Abstract

Recently, we can obtain a practical abstractive document summarization model by fine-tuning a pre-trained language model (PLM). Since the pre-training for PLMs does not consider summarization-specific information such as the target summary length, there is a gap between the pre-training and fine-tuning for PLMs in summarization tasks. To fill the gap, we propose a method for enabling the model to understand the summarization-specific information by predicting the summary length in the encoder and generating a summary of the predicted length in the decoder in fine-tuning. Experimental results on the WikiHow, NYT, and CNN/DM datasets showed that our methods improve ROUGE scores from BART by generating summaries of appropriate lengths. Further, we observed about 3.0, 1.5, and 3.1 point improvements for ROUGE-1, -2, and -L, respectively, from GSum on the WikiHow dataset. Human evaluation results also showed that our methods improve the informativeness and conciseness of summaries.

1 Introduction

Current abstractive summarization models mostly utilize pre-trained language models (PLMs) (Liu and Lapata, 2019; Dou et al., 2021; Liu and Liu, 2021; Narayan et al., 2021; Liu et al., 2022a). Abstractive document summarization requires an encoder to determine the important parts in an input text and a decoder to output a non-redundant summary of the appropriate length relevant to the input. Thus, the characteristics required for an abstractive summarization model differ from those required as a language model, and are not usually considered in the pre-training for PLMs (Devlin et al., 2019; Zhang et al., 2019; Lewis et al., 2020). Hence, we need to fine-tune a PLM with a summarization dataset to treat it as an abstractive summarization model. Unlike training a randomly initialized model, this fine-tuning maintains and inherits the parameters learned as an original language model. Therefore, to learn an abstractive summarization model by fine-tuning a PLM, it is necessary to suppress its characteristics as a language model while enabling it to learn the unique properties of abstractive summarization.

For this purpose, we propose two regularization methods for fine-tuning a PLM to learn abstractive summarization. Figure 1 shows an overview of our methods. The first method is a regularization method that uses the encoder’s hidden states to predict the length of an output summary. When the length is not given for a summary to be generated, we believe it is difficult to determine what volume of important key contents to select from the original document. Thus, fixing the length for a summary can make it easier to select key contents for it. We think humans can also create more informative and concise summaries when a summary length is given. The system should also be better trained for selecting key contents in the original document for a summary in case when it can be provided with the length of the summary.

The second method provides the decoder with the length predicted by the first method and enables it to learn to output a summary of the length. In addition to regularizing the training of the decoder, this method reduces the search space by searching
only for summaries of the appropriate length during generation, and so it is expected to produce a concise and informative summary. Although there have been studies on adjusting the output length of summaries, they have focused on controlling the output length for a given desired length (Kikuchi et al., 2016; Liu et al., 2018; Takase and Okazaki, 2019; Makino et al., 2019; Saito et al., 2020; Yu et al., 2021).

We incorporate a target-length prediction task to the encoder side and then inject the predicted length to the decoder side to generate the final summary.

In an evaluation on the WikiHow, NYT, and CNN/DM datasets, our methods improve the ROUGE scores of BART with appropriate lengths of summaries. On the WikiHow dataset, the performance improvement reached about 3.0, 1.5, and 3.1 points for ROUGE-1, -2, and -L, respectively, from GSum. Human evaluation results also showed that our methods enable the fine-tuning for a PLM to generate informative and concise summaries.

Our contributions are as follows: (1) We propose a regularization method that uses the encoder’s hidden states by predicting the length of a summary. (2) We propose a regularization method that reduces the search space by injecting the predicted length of a summary. (3) Both automatic and human evaluation results show that our novel model that combines (1) and (2) can generate a summary closer to its gold summary length by improving informativeness.

2 Our Methods

We apply our regularization methods to a transformer-based (Vaswani et al., 2017) PLM to generate a summary from a given document.

2.1 Predicting Summary Length

We impose summary-length prediction on the encoder during fine-tuning, and so it is expected to produce a concise and informative summary. Although there have been studies on adjusting the output length of summaries, they have focused on controlling the output length for a given desired length (Kikuchi et al., 2016; Liu et al., 2018; Takase and Okazaki, 2019; Makino et al., 2019; Saito et al., 2020; Yu et al., 2021). We propose a regularization method that reduces the search space by injecting the predicted length of a summary. Note that we replace \( \ell_{\text{pred}} \) with \( \ell_{\text{gold}} \) in the decoder during training.

After that, by using the root-mean-square error (RMSE), the regularization loss for the encoder \( \mathcal{L}_{\text{len}} \) is calculated as follows:

\[
\mathcal{L}_{\text{len}} = \sqrt{(\ell_{\text{pred}} - \ell_{\text{gold}})^2},
\]

where \( \ell_{\text{gold}} \) is the gold length of the target summary.

2.2 Generating a Summary with the Predicted Length

We provide the decoder with the predicted summary length to generate a concise summary of the appropriate length relevant to the given document.

To encode the information of the predicted length into the decoder while keeping its pre-trained information, we insert our Length-Fusion Positional Encoding layer (LFPE), which is a transformer layer, before the decoder. Our LFPE consists of the length-ratio positional encoding (LRPE) (Takase and Okazaki, 2019) and a transformer layer. LRPE converts the position information of an output token \( y_t \) at time \( t \) to a continuous vector \( p_t \) with considering the predicted length \( \ell_{\text{pred}} \) as follows:

\[
p_t = \begin{cases} 
\sin(t/\ell_{\text{pred}}) & (i \equiv 0 \pmod{2}) \\
\cos(t/\ell_{\text{pred}}) & (i \equiv 1 \pmod{2})
\end{cases}
\]

where \( \text{dim} \) is the dimension size of the embedding.

Then, the transformer layer converts \( \{p_1, p_2, ..., p_t\} \) into \( E_t = \{e_1, e_2, ..., e_t\} \) at a decoding time-step \( t \). When adopting LFPE, we replace the original sinusoidal positional encoding of the pre-trained decoder with \( E_t \). After that, the decoder calculates the output probability of \( y_t \) as \( P(y_t|y_{t-1}, \cdots, y_1, x, \ell_{\text{pred}}) \).

Finally, the regularization loss for the decoder \( \mathcal{L}_{\text{gen}} \) is calculated as follows:

\[
\mathcal{L}_{\text{gen}} = -\sum_{t=1}^{m} \log P(y_t|y_{t-1}, \cdots, y_1, x, \ell_{\text{pred}}),
\]

where \( m \) is the number of tokens in the target summary. Note that we replace \( \ell_{\text{pred}} \) with \( \ell_{\text{gold}} \) in the decoder during training.
### Table 1: Statistics of document summarization datasets.
The value in parentheses indicates the variance of target summary lengths.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Training</th>
<th>Valid</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>WikiHow</td>
<td>168,126</td>
<td>6,000</td>
<td>6,000</td>
</tr>
<tr>
<td>NYT</td>
<td>44,382</td>
<td>5,523</td>
<td>6,495</td>
</tr>
<tr>
<td>CNN/DM</td>
<td>287,084</td>
<td>13,367</td>
<td>11,490</td>
</tr>
</tbody>
</table>

Dataset Training Valid Test
WikiHow 168,126 (47.2) 6,000 (51.2) 6,000 (54.4)
NYT 44,382 (28.9) 5,523 (31.2) 6,495 (30.9)
CNN/DM 287,084 (20.5) 13,367 (25.1) 11,490 (22.0)

Table 2: Experimental results on WikiHow, NYT, and CNN/DM. † indicates the improvement is significant (p<0.05) compared with the best baseline score (underlined) on each dataset. * indicates the reported score in the original paper. AVG indicates the average generated summary length.

<table>
<thead>
<tr>
<th>Model</th>
<th>WikiHow</th>
<th>NYT</th>
<th>CNN/DM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEGASUS*</td>
<td>43.06</td>
<td>47.36</td>
<td>57.50</td>
</tr>
<tr>
<td>GSum*</td>
<td>41.74</td>
<td>41.02</td>
<td>42.05</td>
</tr>
<tr>
<td>BART</td>
<td>42.04</td>
<td>41.98</td>
<td>45.50</td>
</tr>
<tr>
<td>BART w/ R_{enc}</td>
<td>44.68†</td>
<td>43.31†</td>
<td>55.12</td>
</tr>
<tr>
<td>BART w/ R_{enc+dec}</td>
<td>45.02†</td>
<td>43.56†</td>
<td>54.44</td>
</tr>
</tbody>
</table>

Table 2: Experimental results on WikiHow, NYT, and CNN/DM. † indicates the improvement is significant (p<0.05) compared with the best baseline score (underlined) on each dataset. * indicates the reported score in the original paper. AVG indicates the average generated summary length.

#### 2.3 Objective Function
To balance the encoder and decoder regularization, we sum the two losses through a hyperparameter \( \lambda \) for calculating the final loss as follows:

\[
\mathcal{L} = \mathcal{L}_{gen} + \lambda \cdot \mathcal{L}_{len}.
\]

#### 3 Experiments

##### 3.1 Experimental Settings

**Datasets**: We used WikiHow (Koupaee and Wang, 2018) in the knowledge base domain and NYT\(^2\) (Sandhaus, 2008) and CNN/DM (Hermann et al., 2015) in the news domain. Table 1 shows the dataset statistics.

**Evaluation Metrics**: We used F-scores of ROUGE-1 (R-1), -2 (R-2), and -L (R-L) in our experiments. To evaluate the quality of the predicted length and the length-controllability, we employed the length variance (VAR): \( \text{VAR} = 0.001 \times \frac{1}{n} \sum_{i=1}^{n} |y_{\text{pred}} - y_{\text{gold}}| \), where \( y_{\text{pred}} \) is the length of the generated summary and \( y_{\text{gold}} \) is the length of the reference summary in word level, respectively.

**Compared Methods**: We used BART-large (Lewis et al., 2020) for constructing baselines and our models by following the previous work (Dou et al., 2021). The proposed models are as follows. **BART w/ R_{enc}** employs our method only for the encoder in §2.1. **BART w/ R_{enc+dec}** employs our methods both for the encoder and the decoder. The baseline models are as follows. **BART** and **PEGASUS** (Zhang et al., 2019) are the original pre-trained BART and PEGASUS. **GSum** (Dou et al., 2021) is a BART-based combination model that utilizes extracted sentences as a guidance signal to consider extractive aspects for a summary. For the guidance signal, it uses the MatchSum model (Zhong et al., 2020).

We followed the hyperparameters of **BART** and **GSUM** for training and testing the baselines and our models. We set \( \lambda \) to 0.1, 0.05, and 0.05 for WikiHow, NYT, and CNN/DM, respectively, on the basis of validation performances.\(^3\)

##### 3.2 Automatic Evaluation

The results are shown in Table 2. We can see that both of our models, **BART w/ R_{enc}** and **BART w/ R_{enc+dec}**, showed significant improvement in ROUGE scores over BART on WikiHow. These scores were higher than the combination model of GSUM and PEGASUS (Zhang et al., 2019), which yields the current best results reported on WikiHow. We analyzed relations between lengths and ROUGE scores. When our **BART w/ R_{enc+dec}** predicted summary lengths closer to gold summary lengths than BART, 95.4% of generated summaries from ours obtained higher R-1 scores than BART. In addition, VAR and AVG scores show that our models can generate summaries closer to the gold summary lengths and can actually reduce the search space in decoding steps. These results indicate that the proposed methods enable BART to generate highly abstractive summaries of appropriate lengths.

We can also confirm that the proposed methods improved summarization performance over BART on NYT\(^4\) and CNN/DM. We can also see that...

---

\(^2\) Detailed pre-processing steps are described in Appendix A.

\(^3\) Further details are described in Appendix B.

\(^4\) There is no reported result for PEGASUS on NYT. For GSum, since the pre-processing could not be made identical, the reported and our scores were a bit different.
our model BART w/ $\text{R}_{\text{enc+dec}}$ showed significant improvement in ROUGE scores over GSUM on NYT. Although GSUM outperformed our BART w/ $\text{R}_{\text{enc+dec}}$ in ROUGE scores on CNN/DM, it could generate summaries closer to the gold summary lengths.

Thus, we tried to investigate what types of datasets our methods can work better on and found that the variance of reference summary lengths might be related to the performance of our models. Based on the observations from Tables 1 and 2, our BART w/ $\text{R}_{\text{enc+dec}}$ can largely improve performances on summarization datasets with a high variance of summary lengths, such as WikiHow and NYT.

### 3.3 Human Evaluation

For human evaluation, we sampled 100 documents each from WikiHow and CNN/DM. By using Amazon Mechanical Turk, we assigned 40 evaluators who obtained both US high school and US bachelor’s degrees to each dataset for grading the results with scores from 1 to 5 (5 is the best) in terms of informativeness (Info) and conciseness (Con).

Table 3 shows the results. These results indicate that BART w/ $\text{R}_{\text{enc+dec}}$ generated more informative summaries than BART, that is consistent with the results from the automatic evaluation. In some cases, the generated summaries with BART are just short summaries on WikiHow due to a high variance of reference summary lengths, and so the Con score is slightly lower than the one for BART w/ $\text{R}_{\text{enc+dec}}$. However, BART w/ $\text{R}_{\text{enc+dec}}$ yields the best overall Info and Con scores, which shows our regularization methods are essential for fine-tuning a PLM to learn abstractive summarization models. We also evaluated GSUM together. BART attained a 0.01 better score for Info than GSUM even on CNN/DM since GSUM focuses on generating faithful summaries with injecting outputs from an extractive summarization model.

We investigated the tendency of the length of generated summaries. Figure 2 shows the relation-ship between gold and generated summary lengths for each model. We used WikiHow because it contains various target summary lengths. When we injected the gold summary length, the length of generated summaries from LFPE (Gold) was almost the same as the gold summaries. These results indicate that LFPE can precisely control various output lengths. In addition, generated summary lengths from BART w/ $\text{R}_{\text{enc+dec}}$ show that the length-prediction layer can also predict various target summary lengths.

Table 4 shows example generated summaries from BART w/ $\text{R}_{\text{enc+dec}}$, BART, and gold summaries on WikiHow.

### 4 Related Work

In summary length control, previous work mostly focuses on controlling models for generating summaries with a predefined length (Kikuchi et al.,...
2016; Liu et al., 2018; Takase and Okazaki, 2019; Makino et al., 2019; Saito et al., 2020; Yu et al., 2021). Our work is novel because it enables a model dynamically predicts the appropriate summary length from the input text without relying on any predefined length.

From the viewpoint of regularization, we can see such a regularization term like $L_{\text{len}}$ in recent works of summarization tasks. Kamigaito et al. (2018); Kamigaito and Okumura (2020) in sentence compression and Ishigaki et al. (2019) in extractive document summarization incorporate dependency tree information into the attention (Kamigaito et al., 2017). Hsu et al. (2018) integrate extractive and abstractive summarization. MatchSum (Zhong et al., 2020) considers the semantic similarity between a document and its extracted summary. BRIO (Liu et al., 2022a) takes multiple similar abstractive summaries into account by contrastive learning in sequence-to-sequence (Edunov et al., 2018). Different from these works, our approach focuses on summary lengths through $L_{\text{len}}$ and can be incorporated into these works by adding $L_{\text{len}}$ to their loss function.

5 Conclusion

To fine-tune a pre-trained language model for abstractive document summarization, we proposed a regularization method that uses the encoder’s hidden states to predict the length of an output summary. We also proposed LFPE, that focuses on generating a summary with a given target length while keeping pre-trained information of the transformer-based model. We used LFPE to regularize the decoder during training to generate a summary with the predicted length.

Automatic evaluation results showed that the proposed methods enable BART to generate summaries of appropriate lengths while improving ROUGE scores. Human evaluation results also showed that the proposed methods enable BART to generate more informative and concise summaries.

6 Limitations

Although our models can largely improve performances on datasets with a high variance of summary lengths, the gain was small for datasets with a low variance of summary lengths. In the future, we will consider external resources to predict a summary length for the datasets with a low variance of target summary lengths. We plan to form document clusters based on each topic since different topics may have different reference since this may improve performances for the datasets with a low variance of summary lengths.

Acknowledgements

We would like to gratefully acknowledge the anonymous reviewers for their helpful comments and feedbacks. This work was supported by Google AI Focused Research Award.

References


Tatsuya Ishigaki, Hidetaka Kamigaito, Hiroya Takamura, and Manabu Okumura. 2019. Discourse-aware


Statistics of the datasets

NYT dataset consists of articles from the New York Times and the associated summaries. We followed the previous preprocessing step and splitting (Kedzie et al., 2018). There are two types of the reference summaries, which are archival abstracts and online teaser means. From this collection, we take all articles that have a concatenated summary length of at least 100 words.

Model details

We introduce the detailed information of the baseline and our models.

We used Fairseq (Ott et al., 2019) for the model implementation. As the pretrained weight, we used bart-large in huggingface. We used the original implementation for GSum. We ran training for the models on two NVIDIA Tesla V100 with the multi-GPU setting. As described in the experimental settings, all hyperparameters were the same as for the large-scale BART in Lewis et al. (2020). Hyperparameter $\lambda$ was set to 0.1, 0.05 and 0.05 for the WikiHow, CNN/DM, and NYT datasets, respectively, on the basis of validation performances.

Length-controllability

We investigated the length-controllability of our LFPE in §2.2 by comparing it with the original BART and LRPE. We also compared these methods with the previously reported scores of GOLC, PALUS, LPAS, and PtLAAM. We used CNN/DM and gave the gold summary length to the models by following the previous work. The results in Table 5 show that LFPE outperformed other methods in terms of ROUGE scores and VAR. Thus, our LFPE can control the output summary length while keeping ROUGE scores and outperform the state-of-the-art length-controllable methods.

Next, we analyzed the effect of length-controllability in actually generated summaries in CNN/DM. Table 6 shows example generated summaries with injecting different lengths into LFPE. In this example, when there is no possibility of dropping a subword, our model paraphrases “10th” to “10” while maintaining the informativeness and grammaticality. From this observation, we can understand that our LFPE controls the summary length through subword-based paraphrasing, which is supported by the decoder’s ability of abstraction.

\begin{table}[H]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Model & R-1 & R-2 & R-L & VAR \\
\hline
GOLC* (Makino et al., 2019) & 38.27 & 16.22 & 34.99 & 5.13 \\
PALUS* (Yu et al., 2021) & 39.82 & 17.31 & 36.20 & 0.01 \\
LPAS* (Saito et al., 2020) & 43.23 & 20.46 & 40.00 & - \\
PtLAAM* (Liu et al., 2022b) & 44.17 & 20.63 & 40.97 & - \\
\hline
BART & 44.48 & 21.41 & 41.19 & 0.78 \\
LRPE (Takase and Okazaki, 2019) & 45.67 & 22.11 & 42.20 & - \\
LFPE (Ours) & 45.93$^\dagger$ & 22.30 & 42.44$^\dagger$ & 0.03 \\
\hline
\end{tabular}
\caption{Experimental results on CNN/DM with using the gold summary length information. The notations are the same as in Table 2.}
\end{table}

\begin{table}[H]
\centering
\begin{tabular}{c|c|c|c|}
\hline
\multicolumn{1}{c|}{\textbf{\emph{\Delta}}} & \multicolumn{1}{c|}{\textbf{Generated Summary}} & \multicolumn{1}{c|}{\textbf{Infrication}} & \multicolumn{1}{c}{\textbf{Length}} \\
\hline
$+1$ & She and her husband are celebrating their 10th wedding anniversary. & +1 & 10th \\
0 & She and her husband are celebrating their 10th anniversary. & 0 & 10th \\
$-1$ & She and her husband are now married 10 years. & -1 & 10th \\
\hline
\end{tabular}
\caption{Example summaries generated from BART with LFPE for different lengths on CNN/DM. $\Delta = +1/-1$ indicates the injected length is larger/smaller than the gold summary.}
\end{table}