summary. We propose a new seq2seq network by using sentence position and term frequency as features. Our system is evaluated with two datasets, an English dataset (CNN/DailyMail) and a

Vietnamese dataset (BaoMoi). Experimental results showed that our system provides better performance compared to existing ones.

The remainder of this paper is organized as follows: Section 2 introduces related works on abstractive summarization, using seq2seq networks. Section 3 describes in detail the baseline model and our proposal to improve this model. Section 4 discusses all issues involving our experiments and evaluation. Finally, Section 5 concludes the paper and gives some insight into future work.

2 Related work

Recent work on abstractive text summarization often uses seq2seq networks, since they are quite suitable and promising in solving this task. Rush et al. [1] applied a Convolutional Neural Network (CNN) seq2seq model with an attention-based encoder for the abstractive summarization task. They used CNN for the encoder, and a context-sensitive attentional feed-forward neural network to generate the summary. The abstractive summarization dataset DUC2004 was used to evaluate their system. The weakness of Rush et al.'s system is that it can only generate headlines (approximately 75 bytes) with many grammatical errors.

Narayan et al. [2] used Topic Sensitive Embeddings and Multi-hop Attention to fix long-range dependencies problems and achieving good results for single-document summarization on a dataset of BBC articles accompanying with single sentence summaries whose length is limited to 90 tokens.

Nallapati et al. [3] used attentional encoder-decoder Recurrent Neural Network (RNN) to create a system that generates longer summaries by capturing the hierarchical document structure with hierarchical attention. In their experiments, they created a large training dataset named CNN/DailyMail, consisting of original texts and their multi-sentence summaries. By testing the system with the new dataset, they establish performance benchmarks for further research.

There are two main weaknesses in early works on abstractive text summarization. First, a word can be generated several times in the output (e.g., “I’m Vietnamese Vietnamese Vietnamese”). Second, numbers and private names, which are considered as out-of-vocabulary (OOV) words by the system, cannot be recovered correctly in the output. For example, if the input is “Peter go to school”, the summary will be “<UNK> go to school” or “John goes to school”. Nallapati et al. [3]’s system encountered the first problem when summarizing long texts. As for the second problem, Nallapati et al. [3] used modeling rare/unseen words that used switching generator-pointer to overcome.

Gu et al. [4] proposed a CopyNet network based on pointer-networks [5] to deal with OOV words. This network is modified by See et al. [6] to create a more powerful system that can solve the OOV problem and word repeat errors by using a pointer-generator network [4] and a distraction mechanism [7]. The limitation of their system is that it does not consider the full article as the input. Instead, it cuts off the last part of the article to guarantee that the important information of the input article will be included in the summary. This process reduces the generality of the system.

The next section will introduce our proposed approach to deal with the problems mentioned above.

3 Proposed model

In this section, we first present Nallapati et al. [3]'s model, which is used as the baseline in our research. Then we briefly introduce two mechanisms used in [6] to solve the weaknesses in Nallapati et al. [3]'s system. Finally, we propose our method to resolve the weakness in See et al. [6]'s system to enhance its generality and its accuracy.

3.1 Baseline model

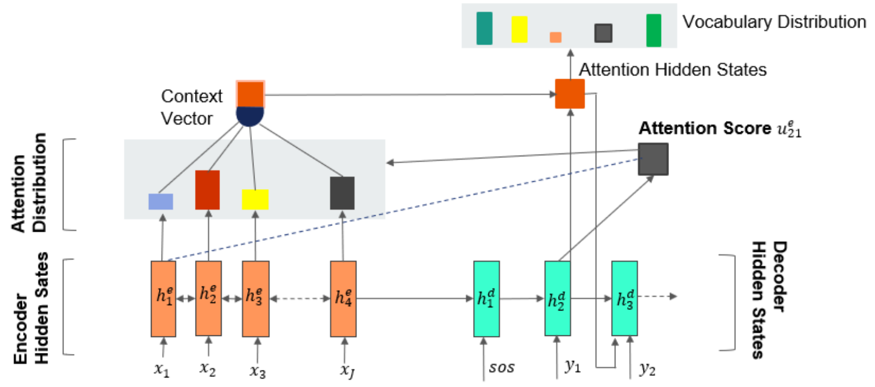

Figure 1. General model. SOS represents the start of a sequence, respectively.

Nallapati et al. [3]'s model is a seq2seq with attention architecture which uses a bidirectional Long Short Term Memory (LSTM) for the encoder and a unidirectional LSTM for the decoder. As shown in **Figure 1**, the encoder reads a sequence of words $x = (x_1, x_2, \dots, x_J)$ from the article, and transforms it to encoder hidden states $h^e = (h^e_1, h^e_2, \dots, h^e_J)$. We denote target summary $y = (y_1, y_2, \dots, y_T)$, on each step t, the decoder receives as input the previous word y_{t-1} of the target summary and uses it to update the decoder hidden state h^d_t.

At each decoding step t for generating output word y_t, an attention score u^e_{tj} is computed based on the encoder hidden state h^e_j and the decoder hidden state h^d_t as in [8]:

$$u^e_{tj} = \vartheta^{\mathrm{T}} \tanh(W_{align}(h^e_j \oplus h^d_t) + \ b_{align}) \tag{1}$$

In which ϑ , W_{align} and b_{align} are learnable parameters.

At each step t, the attention distribution a^e_{tj} over the source words $u^e_{t1}, u^e_{t2}, \dots, u^e_{tJ}$ is calculated as follows:

$$a_{tj}^{e} = \frac{\exp(u_{tj}^{e})}{\sum_{k=1}^{J} \exp(u_{tk}^{e})} \qquad (2)$$

The weighted sum of the encoder hidden states is:

$$c_t^e = \sum_{j=1}^{J} a_{tj}^{e} h_j^e \qquad (3)$$

With the current decoder hidden state h_t^d , we calculate vocabulary distribution as follows:

$$P_{vocab,t} = softmax(W_{d2v}(W_c[c_t^e, h_t^d] + b_c) + b_{d2v} \qquad (4)$$

with W_{d2v} , W_c , b_c, and b_{d2v} are learnable parameters.

Copying from source article Nallapati et al. [3] have pointed out a weakness in the LSTM network in general and in his model in particular, that is, the model cannot recover correctly OOV words (e.g numbers, private names) in the output. See et al. [6] solved this problem by allowing the model to occasionally copy words directly from the source instead of generating a new word based on their attention weights. This is done by a *pointer-generator network* - a specially designed seq2seq attentional model that can generate the summary by copying words in the article or generating words from a fixed vocabulary at the same time. The generation probability of a word $\mathrm{p_{gen,t}}$ at time t is:

$$p_{gen,t} = \sigma(W_{s,c}c_t^e + W_{s,h}h_t^d + W_{s,y}y_t + b_s) \qquad (5)$$

in which $W_{s,c}$, $W_{s,h}$, $W_{s,y}$ and b_s are learnable parameters.
The final distribution that can deal with OOV words is computed as:

$$P(y_t) = p_{gen,t}P_g(y_t) + (1 - p_{gen,t})P_c(y_t) \qquad (6)$$

in which:
$P_g(y_t)$ is the vocabulary distribution corresponding to the "generator" function of *the pointer-generator network.*

$$P_g(y_t) = \begin{cases} P_{vocab,t}(y_t) \ if \ y_t \in V \\ 0 \quad otherwise \end{cases} \qquad (7)$$

with V is the vocabulary
$P_c(y_t)$ represents the attention distribution corresponding to the "copy" function of *the pointer-generator network.*

$$P_c(y_t) = \begin{cases} \sum_{j:x_j=y_t} a_{tj}^{e} \ \ y_t \in V_1 \\ 0 \quad otherwise \end{cases} \qquad (8)$$

with V_1 is the word sequence of the input text

Coverage mechanism The coverage model was first proposed by Tu et al. [9] for the NMT task, then See et al. [6] applied this mechanism for abstract summarization to overcome word repeat errors. In each decoder step t, they calculate the coverage vector cov_t^e as the sum of attention distributions of the previous decoding steps:

$$cov_t^e = \sum_j^{t-1} a_{tj}^e \tag{9}$$

The coverage vector cov_t^e is used to calculate the attention score as:

$$u_{tj}^e = \vartheta^{\mathrm{T}} \tanh(W_{align}(h_j^e \oplus h_t^d \oplus cov_t^e) + \ b_{align}) \tag{10}$$

Besides, See et al. [6] defined a coverage loss to penalize repeatedly attending to the same locations when generating multi-sentence summaries.

3.2 Our proposed model

Rich features According to Pascanu et al. [10], a weakness of the models developed based on RNN is a vanishing gradient problem. That means, when the input is too long, the first part of the text will be forgotten. The LSTM model does not completely solve this problem. Since the main content of articles is often located at the beginning, See et al. [6] deal with this problem by using only the first part of the article to put into the model. However, that solution reduces the generality of the system since not all textual types put important content in the first part of the text. To deal with this problem, we add information about sentence position (POSI) as a feature in the network, to enhance the weight of the first sentences without cutting off the input text. The sequence of input words $x = (x_1, x_2, \ldots, x_J)$ now re-expressed as:

$$x = (x_{1_1}, x_{2_1}, \ldots, x_{J_k}) \tag{11}$$

where x_{j_k} means word x_j of the k^{th} sentence. From there, we can easily create $x_{j_{position}}$ as:

$$x_{j_{position}} = k \tag{12}$$

in which k denotes the sentence position of x_j in the article.

The output word y_t is generated based on the attention distributions of all input words in the encoder side and previous output words. Since we now include the entire article without cutting off the last part, when the array's size grows, the attention distribution of each word will decrease, thus the effect of each attention will be reduced. To fix this, we use term frequency (TF) to help the model focus on important words. The frequency of each word is calculated as follows:

$$tf(x_j, x) = \frac{f(x_j, x)}{\max\{f(x_i, x) \mid i = 1 \rightarrow J\}} \tag{13}$$

where $f(x_j, x)$ is the number of occurrences of x_j in the article, $\max\{f(x_i, x) | i = 1 \rightarrow J\}$ is the highest number of occurrences of any word in the article. Based on this, we denote the frequency of x_j as ${x_j}_{TF}$ as follows:

$$ {x_j}_{TF} = tf\left(x_j, x\right) \tag{14} $$

By using **Rich Features**, we improve the formula to compute the attention score in [6] by a new formula as shown below:

$$ u_{tj}^{e} = \frac{\vartheta^{\mathrm{T}} \tanh\left(W_{align}\left(h_j^e \oplus h_t^d \oplus u_t^e\right) + \ b_{align}\right) {x_j}_{TF}}{{x_j}_{position}} \tag{15} $$

The value of u_{tj}^{e} is inversely proportional to the sentence position. Therefore, sentences at the end of the article will have less effect than sentences at the beginning of the article. Also, the word that has high term frequency will have a high attention score.

Our proposed model is shown in **Figure 2**. Sentence positions and term frequencies are represented as two vectors that have equal lengths with the input article. In our experiments, this length is set to 550 (words) with Vietnamese articles and 800 (words) with English articles. These two vectors are concatenated together with word vectors as input to the encoder.

At each decoder step, information of the sentence positions and the term frequencies are used to compute the attention score as in (15). Then the attention score is used to calculate the attention distribution a_{tj}^{e} as in (2). Because of this, the attention distribution of the first part of the text will be higher than that of the last part.

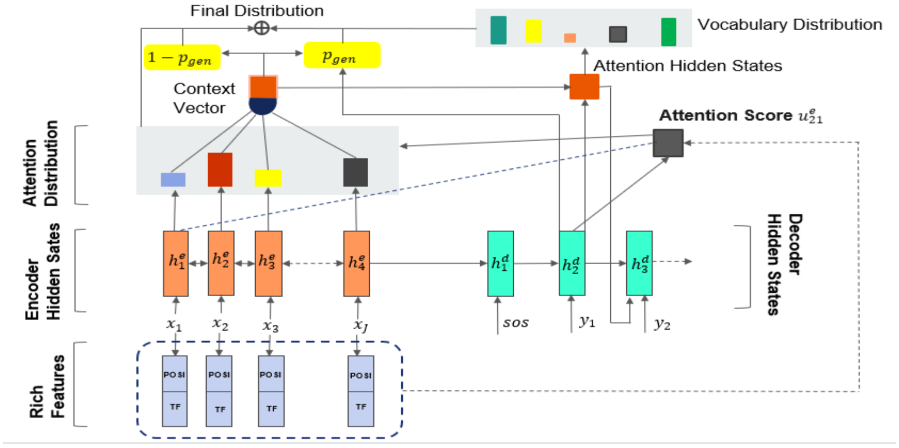

Figure 2. Our proposed model using rich features

4 Experiments and Results

4.1 Dataset

Our experiments were carried out with two datasets: CNN/Daily Mail dataset for English and Baomoi for Vietnamese. The purpose of using the first dataset is to compared results with recent works in abstract summarization. Experiments with the second dataset to evaluate our proposed method on another language and to guaranty the generality of our approach.

The CNN/Daily Mail dataset for abstractive summarization is first established by Nallapati et al. [3]. This dataset contains 287,113 training samples, 13,368 validation samples, and 11,490 test ones. Each sample consists of a news article from CNN or Daily Mail websites accompanying with its summary. The average length of the original articles and their summaries are 781 words and 3.75 sentences (or 56 words), respectively. This dataset has two versions: anonymized and non-anonymized. In the anonymized version, each named entity, e.g., IBM, was replaced by its unique identifier, e.g., @entity10. We use the unanonymized version of the data since it more like the nature of the text summarization task.

There is no available corpus for the Vietnamese text summarization task. Therefore, we have to create a corpus by ourselves. This is done by gathering articles from a Vietnamese online newspaper (http://baomoi.com). The structure of each article consists of 3 parts: headline, abstract, and the article's body. The headline is the first sentence of the article. The abstract is the first paragraph of the article, after the headline. The remaining part is the article's body. The abstract is more likely the key information of the article, which guides readers on whether to continue reading the article or not, rather than a complete summary. In other words, the abstract sometimes can lack important information from the article. However, since we cannot find any better source, Baomoi is still our best choice to be used as the summarization corpus at the moment. We take the article part and the abstract part to serve as the original text and its summary. The average length of the original text and its summary are 503 words and 45 words, respectively.

Length's distributions of abstracts and articles in the Baomoi dataset are shown in **Figure 3** below. The final dataset consists of 1,187,000 news articles, in which 951,000 samples are used for training, 117,000 samples for validation, and 119,000 ones for testing.

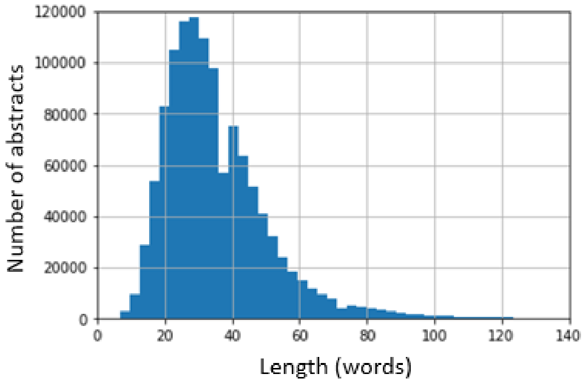

Figure 3.a. The length's distribution of abstracts in the Baomoi dataset

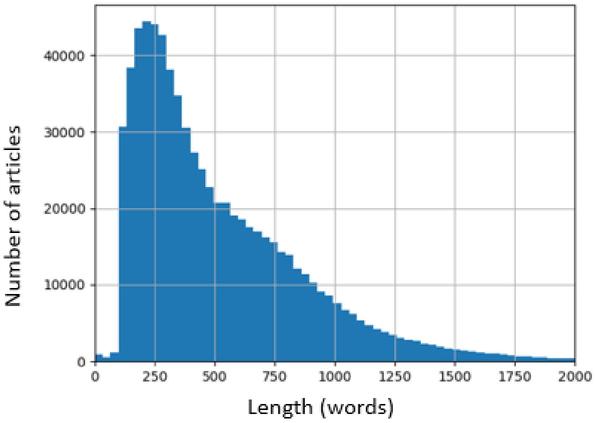

Figure 3.b. The length's distribution of articles in the Baomoi dataset

4.2 Preprocessing data

For the Vietnamese dataset, we cleaned the data by removing words that have no meaning such as the newspaper's address dantri.vn, the author's name at the end of the article, etc. since they do not contribute to the article's content. The articles that are too short (less than 50 characters) were also removed. The tool UETSegment[1] and Stanford CoreNLP[2] were used to tokenize Vietnamese and English text, respectively.

[1] Available at http://github.com/phongnt570/UETsegmenter

[2] Available at http:// stanfordnlp.github.io/CoreNLP/

4.3 Experiments

For each dataset (CNN/Daily Mail and Baomoi), we carried out experiments with four different models:

(i) A baseline sequence-to-sequence RNNs network with attention mechanism, proposed by Nallapati et al. [3]
(ii) The pointer generator network with coverage of [6]
(iii) Our proposed model basing on [6], adding information about sentences' positions
(iv) Our proposed model basing on [6], adding information about sentences' positions and term frequencies

With the first two models, we took the source code from [6] [3], and run them with the two datasets mentioned in Section 4.1. The last two experiments were carried out with our models.

Inputs of our model are a sequence of words from the article, with each word being represented as a one-hot vector. The vocabulary size in these experiments is 50,000 for both English and Vietnamese data. For all experiments, our model has 256-dimensional hidden states and 128-dimensional word embedding. We limit the mini-batches size to 16 and the input article length to 800 words for English and 550 words for Vietnamese. Since English and Vietnamese articles are less than 800 words and 550 words, respectively, the limit of input article lengths as mentioned above is enough for the system to get the full text of the article. We used the Adagrad optimizer [11] with the learning rate of 0.15 and the initial accumulator value of 0.1. We assigned the gradient clipping with a maximum gradient norm of 2, but did not use any form of regularization. When tunning the system, the loss value was used to implement early stopping. In the testing phase, the summary length was limited to 100 for both datasets.

We also carried out an experiment with See et al. [6]'s best model using the CNN/Daily Mail dataset to evaluate the effect of using only the first 400 words as the system's input.

4.4 Results

Our experimental results when using CNN/Daily Mail datasets are shown in **Table 1** below. The standard Rouge metric [12], including the F-score for Rouge-1, Rouge-2, and Rouge-L measures were used to evaluate our systems.

	ROUGE		
	1	2	L
Seq2seq + attn baseline	27.21	10.09	24.48
Pointer-Gen + coverage (*)	29.71	12.13	28.05
Our model: (*) + POSI	31.16	12.66	28.61
Our model: (*) + POSI + TF	**31.89**	**13.01**	**29.97**

Table 1. Results on the CNN/DailyMail dataset – (*) is See et al. [6]'s model

[3] Available at https://github.com/abisee/pointer-generator

When repeated experiments in [6] using the first 400 words of articles as the input, we got a 35.87% Rouge-1 score. However, when using the full article as the input, the Rouge-1 score reduce to 29.71%. This is because when feeding a long text to the model, the first part of the text is “forgot” by the system. Unfortunately, the first part of an article usually keeps the main content of the article. However, summarizing articles in this way will reduce the generality of the system, as in other cases, important information may not locate at the first 400 words of the document.

As can be seen from **Table 1**, when using the full text of articles as the input, both of our proposed models outperform the systems of Nallapati et al. [3] and See et al. [6] in all the three Rouge scores. It indicates that the position is important information in generating a summary. The experimental results also show that term frequencies are a good indicator for summarization tasks using deep learning techniques. When information about sentence positions and term frequencies are added to the model, the Rouge-1 score is significantly improved with 2.18% Rouge-1 higher than that of See et al. [6]’s system.

Table 2 shows our experimental results with the Baomoi dataset.

	ROUGE		
	1	2	L
Seq2seq + attention	26.68	9.34	16.49
Pointer-Gen + coverage (*)	28.34	11.06	18.55
Our model: (*) + POSI	29.47	11.31	18.85
Our model: (*) + POSI + TF	**30.59**	**11.53**	**19.45**

Table 2. Results on the BaoMoi dataset – (*) is See et al. [6]’s model

The results in **Table 2** also pointed out that both of our systems achieve higher Rouge scores than that of the other two systems. Our best model obtained 2.25% Rouge-1 higher than that of See et al. [6] and 3.91% Rouge-1 higher than the baseline.

Tables 3 show an output of See et al. [6]’s model and our best model, using the full article in the CNN/Daily Mail dataset as the input:

REFERENCE SUMMARY: Mary Todd Lowrance, teacher at Moises e Molina high school, turned herself into Dallas independent school district police on Thursday morning.
Dallas isd police said she had been in a relationship with student, who is older than 17 years old, for a couple of months.
She confided in coworker who alerted authorities and police eventually got arrest warrant.
Lowrance was booked into county jail on $ 5,000 bond and has been released from the Dallas county jail, according to county records.
She has been on leave for several weeks while investigators worked on the case, police said.

Pointer-Generator, Coverage (See et al. [6]):

Lowrance Lowrance was accused of a male school on a $ 5,000 bond. Lowrance has been on leave for several weeks while investigators worked on the case. The student has been offered counseling warrant.
Pointer-Generator, Coverage, Rich Features (our system): Mary Todd Lowrance, 49, arrested after being accused of having an improper relationship with a male student older than 17 years old. Miller said the teacher had been in a relationship with the student for a couple of months. Officers learned of the alleged relationship after Lowrance disclosed details to a coworker who then alerted a Dallas isd officer.

Table 3. Outputs of See et al. [6]'s model and our best model, using an input from the CNN/Daily Mail dataset

As can be seen from Table 3, the summary of our proposed system is more informative than that of [6], and it does not have the problem of repeated words.

Experimental results with a Baomoi's article are shown in Table 4 below.

REFERENCE SUMMARY: Một xưởng sản_xuất phi_pháp tại __Tức_Mặc__, Thanh_đảo, Trung_quốc vừa bị phát_hiện sử_dụng 1.000 kg dầu bẩn để chế_biến bánh_trung_thu. *An illegal factory in Tuc_Mac, Qingdao, China has just been found out of using 1,000 kg of dirty oil to process moon cakes.*
Pointer-Generator, Coverage (See et al. [6]): Sự_việc bị phanh_phui vào ngày 30/8 vừa_qua. Cơ_quan an_toàn thực_phẩm, phòng công_thương, công_an và chính_quyền thị_xã đã phối_hợp điều_tra sự_việc bị phanh_phui vào ngày 30/8 vừa_qua. *The incident was uncovered on August 30. Food safety agencies, industrial and commercial bureaus, police and town authorities coordinated to investigate the uncovered incident on August 30.*
Pointer-Generator, Coverage, Rich Features (our system): 1.000 kg dầu bẩn vừa được chuyển tới bởi một xe_tải chở hàng. Theo nguồn_tin thân_cậy, số dầu này có xuất_xứ từ một xưởng chế_biến tại mã_điếm, Giao_Châu. Ngày 31/8, một tổ công_tác gồm phòng an_toàn thực_phẩm, công_an và chính_quyền thị_xã đã phối_hợp điều_tra sự_việc này. *1,000 kg of dirty oil has just been delivered by a freight truck. According to reliable sources, this oil comes from a processing factory in Ma Diem, Giao Chau. On August 31, a working group of the food safety department, the police and the town government coordinated in investigating this case.*

Table 4. Outputs of See et al. [6]'s model and our best model, using an input from the Baomoi dataset

The main information of the article in Table 4 is "*1,000 kg of dirty oil from a processing factory in Ma Diem, Giao Chau has been delivered to an illegal factory in Tuc_Mac, Qingdao to process moon cakes. On August 31, a working group of the food safety department, the police and the town government coordinated in investigating this case*". The reference summary contains most of the above information. The summary generated by [6] does not contain the key point "*1,000 kg of dirty oil*", and only provides half of the necessary information. Also, although the output of [6] is short and lack of the main information, the phrase "*sự_việc bị phanh_phui vào ngày 30/8 vừa_qua|the incident was uncovered on August 30*" is repeated twice. Meanwhile, the summary generated by our system provides more information than that of [6] and does not contains redundance phrases. Besides, our system's output is easier to understand without grammatical errors for both of the English dataset and the Vietnamese dataset.

5 Conclusions

In this paper, we present our work on abstractive summarization, using the pointer generator network with a coverage mechanism, combining with information about sentence positions and term frequencies. Our proposed approach solves the problem of recurrent neural networks that focus on the last part of the input text. Our experimental results with both of the English dataset and the Vietnamese dataset indicate that our approach is language-independent, as our system provides higher Rouge scores than previous systems of Nallapati et al. [3] and See et al. [6]. It also proves that our proposed features (sentence position and term frequency) are important in abstractive summarization tasks using the seq2seq network.

Nevertheless, there are still rooms for future works. Since the quality of abstracts in the Baomoi dataset are not good, it is necessary to build a summarization dataset with better quality. Also, since the structure of a document plays an important role in understanding that document, we will investigate methods to learn that structure, in order to provide a better summary for long texts. Hierarchical Attention Networks [13] is a candidate for this purpose since it is good in capturing document structures.

References

[1] A. M Rush, S. Chopra, and J. Weston. *A neural attention model for abstractive sentence summarization*. arXiv preprint arXiv:1509.00685. 2015.

[2] S. Narayan, S. B. Cohen, and M. Lapata. *Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization*. In EMNLP. 2018.

[3] R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang. *Abstractive text summarization using sequence-to-sequence RNNs and beyond*. In Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, 2016, pages 280–290.

[4] J. Gu, Z. Lu, H. Li, and V. OK Li. 2016. *Incorporating copying mechanism in sequence-to-sequence learning*. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 1631–1640.

[5] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, 2015, pages 2692–2700.

[6] A. See, P. J Liu, and C. D. Manning. *Get to the point: Summarization with pointer generator networks*. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, volume 1, pages 1073–1083.

[7] Q. Chen, X. Zhu, Z. Ling, S. Wei, and H. Jiang. *Distraction-based neural networks for modeling documents.* In International Joint Conference on Artificial Intelligence. 2016.

[8] T. Luong, H. Pham, and C. D. Manning. *Effective approaches to attention-based neural machine translation.* In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 1412–1421.

[9] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li. Modeling coverage for neural machine translation. In *Association for Computational Linguistics.* 2016.

[10] R. Pascanu, T. Mikolov, Y. Bengio. *On the difficulty of training recurrent neural networks.* ICML'13 Proceedings of the 30th International Conference on International Conference on Machine Learning. 2013. Volume 28.

[11] J. Duchi, E. Hazan, and Y. Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research 12:2121–2159.*

[12] C.Y. Lin. *Rouge: A package for automatic evaluation of summaries*. In Text summarization branches out: ACL workshop. 2004.

[13] Z.Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy. *Hierarchical Attention Networks for Document Classification*. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016. pp. 1480–1489