ExtraPhrase: Efficient Data Augmentation for Abstractive Summarization

Mengsay Loem, Sho Takase, Masahiro Kaneko, and **Naoaki Okazaki**
Tokyo Institute of Technology
{mengsay.loem, sho.takase, masahiro.kaneko}@nlp.c.titech.ac.jp
naoaki.okazaki@c.titech.ac.jp

Abstract

Neural models trained with large amount of parallel data have achieved impressive performance in abstractive summarization tasks. However, large-scale parallel corpora are expensive and challenging to construct. In this work, we introduce a low-cost and effective strategy, **ExtraPhrase**, to augment training data for abstractive summarization tasks. ExtraPhrase constructs pseudo training data in two steps: extractive summarization and paraphrasing. We extract major parts of an input text in the extractive summarization step and obtain its diverse expressions with the paraphrasing step. Through experiments, we show that ExtraPhrase improves the performance of abstractive summarization tasks by more than 0.50 points in ROUGE scores compared to the setting without data augmentation. ExtraPhrase also outperforms existing methods such as back-translation and self-training. We also show that ExtraPhrase is significantly effective when the amount of genuine training data is remarkably small, i.e., a low-resource setting. Moreover, ExtraPhrase is more cost-efficient than the existing approaches[1].

1 Introduction

Neural encoder-decoders have achieved remarkable performance in various sequence-to-sequence tasks including machine translation, summarization, and grammatical error correction (Bahdanau et al., 2015; Rush et al., 2015; Yuan and Briscoe, 2016). Recent studies indicated that neural methods are governed by the scaling law for the amount of training data (Koehn and Knowles, 2017; Brown et al., 2020). In short, the more training data we prepare, the better performance a neural model achieves. In this paper, we address increasing the training data for summarization to improve the performance of neural encoder-decoders on abstractive summarization tasks.

[1]The datasets used in our experiments are available at https://github.com/loem-ms/ExtraPhrase.

In sequence-to-sequence tasks, we need a parallel corpus to train neural encoder-decoders. Since it is too costly to construct genuine (i.e., human-generated) parallel corpora, most studies explored the way to construct pseudo training data automatically. Back-translation is a widely used approach to construct pseudo training data for sequence-to-sequence tasks (Sennrich et al., 2016a; Edunov et al., 2018; Caswell et al., 2019). In the back-translation approach, we construct a model generating a source side sentence from a target side sentence, and apply the model to a target side corpus to generate a pseudo source side corpus. In addition to machine translation, back-translation is also used in grammatical error correction (Kiyono et al., 2019) and summarization (Parida and Motlicek, 2019) tasks. However, back-translation on summarization is an unrealistic problem because a model is required to restore deleted information in the given summary without any guide.

He et al. (2020) indicated that the self-training approach, which makes a model generate target sentences from source sentences and use the pairs to train a model, can improve the performance on machine translation and summarization. However, pseudo data generation for summarization by self-training is hard to generate diverse summaries (Gu et al., 2018). Moreover, self-training and back-translation approaches require expensive computational cost because we need to train additional neural encoder-decoders on a large amount of training data to obtain high-quality pseudo data (Imankulova et al., 2019).

To solve these issues, we propose a novel strategy: **ExtraPhrase** consisting of **extra**ctive summarization and para**phrase** to construct pseudo training data for abstractive summarization. Firstly, ExtraPhrase extracts an important part from a source text as a summary without requiring additional model training. Then, we apply a paraphrasing technique to the extracted text to obtain diverse

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop, pages 16 - 24

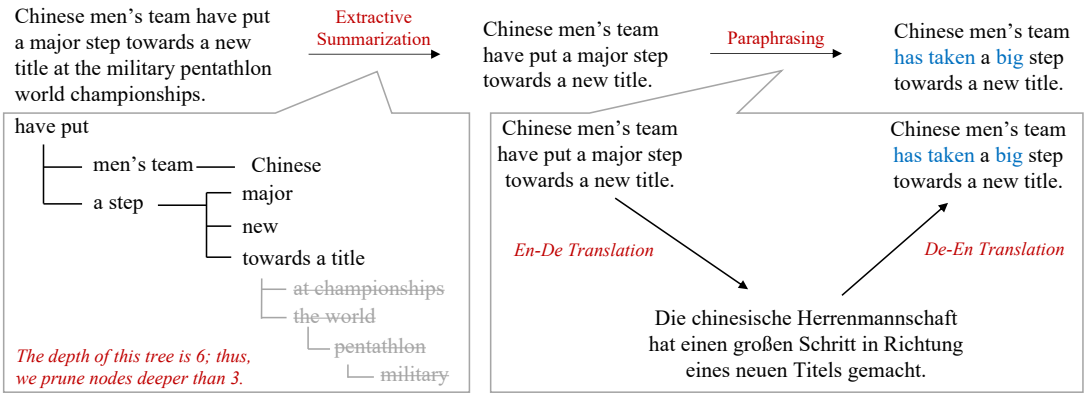

Figure 1: Example of pseudo summary generated by **ExtraPhrase**. The upper part shows output sentences in each step of ExtraPhrase. Paraphrased words after paraphrasing (round-trip translation) in step-2 are highlighted in blue.

pseudo summaries.

We conduct experiments on two summarization tasks: headline generation and document summarization tasks. Experimental results show that pseudo training data constructed by our proposed strategy improves the performance on both tasks. In detail, the pseudo data raises more than 0.50 in ROUGE F1 scores on both tasks. Moreover, we show that ExtraPhrase is robust in low-resource settings and is much more cost-efficient than previous self-training and back-translation approaches.

2 Proposed Method: ExtraPhrase

As described in Section 1, our ExtraPhrase consists of two steps: extractive summarization and paraphrasing. Figure 1 illustrates the overview of ExtraPhrase briefly. ExtraPhrase receives a (genuine) sentence as an input, and generates a pseudo summary corresponding to the input sentence. When we construct a pseudo summary from a document, we independently apply ExtraPhrase to multiple sentences included in the given document.

2.1 Step-1: Extractive Summarization

In this extractive summarization step, we extract important parts of a given source sentence with sentence compression. Previous studies proposed various sentence compression methods such as rule-based methods (Dorr et al., 2003), the approach detecting important parts in a syntax tree (Turner and Charniak, 2005; Filippova and Altun, 2013; Cohn and Lapata, 2009), sequential labeling approach (Hirao et al., 2009), and neural-based methods (Filippova et al., 2015; Kamigaito et al., 2018).

In this study, we adopt the most straightforward approach: a rule-based method based on the syntax tree of the given sentence. Because the rule-based approach does not require any training corpus, we can use it in the situation where we do not have genuine parallel corpus. We emphasize that we can use more sophisticated way if we need because we do not have any restrictions for the summarization method in this step.

We define a rooted subtree of the syntax tree for the given sentence as important parts of the sentence. First, we parse the given sentence to obtain its dependency tree. Follow Filippova and Altun (2013), we combine functional words with their heads on the dependency tree. Then, we prune the dependency tree to obtain a smaller rooted subtree. We can roughly control the output summary length (the number of words) by the depth of the subtree. The left lower part of Figure 1 illustrates these processes. Finally, we linearize the extracted rooted subtree to obtain its sequential representation by following the word order of the original sentence.

2.2 Step-2: Paraphrasing

The constructed summaries by the previous step consist of words included in the source sentences only. To increase the diversity of the summaries, we apply the paraphrasing method to the summaries. For paraphrasing, we adopt the approach using machine translation models (Sun and Zhou, 2012; Mallinson et al., 2017) because some studies published high-quality neural machine translation models (Ott et al., 2018; Ng et al., 2019). In this approach, we obtain paraphrases by conducting round-trip translation that translates a sentence into a different language and the translated sentence into the original language. The right lower part of Figure 1 illustrates this process.

3 Experiments

To investigate the effect of ExtraPhrase, we conduct experiments on two summarization tasks: headline generation and document summarization tasks.

3.1 Datasets

For the headline generation task, we use the de-facto headline generation dataset constructed by Rush et al. (2015). The dataset contains pairs of the first sentence and headline extracted from the annotated English Gigaword (Napoles et al., 2012). We use the same splits for train, valid, and test as Rush et al. (2015). We use the byte pair encoding (Sennrich et al., 2016b) to construct a vocabulary set with the size of 32K by sharing vocabulary between source and target sides.

For the document summarization task, we use CNN/DailyMail dataset (See et al., 2017). The training set contains 280K pairs of news articles and abstractive summary extracted from CNN and DailyMail websites. We construct a vocabulary set with the byte pair encoding (Sennrich et al., 2016b) and set the vocabulary size to 32K with sharing vocabulary between source and target sides.

3.2 Comparison Methods

We compare ExtraPhrase with several existing methods to increase the training data size as follows. We use the training set of each dataset described in Section 3.1 to construct pseudo data.

Oversampling This strategy is the simplest approach to increase the dataset size. We sample source-summary pairs from the genuine training set and add the sampled instances to training data. Thus, the training data constructed by this approach contains genuine data only.

Back-translation In back-translation, we train a neural encoder-decoder that generates a source text from a summary by using each training set. Then, we input summaries in the training set to the neural encoder-decoder to generate corresponding source texts[2]. We use the pairs of pseudo source texts and genuine summaries as pseudo training data.

[2]For the back-translation approach in machine translation, we generate sentences in the source language from monolingual corpus in the target language. In the abstractive summarization, we need summaries as sentences in the target language but it is hard to obtain corpus containing summaries only. Thus, we use genuine summaries in training data as an input of back-translation.

Self-training In self-training, we train a neural encoder-decoder that generates a summary from a source text by using each training set. Then, we input source texts in the training set to the neural encoder-decoder to generate the corresponding summaries. We use the pairs of pseudo summaries and genuine source texts as pseudo training data.

ExtraPhrase We apply ExtraPhrase to each training set. In the headline generation task, we construct pseudo summaries from the source sentence in the training data. Because ExtraPhrase generates pseudo summary in sentence unit, the number of sentences in generated summary is not reduced in the case of multi-sentence source text. Thus, we use the first three sentences in the source document to reduce the number of input sentences beforehand in the document summarization task. As described in Section 2, we apply ExtraPhrase to each sentence one-by-one, and then concatenate them in the original order. In this study, we use spaCy[3] (Honnibal et al., 2020) for dependency parsing and prune nodes whose depths are deeper than half of the dependency tree in the extractive summarization step. For the paraphrasing step, we use English-to-German and German-to-English translation models[4] constructed by Ng et al. (2019). We translate sentences with beam width 5.

For all pseudo training data, we attach a special token, `<Pseudo>`, to the front of the source text because Caswell et al. (2019) indicated that this strategy improves the performance in training on pseudo data.

3.3 Encoder-Decoder Architecture

We use the de-facto standard neural encoder-decoder model, Transformer (Vaswani et al., 2017) in our experiments. We also use the Transformer for back-translation and self-training in addition to each abstractive summarization model. We use the Transformer-base setting described in Vaswani et al. (2017) as our architecture. The setting is widely used in studies on machine translation (Vaswani et al., 2017; Ott et al., 2018). In detail, we use the implementation in the fairseq[5] (Ott et al., 2019) for our experiments.

[3]https://spacy.io/

[4]https://github.com/pytorch/fairseq/tree/main/examples/translation

[5]https://github.com/pytorch/fairseq

Method	Headline Generation				Document Summarization			
	Training Data	R-1	R-2	R-L	Training Data	R-1	R-2	R-L
Genuine only	3.8M	37.95	18.80	35.05	280K	39.76	17.55	36.75
Oversampling	7.6M	38.26	19.14	35.41	560K	40.14	17.86	37.05
Back-translation	7.6M (3.8M)	38.49	19.24	35.63	560K (280K)	39.93	17.74	36.85
Self-training	7.6M (3.8M)	38.32	19.06	35.37	560K (280K)	40.19	17.87	37.21
ExtraPhrase	7.6M (3.8M)	38.51	**19.52**	35.72	560K (280K)	**40.57**	**18.22**	**37.51**
w/o paraphrasing	7.6M (3.8M)	**38.85**	19.43	**35.86**	560K (280K)	40.32	17.94	37.28
w/o extractive	7.6M (3.8M)	38.52	19.32	35.71	560K (280K)	40.33	18.10	37.38

Table 1: ROUGE F1 scores (R-1, 2, and L) for the headline generation and document summarization tasks. The number of genuine training data is shown in parentheses.

3.4 Results

Table 1 shows F1 based ROUGE-1, 2, and L scores for each setting on the headline generation and document summarization tasks. We use the same size of training data for each method except for Genuine only.

Table 1 indicates that Oversampling outperforms Genuine only. This result indicates that the more training data we prepare, the better performance an encoder-decoder achieves even if the training data contains many duplications. For Back-translation and Self-training, they achieve better performance than Genuine only, but their scores are comparable to ones of Oversampling in both tasks. These results imply that the improvements in their approaches are not based on the quality of their generated pseudo data, but based on the increase of training data. Since Back-translation and Self-training require training an additional model to construct pseudo data, these approaches are more costly than Oversampling.

In contrast, our ExtraPhrase achieves better performance than other approaches. In particular, our pseudo training data significantly improves the ROUGE-2 score compared to Genuine only setting in the headline generation. For the document summarization, our pseudo training data significantly improves all ROUGE scores[6]. These results indicate that ExtraPhrase is more effective than existing approaches including oversampling, back-translation, and self-training to construct pseudo data for the abstractive summarization tasks.

In addition to configurations described in Section 3.2, we also report results when using each step of the proposed method to generate pseudo training data to investigate the effect of each step. ExtraPhrase w/o paraphrasing in Table 1 refers to applying only the extractive summarization described in 2.1 on source articles of genuine training data to obtain pseudo summaries. Similarly, ExtraPhrase w/o extractive refers to applying only the paraphrasing described in 2.2 on summaries of genuine training data.

For the headline generation task, ExtraPhrase w/o paraphrasing achieves better performance than Genuine only setting. Surprisingly, although with a small margin, this result also outperforms ExtraPhrase, where the paraphrasing step is applied after the extractive summarization, in ROUGE-1 and ROUGE-L. ExtraPhrase w/o extractive achieves comparable ROUGE-1 and ROUGE-L scores compared to ExtraPhrase, but with a decrease in ROUGE-2 score. However, this result is better than Oversampling, where duplicated data is used, which infers that the paraphrasing step effectively boosts the diversity in augmented training data.

For the document summarization task, summarization performance decreases in both ExtraPhrase w/o paraphrasing and ExtraPhrase w/o extractive. These results imply that ExtraPhrase is better than using each composing step alone.

4 Analysis

4.1 Low-resource Setting

In this section, we investigate the effectiveness of ExtraPhrase when the amount of genuine training data is small.

We randomly sample 1K source text and summary pairs from each training set in the headline generation and document summarization tasks. Then, we conduct the same experiments in Section 3 by using the sampled 1K instances as genuine training data. We construct pseudo training data from the rest of each training data and combine

[6]These results are statistically significant according to Student's t-test ($p < 0.05$) in comparison with Genuine only.

Method	Headline Generation				Document Summarization			
	Training Data	R-1	R-2	R-L	Training Data	R-1	R-2	R-L
Genuine only	1K	4.84	0.58	4.66	1K	2.48	0.29	2.45
Oversampling	3.8M	9.89	1.39	9.30	280K	13.63	0.89	12.63
Back-translation	3.8M (1K)	12.19	2.43	11.31	280K (1K)	9.73	0.50	8.92
Self-training	3.8M (1K)	7.27	1.07	6.98	280K (1K)	14.37	1.52	13.36
ExtraPhrase	3.8M (1K)	**23.58**	**6.56**	**21.12**	280K (1K)	**34.47**	**12.91**	**31.36**
w/o paraphrasing	3.8M (1K)	22.56	5.25	19.87	280K (1K)	32.95	12.07	29.44
Extractive	–	18.72	4.26	17.09	–	28.52	8.02	23.83

Table 2: ROUGE F1 scores (R-1, 2, and L) for the headline generation and document summarization tasks in low-resource setting. The number of genuine training data is shown in parentheses.

Task	Method	BLEU	BERTScore
Headline generation	Self-training	28.64	92.44
	ExtraPhrase	1.51	86.19
Document summarization	Self-training	19.91	90.02
	ExtraPhrase	5.89	87.33

Table 3: BLEU scores and F1 based BERTScores between genuine and pseudo training data.

the pseudo data with the sampled genuine data for training. For Self-training and Back-translation, we train neural encoder-decoders with the sampled 1K instances, and then apply them to the rest of training data for the pseudo data construction.

Table 2 shows the F1 based ROUGE scores of each method on the headline generation and document summarization tasks when we have a small amount of genuine training data. This table indicates that Back-translation and Self-training outperform Genuine only. These results are consistent with the result in Section 3.4. However, the performance improvement by Back-translation and Self-training are smaller compared to ExtraPhrase. These results show that Back-translation and Self-training tend to be ineffective when the amount of genuine training data is small (see appendix A).

For ExtraPhrase, it achieves significantly better performance than others in both tasks. Thus, ExtraPhrase is more effective when the amount of the genuine training data is small. The lowest parts of Table 2 shows the results of ExtraPhrase without paraphrasing for the ablation study. In ExtraPhrase w/o paraphrasing setting, we train the model with genuine and pseudo training data generated by ExtraPhrase without the paraphrasing step. Moreover, Extractive in these parts shows the ROUGE scores of summaries generated by the extractive summarization step. These parts indicate that ExtraPhrase outperforms the one without paraphrasing. Thus, we need the paraphrasing step to improve the quality of the pseudo training data, although the setting excluding paraphrasing significantly outperforms others. Moreover, ROUGE scores of Extractive are much lower than ones of ExtraPhrase. This result implies that we need to train a neural encoder-decoder by using the pseudo data as the training data to generate better abstractive summaries.

4.2 Diversity of Pseudo Summaries

We assume that our ExtraPhrase can generate more diverse summaries in comparison with the self-training approach. To verify this assumption, we compare pseudo summaries generated by Self-training and ExtraPhrase.

Table 3 shows BLEU scores (Papineni et al., 2002) between genuine summaries in each training data and generated pseudo summaries. In addition, this table also shows F1 based BERTScores (Zhang et al., 2020) of them as the indicator of semantic similarities. This table indicates that both BERTScores of Self-training and ExtraPhrase are remarkably high. This result implies that the generated summaries are semantically similar to genuine summaries. Thus, generated summaries are suitable as pseudo data semantically.

In contrast, the BLEU score of ExtraPhrase is much lower than one of Self-training. This result indicates that ExtraPhrase generates pseudo summaries that contain many different phrases from the genuine summaries in comparison with Self-training. Therefore, ExtraPhrase can generate

Task	Method	Training	Generation	Cost
Headline generation	Back-translation	256 H	7 H	333 USD
	Self-training	256 H	4 H	328 USD
	ExtraPhrase	–	7 H	12 USD
Document summarization	Back-translation	384 H	16 H	511 USD
	Self-training	320 H	8 H	417 USD
	ExtraPhrase	–	15 H	26 USD

Table 4: Cost on pseudo data generation using Amazon Elastic Compute Cloud (Amazon EC2). Consuming times are calculated in case of one GPU.

much more diverse summaries than Self-training.

5 Efficiency of Pseudo-data Generation

Our proposed ExtraPhrase does not require additional neural encoder-decoders such as the back-translation and self-training approaches. We discuss the advantage of this property.

Table 4 shows time required by each pseudo data construction method. This table also shows costs when we use Amazon EC2, which is a cloud computing service, to construct pseudo data. This table indicates that Back-translation and Self-training require much time to train their neural encoder-decoders. In contrast, for ExtraPhrase, we do not spend any time on such training. Therefore, ExtraPhrase is much more cost-efficient than others.

6 Related Work

Data Augmentation Back-translation and self-training are widely used techniques in data augmentation for sequence-to-sequence tasks (Sennrich et al., 2016a; Kiyono et al., 2019; Parida and Motlicek, 2019; He et al., 2020).

Sennrich et al. (2016a) proposed back-translation to augment training data for machine translation by translating monolingual data on the target side to generate source side pseudo data. Edunov et al. (2018) reported the effectiveness of the back-translation approach in large-scale monolingual settings for machine translation. In addition, Hoang et al. (2018) introduced an iterative version by repeatedly applying back-translation several times. Back-translation is an effective approach for machine translation but it is unrealistic to apply the approach to abstractive summarization.

In self-training, we train a model on genuine data and apply it to generate pseudo data. Zhang and Zong (2016) applied self-training to enlarge parallel corpus for neural machine translation. He et al. (2020) introduced noisy self-training that uses dropout as the noise while decoding in self-training. These studies reported the effectiveness of self-training but self-training is hard to generate diverse pseudo data (Gu et al., 2018).

Perturbation Using perturbation that is a small difference from genuine data can be regarded as data augmentation (Kobayashi, 2018). Takase and Kiyono (2021) investigated the performance of various perturbations including adversarial perturbations (Goodfellow et al., 2015), word dropout (Gal and Ghahramani, 2016), and word replacement on various sequence-to-sequence tasks. Since these perturbations are orthogonal to our ExtraPhrase, we can combine them with ours. In fact, Takase and Kiyono (2021) reported that simple perturbations such as word dropout are useful on pseudo data generated by back-translation.

7 Conclusion

This paper proposes a novel strategy, ExtraPhrase, to generate pseudo data for abstractive summarization tasks. ExtraPhrase consists of two steps: extractive summarization and paraphrasing. We obtain the important parts of an input by the extractive summarization, and then obtain diverse expressions by the paraphrasing. Experimental results indicate that ExtraPhrase is more effective than other pseudo data generation methods such as back-translation and self-training. Moreover, we show that ExtraPhrase is much more cost-efficient than others in pseudo data construction.

Acknowledgements

This work was partially supported by JSPS KAKENHI Grant Number 19H01118 and JP21K17800. These research results were obtained partially from the commissioned research (No. 225) by National Institute of Information and Communications Technology (NICT), Japan.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 1877–1901.

Isaac Caswell, Ciprian Chelba, and David Grangier. 2019. Tagged back-translation. In *Proceedings of the Fourth Conference on Machine Translation*, pages 53–63.

Trevor Anthony Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal Of Artificial Intelligence Research*, pages 637–674.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT) Text Summarization Workshop*, pages 1–8.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 489–500.

Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 360–368.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1481–1491.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems 29 (NIPS)*, pages 1019–1027.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations (ICLR)*.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *International Conference on Learning Representations (ICLR)*.

Tsutomu Hirao, Jun Suzuki, and Hideki Isozaki. 2009. A syntax-free approach to Japanese sentence compression. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 826–833.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Aizhan Imankulova, Takayuki Sato, and Mamoru Komachi. 2019. Filtered pseudo-parallel corpus improves low-resource neural machine translation. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, pages 1–16.

Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, and Masaaki Nagata. 2018. Higher-order syntactic attention network for longer sentence compression. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1716–1726.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 452–457.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 881–893.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation*, pages 314–319.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT) Demonstrations*, pages 48–53.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Shantipriya Parida and Petr Motlicek. 2019. Abstract text summarization: A low resource challenge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5994–5998.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 379–389.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1073–1083.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 86–96.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1715–1725.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual SMT system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 38–42.

Sho Takase and Shun Kiyono. 2021. Rethinking perturbations in encoder-decoders for fast training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 5767–5780.

Jenine Turner and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 290–297.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 5998–6008.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 380–386.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1535–1545.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations (ICLR)*.

Ratio	Difference	R-1	R-2	R-L
Headline generation				
0.86	-5	35.14	15.13	28.59
Document summarization				
0.81	-297	13.76	1.09	13.07

Table 5: F1 based ROUGE scores (R-1, 2, and L) between source texts generated by back-translation and genuine source texts. Ratio and Difference are comparisons between the number of tokens in generated source texts and genuine ones.

A Quality of Back-translation

As described in Section 1, the back-translation approach for the abstractive summarization task is essentially impossible because it requires restoring source texts from summaries without any additional information. Thus, we investigate the quality of source texts generated by Back-translation.

Table 5 shows the length difference and ratio between genuine and source text generated by Back-translation. This table indicates that the generated source texts are shorter than the original genuine data. This result implies that Back-translation fails to restore the full information in the genuine data. In other words, this result implies that it is difficult to generate source texts from summaries.

Table 5 also shows ROUGE scores of source texts generated by Back-translation when we regard the genuine source texts as the correct instances to investigate whether the generated texts correspond to the genuine data. For the document summarization, ROUGE scores are extremely low. This result also indicates that Back-translation fails to generate source texts.

On the other hand, ROUGE scores on the headline generation are much higher than ones on the document summarization. This result implies that Back-translation might restore the core parts of source texts from summaries. Because the headline generation is the task of generating a headline from a given sentence, the summary (headline) often contains the dominant part of the source sentence. We consider this property causes such high scores.