Summary Level Training of Sentence Rewriting for Abstractive Summarization

Sanghwan Bae, Taeuk Kim, Jihoon Kim and **Sang-goo Lee**
Department of Computer Science and Engineering
Seoul National University, Seoul, Korea
{sanghwan,taeuk,kjh255,sglee}@europa.snu.ac.kr

Abstract

As an attempt to combine extractive and abstractive summarization, *Sentence Rewriting* models adopt the strategy of extracting salient sentences from a document first and then paraphrasing the selected ones to generate a summary. However, the existing models in this framework mostly rely on sentence-level rewards or suboptimal labels, causing a mismatch between a training objective and evaluation metric. In this paper, we present a novel training signal that directly maximizes summary-level ROUGE scores through reinforcement learning. In addition, we incorporate BERT into our model, making good use of its ability on natural language understanding. In extensive experiments, we show that a combination of our proposed model and training procedure obtains new state-of-the-art performance on both CNN/Daily Mail and New York Times datasets. We also demonstrate that it generalizes better on DUC-2002 test set.

1 Introduction

The task of automatic text summarization aims to compress a textual document to a shorter highlight while keeping salient information of the original text. In general, there are two ways to do text summarization: *Extractive* and *Abstractive* (Mani and Maybury, 2001). *Extractive* approaches generate summaries by selecting salient sentences or phrases from a source text, while *abstractive* approaches involve a process of paraphrasing or generating sentences to write a summary.

Recent work (Liu, 2019; Zhang et al., 2019c) demonstrates that it is highly beneficial for extractive summarization models to incorporate pre-trained language models (LMs) such as BERT (Devlin et al., 2019) into their architectures. However, the performance improvement from the pre-trained LMs is known to be relatively small in case of abstractive summarization (Zhang et al., 2019a; Hoang et al., 2019). This discrepancy may be due to the difference between extractive and abstractive approaches in ways of dealing with the task—the former *classifies* whether each sentence to be included in a summary, while the latter *generates* a whole summary from scratch. In other words, as most of the pre-trained LMs are designed to be of help to the tasks which can be categorized as classification including extractive summarization, they are not guaranteed to be advantageous to abstractive summarization models that should be capable of generating language (Wang and Cho, 2019; Zhang et al., 2019b).

On the other hand, recent studies for abstractive summarization (Chen and Bansal, 2018; Hsu et al., 2018; Gehrmann et al., 2018) have attempted to exploit extractive models. Among these, a notable one is Chen and Bansal (2018), in which a sophisticated model called *Reinforce-Selected Sentence Rewriting* is proposed. The model consists of both an extractor and abstractor, where the extractor picks out salient sentences first from a source article, and then the abstractor rewrites and compresses the extracted sentences into a complete summary. It is further fine-tuned by training the extractor with the rewards derived from sentence-level ROUGE scores of the summary generated from the abstractor.

In this paper, we improve the model of Chen and Bansal (2018), addressing two primary issues. Firstly, we argue there is a bottleneck in the existing extractor on the basis of the observation that its performance as an independent summarization model (i.e., without the abstractor) is no better than solid baselines such as selecting the first 3 sentences. To resolve the problem, we present a novel neural extractor exploiting the pre-trained LMs (BERT in this work) which are expected to perform better according to the recent studies (Liu,

Proceedings of the 2nd Workshop on New Frontiers in Summarization, pages 10–20
Hong Kong, China, November 4, 2019. ©2019 Association for Computational Linguistics

2019; Zhang et al., 2019c). Since the extractor is a sort of sentence classifier, we expect that it can make good use of the ability of pre-trained LMs which is proven to be effective in classification.

Secondly, the other point is that there is a mismatch between the training objective and evaluation metric; the previous work utilizes the *sentence-level* ROUGE scores as a reinforcement learning objective, while the final performance of a summarization model is evaluated by the *summary-level* ROUGE scores. Moreover, as Narayan et al. (2018) pointed out, sentences with the highest individual ROUGE scores do not necessarily lead to an optimal summary, since they may contain overlapping contents, causing verbose and redundant summaries. Therefore, we propose to directly use the summary-level ROUGE scores as an objective instead of the sentence-level scores. A potential problem arising from this apprsoach is the sparsity of training signals, because the summary-level ROUGE scores are calculated only once for each training episode. To alleviate this problem, we use *reward shaping* (Ng et al., 1999) to give an intermediate signal for each action, preserving the optimal policy.

We empirically demonstrate the superiority of our approach by achieving new state-of-the-art abstractive summarization results on CNN/Daily Mail and New York Times datasets (Hermann et al., 2015; Durrett et al., 2016). It is worth noting that our approach shows large improvements especially on ROUGE-L score which is considered a means of assessing fluency (Narayan et al., 2018). In addition, our model performs much better than previous work when testing on DUC-2002 dataset, showing better generalization and robustness of our model.

Our contributions in this work are three-fold: a novel successful application of pre-trained transformers for abstractive summarization; suggesting a training method to globally optimize sentence selection; achieving the state-of-the-art results on the benchmark datasets, CNN/Daily Mail and New York Times.

2 Background

2.1 Sentence Rewriting

In this paper, we focus on single-document multi-sentence summarization and propose a neural abstractive model based on the *Sentence Rewriting* framework (Chen and Bansal, 2018; Xu and Durrett, 2019) which consists of two parts: a neural network for the *extractor* and another network for the *abstractor*. The extractor network is designed to extract salient sentences from a source article. The abstractor network rewrites the extracted sentences into a short summary.

2.2 Learning Sentence Selection

The most common way to train extractor to select informative sentences is building extractive oracles as gold targets, and training with cross-entropy (CE) loss. An oracle consists of a set of sentences with the highest possible ROUGE scores. Building oracles is finding an optimal combination of sentences, where there are 2^n possible combinations for each example. Because of this, the exact optimization for ROUGE scores is intractable. Therefore, alternative methods identify the set of sentences with greedy search (Nallapati et al., 2017), sentence-level search (Hsu et al., 2018; Shi et al., 2019) or collective search using the limited number of sentences (Xu and Durrett, 2019), which construct suboptimal oracles. Even if all the optimal oracles are found, training with CE loss using these labels will cause underfitting as it will only maximize probabilities for sentences in label sets and ignore all other sentences.

Alternatively, reinforcement learning (RL) can give room for exploration in the search space. Chen and Bansal (2018), our baseline work, proposed to apply policy gradient methods to train an extractor. This approach makes an end-to-end trainable stochastic computation graph, encouraging the model to select sentences with high ROUGE scores. However, they define a reward for an action (sentence selection) as a sentence-level ROUGE score between the chosen sentence and a sentence in the ground truth summary for that time step. This leads the extractor agent to a suboptimal policy; the set of sentences matching individually with each sentence in a ground truth summary isn't necessarily optimal in terms of summary-level ROUGE score.

Narayan et al. (2018) proposed policy gradient with rewards from summary-level ROUGE. They defined an action as sampling a summary from candidate summaries that contain the limited number of plausible sentences. After training, a sentence is ranked high for selection if it often occurs in high scoring summaries. However, their approach still has a risk of ranking redundant sen-

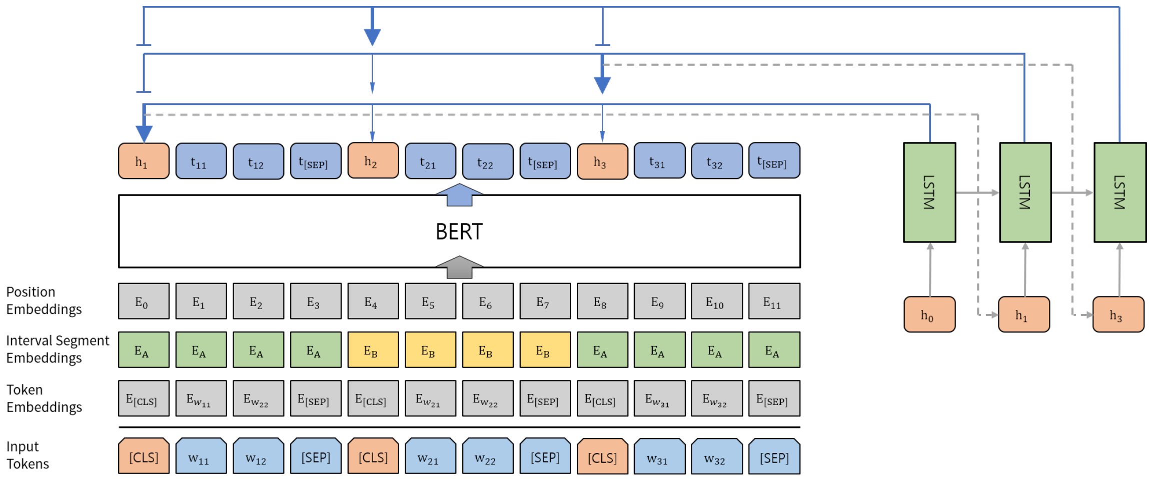

Figure 1: The overview architecture of the extractor netwrok

tences high; if two highly overlapped sentences have salient information, they would be ranked high together, increasing the probability of being sampled in one summary.

To tackle this problem, we propose a training method using reinforcement learning which globally optimizes summary-level ROUGE score and gives intermediate rewards to ease the learning.

2.3 Pre-trained Transformers

Transferring representations from pre-trained transformer language models has been highly successful in the domain of natural language understanding tasks (Radford et al., 2018; Devlin et al., 2019; Radford et al., 2019; Yang et al., 2019). These methods first pre-train highly stacked transformer blocks (Vaswani et al., 2017) on a huge unlabeled corpus, and then fine-tune the models or representations on downstream tasks.

3 Model

Our model consists of two neural network modules, i.e. an extractor and abstractor. The extractor encodes a source document and chooses sentences from the document, and then the abstractor paraphrases the summary candidates. Formally, a single document consists of n sentences $D = \{s_1, s_2, \cdots, s_n\}$. We denote i-th sentence as $s_i = \{w_{i1}, w_{i2}, \cdots, w_{im}\}$ where w_{ij} is the j-th word in s_i. The extractor learns to pick out a subset of D denoted as $\hat{D} = \{\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_k | \hat{s}_i \in D\}$ where k sentences are selected. The abstractor rewrites each of the selected sentences to form a summary $S = \{f(\hat{s}_1), f(\hat{s}_2), \cdots, f(\hat{s}_k)\}$, where f is an abstracting function. And a gold summary consists of l sentences $A = \{a_1, a_2, \cdots, a_l\}$.

3.1 Extractor Network

The extractor is based on the encoder-decoder framework. We adapt BERT for the encoder to exploit contextualized representations from pre-trained transformers. BERT as the encoder maps the input sequence D to sentence representation vectors $H = \{h_1, h_2, \cdots, h_n\}$, where h_i is for the i-th sentence in the document. Then, the decoder utilizes H to extract $\hat{D}$ from D.

3.1.1 Leveraging Pre-trained Transformers

Although we require the encoder to output the representation for each sentence, the output vectors from BERT are grounded to tokens instead of sentences. Therefore, we modify the input sequence and embeddings of BERT as Liu (2019) did.

In the original BERT's configure, a [CLS] token is used to get features from one sentence or a pair of sentences. Since we need a symbol for each sentence representation, we insert the [CLS] token before each sentence. And we add a [SEP] token at the end of each sentence, which is used to differentiate multiple sentences. As a result, the vector for the i-th [CLS] symbol from the top BERT layer corresponds to the i-th sentence representation h_i.

In addition, we add interval segment embeddings as input for BERT to distinguish multiple sentences within a document. For s_i we assign a segment embedding E_A or E_B conditioned on i is odd or even. For example, for a consecutive sequence of sentences s_1, s_2, s_3, s_4, s_5, we assign E_A, E_B, E_A, E_B, E_A in order. All the words in each sentence are assigned to the same segment embedding, *i.e.* segment embeddings for $w_{11}, w_{12}, \cdots, w_{1m}$ is $E_A, E_A, \cdots, E_A$. An illustration for this procedure is shown in Figure 1.

3.1.2 Sentence Selection

We use LSTM Pointer Network (Vinyals et al., 2015) as the decoder to select the extracted sentences based on the above sentence representations. The decoder extracts sentences recurrently, producing a distribution over all of the remaining sentence representations excluding those already selected. Since we use the sequential model which selects one sentence at a time step, our decoder can consider the previously selected sentences. This property is needed to avoid selecting sentences that have overlapping information with the sentences extracted already.

As the decoder structure is almost the same with the previous work, we convey the equations of Chen and Bansal (2018) to avoid confusion, with minor modifications to agree with our notations. Formally, the extraction probability is calculated as:

$$u_{t,i} = v_m^\top \tanh(W_e e_t + W_h h_i) \quad (1)$$

$$P(\hat{s}_t|D, \hat{s}_1, \cdots, \hat{s}_{t-1}) = \text{softmax}(u_t) \quad (2)$$

where e_t is the output of the glimpse operation:

$$c_{t,i} = v_g^\top \tanh(W_{g1} h_i + W_{g2} z_t) \quad (3)$$

$$\alpha_t = \text{softmax}(c_t) \quad (4)$$

$$e_t = \sum_i \alpha_t W_{g1} h_i \quad (5)$$

In Equation 3, z_t is the hidden state of the LSTM decoder at time t (shown in green in Figure 1). All the W and v are trainable parameters.

3.2 Abstractor Network

The abstractor network approximates f, which compresses and paraphrases an extracted document sentence to a concise summary sentence. We use the standard attention based sequence-to-sequence (seq2seq) model (Bahdanau et al., 2015; Luong et al., 2015) with the copying mechanism (See et al., 2017) for handling out-of-vocabulary (OOV) words. Our abstractor is practically identical to the one proposed in Chen and Bansal (2018).

4 Training

In our model, an extractor selects a series of sentences, and then an abstractor paraphrases them. As they work in different ways, we need different training strategies suitable for each of them. Training the abstractor is relatively obvious; maximizing log-likelihood for the next word given the previous ground truth words. However, there are several issues for extractor training. First, the extractor should consider the abstractor's rewriting process when it selects sentences. This causes a *weak supervision* problem (Jehl et al., 2019), since the extractor gets training signals indirectly after paraphrasing processes are finished. In addition, thus this procedure contains sampling or maximum selection, the extractor performs a non-differentiable extraction. Lastly, although our goal is maximizing ROUGE scores, neural models cannot be trained directly by maximum likelihood estimation from them.

To address those issues above, we apply standard policy gradient methods, and we propose a novel training procedure for extractor which guides to the optimal policy in terms of the summary-level ROUGE. As usual in RL for sequence prediction, we pre-train submodules and apply RL to fine-tune the extractor.

4.1 Training Submodules

Extractor Pre-training Starting from a poor random policy makes it difficult to train the extractor agent to converge towards the optimal policy. Thus, we pre-train the network using cross entropy (CE) loss like previous work (Bahdanau et al., 2017; Chen and Bansal, 2018). However, there is no gold label for extractive summarization in most of the summarization datasets. Hence, we employ a greedy approach (Nallapati et al., 2017) to make the extractive oracles, where we add one sentence at a time incrementally to the summary, such that the ROUGE score of the current set of selected sentences is maximized for the entire ground truth summary. This doesn't guarantee optimal, but it is enough to teach the network to select plausible sentences. Formally, the network is trained to minimize the cross-entropy loss as follows:

$$L_{\text{ext}} = -\frac{1}{T}\sum_{t=1}^{T} \log P(s_t^*|D, s_1^*, \cdots, s_{t-1}^*) \quad (6)$$

where s_t^* is the t-th generated oracle sentence.

Abstractor Training For the abstractor training, we should create training pairs for input and target sentences. As the abstractor paraphrases on sentence-level, we take a sentence-level search for each ground-truth summary sentence. We find the most similar document sentence s'_t by:

$$s'_t = \text{argmax}_{s_i}(\text{ROUGE-L}^{\text{sent}}_{F_1}(s_i, a_t)) \tag{7}$$

And then the abstractor is trained as a usual sequence-to-sequence model to minimize the cross-entropy loss:

$$L_{\text{abs}} = -\frac{1}{m}\sum_{j=1}^{m} \log P(w^a_j | w^a_1, \cdots, w^a_{j-1}, \Phi) \tag{8}$$

where w^a_j is the j-th word of the target sentence a_t, and Φ is the encoded representation for s'_t.

4.2 Guiding to the Optimal Policy

To optimize ROUGE metric directly, we assume the extractor as an agent in reinforcement learning paradigm (Sutton et al., 1998). We view the extractor has a stochastic *policy* that generates *actions* (sentence selection) and receives the score of final evaluation metric (summary-level ROUGE in our case) as the *return*

$$R(S) = \text{ROUGE-L}^{\text{summ}}_{F_1}(S, A). \tag{9}$$

While we are ultimately interested in the maximization of the score of a complete summary, simply awarding this score at the last step provides a very sparse training signal. For this reason we define intermediate rewards using *reward shaping* (Ng et al., 1999), which is inspired by Bahdanau et al. (2017)'s attempt for sequence prediction. Namely, we compute summary-level score values for all intermediate summaries:

$$(R(\{\hat{s}_1\}), R(\{\hat{s}_1, \hat{s}_2\}), \cdots, R(\{\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_k\})) \tag{10}$$

The reward for each step r_t is the difference between the consecutive pairs of scores:

$$r_t = R(\{\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_t\}) - R(\{\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_{t-1}\}) \tag{11}$$

This measures an amount of increase or decrease in the summary-level score from selecting $\hat{s}_t$. Using the shaped reward r_t instead of awarding the whole score R at the last step does not change the optimal policy (Ng et al., 1999). We define a discounted future reward for each step as $R_t = \sum_{t=1}^{k} \gamma^t r_{t+1}$, where γ is a discount factor.

Additionally, we add 'stop' action to the action space, by concatenating trainable parameters h_{stop} (the same dimension as h_i) to H. The agent treats it as another candidate to extract. When it selects 'stop', an extracting episode ends and the final return is given. This encourages the model to extract additional sentences only when they are expected to increase the final return.

Following Chen and Bansal (2018), we use the *Advantage Actor Critic* (Mnih et al., 2016) method to train. We add a critic network to estimate a value function $V_t(D, \hat{s}_1, \cdots, \hat{s}_{t-1})$, which then is used to compute advantage of each action (we will omit the current state $(D, \hat{s}_1, \cdots, \hat{s}_{t-1})$ to simplify):

$$A_t(s_i) = Q_t(s_i) - V_t. \tag{12}$$

where $Q_t(s_i)$ is the expected future reward for selecting s_i at the current step t. We maximize this advantage with the policy gradient with the Monte-Carlo sample ($A_t(s_i) \approx R_t - V_t$):

$$\nabla_{\theta_\pi} L_\pi \approx \tfrac{1}{k}\sum_{t=1}^{k} \nabla_{\theta_\pi} \log P(s_i | D, \hat{s}_1, \cdots, \hat{s}_{t-1}) A_t(s_i) \tag{13}$$

where θ_π is the trainable parameters of the actor network (original extractor). And the critic is trained to minimize the square loss:

$$\nabla_{\theta_\psi} L_\psi = \nabla_{\theta_\psi}(V_t - R_t)^2 \tag{14}$$

where θ_ψ is the trainable parameters of the critic network.

5 Experimental Setup

5.1 Datasets

We evaluate the proposed approach on the CNN/Daily Mail (Hermann et al., 2015) and New York Times (Sandhaus, 2008) dataset, which are both standard corpora for multi-sentence abstractive summarization. Additionally, we test generalization of our model on DUC-2002 test set.

CNN/Daily Mail dataset consists of more than 300K news articles and each of them is paired with several highlights. We used the standard splits of Hermann et al. (2015) for training, validation and testing (90,226/1,220/1,093 documents for CNN and 196,961/12,148/10,397 for Daily Mail). We did not anonymize entities. We followed the preprocessing methods in See et al. (2017) after splitting sentences by Stanford CoreNLP (Manning et al., 2014).

The New York Times dataset also consists of many news articles. We followed the dataset splits of Durrett et al. (2016); 100,834 for training and

Models	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**R-AVG**
Extractive				
lead-3 (See et al., 2017)	40.34	17.70	36.57	31.54
REFRESH (Narayan et al., 2018)	40.00	18.20	36.60	31.60
JECS (Xu and Durrett, 2019)	41.70	18.50	37.90	32.70
HiBERT (Zhang et al., 2019c)	42.37	19.95	38.83	33.71
BERTSUM (Liu, 2019)	**43.25**	**20.24**	39.63	**34.37**
BERT-ext (ours)	42.29	19.38	38.63	33.43
BERT-ext + RL (ours)	42.76	19.87	39.11	33.91
Abstractive				
Pointer Generator (See et al., 2017)	39.53	17.28	36.38	31.06
Inconsistency Loss (Hsu et al., 2018)	40.68	17.97	37.13	31.93
Sentence Rewrite (w/o rerank) (Chen and Bansal, 2018)	40.04	17.61	37.59	31.74
Sentence Rewrite (Chen and Bansal, 2018)	40.88	17.80	38.54	32.41
Bottom-Up (Gehrmann et al., 2018)	41.22	18.68	38.34	32.75
Transformer-LM (Hoang et al., 2019)	38.67	17.47	35.79	30.64
Two-Stage BERT (Zhang et al., 2019a)	41.71	**19.49**	38.79	33.33
BERT-ext + abs (ours)	40.14	17.87	37.83	31.95
BERT-ext + abs + rerank (ours)	40.71	17.92	38.51	32.38
BERT-ext + abs + RL (ours)	41.00	18.81	38.51	32.77
BERT-ext + abs + RL + rerank (ours)	**41.90**	19.08	**39.64**	**33.54**

Table 1: Performance on CNN/Daily Mail test set using the full length ROUGE F_1 score. R-AVG calculates average score of ROUGE-1, ROUGE-2 and ROUGE-L.

9,706 for test examples. And we also followed the filtering procedure of them, removing documents with summaries that are shorter than 50 words. The final test set (NYT50) contains 3,452 examples out of the original 9,706.

The DUC-2002 dataset contains 567 document-summary pairs for single-document summarization. As a single document can have multiple summaries, we made one pair per summary. We used this dataset as a test set for our model trained on CNN/Daily Mail dataset to test generalization.

5.2 Implementation Details

Our extractor is built on $\text{BERT}_{\text{BASE}}$ with fine-tuning, smaller version than $\text{BERT}_{\text{LARGE}}$ due to limitation of time and space. We set LSTM hidden size as 256 for all of our models. To initialize word embeddings for our abstractor, we use word2vec (Mikolov et al., 2013) of 128 dimensions trained on the same corpus. We optimize our model with Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For extractor pre-training, we use learning rate schedule following (Vaswani et al., 2017) with $warmup = 10000$:

$$lr = 2e^{-3} \cdot \min(steps^{-0.5}, steps \cdot warmup^{-1.5}).$$

Models	**R-1**	**R-2**	**R-L**
lead-3 (See et al., 2017)	40.34	17.70	36.57
rnn-ext (Chen and Bansal, 2018)	40.17	18.11	36.41
JECS-ext (Xu and Durrett, 2019)	40.70	18.00	36.80
BERT-ext (ours)	**42.29**	**19.38**	**38.63**

Table 2: Comparison of extractor networks.

And we set learning rate $1e^{-3}$ for abstractor and $4e^{-6}$ for RL training. We apply gradient clipping using L2 norm with threshold 2.0. For RL training, we use $\gamma = 0.95$ for the discount factor. To ease learning h_{stop}, we set the reward for the stop action to $\lambda \cdot \text{ROUGE-L}^{\text{summ}}_{F_1}(S, A)$, where λ is a stop coefficient set to 0.08. Our critic network shares the encoder with the actor (extractor) and has the same architecture with it except the output layer, estimating scalar for the state value. And the critic is initialized with the parameters of the pre-trained extractor where it has the same architecture.

5.3 Evaluation

We evaluate the performance of our method using different variants of ROUGE metric computed with respect to the gold summaries. On the CNN/Daily Mail and DUC-2002 dataset, we use standard ROUGE-1, ROUGE-2, and ROUGE-

	R-1	R-2	R-L
Sentence-matching	52.09	28.13	49.74
Greedy Search	55.27	29.24	52.64
Combination Search	55.51	29.33	52.89

Table 3: Comparison of different methods building upper bound for full model.

Models	R-1	R-2	R-L
Sentence-level Reward	40.82	18.63	38.41
Combinatorial Reward	40.85	18.77	38.44
Sentence-level Reward + rerank	41.58	18.72	39.31
Combinatorial Reward + rerank	**41.90**	**19.08**	**39.64**

Table 4: Comparison of RL training.

L (Lin, 2004) on full length F_1 with stemming as previous work did (Nallapati et al., 2017; See et al., 2017; Chen and Bansal, 2018). On NYT50 dataset, following Durrett et al. (2016) and Paulus et al. (2018), we used the limited length ROUGE recall metric, truncating the generated summary to the length of the ground truth summary.

6 Results

6.1 CNN/Daily Mail

Table 1 shows the experimental results on CNN/Daily Mail dataset, with extractive models in the top block and abstractive models in the bottom block. For comparison, we list the performance of many recent approaches with ours.

Extractive Summarization As See et al. (2017) showed, the first 3 sentences (lead-3) in an article form a strong summarization baseline in CNN/Daily Mail dataset. Therefore, the very first objective of extractive models is to outperform the simple method which always returns 3 or 4 sentences at the top. However, as Table 2 shows, ROUGE scores of lead baselines and extractors from previous work in *Sentence Rewrite* framework (Chen and Bansal, 2018; Xu and Durrett, 2019) are almost tie. We can easily conjecture that the limited performances of their full model are due to their extractor networks. Our extractor network with BERT (BERT-ext), as a single model, outperforms those models with large margins. Adding reinforcement learning (BERT-ext + RL) gives higher performance, which is competitive with other extractive approaches using pre-trained Transformers (see Table 1). This shows the effectiveness of our learning method.

Abstractive Summarization Our abstractive approaches combine the extractor with the abstractor. The combined model (BERT-ext + abs) without additional RL training outperforms the Sentence Rewrite model (Chen and Bansal, 2018) without reranking, showing the effectiveness of our extractor network. With the proposed RL training procedure (BERT-ext + abs + RL), our model exceeds the best model of Chen and Bansal (2018). In addition, the result is better than those of all the other abstractive methods exploiting extractive approaches in them (Hsu et al., 2018; Chen and Bansal, 2018; Gehrmann et al., 2018).

Redundancy Control Although the proposed RL training inherently gives training signals that induce the model to avoid redundancy across sentences, there can be still remaining overlaps between extracted sentences. We found that the additional methods reducing redundancies can improve the summarization quality, especially on CNN/Daily Mail dataset.

We tried Trigram Blocking (Liu, 2019) for extractor and Reranking (Chen and Bansal, 2018) for abstractor, and we empirically found that the reranking only improves the performance. This helps the model to compress the extracted sentences focusing on disjoint information, even if there are some partial overlaps between the sentences. Our best abstractive model (BERT-ext + abs + RL + rerank) achieves the new state-of-the-art performance for abstractive summarization in terms of average ROUGE score, with large margins on ROUGE-L.

However, we empirically found that the reranking method has no effect or has negative effect on NYT50 or DUC-2002 dataset. Hence, we don't apply it for the remaining datasets.

Combinatorial Reward Before seeing the effects of our summary-level rewards on final results, we check the upper bounds of different training signals for the full model. All the document sentences are paraphrased with our trained abstractor, and then we find the best set for each search method. *Sentence-matching* finds sentences with the highest ROUGE-L score for each sentence in the gold summary. This search method matches with the best reward from Chen and Bansal (2018). *Greedy Search* is the same method explained for extractor pre-training in section 4.1. *Combination Search* selects a set of sentences

Models	Relevance	Readability	Total
Sentence Rewrite (Chen and Bansal, 2018)	56	59	115
BERTSUM (Liu, 2019)	58	60	118
BERT-ext + abs + RL + rerank (ours)	**66**	**61**	**127**

Table 5: Results of human evaluation.

which has the highest summary-level ROUGE-L score, from all the possible combinations of sentences. Due to time constraints, we limited the maximum number of sentences to 5. This method corresponds to our final return in RL training.

Table 3 shows the summary-level ROUGE scores of previously explained methods. We see considerable gaps between Sentence-matching and Greedy Search, while the scores of Greedy Search are close to those of Combination Search. Note that since we limited the number of sentences for Combination Search, the exact scores for it would be higher. The scores can be interpreted to be upper bounds for corresponding training methods. This result supports our training strategy; pretraining with Greedy Search and final optimization with the combinatorial return.

Additionally, we experiment to verify the contribution of our training method. We train the same model with different training signals; Sentence-level reward from Chen and Bansal (2018) and combinatorial reward from ours. The results are shown in Table 4. Both with and without reranking, the models trained with the combinatorial reward consistently outperform those trained with the sentence-level reward.

Human Evaluation We also conduct human evaluation to ensure robustness of our training procedure. We measure relevance and readability of the summaries. Relevance is based on the summary containing important, salient information from the input article, being correct by avoiding contradictory/unrelated information, and avoiding repeated/redundant information. Readability is based on the summarys fluency, grammaticality, and coherence. To evaluate both these criteria, we design a Amazon Mechanical Turk experiment based on ranking method, inspired by Kiritchenko and Mohammad (2017). We randomly select 20 samples from the CNN/Daily Mail test set and ask the human testers (3 for each sample) to rank summaries (for relevance and readability) produced by 3 different models: our final model, that of Chen and Bansal (2018) and that of Liu (2019). 2, 1 and 0 points were given according to the ranking. The models were anonymized and randomly shuffled. Following previous work, the input article and ground truth summaries are also shown to the human participants in addition to the three model summaries. From the results shown in Table 5, we can see that our model is better in relevance compared to others. In terms of readability, there was no noticeable difference.

Models	R-1	R-2	R-L
Extractive			
First sentences (Durrett et al., 2016)	28.60	17.30	-
First k words (Durrett et al., 2016)	35.70	21.60	-
Full (Durrett et al., 2016)	42.20	24.90	-
BERTSUM (Liu, 2019)	**46.66**	26.35	42.62
Abstractive			
Deep Reinforced (Paulus et al., 2018)	42.94	26.02	-
Two-Stage BERT (Zhang et al., 2019a)	45.33	26.53	-
BERT-ext + abs (ours)	44.41	24.61	41.40
BERT-ext + abs + RL (ours)	**46.63**	**26.76**	**43.38**

Table 6: Performance on NYT50 test set using the limited length ROUGE recall score.

Models	R-1	R-2	R-L
Pointer Generator (See et al., 2017)	37.22	15.78	33.90
Sentence Rewrite (Chen and Bansal, 2018)	39.46	17.34	36.72
BERT-ext + abs + RL (ours)	**43.39**	**19.38**	**40.14**

Table 7: Performance on DUC-2002 test set using the full length ROUGE F_1 score.

6.2 New York Times corpus

Table 6 gives the results on NYT50 dataset. We see our BERT-ext + abs + RL outperforms all the extractive and abstractive models, except ROUGE-1 from Liu (2019). Comparing with two recent models that adapted BERT on their summarization models (Liu, 2019; Zhang et al., 2019a), we can say that we proposed another method successfully leveraging BERT for summarization. In addition, the experiment proves the effectiveness of our RL training, with about 2 point improvement for each ROUGE metric.

6.3 DUC-2002

We also evaluated the models trained on the CNN/Daily Mail dataset on the out-of-domain DUC-2002 test set as shown in Table 7. BERT-ext + abs + RL outperforms baseline models with large margins on all of the ROUGE scores. This result shows that our model generalizes better.

7 Related Work

There has been a variety of deep neural network models for abstractive document summarization. One of the most dominant structures is the sequence-to-sequence (seq2seq) models with attention mechanism (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016). See et al. (2017) introduced Pointer Generator network that implicitly combines the abstraction with the extraction, using copy mechanism (Gu et al., 2016; Zeng et al., 2016). More recently, there have been several studies that have attempted to improve the performance of the abstractive summarization by explicitly combining them with extractive models. Some notable examples include the use of inconsistency loss (Hsu et al., 2018), key phrase extraction (Li et al., 2018; Gehrmann et al., 2018), and sentence extraction with rewriting (Chen and Bansal, 2018). Our model improves Sentence Rewriting with BERT as an extractor and summary-level rewards to optimize the extractor.

Reinforcement learning has been shown to be effective to directly optimize a non-differentiable objective in language generation including text summarization (Ranzato et al., 2016; Bahdanau et al., 2017; Paulus et al., 2018; Celikyilmaz et al., 2018; Narayan et al., 2018). Bahdanau et al. (2017) use actor-critic methods for language generation, using reward shaping (Ng et al., 1999) to solve the sparsity of training signals. Inspired by this, we generalize it to sentence extraction to give per step reward preserving optimality.

8 Conclusions

We have improved Sentence Rewriting approaches for abstractive summarization, proposing a novel extractor architecture exploiting BERT and a novel training procedure which globally optimizes summary-level ROUGE metric. Our approach achieves the new state-of-the-art on both CNN/Daily Mail and New York Times datasets as well as much better generalization on DUC-2002 test set.

Acknowledgments

We thank anonymous reviewers for their constructive and fruitful comments. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF2016M3C4A7952587).

References

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In (Bengio and LeCun, 2015).

Yoshua Bengio and Yann LeCun, editors. 2015. *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675, New Orleans, Louisiana. Association for Computational Linguistics.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98, San Diego, California. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1998–2008, Berlin, Germany. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. 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)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

Andrew Hoang, Antoine Bosselut, Asli Celikyilmaz, and Yejin Choi. 2019. Efficient adaptation of pretrained transformers for abstractive summarization. *arXiv preprint arXiv:1906.00138*.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.

Laura Jehl, Carolin Lawrence, and Stefan Riezler. 2019. Neural sequence-to-sequence models from weak feedback with bipolar ramp loss. *Transactions of the Association for Computational Linguistics*, 7:233–248.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In (Bengio and LeCun, 2015).

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Chenliang Li, Weiran Xu, Si Li, and Sheng Gao. 2018. Guiding generation for abstractive text summarization based on key information guide network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 55–60, New Orleans, Louisiana. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu. 2019. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Inderjeet Mani and Mark T Maybury. 2001. Automatic summarization.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759, New Orleans, Louisiana. Association for Computational Linguistics.

Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In

Proceedings of the Sixteenth International Conference on Machine Learning, ICML '99, pages 278–287, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

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*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Evan Sandhaus. 2008. The new york times annotated corpus.

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 (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Jiaxin Shi, Chen Liang, Lei Hou, Juanzi Li, Zhiyuan Liu, and Hanwang Zhang. 2019. Deepchannel: Salience estimation by contrastive learning for extractive document summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6999–7006.

Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. *arXiv preprint arXiv:1902.00863*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Wenyuan Zeng, Wenjie Luo, Sanja Fidler, and Raquel Urtasun. 2016. Efficient summarization with read-again and copy mechanism. *arXiv preprint arXiv:1611.03382*.

Haoyu Zhang, Yeyun Gong, Yu Yan, Nan Duan, Jianjun Xu, Ji Wang, Ming Gong, and Ming Zhou. 2019a. Pretraining-based natural language generation for text summarization. *arXiv preprint arXiv:1902.09243*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019b. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019c. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.