information

MDPI

Article

Document Summarization Based on Coverage with Noise Injection and Word Association

Heechan Kim [1] and Soowon Lee [2,*]

[1] Department of Software Convergence, Soongsil University, Seoul 06978, Korea; heathkim@soongsil.ac.kr
[2] School of Software, Soongsil University, Seoul 06978, Korea
* Correspondence: swlee@ssu.ac.kr

Received: 22 October 2020; Accepted: 18 November 2020; Published: 19 November 2020

Abstract: Automatic document summarization is a field of natural language processing that is rapidly improving with the development of end-to-end deep learning models. In this paper, we propose a novel summarization model that consists of three methods. The first is a coverage method based on noise injection that makes the attention mechanism select only important words by defining previous context information as noise. This alleviates the problem that the summarization model generates the same word sequence repeatedly. The second is a word association method to update the information of each word by comparing the information of the current step with the information of all previous decoding steps. According to following words, this catches a change in the meaning of the word that has been already decoded. The third is a method using a suppression loss function that explicitly minimizes the probabilities of non-answer words. The proposed summarization model showed good performance on some recall-oriented understudy for gisting evaluation (ROUGE) metrics compared to the state-of-the-art models in the CNN/Daily Mail summarization task, and the results were achieved with very few learning steps compared to the state-of-the-art models.

Keywords: automatic summarization; natural language processing; deep learning

1. Introduction

Automatic document summarization is a research field that extracts important information from documents in natural language processing [1]. As the volume of text data is rapidly increasing, the importance of summarization research is increasing, with the need for only important information to be extracted. Automatic summarization can be divided into abstract summarization and extractive summarization based on how the summary is generated. Abstractive summarization constructs a summary by generating a sequence of important words related to an input document. Extractive summarization constructs a summary by measuring saliences of sentences or words in an input document and selecting the sentences or words having the highest salience. In this paper, we focus on abstractive summarization.

An abstractive summarization model based on deep learning has an end-to-end structure that can directly learn relationships between an input document and a summary. This is an encoder-decoder structure implemented by attentional sequence-to-sequence models [2–11] or transformer-based models [12,13].

In this paper, we propose a summarization model to solve three problems of automatic summarization. The first problem is that the summarization model repeatedly generates the same subsequence as the previously generated word sequence, which is called a repetition problem [4,14]. As the summary comprises the word sequence that contains only important information from the input document, duplicate information in the summary should be minimized.

The repetition problem is due to the nature of a recurrent neural network, which is used in the sequence-to-sequence model. When the model is given information that is similar to previously given information, the model regenerates the same words already generated. To solve the repetition problem, models are suggested by [4,14] that use a positional coverage method to update the attention mechanism so that a word is not selected based on the positional information of the affected word scored by the attention distribution. The automatic summarization model suggested by [4], however, has a possibility of selecting an unimportant word because the summary is shorter than the input document. To alleviate this problem, Kim and Lee suggested a model using a context-based coverage method, in which the context is information of the input document that depends on the decoding step [5].

In order to measure coverage more effectively from the point of view of automatic summarization, we intend to improve the existing context-based coverage method to be robust to unimportant information. To achieve this, we propose a coverage method based on noise injection, in which noise refers to adaptive noise that changes according to the context information rather than a random variable and the coverage is defined based on the context and the noise. The coverage is added to the attention mechanism and makes the attention mechanism robust to unimportant information. Through adding coverage to the attention mechanism, the summarization model is trained to include only important information in the context.

The second problem is that previous summarization models calculate the word generation and pointing probability only using the information in the corresponding decode step. When a human writes a text, the subsequent word is chosen by taking into account all of previous words. This is also true in the summary. The network structures of previous summarization models make it difficult to reuse the information of words that have already been decoded. To solve this problem, Paulus et al. suggested a model applying an intra-attention mechanism that is operated within the decoder [6]. The intra-attention mechanism can be operated in both the encoder and decoder in Transformer-based models [12,13,15–18]. However, these models are only focused on the attention scores, which are the saliencies of words. The information of the already decoded words is not updated according to the information of the current decoding step. In this case, there is a limitation in delivering the information necessary for the corresponding step. In particular, Transformer-based models [16,17] focus on extractive summarization, which is far from the focus of this paper.

To overcome the limitation of existing research, we propose a word association method to update the information of each word by comparing the information of other words that were previously generated. Each piece of updated information is projected into a new dimension. Finally, all updated information of words is modeled as an associated context as a single vector. The associated context affects the final word probability distribution to produce a suitable word for a summary.

The third problem is that the probabilities of words that are not correct answers are not directly reflected in the learning process because the classification model learns through one-hot encoded answers. A summarization model trains its own weights by minimizing a negative log likelihood (NLL) loss so that the probabilities of occurrence of the correct answer words are maximized. When using one-hot encoded answers, the NLL loss reflects only the likelihood of the correct answer word, thus we need an additional penalty for misclassification.

To reflect a misclassification penalty in the summarization model, we propose a suppression loss function that can minimize the probability of occurrence of words that are not the correct answers. The suppression loss function is defined as an average of the positive log likelihood of words that are not correct answers and is applied in the form of a regularizer of the existing NLL loss function during training. The proposed model consists of the above three methods.

The rest of this paper is organized as follows. Existing summarization models are described in Section 2. The details of the proposed model are explained in Section 3. The experimental results of the proposed model using a CNN/Daily Mail dataset are presented in Section 4. Finally, conclusions and future work are discussed in Section 5.

2. Related Works

In an automatic summarization model based on an artificial neural network, investigations have been made to find the cause of the repetition problem in the structure of the sequence-to-sequence model. In neural machine translation, Tu et al. judged that the reason for the repetition problem is that the attention mechanism repeatedly gives high scores to the same input words. To solve the bias of the attention mechanism, Tu et al. suggested a new coverage that was defined as the cumulative sum of the attention distributions for the input word sequence in each decode step from the beginning to the previous step [14]. Thus, the coverage has information on the positional importance of the input words.

To solve the repetition problem in automatic summarization, See et al. suggested a summarization model that used the positional coverage proposed in machine translation [4,14]. A summary has the property that the length of the summary is very short compared to the length of the input word sequence. This is because the summary contains only the important content in the input document, which inevitably leads to loss of information about the non-critical content. For this reason, the summarization model using the positional coverage method has limitations. To overcome this limitation, Kim and Lee suggested a context-based coverage method to measure the coverage of the summary [5]. Context-based coverage was defined as the cumulative sum of the context up to the previous step and was added into the attention mechanism to select the next word containing information that had not been considered yet. The context is the weighted sum of the information of the words in the input document by the attention distribution.

The second problem is that the structures of previous summarization models make it difficult to reuse the information of words that have already been decoded in the corresponding decode step. From this point of view, Paulus et al. proposed a summarization model that uses an intra-attention mechanism that operates within the decoder [6]. In this intra-attention mechanism, an attention distribution was calculated using the current and previous information in the decoder and represents importance indices of words in the decoder. However, with this method, it is hard to determine important information of words at the current step from already decoded words in the decoder. The information of already decoded words is not updated according to the information of the current decoding step, which is a limitation in delivering information necessary for the corresponding step.

To effectively train the summarization model, various types of loss function applied in the model were investigated. See et al. suggested a coverage penalty to effectively learn the coverage mechanism [4]. Chung et al. suggested mechanisms and penalties to point words in the same sequence as the input document and a word near the word that has already been selected [8].

Various models have been proposed to improve summarization performance. Gehrmann et al. suggested a summarization model that has two sub-models: one is the binary classification model that only selects salient words and the other is the summarization model [4]. Because of the first sub-model, the second sub-model works only on the important words of the input document [9]. To maximize the performance of summarization directly, models based on reinforcement learning were suggested by [6,10,11]. In detail, the models were trained by a policy gradient method [19] that directly optimizes a recall-oriented understudy for gisting evaluation (ROUGE) metric [20], a performance measure in automatic summarization. The baseline for the REINFORCE algorithm [21] of these models was followed as a self-critical sequence-learning approach [22] that calculates the rewards of two sequences generated by selecting greedy policy and sampling from the policy distribution. Especially, Pasunuru and Bansal suggested additional loss function depending on logical entailment between a summary and an input document [11].

In addition to the summarization model based on the sequence-to-sequence structure, You et al. suggested a summarization model based on the transformer [12] to measure the saliency of words in both the encoder and decoder and to adjust the attention mechanisms according to the saliencies [13].

3. Proposed Model

The basic network structure of the proposed model is based on a modified network by adding the general context [7] to the pointer-generator [4], which is an extended form of the attentional sequence-to-sequence model based on the ideas in [2,23,24]. The general context is independent from the decoding step. As the attention distribution depends on the decoding step, the context defined using the attention distribution [3] is represented as a local context to eliminate ambiguity with the general context.

The complete network structure of the proposed model is extended from the basic network structure by adding two networks for the coverage method based on noise injection and the word association method, as illustrated in Figure 1. The word probability distribution in each step is defined by using the general context containing only the information in the encoder, the local context containing the information in the encoder and the decoder, and the associated context containing only the information in the decoder.

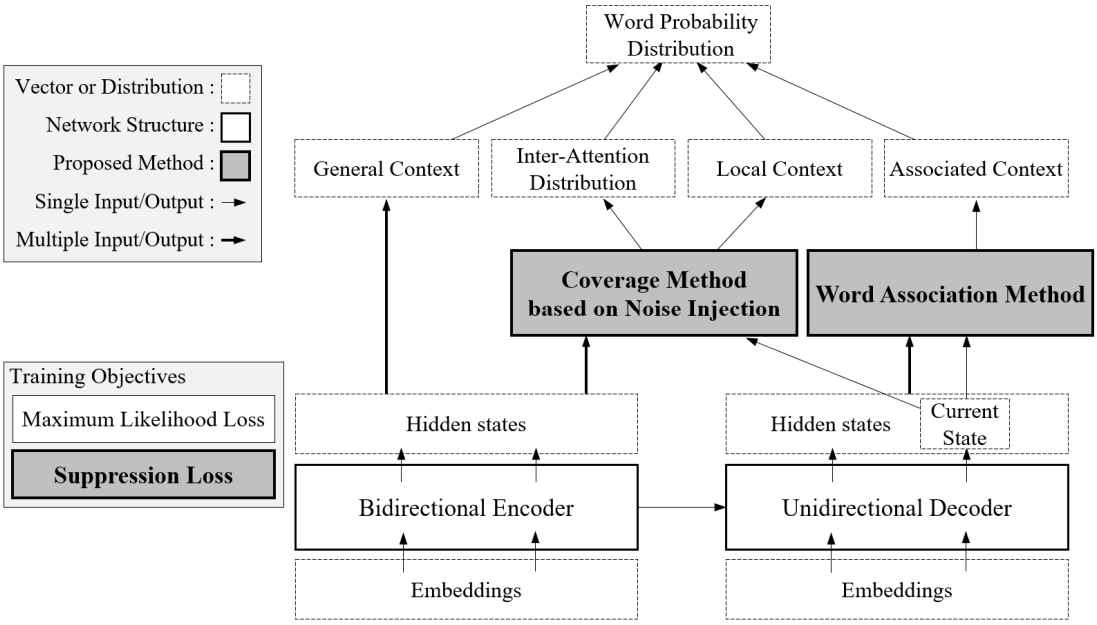

Figure 1. Structure of the proposed model.

In Figure 1, the methods proposed in this paper are shown in grey. For readability, the notation used in this section is presented differently from the original papers, but the implications are the same.

3.1. Notation and Basic Network

The proposed model consists of a multi-layered bidirectional encoder and a single-layered unidirectional decoder. The encoder and decoder use a long short-term memory (LSTM) network [25] as a cell. Input words $\dot{x}_{1,\ldots,i,\ldots,I}$ and their embeddings $x_{1,\ldots,i,\ldots,I}$ are given as the input of the bidirectional LSTM. A hidden state h_i is defined as the concatenation of the forward and the backward hidden states. Likewise, a cell state s_i is defined similarly. The hidden and cell states of the encoder in each direction are n_E-dimensional real vectors, so that the hidden and cell states h_i, s_i are $2n_E$-dimensional real vectors.

As the encoder has multiple layers of the bidirectional LSTM, $h_i^{(l)}$ represents the hidden state of the l-th layer and L represents the total number of layers. The input of the first layer is the embedding vector of the word, x_i, and the input of the other layers is given as the concatenation of the hidden state of the previous layer $h_i^{(l-1)}$ and the embedding vector x_i. The final hidden state of the encoder h_i^E

is defined as the concatenation of the hidden vectors of each layer $h_i^{(l)}$. The final cell state is defined similarly. The initial hidden and cell states of each layer are set with zero vectors.

Target words $\dot{y}_{1,\ldots,t,\ldots,T}$ and their embeddings $y_{1,\ldots,t,\ldots,T}$ are given as the input of the unidirectional decoder. The hidden and cell states of the decoder h_t^D, s_t^D are n_D-dimensional real vectors. The initial hidden and cell states of the decoder are initialized with the last hidden and cell states of the encoder. The number of dimensions of the final hidden and cell states of the encoder does not match the number of dimensions of the hidden and cell states of the decoder. In order to reduce the number of dimensions of the hidden and cell states of the encoder to a suitable number, the initial hidden and cell states of the decoder are defined as affine transformations of the hidden and cell states of the encoder.

Since the proposed model uses a multi-layered encoder, it is necessary to modify the inter-attention mechanism. Although there can be many variations, we define the independent inter-attention mechanism for each layer to model from the grammatical to the semantic information [26]. The local context c^L is a concatenation of the local context of each layer $c^{L(l)}$ as defined as follows:

$$\begin{aligned} c_t^L &= \left[c_t^{L(1)}, \ldots, c_t^{L(l)}, \ldots, c_t^{L(L)}\right], \\ c_t^{L(l)} &= \sum_{i=1}^{I} \alpha_{it}^{(l)} h_i^{(l)}, \\ \alpha_{it}^{(l)} &= \frac{\exp\left(e_{it}^{(l)}\right)}{\sum_k \exp\left(e_{kt}^{(l)}\right)}, \end{aligned} \tag{1}$$

where an inter-attention score $\alpha_{it}^{(l)}$ is defined as the softmax of inter-attention energy $e_{it}^{(l)}$. The summary should have the same meaning as the input document. According to the range of the attention mechanism, we classify the attention between the encoder and the decoder as inter-attention and the attention within the decoder as intra-attention. To reflect the overall meaning of the input document to the summary, it is essential to consider the information of the encoder to the decoder independent to the decoding steps. A general context, defined as the arithmetic mean of the hidden states of the encoder, was proposed to consider the overall meaning of the input document [7]. As the model proposed in this paper has a multi-layered encoder, the general context c^G is redefined as a concatenation of the general context of each layer $c^{G(l)}$, which is defined as an arithmetic mean of hidden states of each layer of the encoder. Since Kim and Lee added the general context to the attention mechanism [7], there was a limitation that the overall information of the input document did not sufficiently affect the word probability distribution. To overcome this limitation, the word probability distribution P is defined as the weighted sum of the word generation distribution P^V from the vocabulary V and the word pointing distribution P^p from the input document by the word generation probability p^g as follows:

$$\begin{aligned} P_t\left(\dot{y}\right) &= p_t^g P_t^V\left(\dot{y}\right) + \left(1 - p_t^g\right) P_t^p\left(\dot{y}\right), \\ p_t^g &= \sigma\left(w_{g1}^\top c^G + w_{g2}^\top c_t^L + w_{g3}^\top s_t^D + b_g\right) \in \mathbb{R}, \\ P_t^V\left(\dot{y}\right) &= \frac{\exp\left(V_t\left(\dot{y}\right)\right)}{\sum_k exp(V_t(k))}, \\ P_t^p\left(\dot{y}\right) &= \sum_{k=1}^{L} \sum_{i:\dot{y}_i=\dot{y}} \alpha_{it}^{(k)} \\ V_t &= W_{v4}\sigma\left(W_{v1}c^G + W_{v2}c_t^L + W_{v3}c_t^A + b_{v1}\right) + b_{v2}, \end{aligned} \tag{2}$$

where the pointing probability of the word $\dot{y}$ in the input document $P^p\left(\dot{y}\right)$ is defined as the sum of the attention score of each layer that represents the word $\dot{y}$; σ represents an activation function; the weights $w_{g1}, w_{g2} \in \mathbb{R}^{2*L*n_E}$, $w_{g3} \in \mathbb{R}^{n_D}$, $b_g \in \mathbb{R}$, $W_{v1}, W_{v2} \in \mathbb{R}^{n_V \times 2*L*n_E}$, $W_{v3} \in \mathbb{R}^{n_V \times n_D}$, $b_{v1} \in \mathbb{R}^{n_V}$, $W_{v4} \in \mathbb{R}^{|V| \times n_V}$, and $b_{v2} \in \mathbb{R}^{|V|}$ are learnable parameters. The details of the associated context c^A are described in Section 3.3.

3.2. Coverage Method Based on Noise Injection

The summarization model is trained so that it can extract important information using the given input document and summary. This property is related to a coverage method that solves the repetition problem. A coverage method for summarization should pick out important words that have not yet been summarized among the input words. The model using the context-based coverage method [5] showed slightly better performance than that using the positional coverage method [4].

In this paper, we consider that this limitation of little performance gain occurs because the existing context-based coverage method did not explicitly manage the context information used in the previous steps. To overcome this limitation, we propose a coverage method based on noise injection that deals with the previously used context information as the noise. The coverage method based on noise injection makes the model robust to unimportant information in the local context according to the decoding step so that the coverage works more effectively [27]. The coverage is defined separately for each layer as follows:

$$
\begin{aligned}
r_t^{(l)} &= \begin{cases} c_{t-1}^{L(l)} + \varepsilon_t^{(l)}, & t > 2 \\ \{0\}^{2*n_E}, & \text{otherwise} \end{cases} \\
\varepsilon_t^{(l)} &= \sum_{k=1}^{t-2} c_k^{L(l)} \\
e_{it}^{(l)} &= w_e^{(l)\top} \tanh\Big(W_{e1}^{(l)} r_t^{(l)} + W_{e2}^{(l)} s_t^D + W_{e3}^{(l)} s_i^{(l)} + b_e^{(l)}\Big),
\end{aligned}
\tag{3}
$$

where the coverage based on noise injection in the l-th layer $r_t^{(l)}$ is defined as the sum of the local context of the previous step $c_{t-1}^{L(l)}$ and the noise $\varepsilon_t^{(l)}$. To select the word with important information by the intra-attention mechanism at step t, one piece of required information is the local context at the previous step $c_{t-1}^{L(l)}$ that was used to generate the word for the summary. Noise $\varepsilon_t^{(l)}$ is information that is likely not needed at the current step and is defined as the sum of the local contexts from the beginning to the step $t-2$, $c_{1,\ldots,t-2}^{L(l)}$, which is information that has already been used for word generation. This noise can be seen as a mixture of information that is needed or not needed for summarization, depending on an embedding of a word and a decoding step. With this dependency on the local context, this noise can be seen as a context-based adaptive noise that changes depending on input information and it only applies during training. As the coverage is applied to the inter-attention mechanism, the inter-attention mechanism can focus on words with information that has not yet been summarized. The weights $W_{e1}^{(l)}, W_{e3}^{(l)} \in \mathbb{R}^{n_A \times 2*n_E}$, $W_{e2}^{(l)} \in \mathbb{R}^{n_A \times n_D}$, and $b_e^{(l)}, w_e^{(l)} \in \mathbb{R}^{n_A}$ are learnable parameters. The whole process of the coverage method is shown in Figure 2.

The context-based adaptive noise $\varepsilon_t^{(l)}$ works within the model in the following way. The local context is information weighted by the inter-attention distribution that is the relevance between the information of all input words and the information of the current decoding step. When the cumulative local context is used as the noise as in the proposed method, the weights in the inter-attention mechanism will be trained to suppress the dimensions of the local context with unnecessary information to generate words for the summary so that the inter-attention mechanism can be robust to unimportant context information. As a result, all weights in the model also learn to reflect only the necessary information, as the inter-attention mechanism utilizes information of both the encoder and the decoder.

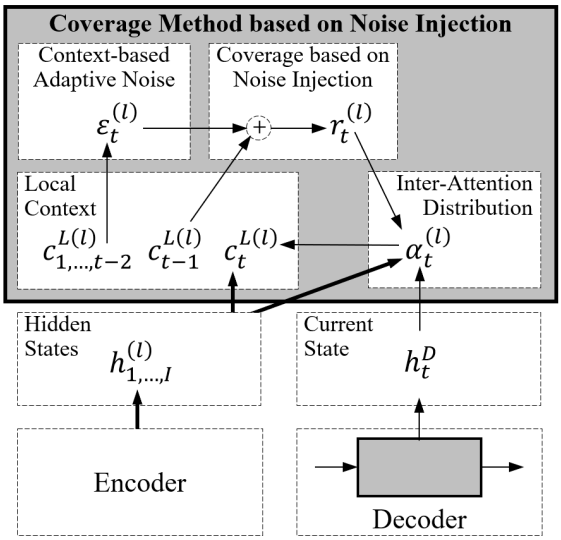

Figure 2. Detailed process of the coverage method based on noise injection.

3.3. Word Association Method

To use the information of words that have already been decoded, Paulus et al. used the intra-attention mechanism based on the sequence-to-sequence model [11] and You et al. used the self-attention mechanism based on the transformer model [13]. These models focus on the relationship between the information of one word and that of others and do not update the information of already decoded words according to the information of the current decoding step. Thus, the previous models have a limitation in that the decoder may not receive accurate information for the corresponding step.

To overcome the limitation of the previous models, we propose a word association method that explicitly specifies the updated information of words according to the information of other words. The word association method updates the information of the words in all decoding steps by comparing the information of the word in the current step and all previous steps and abstracting the information of all updated information into a single vector, an associated context, by using the existing intra-attention mechanism. The intra-attention mechanism works within only the decoder, unlike the inter-attention mechanism. The associated context in the t-th step c_t^A is defined as follows:

$$\begin{aligned} c_t^A &= \sum_{k=1}^{t} \beta_{kt} \widetilde{h}_{kt}^D \\ \beta_{kt} &= \frac{\exp\left(f_{kt}^{(l)}\right)}{\sum_j \exp\left(f_{jt}^{(l)}\right)} \in \mathbb{R}, \qquad k \le t \\ f_{kt} &= w_f^{(l)\top} \tanh\left(W_{f1} h_k^D + W_{f2} h_t^D + b_f\right) \in \mathbb{R}, \qquad k \le t \\ \widetilde{h}_{kt}^D &= \sigma\left(W_{r1} h_k^D + W_{r2} h_t^D + b_r\right) \in \mathbb{R}^{n_D}, \qquad k \le t, \end{aligned} \tag{4}$$

where the associated context in the t-th step c_t^A is defined as the weighted sum of the updated hidden states of each k-th step for the t-th step $\widetilde{h}_{kt}^D$ by the its intra-attention score β_{kt}. The intra-attention score β_{kt} between the k-th and t-th steps is defined as the softmax of intra-attention energy f_{kt}, which is defined using the hidden states of the k-th and t-th steps in the decoder h_k^D and h_t^D. The updated hidden state of the k-th step for the t-th step $\widetilde{h}_{kt}^D$ is defined by using the hidden states h_k^D and h_t^D of the decoder as the intra-attention energy; however, it is an n_D-dimensional real vector, like the hidden state of the decoder. In the intra-attention mechanism, k is always less than or equal to t. The weights $W_{f1}, W_{f2} \in \mathbb{R}^{n_A \times n_D}$, $W_{r1}, W_{r2} \in \mathbb{R}^{n_D \times n_D}$, $b_f, w_f \in \mathbb{R}^{n_A}$, and $b_r \in \mathbb{R}^{n_D}$ are learnable parameters. The whole process of the word association method is shown in Figure 3.

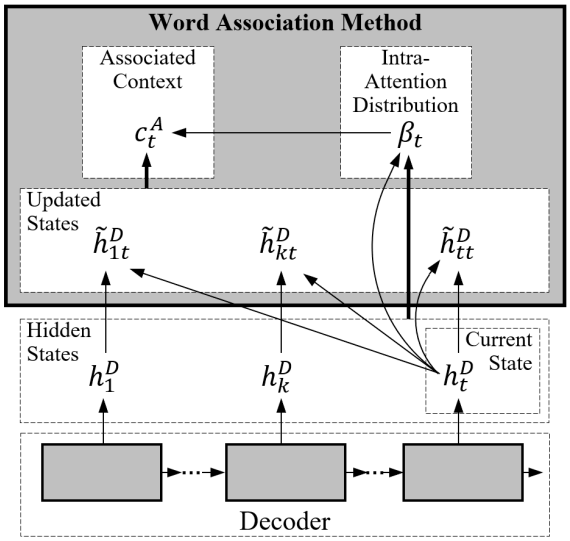

Figure 3. Detailed process of the word association method.

3.4. Suppression Loss Function

In general, a good classification model produces a high probability for the correct answer class and a low probability for the wrong answer class. The neural model that classifies data using the category distribution is trained to increase the probability of the correct answer class in the aspect of maximum likelihood estimation. In automatic summarization, the summarization model is trained by minimizing the negative log likelihood (NLL) loss to maximize the probabilities of the generation of the summary word sequence y^*. The NLL loss function $loss_{ML}$ is defined as follows:

$$loss_{ML}(y^*) = -\sum_{t=1}^{T} \log\left(P_t\left(y_t^*\right)\right) \tag{5}$$

As the summarization model is trained by one-hot encoded answers, this NLL loss function $loss_{ML}(y^*)$ uses only the probability of the correct word in the distribution output by the model. In this case, there is the limitation that the probabilities of the wrong words are not used in the learning process. In order to use the probabilities of the wrong words, we propose the suppression loss function, reflecting a penalty for misclassification. This suppression loss minimizes the probabilities of the wrong words and is used to train the summarization model along with the NLL loss. The suppression loss function $loss_S$ is defined as follows:

$$loss_S(y^*) = \frac{1}{|V| + |\neg V| - 1} \sum_{t=1}^{T} \sum_{v \in \neg y_t^*} \log(P_t(v)), \qquad \neg y_t^* = \left\{y \middle| y \neq y_t^*\right\}, \tag{6}$$

where the suppression loss function $loss_S$ is defined as the mean of the positive log likelihood of the words v that are in the set of words not in the summary $\neg y_t^*$. When calculating the mean, the number of non-answer words is $|V| + |\neg V| - 1$, where $|V|$ and $|\neg V|$ are the total number of the vocabulary V and the total number of the out-of-vocabulary $\neg V$ words in y^*, respectively, excluding the single correct word. The final loss function *Loss* is defined as follows:

$$Loss(y^*; \lambda_R) = loss_{ML}(y^*) + \lambda_R loss_S(y^*). \tag{7}$$

where the impact of the suppression loss is controlled by the regularization parameter λ_R and the parameter λ_R is determined through a validation.

3.5. Decoding Algorithm

In this study, we used the beam-search algorithm to generate the most likely summary, with a constraint excluding the previously generated trigram proposed by [6]. In addition to this constraint, in order to block the continuous generation of the same unigram or bigram, we add a new constraint that excludes consecutive generation of a unigram or bigram in the beam-search algorithm. Furthermore, the unknown token is excluded.

We use the score based on the length penalty for the beam-search [28]. The score and the penalty are defined as follows:

$$\begin{aligned} s(\hat{y}) &= \frac{-\sum_{k-1}^{t'} \log(P_k(\hat{y}_k))}{lp(\hat{y})}, \\ lp(\hat{y}) &= \frac{(5+|\hat{y}|)^{\lambda_l}}{(5+1)^{\lambda_l}}, \end{aligned} \tag{8}$$

where the impact of the length penalty is determined by the hyperparameter for the penalty λ_l.

4. Experiments

4.1. Dataset

To evaluate the performance of the proposed model, we chose the CNN/Daily Mail dataset [29], which is a set of news items widely used for abstractive summarization model learning. Each data element in the dataset consists of a pair containing an article (only news body) and a summary. The summary consists of the highlights written by the author of the news item.

We used the non-anonymized version of the dataset like other research [4–11,13], that is, without a named entity recognition, part-of-speech tagging, and so on. Each data element was space tokenized and changed to lowercase. Special characters that are attached to other characters were also tokenized. We also used the same split dataset, which consists of 287,226 pairs for training, 13,368 pairs for validation, and 11,490 pairs for testing. The training data was shuffled every epoch.

4.2. Experimental Settings

The hyperparameters for the proposed model, similar to other studies, were set as large as the experimental environment permits, and the detailed settings were as follows. The encoder consisted of two layers. The number of dimensions of word embedding, the number of dimensions of the states in each LSTM for the encoder n_E, and the number of dimensions for the attention mechanism n_A were set to 128. The number of dimensions of the states in the LSTM for the decoder n_D and the number of dimensions for the vocabulary distribution n_V were set to 256. The vocabulary was defined as the top 50,000 words that appear most frequently in the training dataset and the same vocabulary was used in both the encoder and the decoder. As a result, the total number of learnable parameters of the proposed model is 21,353,553. The maximum lengths of the input document and the summary were set to 400 and 100, respectively. In the testing, the beam-search algorithm was performed to 120 steps, as in [4]. For objective comparison with other models, the summary consisted of only the first 100 words, the same as other models.

The learnable parameters were initialized as in the experiment in [4]. Specifically, the parameters for the attention mechanisms were initialized using a random uniform distribution, with −0.02 as the minimum and 0.02 as the maximum. The parameters for the rest, except biases, were initialized with a truncated normal distribution with the zero mean and 0.0001 of the standard deviation. The parameters in the LSTM cells were initialized by Glorot Uniform distribution [30]. All biases were initialized by a zero vector. We used the Adam optimizer [31] to train the proposed model with the parameters, which are a learning rate of 0.001 and β_1 and β_2 of 0.9 and 0.999, respectively. We applied the gradient clipping method [32] with the global norm and a max of 2 to suppress a gradient exploding problem. In addition, we clipped the final word probabilities with a minimum of 1×10^{-10} for computational stability. We trained the proposed model using a single NVIDIA RTX 2080

Ti graphics processing unit. The proposed model was trained up to seven epochs with a batch size of 32 and it took about 105 min per epoch.

4.3. Evaluation Measure: ROUGE Metric

In automatic summarization, the final goal of the summarization model is to create a summary like a human-written summary; thus, the model is trained with the human-written summary as the correct word sequence that is a gold standard. In this sense, Lin suggested ROUGE metrics to quantify the performance of the summarization model [20]. ROUGE metrics measure how much the generated summary matches the gold standard, and there are several variations depending on the unit of matching. We used the metrics ROUGE-1, ROUGE-2, and ROUGE-L, like previous summarization research studies. ROUGE-N, which covers ROUGE-1 and ROUGE-2, is a measure that evaluates the ratio of n-gram units between the gold standard and the generated summary from the model, and is defined as the F1 score as follows:

$$\begin{aligned} \text{ROUGE} - \text{N} &= 2\frac{R_n P_n}{R_n + P_n}, \\ R_n &= \frac{\sum_{g \in G} \sum_{ng_n \in g} Count_{match}(ng_n)}{\sum_{g \in G} \sum_{ng_n \in g} Count(ng_n, G)}, \\ P_n &= \frac{\sum_{g \in G} \sum_{ng_n \in g} Count_{match}(ng_n)}{\sum_{m \in M} \sum_{ng_n \in m} Count(ng_n, M)}, \end{aligned} \tag{9}$$

where G and M represent the gold standard and the generated summary from the model, respectively; g and m represent the sets of sentences in each summary; ng_n represents the n-gram in the sentence; $Count_{match}(ng_n)$ represents the number of times the n-gram ng_n appeared in both the gold standard and the generated summary; and $Count(ng_n, G)$ and $Count(ng_n, M)$ represent the number of times the n-gram ng_n appeared in each summary. The recall of ROUGE-N R_n is defined as the ratio of the sum of the number of n-grams in both summaries to the sum of the number of n-grams in the gold standard only. Similarly, the precision of ROUGE-N P_n is defined as the ratio of the sum of the number of n-grams in both summaries to the sum of the number of n-grams in the generated summary only.

ROUGE-L is a measure based on the longest common subsequence (LCS) of the words between the gold standard and the generated summary and is defined as the F1 score as follows:

$$\begin{aligned} \text{ROUGE} - \text{L} &= \frac{2R_{lcs}P_{lcs}}{R_{lcs} + P_{lcs}}, \\ R_{lcs} &= \frac{\sum_{g \in G} |\cup_{m \in M} LCS(g,m)|}{n_G}, \\ P_{lcs} &= \frac{\sum_{g \in G} |\cup_{m \in M} LCS(g,m)|}{n_M}, \end{aligned} \tag{10}$$

where $LCS(g, m)$ represents the LCS between the sentences g and m. The recall based on the LCS R_{lcs} is defined as the ratio of the sum of the number of elements in the union of the LCS between g and $m \in M$ to the number of words in the gold standard n_G. Similarly, the precision based on the LCS P_{lcs} is defined as the ratio of the sum of the number of elements in the union of the LCS between g and $m \in M$ to the number of words in the generated summary n_M.

We used the libraries of the ROUGE metric for the experiments, including the core library (ROUGE-1.5.5) implemented with Perl and the wrapper (pyrouge) implemented with Python. We used the arguments '-n 2 -l 100' for ROUGE-1.5.5, which calculated the ROUGE-N scores up to ROUGE-2 and set the max length of a summary to 100 words.

4.4. Optimal Parameter Search

A deep learning model with many parameters is likely to overfit. To prevent overfitting, we chose the optimal parameters through early stopping. The trained parameters are saved and validated at the end of every epoch. The parameter space to be searched for the hyperparameters for the suppression loss λ_R and the length penalty λ_l and the index of epoch is too large. To reduce the space, we found the

optimal hyperparameters step by step. First, we found the optimal hyperparameter for the suppression loss λ_R^* and the best epoch with a beam-size of three and $lp(\hat{y}) = |\hat{y}|$. The set of parameters with the highest ROUGE-L score for validation data was chosen as the set of optimal parameters for the model. High ROUGE-L means that the summary that most closely resembles the golden standard has been generated. For the validation, the model generated a summary which was the same as the testing. The hyperparameter for the suppression loss λ_R was set from 0.1 to 1.1 in 0.1 steps, and the index of epoch was set from 2 to 7. As a result of the validation with the validation data, the ROUGE-L scores are shown in Appendix A Table A1.

The optimal hyperparameter for the suppression loss λ_R^* and the optimal index of epoch were confirmed as 0.5 and 3, respectively. Second, we found the optimal hyperparameter for the length penalty λ_l^* using the optimal hyperparameter for the model with λ_R^* of 0.5 and the best index of epoch of 3. The hyperparameter for the length penalty λ_l was set from 0.7 to 1.7 in 0.1 steps. As a result of the validation with the validation data, the ROUGE-1, ROUGE-2, and ROUGE-L scores are shown in Table A2.

As a result of the validation, the optimal hyperparameter for the length penalty λ_l^* was confirmed as 1.4 with a 38.88 ROUGE-L score. The beam-size was set to 10 for the testing, as in other research [9,13]. The final performance evaluation of the proposed model is based on the optimal parameters and hyperparameters found.

4.5. Experimental Results

4.5.1. Quantitative Results

We chose the following state-of-the-art summarization models for performance comparison: pointer-generator without coverage and with coverage [4], context-based coverage [5], a reinforcement learning-based model (RL and ML+RL with intra-attention) [6], monotonic alignments [8], bottom-up summarization [9], deep communicating agents (ML, ML+RL) [10], multi-reward reinforced summarization (ROUGESal+Ent) [11], and extended transformer model for abstractive document summarization (ETADS) [13]. Table 1 shows the ROUGE F1 scores of the proposed model in the last row and other models sorted in ascending order based on the ROUGE-2 scores as in other summarization research. The best scores in each measure are marked in bold.

Table 1. Recall-oriented understudy for gisting evaluation (ROUGE) F1 scores of automatic summarization models.

Various Models	ROUGE-1	ROUGE-2	ROUGE-L
Pointer-generator without coverage [4]	36.44	15.66	33.42
RL, with intra-attention [6]	41.16	15.75	**39.08**
ML+RL, with intra-attention [6]	39.87	15.82	36.90
Monotonic alignments [8]	39.91	17.06	36.24
Pointer-generator with coverage [4]	39.53	17.28	36.38
Context-based coverage [5]	39.64	17.54	36.52
Lead-3 [4]	40.34	17.70	36.57
ROUGESal+Ent (ML+RL) [12]	40.43	18.00	37.10
Deep communicating agents (ML) [11]	41.11	18.21	36.03
Bottom-up summarization [10]	41.22	18.68	38.34
ETADS [13]	**41.75**	19.01	38.89
Deep communicating agents (ML+RL) [11]	41.69	**19.47**	37.92
Proposed model	41.63	19.14	38.84

Note: RL, reinforcement learning; ML, maximum likelihood; Sal+Ent, Saliency+Entailment; ETADS, extended transformer model for abstractive document summarization.

As a result of the experiment, the proposed model recorded a 41.63 ROUGE-1, 19.14 ROUGE-2, and 38.84 ROUGE-L score, outperforming most of the previous state-of-the-art models. The proposed

model performed better in some ROUGE metrics compared to the most recent state-of-the-art models. Among all comparative models, the proposed model achieved the third rank for ROUGE-1, second rank for ROUGE-2, and third rank for ROUGE-L. Through this result, it can be confirmed that the proposed model generally shows good performance as a single model for all metrics. Compared to the ETADS model, the proposed model recorded a 0.13 higher ROUGE-2 score and 0.12 lower ROUGE-1 and 0.05 lower ROUGE-L scores. The ETADS model was trained by additionally applying the dropout method [33] to the network and the Noam decay strategy [12] to the learning rate. The proposed model, however, recorded this performance through a relatively simple learning strategy that did not apply these methods. Compared to the deep communicating agents (ML+RL) model, the proposed model recorded a 0.92 higher ROUGE-L score and a 0.06 lower ROUGE-1 and 0.33 lower ROUGE-2 scores. The deep communicating agents (ML+RL) model is a reinforcement learning-based model that directly maximizes the ROUGE metrics as rewards and uses the initial embeddings by the pre-trained global vectors for word representation (GloVe) [34]. The proposed model, however, was optimized only by maximum likelihood estimation with the penalty and the embeddings were initialized at random, so the performance was achieved using relatively simple settings. Under these conditions, the performance of the proposed model is remarkable.

The proposed model exceeded the state-of-the-art models in terms of convergence speed. Both the ETADS and the deep communicating agents models were tested based on parameters that learned up to 200,000 steps. As the batch size was set to 32, the proposed model learned 8791 steps in each epoch. As the best parameter was determined in the third epoch, the proposed model with the best parameters was totally trained in just 26,913 steps. That means that the proposed model achieved competitive performance, with only about 13% of the training steps compared to the state-of-the-art models. In view of this fast convergence speed and the absence of other additional methods, it can be judged that the three proposed methods in the model are well defined for automatic summarization.

4.5.2. Qualitative Results

In order to compare the characteristics of the summary generated by the proposed model, we chose models in which the generated summaries are published by the own authors as the baseline models. Since it is difficult to evaluate the quality of the generated summaries only with ROUGE scores, the summaries are also evaluated qualitatively in research [16,18]. The baseline models are pointer-generator with coverage and bottom-up summarization. A sample news article selected from the test dataset for generating summaries is shown in Table 2.

Table 2. A sample news article in the test dataset.

Article
a 46-year-old man was sentenced to life in prison on monday after shooting dead a father and son because they were related to a driver who killed his nine-year-old sister in a crash 45 years ago
alfred guy vuozzo *swore loudly as he was told he would not be eligible for parole* for 35 years *for murdering brent mcguigan, 68, and his son, brendon, 39, on prince edward island last august*
as he was escorted from the courtroom, he screamed: 'you've sentenced me to life and I sent them to death', while the judge called the brutal double-murder an act of 'hatred and misdirected vengeance'
vuozzo was two years old when his older sister, cathy, was killed in a crash in 1970
Brent's father, herbert, who was behind the wheel, later *received a nine-month sentence for dangerous driving*
scroll down for video
'revenge': *alfred guy vuozzo, 46,* has been sentenced to life in prison after shooting dead brent mcguigan, 68, and his son, brendon -lrb- both pictured -rrb-, 39, because they were related to a driver who killed his nine-year-old sister in a crash 45 years ago

In Table 2, underlined, italicized, and shaded text represents the referenced text by the pointer-generator with coverage, bottom-up summarization, and the proposed models, respectively. The summaries generated by these models are shown as Table 3.

Table 3. Summaries generated by the proposed model and the baseline models.

Models	Summaries
Proposed model	alfred guy vuozzo swore loudly as he was told he would not be eligible for parole for 35 years. he was sentenced to life in prison after shooting dead brent mcguigan, 68, and his son, brendon, 39, on prince edward island last august. judge called the brutal double-murder an act of 'hatred and misdirected vengeance'
BU	alfred guy vuozzo, 46, swore loudly as he was told he would not be eligible for parole for 35 years. vuozzo was two years old when his older sister, cathy, was killed in a crash in 1970. Brent's father, herbert, received a nine-month sentence for dangerous driving.
PGC	*alfred guy vuozzo, 46, swore loudly as he was told he would not be eligible for parole for murdering brent mcguigan, 68, and his son, brendon, 39, on prince edward island last august. vuozzo was two years old when his older sister, cathy, was killed in a crash in 1970. Brent's father, herbert, who was behind the wheel, received a nine-month sentence for dangerous driving.*

Note: BU represents Bottom-up summarization [10]; PGC represents Pointer-generator with coverage [4].

As shown in Table 3, these models generated the summary by highlighting the input document. There was, however, a difference between the baseline models and the proposed model for generating summaries. In the sample news, the situation of a person, Alfred, was described in several sentences. Different from the baseline models, the proposed model tended to mix the phrases in the sentences with the same subject. Especially, as in the second sentence generated by the proposed model, the proposed model could express a proper noun as a pronoun.

From these results, it can be seen that the proposed model produces a more abstractive summary. This is in line with the point that the ROUGE-L score of the proposed model is higher than that of the baseline models. Due to this characteristic, however, there are cases where the proposed model mismatched the objects and their explanations in the news where several objects appear, or the matches and their scores in the sports news.

4.6. Comparison between Proposed Methods

Three methods are proposed in this paper: the coverage method based on noise injection, the word association method, and the suppression loss function. To investigate how each of these three proposed methods affects performance improvement, we conducted experiments using models composed of only a subset of the proposed methods. The models are three in total. The first model consists of only the coverage method (C model). The second model consists of the coverage method and the word association method (C-A model). The third model consists of the coverage method and the suppression loss (C-L model). These three models are indicators for determining the impact of each proposed method on the performance of the overall model. Selecting the optimal parameters of the three models proceeded in the same way as described in Section 4.3. In the cases of the C model and the C-A model, which do not include the suppression loss function, the hyperparameter for suppression loss λ_R was not searched. The results of the validation of the optimal parameters for the models are shown in Table A3. The highest ROUGE-L scores in each model are marked in bold.

As a result of the validation, the optimal parameters of each model were determined as follows. The optimal index of epoch for the C model was confirmed as 5 with a ROUGE-L score of 37.67. The optimal index of epoch for the C-A model was confirmed as 2 with a ROUGE-L score of 37.89. The optimal hyperparameter for the suppression loss and the optimal index of epoch for the C-L model were confirmed as 0.5 and 5, respectively, with a ROUGE-L score of 38.37. Based on the optimal parameters of each model, the optimal hyperparameter for the length penalty was validated separately. The results of the validation for the hyperparameter are shown in Table A4.

When finding the optimal length penalty for the C-A model, there were cases where the same ROUGE-L was recorded in the validation. This means that there was no change in the ROUGE-L score according to the values of the length penalty for the beam search algorithm. To select one value, the optimal hyperparameter for the length penalty was selected as the highest ROUGE-2 score among the cases with the same ROUGE-L.

For comparison between the proposed methods, we used the model suggested by [7] as a baseline, which is similar to the basic network of the proposed model. The results of the testing are shown in Table 4.

Table 4. ROUGE F1 scores of models containing a subset or all of the proposed methods.

Various Models	ROUGE-1	ROUGE-2	ROUGE-L
Baseline	39.93	17.71	36.68
C model	41.04	18.62	38.07
C-A model	40.84	18.78	37.85
C-L model	41.70	19.00	38.67
C-A-L model (Proposed Model)	41.63	19.14	38.84

In Table 4, from the result of the C model, the coverage method improved the performance for ROUGE-1 by 1.11, for ROUGE-2 by 0.91, and for ROUGE-L by 1.39 compared to the baseline. From the result of the C-A model, compared to the C model, the association method improved the performance for ROUGE-2 by 0.16, and in the other metrics, worsened the performance for ROUGE-1 by 0.2 and for ROUGE-L by 0.22. From the result of the C-L model, compared to the C model, the suppression loss function improved the performance of each ROUGE metric by 0.66, 0.38, and 0.60, respectively. In the model in which all proposed methods were used, compared to the C-L model, ROUGE-1 decreased by 0.07, but ROUGE-2 and ROUGE-L rose by 0.14 and 0.17, respectively. From these results, it can be said that the synergy between the proposed methods is good.

5. Conclusions

In this paper, we studied methods to automatically generate a summary which contains important information for given news data. To solve the problems in previous research on automatic summarization, we proposed a coverage method based on noise injection, a word association method, and a suppression loss function that utilizes misclassification information as a penalty. The proposed model, consisting of the proposed three methods, achieved a ROUGE-2 score of 19.14 and a ROUGE-L score of 38.84 on the benchmark CNN/DailyMail news dataset, and these performance results are better in some ROUGE metrics than current state-of-the-art models. In addition, compared to the state-of-the-art models, the proposed model achieved comparable performance with only 13% of the learning steps, and it was confirmed that the convergence speed was very fast. From these results, we can conclude that the synergy between the proposed methods is very effective.

During the analysis of the experimental results, we observed that the summary was often generated in a form that did not match the meaning of the input document. To minimize the distortion of the information in the summary, in a future work, we will study how to more clearly define the relationship between the information of contents in the input document and the summary for the pointing method.

Author Contributions: Conceptualization, H.K. and S.L.; Methodology, H.K.; Software, H.K.; Validation, H.K. and S.L.; Formal Analysis, S.L.; Investigation, H.K.; Resources, S.L.; Data Curation, H.K.; Writing Original Draft Preparation, H.K.; Writing Review & Editing, S.L.; Visualization, H.K.; Supervision, S.L.; Project Administration, S.L.; Funding Acquisition, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Ministry of Science and ICT (MSIT), Korea, under the Information Technology Research Center (ITRC) support program (IITP-2020-2018-0-01419) supervised by the Institute for Information and communications Technology Promotion (IITP).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. The Tables about Validation Results

Table A1. Results of validation for each epoch and λ_R.

ROUGE-L		Epoch					
		2	**3**	**4**	**5**	**6**	**7**
λ_R	0.1	38.26	38.41	38.68	38.59	38.49	38.45
	0.2	38.34	38.40	38.20	38.19	38.17	37.78
	0.3	37.16	37.23	37.98	38.07	38.13	37.77
	0.4	35.99	36.85	36.41	36.50	36.17	36.07
	0.5	38.09	**38.73**	38.47	38.36	38.26	38.41
	0.6	37.16	37.26	37.75	37.28	37.30	37.37
	0.7	37.17	37.55	38.24	38.03	38.22	38.20
	0.8	37.57	37.91	38.14	38.45	37.53	38.03
	0.9	36.96	37.35	37.18	36.44	37.08	36.84
	1.0	37.49	37.48	37.26	37.12	37.65	37.63
	1.1	37.65	37.87	36.73	37.70	37.97	37.84

Note: The best score is marked in bold.

Table A2. Results of validation for each λ_I.

λ_I	ROUGE-L
0.7	38.67
0.8	38.68
0.9	38.69
1.0	38.71
1.1	38.75
1.2	38.80
1.3	38.83
1.4	38.87
1.5	**38.88**
1.6	38.86
1.7	38.82

Note: The best score is marked in bold.

Table A3. Results of the validation of the optimal parameters for various combinations of proposed method.

ROUGE-L		Epoch					
Models		**2**	**3**	**4**	**5**	**6**	**7**
C model		36.28	37.07	36.30	**37.67**	35.53	37.21
C-A model		37.21	**37.89**	37.38	37.50	37.70	37.04
C-L model							
λ_R	0.1	36.71	36.80	35.80	37.12	36.05	36.65
	0.2	36.74	36.45	35.83	37.42	36.55	36.68
	0.3	36.91	36.94	35.80	37.30	36.22	36.93
	0.4	35.64	36.70	35.71	36.68	35.91	36.80
	0.5	35.89	37.73	36.39	**38.37**	36.42	37.36
	0.6	35.99	36.99	35.92	37.56	36.08	36.89
	0.7	36.71	36.57	35.69	36.84	36.05	36.73
	0.8	36.39	36.09	35.04	36.69	35.76	36.75
	0.9	36.37	37.32	36.44	37.74	36.08	37.36
	1.0	35.91	37.08	36.02	37.67	35.83	36.99
	1.1	36.29	36.12	35.44	37.07	36.46	36.46

Note: The C-model used only the proposed noise injection coverage method; the C-A model used the proposed coverage method and the proposed word association method; the C-L model used the proposed coverage method and the proposed suppression loss function. The best scores for each model are marked in bold.

Table A4. Results of the validation for the hyperparameter the length penalty in each model.

Models	λ_l^*	ROUGE-L
C model	1.7	37.90
C-A model	1.1	37.88
C-L model	1.7	38.90

References

1. Radev, D.R.; Hovy, E.; McKeown, K. Introduction to the special issue on summarization. *Comput. Linguist.* **2002**, *28*, 399–408. [CrossRef]
2. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Bangkok, Thailand, 23–27 November 2014; Volume 2, pp. 3104–3112.
3. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
4. See, A.; Liu, P.J.; Manning, C.D. Get to the point: Summarization with pointer-generator networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1, pp. 1073–1083.
5. Kim, H.; Lee, S. A context based coverage model for abstractive document summarization. In Proceedings of the 10th International Conference on Information and Communication Technology Convergence, Jeju Island, Korea, 16–18 October 2019; pp. 1129–1132.
6. Paulus, R.; Xiong, C.; Socher, R. A deep reinforced model for abstractive summarization. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
7. Kim, H.; Lee, S. Document summarization model based on general context in RNN. *J. Inf. Process. Syst.* **2019**, *15*, 1378–1391.
8. Chung, T.; Liu, Y.; Xu, B. Monotonic alignments for summarization. *Knowl. Based Syst.* **2020**, *192*, 105363. [CrossRef]
9. Gehrmann, S.; Deng, Y.; Rush, A. Bottom-up abstractive summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 4098–4109.
10. Celikyilmaz, A.; Bosselut, A.; He, X.; Choi, Y. 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, New Orleans, LA, USA, 1–6 June 2018; Volume 1, pp. 1662–1675.
11. Pasunuru, R.; Bansal, M. Multi-reward reinforced summarization with saliency and entailment. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; Volume 2, pp. 646–653.
12. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
13. You, Y.; Jia, W.; Liu, T.; Yang, W. Improving abstractive document summarization with salient information modeling. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 2132–2141.
14. Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; Li, H. Modeling coverage for neural machine translation. In Proceedings of the 54th Annual meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; Volume 1, pp. 76–85.
15. Brown, T.B.; Mann, B.; Ryer, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam., P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *arXiv* **2020**, arXiv:2005.14165.
16. Gu, Y.; Hu, Y. Extractive summarization with very deep pretrained language model. *Int. J. Artif. Intell. Appl.* **2019**, *10*, 2732. [CrossRef]
17. Miller, D. Leveraging BERT for extractive text summarization on lectures. *arXiv* **2019**, arXiv:1906.04165.

18. Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D.M.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; Christiano, P. Learning to summarize with human feedback. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020; Volume 33.
19. Sutton, R.S.; McAllester, D.A.; Singh, S.P.; Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In Proceedings of the 12th International Conference on Neural Information Processing Systems; 1999; pp. 1057–1063. Available online: https://proceedings.neurips.cc/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf (accessed on 18 November 2020).
20. Lin, C.Y. ROUGE: A package for automatic evaluation of summaries. In Proceedings of the Workshop on Text Summarization Branches Out, Barcelona, Spain, 25–26 July 2004; pp. 74–81.
21. Williams, R.J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* **1992**, *8*, 229–256. [CrossRef]
22. Rennie, S.J.; Marcheret, E.; Mroueh, Y.; Ross, J.; Goel, V. Self-critical sequence training for image captioning. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1179–1195.
23. Gulcehre, C.; Ahn, S.; Nallapati, R.; Zhou, B.; Bengio, Y. Pointing the unknown words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; Volume 1, pp. 140–149.
24. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer networks. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; Volume 2, pp. 2692–2700.
25. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
26. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. *arXiv* **2018**, arXiv:1802.05365.
27. Dieng, A.B.; Ranganath, R.; Altosaar, J.; Blei, D.M. Noisin: Unbiased regularization for recurrent neural networks. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 3, pp. 2030–2039.
28. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv* **2016**, arXiv:1609.08144.
29. Hermann, K.M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching machines to read and comprehend. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; Volume 1, pp. 1693–1701.
30. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
31. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
32. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; Volume 28, pp. 1310–1318.
33. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
34. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.