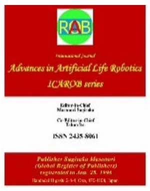
SUGISAKA MASANORI

Journal of Advances in Artificial Life Robotics
Vol. 1(2); September (2020), pp. 96–100
ON LINE ISSN 2435-8061; ISSN-L 2435-8061
https://alife-robotics.org/jallr.html



Research Article

Effectiveness of Data Augmentation in Pointer-Generator Model

Tomohito Ouchi, Masayoshi Tabuse
Graduate School of Life and Environmental Sciences, Kyoto Prefectural University, 1-5 Shimogamohangi-cho, Sakyo-ku, Kyoto 606-8522, Japan

ARTICLE INFO

Article History

Received 13 November 2019
Accepted 21 July 2020

Keywords

Automatic summarization
Data augmentation
Pointer-Generator Model

ABSTRACT

We propose a new data augmentation method in automatic summarization system, especially the Pointer-Generator model. A large corpus is required to create an automatic summarization system using deep learning. However, in the field of natural language processing, especially in the field of automatic summarization, there are not many data sets that are sufficient to train automatic summarization system. Therefore, we propose a new method of data augmentation. We use the Pointer-Generator model. First, we determine the importance of each sentence in an article using topic model. In order to augment the data, we remove the least important sentence from an input article and use it as a new article. We examine the effectiveness of our proposed data augmentation method in automatic summarization system.

1. Introduction

In recent years, the amount of information on the net of the world has increased exponentially, and it is expected to become 44 ZB in 2020 [1]. Under the circumstance, the technique of automatic summarization becomes indispensable for efficient selection of information. However, in the field of automatic summarization, there are not many data sets that are sufficient to train automatic summarization system. Therefore, we consider whether the technique of data augmentation commonly used in image processing can be applied to natural language processing.

Several studies have applied data augmentation to natural language processing. For example, in document classification the data augmentation methods of the studies are replacement with synonyms, replacement by calculating similarity, replacement with antonyms, replacement of words with words in sentences and random deletion, etc. Also, in machine translation, there is the Back Translation method [2]. However, none have yet been adapted to an automatic summarization system. The nature of automatic summarization is that there is little influence on the generated summary whether there is redundant part or not. Further, in image processing, data may be expanded by changing the background. This assumes that the background has no direct impact on the target task. Therefore, we focused on change of background which is one of data expansion of image processing, and expanded data by deleting redundant sentences of original article in automatic summary.

Automatic summarization is classified into two approaches. One is an extractive summarization in which words, phrases or sentences of input articles are directly extracted and combined to create a summary. The other is an abstractive summarization in which an input article is temporarily converted to an intermediate vector and then a summary is generated based on the intermediate vector. In extractive summarization, there is a problem of how to connect extracted words, phrases or sentences. And there is also the possibility that in extractive summarization it is unknown what the directive word is pointing at. On the other hand, in abstractive summarization, since it is generated from an internal semantic representation, there

Corresponding author's E-mail: t_ouchi@mei.kpu.ac.jp, tabuse@kpu.ac.jp

is a possibility that it becomes one integrated summary. It also learns grammatical information, so it is easy to connect words and phrases in contrast to an extractive summarization. However, the abstract summarization needs to be learned with a large number of pairs of input articles and generation summaries, which takes time and effort to prepare the corpus.

In the previous research [3], we showed data augmentation is effective in abstractive summary. But in that research, using model is only a union of Encoder-Decoder model and attention mechanism. Therefore, in this study, we confirm the effect of data augmentation using the latest model, the Pointer-Generator model [4]. The automatic summarization system used in this study is described in Section 2. The method of data augmentation used in this research is described in Section 3. Experiment and evaluation are described in Section 4. Discussion and summary are given in Section 5.

2. Pointer-Generator model

The pointer-Generator model [4] shown in Fig. 1 processes according to the following procedure.

1. Encode input text (Source Text) with bidirectional LSTM. Convert variable length data to fixed length vector. (Red part)
2. The word generation probability (Green part) is obtained by using the attention mechanism (Blue part) with the decoder LSTM. By performing this process, the summary should be generated using the information of the entire input sentence as much as possible.
3. Add the attention value in the encoding part to the word generation probability distribution in the decoding part shown as the final distribution in Fig. 1 (copy mechanism). With the copy mechanism, it is possible to generate words that are not included in the output vocabulary of the model for inefficient calculations, delete low-frequency words in advance.
4. Select the final generated word from final distribution.
5. Repeat steps 2-4 to generate words sequentially. If the token that represents the end of the sentence is selected, the decoding ends there.

Furthermore, Attention is attenuated where the value of attention was larger than before (coverage mechanism).

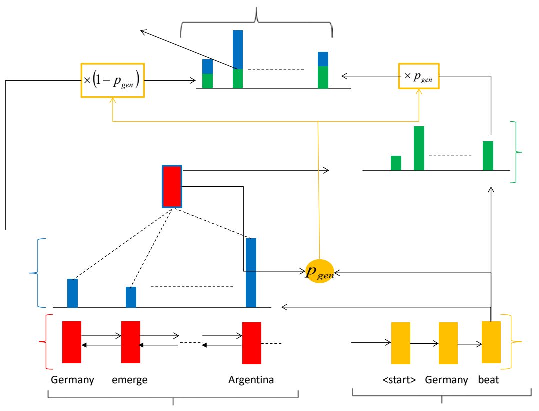


Fig. 1. Pointer-Generator model [4].

3. Data augmentation Method

We measure the importance for each sentence of the input article and use the article generated excluding the lowest important sentence as new data.

Refer to [5] for how to determine importance for each sentence in the input article. First, the topic model is created from the training data set, and the sentence weight is determined for each topic of the input article. Then, the sum of sentence weights for all topics is taken as the final weight of the sentence, and it was taken as the importance of the sentence.

$$b_{t,i} = \frac{W_{t,i}}{\sqrt{N_i}} \tag{1}$$

$$b_i = \sum_t b_{t,i} \tag{2}$$

Firstly the weight $b_{t,i}$ of the sentence i with respect to the topic t is examined. In Equation (1), $b_{t,i}$ multiplies the sum $W_{t,i}$ of an importance of the words constituting the sentence i by the inverse of the square root of the total number N_i of words as a coefficient. This coefficient is for weighting sentences that are not influenced by the sentence length. In Equation (2), we let the sum of $b_{t,i}$ be the final weight of sentence i.

4. Experiment and Evaluation

In this section, we evaluate the effectiveness of the data augmentation proposed in Section 3.

4.1. *Dataset*

We used CNN/DailyMail dataset [6]. This dataset has 287,226 training pairs, 13,368 validation pairs and 11,490 test pairs. In the experiment, 10,000 articles, 45,000 articles, 90,000 articles, 180,000 articles, and 287,226 articles are prepared as training data.

4.2. *Evaluation method*

ROUGE [7] is used as a general evaluation method of automatic summarization, which is evaluated by matching of reference summary and generated summary word. We used ROUGE-1, ROUGE-2, ROUGE-L. ROUGE-1 and ROUGE-2 evaluate unigram and bigram coincidence, respectively. ROUGE-L evaluates matching the longest common subsequence. Each ROUGE evaluation method has F value, recall, and precision. "Recall" means how many words of the generated summary are included in the reference summary, and "precision" means how many words of the reference summary are included in the generated summary. The F value is obtained by Equation (3).

$$F = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

4.3. *Parameter settings*

In this research we use the program written by PyTorch[8]. It has been verified that this program can achieve the same result as [4]. The hidden layer vector size was set to 256 and the embedded vector size was set to 128. The batch size was set to 8. In the original paper, the batch size is 16, so double learning is required to learn the same number of articles. The beam size was set to 4. The beam search will be described later. The number of vocabulary was set to 50,000. The learning rate was set to 0.15.

In this program, the number of words used to encode an input article is limited to 400. However, in the case there is no sentence with the lowest importance within 400 words from the beginning when we use the data augmentation method, since the sentence is omitted there is no effect on learning. Therefore, I found the article with the most words among the articles used in the training data. The number of words with the most words was 2,380. And the upper limit of the number of words used in encoding the input article was set to 2,380. Table 1 show the values of ROUGE when the maximum number of words is 400 and 2,380. In the Table 1, f, r, and p represent the F value, recall, and precision, respectively.

Table 1. the values of ROUGE when the maximum number of words is 400 and 2,380

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
400	0.3935	0.4372	0.3800	0.1709	0.1891	0.1662
2380	0.3958	0.4181	0.3994	0.1741	0.1832	0.1770

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
400	0.3616	0.4014	0.3493
2380	0.3644	0.3846	0.3679

Table 1 shows when the upper limit of the number of words is increased from 400 to 2380, the value of ROUGE increases slightly. In the following, the experiment is performed with the upper limit of the number of words set to 2,380.

In the research of [4], after learning with the copy mechanism, it learns with the coverage mechanism. Therefore, we use copy mechanism and coverage mechanism in this research. Table 2 shows how many times batch-sized articles are learned.

Table 2. the number of iteration times each articles was learned

	iteration in copy mechanism	iteration in coverage mechanism
10,000 articles	20,000 times	300 times
45,000 articles	80,000 times	1,000 times
90,000 articles	160,000 times	2,000 times
180,000 articles	320,000 times	4,000 times
287,226 articles	460,000 times	6,000 times

The Pointer-Generator program used in this research takes about 0.27 seconds to learn 1 batch using GeForce GTX 1080 Ti GPU. Therefore, it takes about an hour and a half to learn 10,000 articles. Similarly, it takes about 6 hours, 12 hours, 24 hours, 35 hours to learn 45,000 articles, 90,000 articles, 180,000 articles, 287,226 articles. While adding extended articles, the total epoch count is halved, therefore the learning time is almost the same.

4.4. *Beam Search*

In our research we use the beam search. The beam search contrasts with the greedy method. This is because, when generating a word, one word with the highest generation probability is selected in the greedy method, while in the beam search, processing is performed while holding the top K words. We make the final summarizations by multiplying the probabilities of each word generation, and make the highest one the final summarization. In this experiment, this K value is set to 4.

4.5. *Result*

The ROUGE evaluation is shown below.

Table 3. Results on 10,000 articles. 'original' is original data and 'augmented' is augmented data.

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
original	0.3226	0.3063	0.3665	0.1209	0.1142	0.1389
argmented	0.3335	0.3181	0.3761	0.1275	0.1210	0.1452

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
original	0.2994	0.2841	0.3403
argmented	0.3085	0.2940	0.3482

Table 4. Results on 45,000 articles

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
original	0.3398	0.3322	0.3731	0.1304	0.1269	0.1446
argmented	0.3506	0.3556	0.3698	0.1378	0.1393	0.1464

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
original	0.3155	0.3082	0.3468
argmented	0.3249	0.3293	0.3428

Table 5. Results on 90,000 articles

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
original	0.3538	0.3507	0.3830	0.1404	0.1387	0.1531
argmented	0.3627	0.3565	0.3945	0.1495	0.1467	0.1636

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
original	0.3288	0.3258	0.3562
argmented	0.3372	0.3313	0.3671

Table 6. Results on 180,000 articles

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
original	0.3758	0.3884	0.3893	0.1587	0.1635	0.1655
argmented	0.3827	0.4027	0.3874	0.1628	0.1711	0.1655

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
original	0.3475	0.3588	0.3604
argmented	0.3515	0.3698	0.3561

Table 7. Results on 287,226 articles

	ROUGE-1-f	ROUGE-1-r	ROUGE-1-p	ROUGE-2-f	ROUGE-2-r	ROUGE-2-p
original	0.3918	0.4121	0.3958	0.1703	0.1784	0.1733
argmented	0.3941	0.4288	0.3859	0.1721	0.1869	0.1693

	ROUGE-L-f	ROUGE-L-r	ROUGE-L-p
original	0.3593	0.3776	0.3633
argmented	0.3625	0.3941	0.3551

4.6. *Discussion*

Tables 3,4,5,6,7 show that the data augmentation is effective in all articles. However, as the number of articles increases, the data augmentation effect fades. Table 8 shows the value obtained by subtracting the F value of ROUGE of 'original' from that of 'augmented'.

Since all the values in Table 8 are positive, it is better to extend in each the number of articles. Also, looking at ROUGE-1, it gradually decreases as the number of articles increases. Looking at ROUGE-2, it goes up to 90,000 articles and decreases from there. Looking at ROUGE-L, it goes up to 45,000 articles and decreases from there.

The reason why the data augmentation effect is less likely to appear as the number of articles increases is as follows. In a sufficient number of articles, learning became saturated and the data augmentation is not very effective. On the other hand, in the case where there were not many articles, the data augmentation is effective due to lack of training data. Therefore, it can be said that this method is more effective when the number of articles is small.

Table 8. the value obtained by subtracting the F value of Rouge of 'original' from that of 'augmented'

	10,000articles	45,000articles	90,000articles	180,000articles	287,226articles
ROUGE-1-f	0.0109	0.0108	0.0089	0.0069	0.0023
ROUGE-2-f	0.0066	0.0074	0.0091	0.0041	0.0018
ROUGE-L-f	0.0091	0.0094	0.0084	0.0040	0.0032

5. Conclusion

In this experiment, it is found that the value of ROUGE is improved by learning on adding an article from which the sentence with the least importance is extracted, rather than learning normally. Moreover, since this method has the same number of learning times as the case where it is not augmented, it succeeded in improving the value of ROUGE without changing the learning time. In addition, compared to previous research [3], the overall value of ROUGE was much better, and it turns out that it can also be applied to a more accurate automatic summarization system. Moreover, the experiment of this research shows that the effect of the data augmentation was small under the same conditions as the previous research [4]. However, it has also been found that reducing the number of articles increases the effect of the data augmentation.

A future task is to increase the application range of this method. There are several models developed from the Pointer-Generator model [9],[10]. I would like to examine whether proposed method is effective. Also, in this research, we used the topic model to determine the importance of sentences, therefore, I would also like to examine whether the data augmentation is more effective when we use ROUGE to determine the importance of sentences. Furthermore, in this research, the sentence with the lowest importance was extracted, therefore, it is possible to extract only the sentence with the highest importance and make it an extended article. I would like to consider such a new extension method as a future research subject.

References

1. Data Growth, Business Opportunities, and the IT Imperatives
2. Sergey Edunov, Myle Ott, Michael Auli, David Grangier, "Understanding Back-Translation at Scale", arXiv:1808.0938lv2[cs.CL] (2018).

3. T. Ouchi, M. Tabuse, "Effectiveness of Data Augmentation in Automatic Summarization System", ICAROB2019, 177-180 (2019)
4. Abigail See, Peter J. Liu, Christopher D. Manning, "Get To The Point: Summarization with Pointer-Generator Networks", arXiv:1704.04368v2(2017).
5. H. Sigematsu, I. Kobayashi, "Generation of abstracts considering importance of potential topics", Proceedings of the Annual Meeting of the Association for Natural Language Processing (2012) (In Japanese).
6. DeepMind Q&A Dataset
7. Chin-Yew Lin, "ROUGE: A Package for Automatic Evaluation of Summaries", Proceeding of the Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004, Barcelona, Spain (2004).
8. pointer_summarizer
9. Mike Lewis, Yinhan Liu, Naman Goyal, et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension", arXiv:1910.13461v1(2019).
10. Li Dong, Nan Yang, et al., "Unified Language Model Pre-training for Natural Language Understanding and Generation", arXiv:1905.03197v3(2019).

Authors introduction

Mr. Tomohito Ouchi

He received his Master's degree from Department of Environmental Science, Graduate School of Life and Environmental Sciences, Kyoto Prefectural University, Japan in 2019. He is currently a Doctoral course student in Kyoto Prefectural University, Japan.

Dr. Masayoshi Tabuse

He received his M.S. and Ph.D. degrees from Kobe University in 1985 and 1988 respectively. From June 1992 to March 2003, he had worked in Miyazaki University. Since April 2003, he has been in Kyoto Prefectural University. His current research interests are machine learning, computer vision and natural language processing. IPSJ, IEICE and RSJ member.