J Inf Process Syst, Vol.15, No.6, pp.1378~1391, December 2019
https://doi.org/10.3745/JIPS.02.0123

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Document Summarization Model Based on General Context in RNN

Heechan Kim* and Soowon Lee**

Abstract

In recent years, automatic document summarization has been widely studied in the field of natural language processing thanks to the remarkable developments made using deep learning models. To decode a word, existing models for abstractive summarization usually represent the context of a document using the weighted hidden states of each input word when they decode it. Because the weights change at each decoding step, these weights reflect only the local context of a document. Therefore, it is difficult to generate a summary that reflects the overall context of a document. To solve this problem, we introduce the notion of a general context and propose a model for summarization based on it. The general context reflects overall context of the document that is independent of each decoding step. Experimental results using the CNN/Daily Mail dataset show that the proposed model outperforms existing models.

Keywords

Document Summarization, General Context, Natural Language Processing, Sequence-to-Sequence Model

1. Introduction

Automatic document summarization is a research field focused on how to organize the important contents of a document automatically [1]. In general, automatic summarization can be categorized into extractive summarization, which selects salient words or sentences, and abstractive summarization, which generates word sequences. There are several models for extractive summarization. One is a binary classification model that judges whether a word or a sentence is important using a word scoring method based on features such as word frequency or word position [2]. Another is a graph-based model that represents a document as a graph, a sentence or a word as a node, and the similarity between sentences or words as an edge [3]. In a graph-based model, the importance of a word or sentence is measured using graph centrality.

In abstractive summarization, sequence-to-sequence models have performed well, especially for single document summarization [4-7]. If an input sequence is too long in the sequence-to-sequence model, this sequence is difficult to influence the decoding steps that generate words for the summary. To overcome this problem, attention mechanisms [8,9], which measure the importance of the input words to the output word, have been proposed.

Manuscript received July 23, 2019; accepted November 5, 2019.
Corresponding Author: Soowon Lee (swlee@ssu.ac.kr)
* Dept. of Software Convergence, Soongsil University, Seoul, Korea (heathkim@ssu.ac.kr)
**School of Software, Soongsil University, Seoul, Korea (swlee@ssu.ac.kr)

In neural network models, the word appearance is usually represented by a multinomial distribution. This means that a single word is mapped to each category of the distribution and the number of words represented by the distribution is limited by a hyperparameter. Thus, there is an out-of-vocabulary problem, in that if an output word is not in the category of the distribution, then the word is mapped to a category for a special word called the unknown token.

To avoid the generation of unknown tokens in the decoding steps, models have been proposed that instead point to the next word in the input document. The authors of [6,10,11] proposed models that choose the distribution of the word generation or the distribution of the word pointing with a switching architecture. In [12], a model was proposed that produces a next-word distribution by merging the distributions of the word generation and pointing.

The existing models for abstractive summarization represent the context of a document as the weighted hidden state of each input word. Because the weights change for each decoding step, they reflect only the local context of a document, and they generate a summary that does not reflect the overall context of a document. To solve this problem, we introduce the notion of general context and propose a novel summarization model based on it. The general context is defined as an evenly weighted hidden state of the encoder that is independent of each decoding step.

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 the CNN/Daily Mail dataset are presented in Section 4. Finally, the conclusion and future work are discussed in Section 5.

2. Related Work

The sequence-to-sequence model is widely used in various natural language processing applications such as machine translation [4], question answering [13], and text summarization [5-7]. This model uses a long short-term memory (LSTM) network [14] or gated recurrent unit [15] as a cell of recurrent neural network (RNN) to encode input token sequences into fixed size representations and to generate new tokens from the representations.

Although the structure of an abstractive summarization model is similar to the structure of a machine translation model, the following differences need to be considered. The lengths of the input and output document are almost the same in machine translation. This means that each input word is mapped to a corresponding output word with a one-to-one relationship. In contrast, summarization has the characteristic that an output document (a summary) is very short compared with the input document. Because of this, it is difficult to learn the relationships between input and output words using a vanilla sequence-to-sequence model [4].

The sequence-to-sequence model has three main issues. The first is the loss of information about the input word as the decoding step proceeds. The second is the out-of-vocabulary problem, in which a word not in the vocabulary is found in the input or output document. The third is the repetition problem, in which if a decoded word is the same as some already decoded words, then the next sequence after the word is reproduced.

To solve the loss of information in machine translation, [8,9] proposed an attention mechanism that reflects the importance of input tokens at the decoding step using the hidden states of RNN cells on the

encoder side of sequence-to-sequence model.

To solve the out of vocabulary problem, [6,11] proposed models that choose the distribution of word generation or the distribution of word pointing with a switching architecture based on a multi-layer perceptron with binary output. The authors of [12] proposed an enhanced model (pointer-generator) based on a pointer network [10] and a coverage mechanism [16] to solve the repetition problem as well as the out of vocabulary problem. This model produces the next word distribution as the weighted sum of the distributions of word pointing and word generation using a switching probability. A model that learns whether to use word pointing or word generation through a reinforcement learning process was proposed in [17].

3. Model Architecture

In this section, we describe the details of the notion of a general context and the proposed model. In this paper, we use LSTM as the RNN cell, defined as follows.

$$\begin{aligned} f_t &= \sigma\left(W_f x_t + W_f' h_{t-1} + b_f\right) \\ i_t &= \sigma(W_i x_t + W_i' h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + W_o' h_{t-1} + b_o) \\ g_t &= \tanh(W_g x_t + W_g' h_{t-1} + b_g) \\ s_t &= f_t \odot s_{t-1} + i_t \odot g_t \\ h_t &= o_t \odot \tanh(s_t) \end{aligned} \tag{1}$$

We denote the number of dimensions of the hidden state of the RNN cell as d_h, and the number of dimensions of the word embedding as d_e. We also represent the vocabulary as $V \subset \{w|w \in D\}$, where D represents an input dataset. The proposed model is based on the sequence-to-sequence model with an attention mechanism and a pointer-generator without coverage.

3.1 Sequence-to-Sequence Structure with Attention Mechanism

In the proposed model, the attentional sequence-to-sequence structure consists of a bi-directional encoder and a uni-directional decoder based on LSTM cells. The bi-directional encoder receives the word sequence $w^e_{1\ldots i\ldots I}$ of the input document and uses them to produce hidden states $h^e_{1\ldots i\ldots I}$ at each step. The final hidden state of encoder h^e_i at each step is defined by concatenating the forward and backward hidden states $\vec{h}^e_i$ and $\overleftarrow{h}^e_i$. The uni-directional decoder also receives the word sequence $w^d_{1\ldots t\ldots T}$ of the output document and uses them, along with the previous cell states and hidden states, to produce cell states $s^d_{1\ldots t\ldots T}$ and hidden states $h^d_{1\ldots t\ldots T}$. An associated energy e and attention a are respectively defined as follows [8].

$$e^t_i = v^T \tanh(W_s s^d_t + W_h h^e_i + b) \tag{2}$$

$$a^t_i = \text{softmax}(e^t_i) = \frac{\exp\left(e^t_i\right)}{\sum_k \exp\left(e^t_k\right)} \tag{3}$$

In Eq. (2), $v \in \mathbb{R}^{d_h}, W_s \in \mathbb{R}^{d_h \times d_h}$, and $W_h \in \mathbb{R}^{d_h \times 2d_h}$ are learnable parameters. The number of dimensions of the state of the encoder is twice that of the decoder; therefore, the state of the encoder is reduced by the weight matrix W_h. It can be assumed that the attention values measure the importance of the input words against a currently decoded word. The attention vector $a^t \in \mathbb{R}^I$ can be handled like a multinomial distribution. In the distribution, the input words are mapped to the categories of the distribution in the input order and the probability of the category represents the importance of the input word with respect to the current decoding word.

The context feature is defined as a weighted sum of the hidden states of the encoder, as follows.

$$c_t^L = \sum_{k=1}^{I} a_k^t h_k^e \tag{4}$$

We call the context feature a local context c_t^L to distinguish it from the proposed notion of a general context. In Eq. (4), the local context c_t^L depends on a decoding step t.

The local context c_{t-1}^L at a previous step $t-1$ is additionally given to the decoder's input with the word embedding vector x_t; therefore, new decoder input $\tilde{x}_t$ replaces the existing decoder input x_t, and is defined as follows.

$$\tilde{x}_t = W_x[x_t | c_{t-1}^L] \tag{5}$$

In Eq. (5), $W_x \in \mathbb{R}^{d_h \times (d_e + 2d_h)}$ is a learnable parameter and $[\cdot \mid \cdot]$ represents the concatenation of two vectors.

To simultaneously consider the hidden state of the decoder h_t^d and the local context c_t^L at a current step t, a vocabulary distribution for generating next word P_{vocab} is defined as follows.

$$P_{vocab} = \text{softmax}(W_V(W_v[c_t^L | h_t^d] + b) + b') \tag{6}$$

In Eq. (6), $W_V \in \mathbb{R}^{|V| \times d_h}$ and $W_v \in \mathbb{R}^{d_h \times 3d_h}$ are learnable parameters, and the weight W_V produces the values for the multinomial vocabulary distribution from the previously calculated vector.

The probability $P(w)$ of generating word w is defined as $P_{vocab}(w)$, which is the probability that the specific category is the meaning of word w. A step loss function $l(t)$, which is the negative log likelihood of a target word w_t^*, is defined as follows.

$$l(t) = -\log P_{vocab}(w_t^*) \tag{7}$$

A total loss l^* is computed as an average of the step losses, as follows.

$$l^* = \frac{1}{T} \sum_{t=1}^{T} l(t) \tag{8}$$

3.2 Pointer-Generator

A pointer-generator was proposed by [12] to solve the out-of-vocabulary and repetition problems. A pointer-generator without coverage model consists of two models: the attentional sequence-to-sequence model and a pointer network [10]. This model can handle the unknown token as an existing word in the input words through a weighting based on the attention distribution. A pointer-generator with coverage

(PGC) model is an expanded model that adds the coverage mechanism [16] to the pointer-generator without coverage model to avoid generating a word in the previously decoded words.

To solve the out-of-vocabulary problem, the pointer-generator produces an extended next word probability by merging the two distributions. One is the generation distribution of the word based on vocabulary P_{vocab}. The other is the pointing distribution of a word from the input words based on the attention distribution. Both distributions are merged into an extended distribution P_{pg} weighted by a generation probability p_{gen}, which is defined as follows.

$$p_{gen} = \sigma(w_c^T c_t^L + w_s^T s_t^d + w_h^T h_t^d + w_x^T \tilde{x}_t + b) \tag{9}$$

The generation probability p_{gen} is computed based on the cell state s_t^d and hidden state h_t^d of the decoder, the decoder input $\tilde{x}_t$, and the local context c_t^L. In Eq. (9), $w_c \in \mathbb{R}^{2d_h}$, $w_s \in \mathbb{R}^{d_h}$, $w_h \in \mathbb{R}^{d_h}$, and $w_x \in \mathbb{R}^{d_e}$ are learnable parameters, and $\sigma()$ represents a sigmoid function.

The probability of a next word w in the extended probability $P_{pg}(w)$ is a sum of several probabilities weighted by the generation probability p_{gen}: the generating probability of word w, $P_{vocab}(w)$, and the sum of attention values for word w, which is defined as follows.

$$P_{pg}(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \tag{10}$$

In Eq. (10), if word w is not in the vocabulary, then $P_{vocab}(w)$ is 0. By contrast, if word w is not in the input words, then $\sum_{i:w_i=w} a_i^t$ is 0.

To solve the repetition problem, the coverage mechanism in machine translation [16] is merged into the pointer-generator for abstractive summarization. A coverage vector a_{cv}^t is an accumulated sum of the vector of attentions until the previous step $t-1$ and is defined as follows.

$$a_{cv}^t = \sum_{t\prime=0}^{t-1} a^{t\prime} \tag{11}$$

The coverage a_{cv}^t measures how well the decoded words cover the input words. To reflect the coverage in each decoding step, the associated energy is redefined as follows, where $w_{cv} \in \mathbb{R}^{d_h}$ is a learnable parameter.

$$e'^t_i = v^T \tanh(W_s s_t^d + W_h h_i^e + w_{cv} a_{cv}^t + b) \tag{12}$$

Despite the coverage mechanism, this model still generates the same subsequence, which is an already decoded pattern, repeatedly. To solve this problem, a coverage loss function $l_{cv}(t)$ is defined as follows.

$$l_{cv}(t) = \sum_i \min(a_i^t, a_{cv}^t) \tag{13}$$

To add the coverage loss, a modified step loss function $\tilde{l}(t)$ is defined as follows, where the coverage loss is weighted by a hyperparameter λ_{cv}.

$$\tilde{l}(t) = -\log P_{pg}(w_t^*) + \lambda_{cv} l_{cv}(t) \tag{14}$$

3.3 General Context based Decoding

It is assumed that the process of writing a summary manually roughly consists of two actions. The first is to choose (point or generate) important words that represent the information of the input document. The second is to ensure that the words are true to the overall context of the input document [18]. The previously described local context c_t^l and coverage a_{cv}^t are features that depend on each decoding step t. Because the attentions change for each decoding step, these features reflect only the local context of an input document. Therefore, the overall context of the input document may not be considered enough in each decoding step.

To solve this problem, we introduce the notion of a general context and propose a model based on the general context. The structure of the overall context of the input document consists of a sequence of descriptive information and relationships between the sequential information. In this paper, we define the general context as a n-dimensional vector representation. The general context c^G is defined as an evenly weighted sum of hidden states of the encoder, as follows.

$$c^G = \frac{1}{I}\sum_i h_i^e \tag{15}$$

Thus, the general context is independent from each decoding step.

In the proposed model, the general context affects the distribution of the next word through an attention mechanism at every decoding step. We modify the above-mentioned associated energy by adding the general context c^G, as defined as follows.

$$\tilde{e}_i^t = v^T \tanh(W_s s_t^d + W_h h_i^e + W_{c^G} c^G + b) \tag{16}$$

In Eq. (16), $W_{c^G} \in \mathbb{R}^{d_h \times 2d_h}$ is a learnable parameter, and it reduces the vector dimensionality.

The whole structure of the proposed model is shown in Fig. 1, where the circles, shaded circles, rounded squares, and bold rectangles represent scalars, calculations, vectors, and multinomial distributions, respectively. As shown in Fig. 1, the attention is updated by many features, including the general context c^G, and influences the features of the next step, excluding the general context c^G.

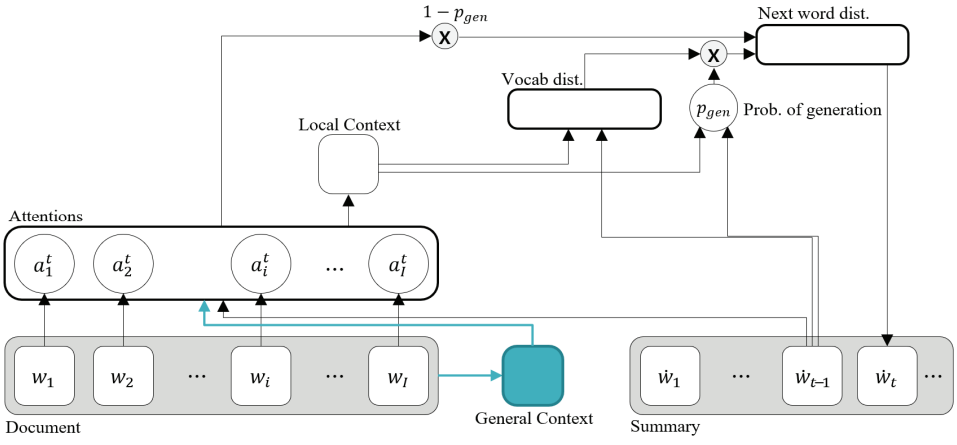

Fig. 1. The structure of the proposed model.

4. Experiments

4.1 Dataset

We used the CNN/Daily Mail dataset [13] to compare the proposed model with others, and we used the highlights of the news, which were written by the author of the news, as a golden standard (GS). Like [12], we used the non-anonymized version of CNN/Daily Mail dataset, and we also split the dataset into 287,226 pairs for training, 13,368 pairs for validation, and 11,490 pairs for testing. In the dataset, documents were space tokenized, and special characters and all punctuation marks in the dataset were removed except for periods, single/double quotations, and parentheses. Stopwords remained in the dataset. Named entity recognition was not applied. We shuffled the training data randomly in every epoch.

4.2 Hyperparameter Setting & Environment

We set the hyperparameters of the proposed model as follows. The number of dimensions of the hidden states was set to 256, the number of dimensions of the word embedding was set to 128, and the vocabulary consists of the top 50k words that appeared frequently in the training dataset. Word embeddings were initialized randomly by a truncated normal distribution with (0, 10^{-4}). We used the Adagrad optimizer [19] with a learning rate of 0.15 and an initial accumulator value of 0.1. We used gradient clipping with a maximum gradient norm of 2.

The hyperparameter for coverage loss λ_{cv} was set to 1, and the initial cell state of the decoder s_0^d was initialized by the last cell state of encoder s_l^e. We trained the model on a single GeForce RTX 2080 Ti GPU with a batch size of 32. The maximum lengths of the input documents and summaries for training, validation, and testing were set to 400 and 100 words, respectively.

To generate summaries, we used the two variants of beam search algorithm. One is the vanilla beam search algorithm, while the other is the modified beam search algorithm introduced in [17], which prevents the generation of repetition words by excluding the word included in the trigram in the already decoded words.

4.3 Model Selection

We trained and tested two variants of the proposed model: a model consisting of the general context only, referred to as the general context based decoder (GCD), and a model using general context, coverage, and coverage loss, called the GCD with coverage (GCDC). In detail, the GCD model consists of the general context and the pointer-generator without coverage model and the GCDC model consists of the general context and the PGC model. The pointer-generator without coverage model consists of Eqs. (1) to (10) and the PGC model consists of Eqs. (1) to (14).

We validated the variant models at every epoch, and we selected the two versions of the models for testing. One is the model that had the lowest validation loss, and the other is the model that achieved the highest ROUGE-2 score [20] in the validation data (Details of the ROUGE metric are introduced in the next section). In automatic summarization, the models were evaluated by the ROUGE score using the generated summaries. To measure the ROUGE score of the model, we should generate the summaries using the model. However, when generating the summary, we must also first select a hyperparameter

specifying the beam width for the beam search algorithm. Generating a summary is a time-consuming task, and there are many combinations that can be generated according to the number of epochs and beam width. To simplify the experiment, we generated the summaries for every epoch using the modified beam search algorithm with a fixed beam width of four and determined the best-performing model (the one with the highest ROUGE-2 score). Next, we generated the summaries using various beam widths from three to six using the best-performing model and selected the best beam width based on the ROUGE-2 score.

Using our experiment settings, it took about three hours to train the model per epoch. As a result of validating the GCD and GCDC models in the case of the validation loss, we chose the models trained for 27 epochs in both cases. The validation losses are shown in Fig. 2, where the red dashed vertical line indicates 27 epochs. For the ROUGE score, we chose the model trained for 24 epochs in the GCD model and chose the model trained for 19 epochs in the GCDC model. The ROUGE scores are also shown in Fig. 2, where the blue dashed vertical lines indicate 19 and 24 epochs. We also chose a beam width of three for the modified beam search algorithm in both the GCD and GCDC models.

To evaluate the performance of the proposed model, it was necessary to generate a summary using the PGC model. To generate the summary, we used the code published by [12] for training and validating the model. This model may generate somewhat different summaries than the origin because of our model selection method.

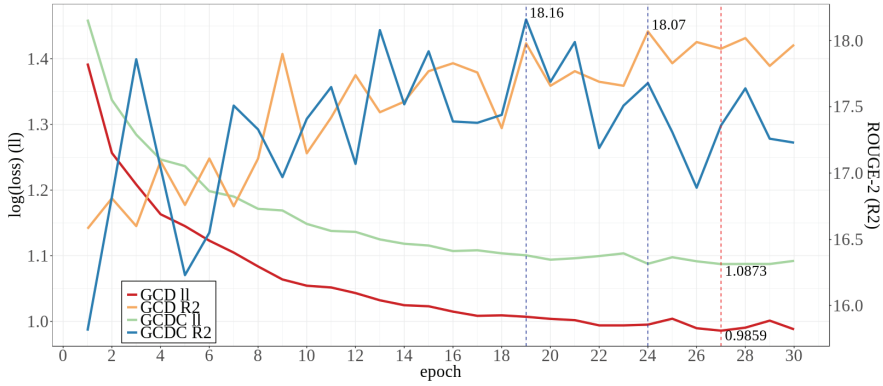

Fig. 2. Validation losses and ROUGE scores of various proposed models per epoch.

4.4 Quantitative Result

To evaluate the performance of the various models, we use the ROUGE metric. ROUGE-1 and ROUGE-2 refer to overlap ratios of unigram and bigram between the generated summary and the golden standard and these are defined as follows.

$$\text{ROUGE} - \text{N} = \frac{\sum_{S \in G} \sum_{g_n \in S} Count_{match}(g_n)}{\sum_{S \in G} \sum_{g_n \in S} Count(g_n)} \tag{17}$$

In Eq. (17), G and S represent the golden standard and the set of n-grams, respectively. In addition, g_n represents an n-gram, and $Count_{match}(g_n)$ represents a function that counts the number of occurrences of the n-gram g_n in both in the golden standard and generated summary.

The ROUGE-L is the F1 score based on the longest common subsequence (LCS) of words between two summaries. It can be computed as follows.

$$\text{ROUGE} - \text{L} = \frac{2R_{lcs}P_{lcs}}{R_{lcs} + P_{lcs}}$$
$$R_{lcs} = \frac{\sum_{g \in G} |\cup_{c \in C} LCS(g,c)|}{m}, P_{lcs} = \frac{\sum_{g \in G} |\cup_{c \in C} LCS(g,c)|}{n} \quad (18)$$

In Eq. (18), C represents the summary generated by the model and c represents the sentence in the generated summary. In addition, $LCS(a,b)$ represents the longest common subsequence of words in sentences a and b. The recall for LCS R_{lcs} is defined as the sum of the number of elements of the union of $LCS(g,c)$ for golden standard G and generated summary C divided by the total number of words m in golden standard G. Similarly, the precision for LCS P_{lcs} is defined as the sum of the number of elements of the union of $LCS(g,c)$ for golden standard G and generated summary C divided by the total number of words n in generated summary C.

In this experiment, we used the `ROUGE-1.5.5` module and the `pyrouge` package with the parameters "-a -n 2." The models used for comparison with the proposed model are a baseline method (Lead-3), which produces the leading three sentences of the document as a summary [7], Nallapati2016 [6], SummaRuNNer-abs and SummaRuNNer [7], Pointer-Generator with and without coverage [12], and two reinforcement learning-based models (RL and ML+RL with intra-attention) proposed by [17].

Table 1 shows the ROUGE F1 scores of variants of the proposed model and other models. The models were sorted in ascending order by the ROUGE-2 score. In Table 1, a dagger mark (†) represents the use of the modified beam search algorithm in the decoding phase. A double dagger mark (‡) represents the validated model based on the ROUGE score.

Table 1. ROUGE F1 scores of various models

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead-3	39.20	15.70	35.50
Nallapati2016	35.46	13.30	32.65
SummaRuNNer-abs	37.50	14.50	33.40
Pointer-Generator without coverage	36.44	15.66	33.42
RL with intra-attention	41.16	15.75	39.08
ML+RL with intra-attention	39.87	15.82	36.90
SummaRuNNer	39.60	16.20	35.30
Pointer-generator with coverage (PGC)	39.53	17.28	36.38
General context based decoder (GCD)	37.45	16.30	34.32
General context based decoder†	39.73	17.56	36.57
General context based decoder†‡	39.93	17.71	36.68
General context based decoder with coverage (GCDC)	38.90	16.83	35.60
General context based decoder with coverage†	39.15	16.96	35.85
General context based decoder with coverage†‡	40.11	17.54	36.66

The GCD†‡ model outperforms the 2017 state-of-the-art abstractive models for two ROUGE scores, yielding values of 39.93, 17.71, and 36.68, for ROUGE-1, ROUGE-2, and ROUGE-L, respectively. The GCD model obtained a ROUGE-2 score that is higher than that of the pointer-generator without coverage

model. The GCDC model obtained a ROUGE-2 score higher than that of the GCD model, but lower than those of the state-of-the-art models. In this case, it seems that the general context acts like some noise with respect to the coverage because the two features simultaneously affect the attention. Compared to the GCD and GCDC models, the ROUGE-2 score of the GCD† and GCDC† models that use the modified beam search algorithm improved by 1.26 and 0.13, respectively. In this case, it seems that the modified beam search plays the role of the existing coverage mechanism and coverage loss. The results indicate that the synergy between the general context and the modified beam search algorithm is strong.

The GCD†‡ and GCDC†‡ models that were validated by the ROUGE score achieved better ROUGE-2 scores that are 0.15 and 0.58 higher, respectively, than the GCD† and GCDC† models that were validated by the validation loss. Because the summary is generated by the beam search algorithm, which is independent of the learning process of the model, it is reasonable to consider the interactions between the model and the beam search algorithm in the decoding phase when selecting the optimal model.

The pointer-generator with coverage model usually generates summaries by highlighting the input documents. Instead of simply pointing out only some words or a small phrase, the model generates summaries similar to extractive summaries by copying long word sequences from the input document.

To determine how the proposed model works for abstractive summarization, we count the n-grams of the copied phrases in the generated summaries and present the results in Fig. 3, in which the horizontal axis represents the sorted n-grams and the vertical axis represents the ratio of the n-grams in the copied phrases.

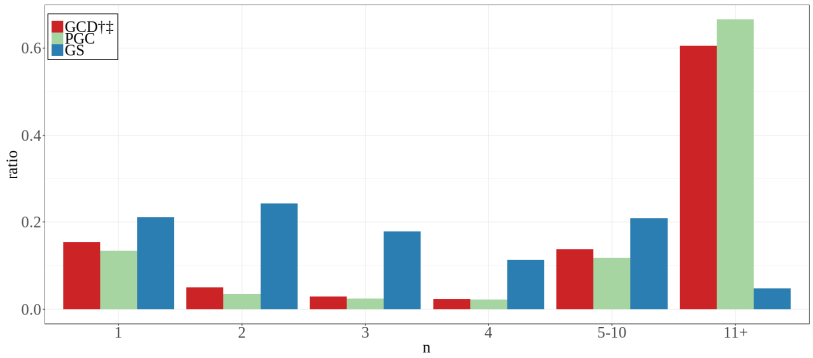

Fig. 3. Ratio of n-grams in copied phrases.

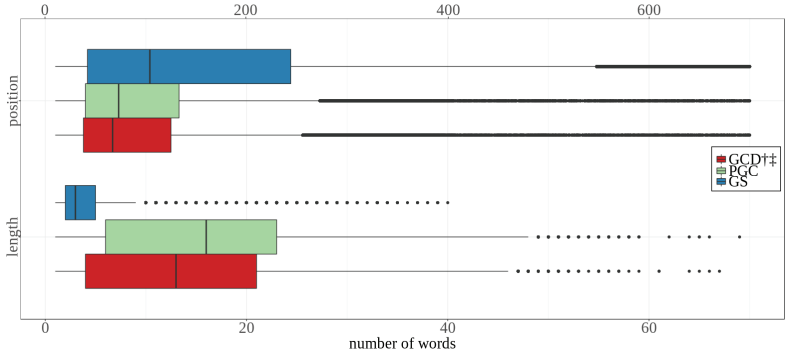

Fig. 4. Distributions of copied sentence positions and their lengths in the summary obtained by various models.

In Fig. 3, the golden standard summaries only copy some words or short phrases, not a whole sentence or more. The GCD†‡ model copies a few more unigrams and bigrams from the input documents and but fewer 11-grams than the PGC model. This could be because the proposed model generates summaries that are more abstractive than those generated by the PGC model. To verify this, we checked the distribution of the sentence lengths in the generated summaries as well as the distribution of the positions of the sentences copied from the input document. The results are shown in Fig. 4, where the vertical lines in the colored boxes from left to right represent the first quartile (Q1), median, and third quartile (Q3). The horizontal line across the box represents the range, which is from Q1 − 1.5 IQR (interquartile range) to Q3 + 1.5 IQR. The black dots represent outliers. The upper x-axis indicates position, while the lower x-axis indicates length. For readability, we only report data below 700 words in position.

The position of the copied sentences in the summary of the GCD†‡ and PGC models, and the golden standard are 116.44±102.98, 115.97±100.76, and 213.92±210.17 words, respectively. Because humans freely refer to any part of the input document, it seems that the mean and standard deviation for the golden standard are higher than those of the others. The positions of the GCD†‡ and PGC models are almost the same, so we used a *t*-test to determine whether the positions of the summary generated by the two models are the same. The result of the *t*-test with a *p*-value of 0.7044 indicates that the positions are not significantly different.

The lengths of the summary generated by the GCD†‡ and PGC models, and the golden standard are 13.97±10.20, 15.78±10.71, and 3.77±3.51 words, respectively. We can see from the values obtained for the golden standard that humans only copy some words from the input document to create the summary. In this case, the lengths of the proposed model and the existing model are a little different. We also used a *t*-test to determine whether the lengths of summary generated by the two models are same; the result of the *t*-test with a *p*-value <0.0001 indicates that the lengths are not same. From these results, we can conclude that our model produces more abstractive summaries than the PGC model because our model copies fewer words from the body of the original text.

4.5 Qualitative Result

To more closely look at the characteristics of the generated summary by the proposed model, we choose several documents as examples from the test data. The criteria for document selection were not only based on the ROUGE-2 score. To avoid trivial cases that consist of whole sentences, like highlighting, we choose two summaries that have good ROUGE scores and consist of combinations of parts of sentences (The parts of the news articles could not be included in the paper because of copyright issues. For readability, we replaced the parts of the news articles with URL links to the articles. A: https://www.dailymail.co.uk/news/article-3048117/Hiker-27-slips-falls-500-feet-death-popular-Hawaii-beauty-spot.html, B: https://www.dailymail.co.uk/news/article-3056287/Ukip-deputy-chairman-vows-step-party-leadership-Farage-forced-ill-health.html). The input document of summary A is news about an accident of a hiker in Hawaii. The input document of summary B is news about a possible change in the UK Independence Party's leader. The summaries generated by the GCD†‡ model are shown in Table 2. In this table, a spanned text means that it is copied from another sentence. A hat (^) means that there is a phrase that was skipped in that sentence. A strikethrough text indicates a repeated phrase.

The first sentence of summary A is a union of three small phrases, but the order of these small phrases

is twisted. The second sentence of the summary is also a union of two phrases in reverse order. The last sentence is copied from the input document. The first sentence of summary B is a union of two phrases in reverse order. First phrase in the sentence has skipped a part of the original sentence. The second sentence of the summary is the first part of the sentence in the input document. The third sentence has a partially repeated phrase. Here, the trigram exception is not applicable because of the difference in the middle word "his." The last sentence also has a skipped part. Compared with the input documents, the sentences in the generated summaries are combinations of several parts of sentences that make sense. In addition, the skipped phrases are unimportant phrase such as "(pictured right)." We think it is a remarkable case.

Table 2. Summaries of the example

ID	Generated summaries
A	Darlene Feliciano, 27, was found unresponsive about 500 feet below a hole in the trail known locally as the Puka - a popular spot among photographers. The 27-year-old was out on the Makapuu 'Tom-Tom' trail overlooking Sea Life Park along Kalanianaole Highway with a male friend when she slipped and fell. Firefighters received a distress call at around 12.45 pm Friday.
B	Suzanne Evans said ^ she would step in as leader if Nigel Farage quits because of ill health. He is suffering from a recurring spinal injury that means he is on medication. ^ But she played down his medical problems and ~~played down medical problems.~~ Miss Evans ^ is the party's deputy chairman.

5. Conclusion

In this paper, to consider the overall context of the document in single document summarization, we proposed a model based on the general context in an RNN. The general context is defined as an evenly weighted sum of the hidden states of the encoder, and it ensures that the words are true to the overall context of the input document.

The proposed model was shown to outperform the 2017 state-of-the-art abstractive models in experiments on the CNN/Daily Mail dataset, achieving a ROUGE-2 score of 17.71. From the experimental results of variants of the proposed model, we conclude that our intuition of the general context is reasonable and the synergy between the general context and the modified beam search algorithm is very effective. To concentrate on validating that the concept of a general context is a worthwhile approach, we chose a simple n-dimensional vector representation for the general context. We reserve the task of expanding the representation of the general context to something such as a graph topology for future work.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. NRF-2017R1D1A1B03031383).

References

[1] D. R. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization," *Computational linguistics*, vol. 28, no. 4, pp. 399-408, 2002.

[2] K. Hong, J. M. Conroy, B. Favre, A. Kulesza, H. Lin, and A. Nenkova, "A repository of state of the art and competitive baseline summaries for generic news summarization," in *Proceedings of the 9th International Conference on Language Resources and Evaluation*, Reykjavik, Iceland, 2014, pp. 1608-1616.

[3] G. Erkan and D. R. Radev, "Lexrank: graph-based lexical centrality as salience in text summarization," *Journal of Artificial Intelligence Research*, vol. 22, no. 1, pp. 457-479, 2004.

[4] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 3104-3112.

[5] S. Chopra, M. Auli, and A. M. Rush, "Abstractive sentence summarization with attentive recurrent neural networks," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, CA, 2016, pp. 93-98.

[6] R. Nallapati, B. Zhou, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany, 2016, pp. 280-290.

[7] R. Nallapati, F. Zhai, and B. Zhou, "SummaRuNNer: a recurrent neural network based sequence model for extractive summarization of documents," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, San Francisco, CA, 2017, pp. 3075-3081.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, 2015.

[9] M. T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 1412-1421.

[10] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Montreal, Canada, 2015, pp. 2692-2700.

[11] C. Gulcehre, S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio, "Pointing the unknown words," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, 2016, pp. 140-149.

[12] A. See, P. J. Liu, and C. D. Manning, "Get to the point: summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, 2017, pp. 1073-1083.

[13] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Montreal, Canada, 2015, pp. 1693-1701.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[15] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, 2014, pp. 1724-1734.

[16] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," in *Proceedings of the 54th Annual meeting of the Association for Computational Linguistics*, Berlin, Germany, 2016, pp. 76-85.

[17] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," in *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[18] V. Anderson and S. Hidi, "Teaching students to summarize," *Educational Leadership*, vol. 46, no. 4, pp. 26-28, 1988.

[19] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121-2159, 2011.

[20] C. Y. Lin, "Rouge: a package for automatic evaluation of summaries," in *Proceedings of the Workshop on Text Summarization Branches Out*, Barcelona, Spain, 2004, pp. 74-81.

Heechan Kim https://orcid.org/0000-0002-7564-2230

He received his B.S. and M.S. degrees from the Department of Computer Science and Engineering, Soongsil University, South Korea, in 2013 and 2015, respectively. Since March 2015, he has been with the Department of Software Convergence, Soongsil University, as a PhD candidate.

Soowon Lee https://orcid.org/0000-0001-5863-1188

He is a Full Professor of the School of Software, Soongsil University, South Korea. He was the Chief Editor of the Korean Institute of Information Scientists and Engineers from 2008 to 2009 and has been the Vice Chairman of the Korea Business Intelligence Data Mining Society since 2008. He received his B.S. in computer science and statistics from Seoul National University in 1982, his M.S. in computer science from KAIST in 1984, and his Ph.D. in computer science from the University of Southern California in 1994. His research interests fall into the areas of data science, text mining, and machine learning.