applied sciences

MDPI

Article

A Topical Category-Aware Neural Text Summarizer

So-Eon Kim, Nazira Kaibalina and Seong-Bae Park *

Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, Korea; sekim0211@khu.ac.kr (S.-E.K.); nazira.kaibalina@khu.ac.kr (N.K.)
* Correspondence: sbpark71@khu.ac.kr

Received: 17 June 2020; Accepted: 3 August 2020; Published: 5 August 2020

Abstract: The advent of the sequence-to-sequence model and the attention mechanism has increased the comprehension and readability of automatically generated summaries. However, most previous studies on text summarization have focused on generating or extracting sentences only from an original text, even though every text has a latent topic category. That is, even if a topic category helps improve the summarization quality, there have been no efforts to utilize such information in text summarization. Therefore, this paper proposes a novel topical category-aware neural text summarizer which is differentiated from legacy neural summarizers in that it reflects the topic category of an original text into generating a summary. The proposed summarizer adopts the class activation map (CAM) as topical influence of the words in the original text. Since the CAM excerpts the words relevant to a specific category from the text, it allows the attention mechanism to be influenced by the topic category. As a result, the proposed neural summarizer reflects the topical information of a text as well as the content information into a summary by combining the attention mechanism and CAM. The experiments on The New York Times Annotated Corpus show that the proposed model outperforms the legacy attention-based sequence-to-sequence model, which proves that it is effective at reflecting a topic category into automatic summarization.

Keywords: text summarization; class activation map; attention mechanism; topic category; text readability

1. Introduction

Due to the fast generation of text data by various media, it is becoming nearly impossible for a person to read all texts directly, which leads to the need for automatic text summarization. Thus, automatic text summarization is nowadays used widely in many applications of natural language processing, but is divided into two types depending on how a summary is generated. One type is the extractive summarization [1–4] which selects important words, phrases, or sentences from an original text and combines them to create a summary. Since it uses the content of the original text directly, it is regarded to be a relatively easy way to summarize a text. However, the summaries by extractive summarization are likely to have low cohesion or readability. The other is the abstractive summarization [5–7] which generates summaries by comprehending the content of an original text. Since it is based on an overall understanding of the context, the problem of generating inconsistent summarization rarely occurs. In addition, the summaries by this type are likely to have high cohesion or readability, since abstractive summarization generates summaries by paraphrasing. However, abstractive summarizers have to write concise sentences through understanding of whole content. Therefore, it is in general more difficult to design an abstractive summarizer than to design an extractive one.

Because of low difficulty, the early research on automatic text summarization has focused mainly on the extractive summarization [1–4]. However, vigorous research about sequence-to-sequence

models [8] makes it easy to generate a context vector that contains an entire content of an original text, where a sequence-to-sequence model is a neural network which is composed of an encoder and a decoder. When a sequence-to-sequence model is used as a summarization model, the encoder takes a sequence of original text as its input and generates a context vector as its output. Since the context vector is created by referencing all sequences of the input, the context vector encompasses the entire content of the input. Then, the decoder takes the context vector as its input and generates a summary as its output. This operation of the sequence-to-sequence model fits well with the abstractive summarization. As a result, the number of studies on neural abstractive summarization increases gradually [5,7,9–11].

One problem of sequence-to-sequence models for abstractive summarization is that they do not reflect any topical information into summarization even though most texts have one or more topical themes. Especially, the topic category of a text exists actually, though it is not revealed explicitly. As a result, the summaries by human beings are affected by a topic category. Table 1 shows two example summaries by human beings that prove that a topic category is critical information for text summarization. In this table, ***S*** and ***R*** indicate a source text and its summarized text, respectively. The first example is about music. The phrases related with music such as "two concerts at the Metropolitan Museum of Art" and "pianist" remain in the summary. On the other hand, the second example is about politics. Thus, the words such as "Governor" and "campaign" survive in the summary. As shown in these examples, it is of importance to consider a topic category in summarizing a text.

Table 1. Two examples on text summarization that show the importance of topic category.

S: A review on Tuesday about two concerts at the Metropolitan Museum of Art misstated the surname of a pianist at two points. As noted elsewhere in the review, he is John Bell Young, not John Bell. ***R***: Review of two concerts at Metropolitan Museum of Art: Pianist's name is John Bell Young.
S: Most campaigns slow down in the heat of the summer, but voters itching for the governor's race to pick up speed can now turn to the Internet. Governor Whitman unveiled a campaign web site yesterday that offers video and audio clips as well as the text of speeches, position papers and a biography. The web page, at www.christie97.org, even shows supporters how to make their own campaign buttons. ***R***: Governor Whitman unveils campaign site on Internet.

This paper proposes a novel neural summarizer which compresses a text according to its topic category. The class activation map (CAM) [12] is adopted to incorporate a topic category into a summary. It is obtained by applying the global average pooling (GAP) to a convolutional neural network (CNN) so that it predicts which parts of an input instance should be highlighted to classify the instance to a specific topic category. The proposed summarizer is basically a neural encoder-decoder model of which encoder and decoder are a long short-term memory (LSTM). After a text-CNN identifies the topic category of a given text, the category is used to generate the CAM of the text. Since the map identifies relevant words to the topic category, it is regarded as a sort of attention mechanism for an encoder-decoder model. The LSTM decoder of the proposed summarizer produces a summary using the context vector from the LSTM encoder like other encoder-decoder models. The main difference from other models is that the proposed decoder uses the combination of class-activation map and legacy attention as attention weights in writing a summary. The class-activation map provides relevancy weights to the words in the original text for the topic category and legacy attention [13] does relevancy weights for generating a summary. As a result, the summary does not simply summarize a text but also reflects the topic category into a summary.

The contribution of this paper is three-folds. Note that the proposed model learns the relation between a text and its topic category. Thus, the first contribution is that it can produce a summary for a desired topic category without learning from topic-specific summaries, which leads to saving on the costs of preparing topic-specific summaries. The second is that CAM is applied to text summarization for the first time. The proposed model combines CAM and the attention mechanism

within an encoder-decoder model. The last is that it is proven empirically that the use of topic category increases the summarization performance. According to the experiments on the New York Times corpus, the proposed topic category-aware summarizer outperforms the legacy attention-based sequence-to-sequence model.

The rest of this paper is organized as follows. Section 2 introduces the previous studies on automatic text summarization. Section 3 presents the proposed approach to extracting topical information from a text, and Section 4 explains the proposed automatic text summarization model which uses the topical information. Section 5 gives the experimental results, and finally Section 6 draws conclusions and future work.

2. Related Work

The main approach to text summarization at the early stage was the extractive summarization which extracts key words or sentences from an original text and then generates a summary by rearranging them. Before appearance of neural sequence-to-sequence models, two types of methods were mainly used to sort out important sentences from an original text. One is to use specific sentence features for scoring sentences such as term frequency, TF·IDF weight, sentence length, and sentence position [14], and the other is to adopt a specific ranking method such as a structure-based, a vector-based, or a graph-based method [15]. After development of the neural summarization [8,16], there have been many studies that use a neural network for sentence modeling and scoring [1,17] or use a neural sequence classifier to determine which sentences should be included in the summary [10]. However, the extractive summarization suffers from two kinds of problems. One is that it generates a grammatically incorrect output when over-extraction occurs, and the other is that the summary does not have a natural flow when uncorrelated words or sentences are extracted.

The abstractive summarization which contrasts with the extractive summarization generates a summary through understanding an original text and producing abstract sentences that reflect the understanding. This kind of summarization has been studied relatively less than the extractive summarization because of its difficulty. However, the research on abstractive summarization has been active after Rush et al. applied neural networks to abstractive summarization and showed significant performance improvement over the extractive summarization [9]. Chopra et al. used a recurrent neural network (RNN) for the first time to generate a summary and proved empirically that RNN is more suitable for abstractive summarization than feed-forward networks [18]. On the other hand, Nallapati et al. focused on the sequence processing ability of RNN and thus proposed a sequence-to-sequence model in which a RNN is used as both an encoder and a decoder [2]. While RNNs show promising results, they suffer from the out-of-vocabulary (OOV) problem and tend to generate the same words repeatedly. Thus, See et al. solved the OOV problem by the pointer network [19] and the repetition problem by coverage mechanism [20] which discourages repetitions by keeping track of what has been summarized [5].

Topic category is regarded as a key information of a text. Thus, it is not unusual to improve the summary quality by controlling a summarizer to reflect a topic category into a summary [21–24]. Krishna et al. proposed a neural summarizer that takes an article along with its topic as its input [21], but the summarizer has a problem of requiring multiple summaries of a document for every topic. It is expensive to prepare the data in which a single text has multiple summaries for various topics, since it is a very tedious and laborious task to write summaries manually for every topic. Thus, they proposed a method to create such data artificially in this work. However, the quality of automatically generated summaries is less reliable than that of manual summaries. Therefore, in this paper, we propose a model that reflects a topic into a summary, but does not require a different summary for every different topic.

3. Class Activation Map for Text-CNN

3.1. Text-CNN

Convolutional neural networks (CNN) have been mainly applied to image processing. Affected by great success of CNN in many applications of computer vision, Kim recently proposed the text-CNN [25] to apply CNNs to text classification such as sentiment analysis and sentence classification. The text-CNN consists of a single convolution layer, a single max pooling layer, and one or more fully connected layers.

Let $\mathbf{X} = \langle x_1, x_2, \ldots, x_n \rangle$ be a text where x_i is the i-th word of $\mathbf{X}$, and $\mathbf{x}_i \in \mathbb{R}^d$ be the d-dimensional vector representation to x_i. When $\mathbf{x}_{i:i+k_h-1}$ is a word window of size k_h from x_i to x_{i+k_h-1}, it is represented as

$$\mathbf{x}_{i:i+k_h-1} = \mathbf{x}_i \oplus \ldots \oplus \mathbf{x}_{i+k_h-1},$$

where $\oplus$ is the concatenation operator. The convolution layer creates a feature map $\mathbf{C}$ of which element is again a vector generated by sliding $\mathbf{X}$ with a k_h-size window and a kernel, where there are H kinds of window sizes and J kinds of kernels. That is, $\mathbf{c}_j^{k_h}$, the element of $\mathbf{C}$ with a window size k_h ($1 \leq h \leq H$) and the j-th ($1 \leq j \leq J$) kernel is

$$\mathbf{c}_j^{k_h} = \left[c_j^{k_h}(1), c_j^{k_h}(2), \ldots, c_j^{k_h}(n - k_h + 1) \right]. \tag{1}$$

Here, $c_j^{k_h}(i)$ is computed by

$$c_j^{k_h}(i) = f(\mathbf{w}_j^{k_h} \cdot \mathbf{x}_{i:i+k_h-1} + b),$$

where $b \in \mathbb{R}$ is a bias, f is a non-linear activation function such as ReLU, and $\mathbf{w}_j^{k_h} \in \mathbb{R}^{k_h \cdot d}$ is the j-th kernel applied to $\mathbf{x}_{i:i+k_h-1}$. Since the number of $\mathbf{c}_j^{k_h}$'s is $H \cdot J$, the feature map $\mathbf{C}$ becomes

$$\mathbf{C} = \left[\mathbf{c}_1^{k_1}, \ldots, \mathbf{c}_J^{k_1}, \mathbf{c}_1^{k_2}, \ldots, \mathbf{c}_J^{k_2}, \ldots, \mathbf{c}_1^{k_H}, \ldots, \mathbf{c}_J^{k_H} \right], \tag{2}$$

where $|\mathbf{C}| = H \cdot J$.

The max pooling layer represents the final feature vector

$$\hat{\mathbf{C}} = \left[\hat{c}_1^{k_1}, \ldots, \hat{c}_J^{k_1}, \hat{c}_1^{k_2}, \ldots, \hat{c}_J^{k_2}, \ldots, \hat{c}_1^{k_H}, \ldots, \hat{c}_J^{k_H} \right].$$

Here, $\hat{c}_j^{k_h} = \max(\mathbf{c}_j^{k_h})$ where $\mathbf{c}_j^{k_h}$ is defined in Equation (1). From the vector $\hat{\mathbf{C}}$, the last fully connected softmax layer determines the class label of the input $\mathbf{X}$. The text-CNN is trained with a training set $D = \{(\mathbf{X}_1, \tau_1), \ldots, (\mathbf{X}_N, \tau_N)\}$ where $\mathbf{X}_m$ is the m-th text and τ_m is the topic category of $\mathbf{X}_m$. It is trained to minimize a risk functional of the cross-entropy loss over D.

3.2. Class Activation Map

Class activation map (CAM) [12] is one of the interpretations of convolutional neural networks (CNN). CAM is created by introducing the global average pooling (GAP) layer into a CNN instead of the max pooling layer. Thus, the CNN for CAM consists of a convolution layer, a global average pooling layer, and a fully connected layer.

Figure 1 depicts the general architecture of CAM. This architecture is identical to the text-CNN except the global average pooling layer. That is, from the vector **C** in Equation (2), the final feature vector **g** of CAM

$$\mathbf{g} = \left[g_1^{k_1}, \ldots, g_J^{k_1}, g_1^{k_2}, \ldots, g_J^{k_2}, \ldots, g_1^{k_H}, \ldots, g_J^{k_H}\right]$$

is obtained by computing $g_j^{k_h}$ as

$$g_j^{k_h} = \sum_{u=1}^{n-k_h+1} c_j^{k_h}(u). \tag{3}$$

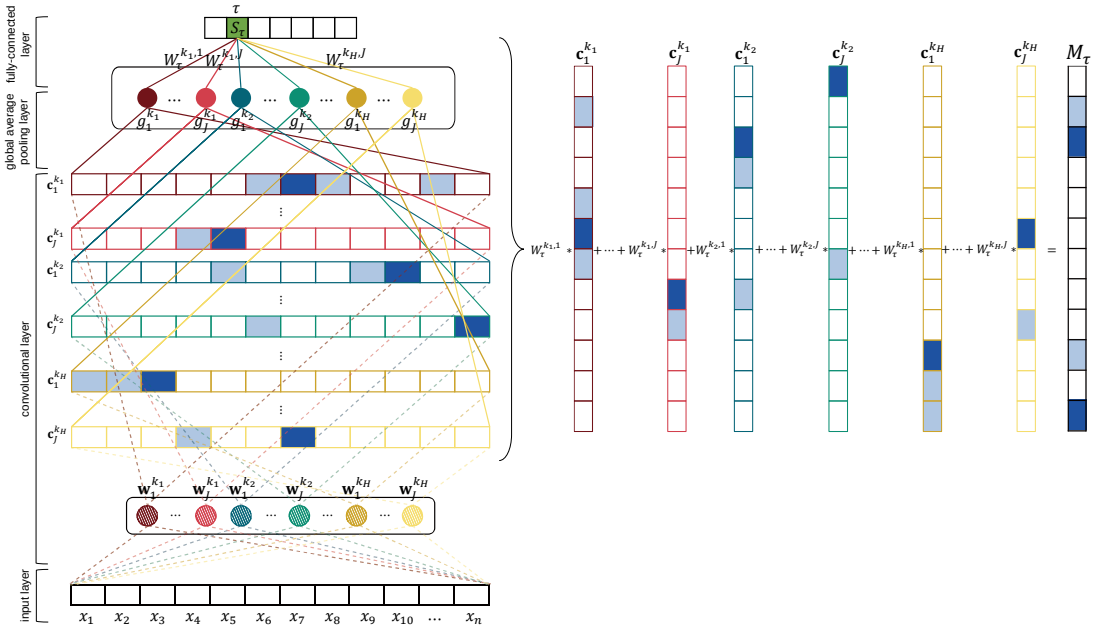

Figure 1. The architecture of class activation map (CAM) applied to text classification.

The possibility of each class τ for the input **X** by CAM is determined through a fully connected layer. That is, the possibility S_τ is

$$S_\tau = \sum_{h,j} W_\tau^{k_h,j} g_j^{k_h},$$

where $W_\tau^{k_h,j}$ is a weight of $g_j^{k_h}$ to a class τ. Since a bias term does not affect classification performance significantly, there is no bias term in S_τ. CAM is trained in the same manner to training text-CNN with the training set D.

From Equation (3), S_τ can be formulated with $c_j^{k_h}$'s as

$$\begin{aligned} S_\tau &= \sum_{h,j} W_\tau^{k_h,j} g_j^{k_h} \\ &= \sum_{h,j} W_\tau^{k_h,j} \sum_u c_j^{k_h}(u) \\ &= \sum_u \sum_{h,j} W_\tau^{k_h,j} c_j^{k_h}(u). \end{aligned}$$

By defining

$$M_\tau(u) = \sum_{h,j} W_\tau^{k_h,j} c_j^{k_h}(u), \tag{4}$$

S_τ becomes $S_\tau = \sum_u M_\tau(u)$. Thus, $M_\tau(u)$ indicates the importance of a word u in classifying $\mathbf{X}$ to class τ. To complete $M_\tau(u)$, $\mathbf{c}_j^{k_h}$ in Equation (1) should be of the same size regardless of window size. For this, the class activation map is upsampled to the length of the input text $\mathbf{X}$.

CNN is known to have an object detection capability in computer vision [26], but is poor at detecting the edge of objects due to its max pooling. On the other hand, the global average pooling enforces CAM to detect the inner side of objects. If CAM is applied to a text, the words or phrases that have a high impact on a target class are found out. Therefore, CAM can be regarded as word weights for the target class.

4. Topic Category-Aware Neural Summarizer

The proposed topic category-aware neural summarizer is basically a sequence-to-sequence model which consists of two recurrent neural networks (RNNs) of an encoder and a decoder as shown in Figure 2. The encoder takes the words of an original text $\mathbf{X} = \langle x_1, \ldots, x_n \rangle$ and generates a context vector z which implicitly contains the content of $\mathbf{X}$. Since it is a RNN, its hidden state vector at time t, h_t, is computed with the t-th word x_t and the hidden state vector h_{t-1} at time $t-1$ by

$$h_t = f_1(x_t, h_{t-1}),$$

and the context vector z is computed with all hidden state vectors by

$$z = f_2(\{h_1, h_2, \cdots, h_n\}), \tag{5}$$

where f_1 and f_2 are nonlinear activation functions.

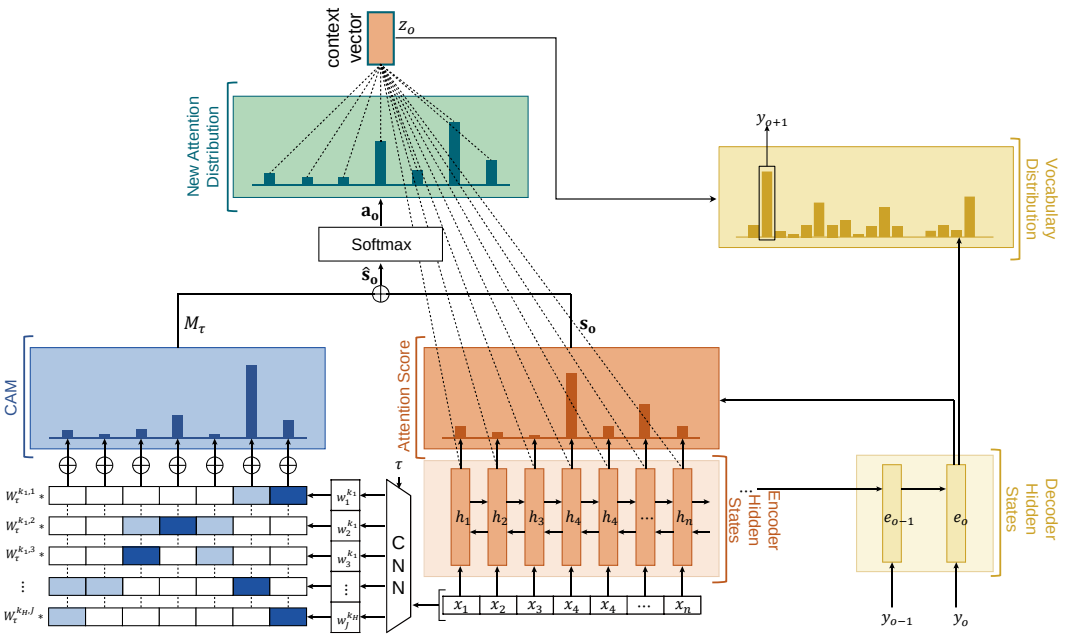

Figure 2. Overall structure of topic category-aware neural summarizer.

The decoder generates a sequence of words by predicting the conditional probability of the target word y_o at time o with the context vector z, its hidden state vector e_{o-1} at time $o-1$, and the output of the decoder y_{o-1} at time $o-1$. That is,

$$p(y_o|\{h_1, h_2, \cdots, h_n\}, z) = f_3(z, y_{o-1}, e_{o-1}), \tag{6}$$

where f_3 is a nonlinear activation function. The current hidden state vector e_o is also updated with z, e_{o-1}, and y_{o-1} by

$$e_o = f_4(z, y_{o-1}, e_{o-1}), \tag{7}$$

where f_4 is a nonlinear activation function.

The attention mechanism [13] is a patent way of leveraging the performance of vanilla sequence-to-sequence models by updating z in Equation (5) at every step of the decoder. When the hidden state vector of the decoder at time o is e_o, the attention score s_o^t represents the similarity between e_{o-1} and h_t, the t-th hidden state of the encoder. Since there are same number of attention scores with the hidden states of the encoder at every time o, the attention score $\mathbf{s}_o$ is represented as

$$\mathbf{s}_o = \left[s_o^1, s_o^2, \ldots, s_o^{n-1}, s_o^n\right].$$

It is computed using an alignment model $\alpha(e_{o-1}, h_t)$. The proposed model adopts, as its alignment model, $v^T \tanh(Wh_t + Ve_{o-1})$ where v, V, and W are all learnable parameters following the study of See et al. [5].

Since $M_\tau(t)$ in Equation (4) is the importance of a word t when the topic of $\mathbf{X}$ is τ, it can be used as another attention score. Thus, in order to reflect both $M_\tau(t)$ and s_o^t into generating a summary, the modified attention score $\hat{s}_o^t$ is defined as

$$\hat{s}_o^t = s_o^t + \varepsilon \cdot M_\tau(t), \tag{8}$$

where ε is a hyper-parameter for avoiding extreme attention scores. In this equation, s_o^t represents how much the t-th input word should be reflected in generating the o-th word, and $M_\tau(t)$ does how much it is related to the topic τ. Thus, if a word in the original text is significant, but has nothing to do with the topic, then it is adjusted to have less effect to the summary. Similarly, the word which is deeply related to the topic but less important in the text would also affect less to the summary. As a result, the decoder focuses more on the topic-related and important words of the original text.

The final attention weight vector $\mathbf{a}_o = \left[a_o^1, a_o^2, \ldots, a_o^n\right]$ at time o is calculated by applying the softmax function to $\hat{\mathbf{s}}_o = \left[\hat{s}_o^1, \hat{s}_o^2, \ldots, \hat{s}_o^n\right]$, where

$$a_o^t = \frac{exp(\hat{s}_o^t)}{\sum_{k=1}^n exp(\hat{s}_o^k)}. \tag{9}$$

Then, instead of Equations (5) and (7), the context vector z_o and the hidden state vector e_o of the decoder is computed as

$$\begin{aligned} z_o &= \sum_{t=1}^{n} a_o^t \cdot h_t, \\ e_o &= f_4(z_o, y_{o-1}, e_{o-1}). \end{aligned}$$

The next target word y_{o+1} is chosen from the distribution of candidate words which is computed as

$$p(y_{o+1}|\{h_1, h_2, \cdots, h_n\}, z_o) = f_3(z_o, y_o, e_o),$$

where f_3 is an activation function in Equation (6).

5. Experiments

5.1. Data Set

The New York Times Annotated Corpus (NYT Corpus) [27] which is a collection of The New York Times articles from 1 January 1987 to 19 June 2007 is a representative corpus for text summarization which has a category and a summary for each article. The number of articles in this corpus is 1,238,889, and each article in this corpus is annotated with a summary and two types of topic categories. One topic type is a general online descriptor and the other is online section. The general online descriptor is known to be automatically assigned by an algorithm and then manually verified by a production staff. The online section of an article shows where the article is posted on `NYTimes.com`, and represents a coarse topic. In the experiments below, the general online descriptor is adopted as a topic category of articles, since it classifies the articles more finely and precisely than the online section. The NYT Corpus has 776 general online descriptors, but only top seven descriptors are used for the sake of learning efficiency. The descriptors used are "Politics and Government", "Finances", "Medicine and Health", "Books and Literature", "Music", "Baseball", and "Education and Schools".

Several pre-processing steps are applied to prepare a data set for text summarization from the corpus. Note that all articles in the corpus do not have a summary. Thus, the articles without a summary are first excluded from the data set. In addition, the articles whose length is less than 15 words or which have a summary of fewer than three words are excluded since such articles do not provide enough context. The number of articles which have a summary is 460,419, but only 183,721 articles remain after these pre-processing steps. Then, all articles and summaries are limited to have up to 800 and 100 words respectively following the study of Paulus et al. [6]. All alphabets are represented as lower-cases and some garbage words in the summaries are removed. The markers of "photo", "graph", "chart", "map", "table", "drawing" at the end of the summaries are also removed. Table 2 shows a simple statistics on the data set after all these-processing steps. The average number of tokens in articles is 500.75 and that in summaries is 40.74. The data set is further split into a training, a validation, and a test set after random shuffling as shown in Table 3. 80% (146,977 articles) of the articles in the data set is used as a training set, 10% (18,372 articles) as a validation set, and the remaining 10% (18,372 articles) as a test set.

Table 2. Simple statistics on the New York Times (NYT) Corpus.

Category	No. of Articles	Avg. Tokens in Articles	Avg. Tokens in Summaries
Politics & Government	66,698	500.11	47.62
Finances	28,540	479.65	46.46
Medicine & Health	25,516	467.85	45.04
Books & Literature	19,807	545.64	27.67
Music	17,138	504.06	27.83
Baseball	13,051	623.50	24.23
Education & Schools	12,971	419.51	38.05
Total	183,721	500.75	40.74

Table 3. Smple statistics on the training, validation, and test set.

Category	No. of Articles	Avg. Tokens in Articles	Avg. Tokens in Summaries
Training set	146,977	501.05	40.78
Validation set	18,372	497.04	40.43
Test set	18,372	501.01	40.73
Total	183,721	500.75	40.74

5.2. Experiment Settings

The proposed model consists of two sub-models that are the CAM-generation model and the summarization model. Both models are trained with the same training set, and verified with the same test set. The vocabulary is built with the tokens which appear at least two times in articles and summaries. As a result, the vocabulary sizes for articles and summaries are 168,387 and 61,375, respectively.

5.2.1. CAM-Generation Model

A vanilla text-CNN is adopted to determine the topic category of articles. 128-dimensional word embedding vectors are used for the word-embedding layer. The word embedding vectors are learned from scratch during training. Kernel sizes of three, four, and five are used for one hundred kernels. The Adam optimizer [28] is used for optimizing the CNN with the learning rate of 1×10^{-2}. The cross entropy loss is chosen as an objective function.

5.2.2. Summarization Model

The proposed sequence-to-sequence model in Figure 2 consists of a bi-directional RNN encoder and a uni-directional RNN decoder. Both the encoder and the decoder have a 128-dimensional hidden layer, and Gated Recurrent Unit [29] is adopted for their cells. 128-dimensional word embedding vectors are used for the word-embedding layer as similar as the CAM-Generation Model. The model uses the teacher forcing algorithm [30] at its training step with 50% probability. The teacher forcing algorithm forces the decoder to adopt a ground-truth word with a certain probability instead of generating a word using the previously generated word when generating a current token word. The dropout [31] is applied to both the encoder and the decoder with 50% probability. The proposed model is trained also with the Adam optimizer with learning rate of 1×10^{-3}, and the cross entropy loss is used for its objective function. At test time, a summary is generated using the beam search with a beam size of four.

5.3. Baseline and Evaluation Metric

The baseline models of the proposed model are the vanilla sequence-to-sequence model (seq-baseline) and the vanilla attention-based sequence-to-sequence model (attn-baseline). All settings of the two baselines are the same as the proposed model except the attention. The seq-baseline does not use any attention and the attn-baseline does not adopt the CAM-based attention. That is, in the attn-baseline, the attention weight is calculated as

$$a_o^t = \frac{exp(s_o^t)}{\sum_{k=1}^{n} exp(s_o^k)},$$

instead of Equation (9).

ROUGE [32], a set of simple metrics, is adopted for the evaluation of the proposed model. Especially, ROUGE-1, ROUGE-2 and ROUGE-L within ROUGE are used. ROUGE-1 measures the unigram-overlap, ROUGE-2 measures the bigram-overlap, and ROUGE-L measures the longest common subsequence between a reference summary and a generated summary. The original ROUGE measures the performance of summary sentences based on recall so that the longer a summary is, the higher score it obtains. To solve this problem, Nallapati et al. proposed the full-length F1 variant ROUGE which puts some penalty on longer summary sentences [2]. Therefore, the full-length F1 variant ROUGE is adopted as an evaluation metric.

5.4. Experimental Results

Figure 3 shows the ROUGE-1 change according to ε values in Equation (8). When ε is zero, CAM is not reflected at all into attention weights. Then, the proposed model becomes equivalent to the attn baseline model. When ε increases from 0 to 0.4, the ROUGE-1 score increases monotonically. On the

other hand, when ε is larger than 0.4, the score decreases steeply. This implies that the effect of CAM is maximized at $\varepsilon = 0.4$. Thus, all experiments below are performed with $\varepsilon = 0.4$.

Table 4 shows that the proposed model outperforms the two baselines. The seq-baseline achieves 39.6703 of ROUGE-1, 17.4243 of ROUGE-2, and 31.0626 of ROUGE-L and the attn-baseline achieves 40.3402, 23.1140, and 36.2059, respectively. The attn-baseline shows 0.6699, 5.6897, and 5.1433 higher performances than the seq-baseline. This is because the attn-baseline uses different context vectors for every position of word generation while the seq-baseline clings to an identical one. On the other hand, the proposed model achieves 43.2400 of ROUGE-1, 24.6724 of ROUGE-2, and 37.8523 of ROUGE-L, which is 2.8998, 1.5584, and 1.6464 higher than the attn-baseline. The main difference between the attn-baseline and the proposed model is whether or not CAM is reflected into the attention, and ROUGE measures the overlaps between the summary written by a human being and the summary generated by a machine. This implies that the reflection of topical information to the attention helps generate a summary that is closer to the human-made summary. Another thing to note is that the performance difference between the proposed model and the attn-baseline in ROUGE-1 is larger than those in ROUGE-2 and ROUGE-L. This is because CAM assigns its attention at a word level. Since CAM is originated from computer vision, it regards a word as a pixel in an image. That is, it focuses on the topic-related words independently. As a result, the surrounding words of a topic-related word are not paid sufficient attention to by CAM.

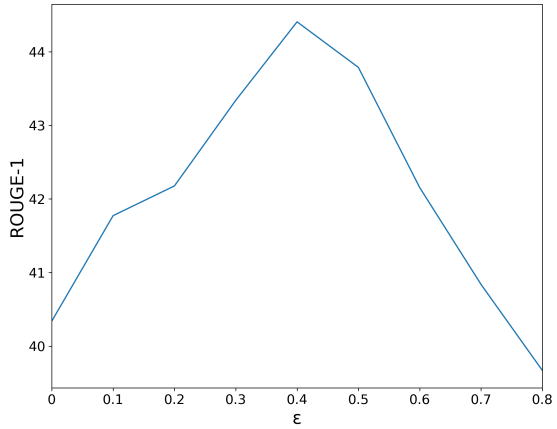

Figure 3. The change of ROUGE-1 scores according to ϵ on the validation set.

Table 4. ROUGE score on the test set.

Model	ROUGE		
	1	**2**	**L**
Seq-Baseline	39.6703	17.4243	31.0626
Attn-Baseline	40.3402	23.1140	36.2059
Proposed Model	**43.2400**	**24.6724**	**37.8523**

There are several previous studies that adopted the NYT corpus for learning their own summarization model. Table 5 compares the proposed model with the previous models by Paulus et al. [6], and by Celikyilmaz et al. [33]. It shows the result obtained by training each model using the same data as the proposed model. Paulus et al. achieved 42.53 of ROUGE-1, 24.62 of ROUGE-2, and 36.18 of ROUGE-L and Celikyilmaz et al. achieved 43.01, 25.12, and 36.93, respectively. The proposed model shows the highest scores in all metrics except ROUGE-2. Thus, this table proves that the proposed model generates high-quality summaries.

Table 5. Comparison of the proposed model with previous models trained with the NYT corpus.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Paulus et al. [6]	42.53	24.62	36.18
Celikyilmaz et al. [33]	43.01	**25.12**	36.93
Proposed Model	**43.24**	24.67	**37.85**

Figure 4 depicts the attention heatmaps between a source text and a generated summary, where Figure 4a is generated by the proposed model and 4b is generated by the attn-baseline. The X-axis of the heatmap represents the tokens of a source text and the Y-axis is the tokens of a summary. The closer the color of the heatmap is to yellow, the higher the attention weight is. Thus, the color of purple implies low attention weight. When comparing the token "department", the second token of the source text in both figures, its color in Figure 4a is nearer to yellow than that in Figure 4b, which implies that the importance of "department" increases by CAM when the first token is generated. Since the category of the source text is "Politics and Government", CAM enhances its attention to the token "department". As a result, the phrase "defense dept" appears in the summary of the proposed model. However, it does not appear in the summary of the attn-baseline since it does not consider the topic category of the source text. In consequence, the summary by the proposed model is more similar to the golden target "defense dept confirms death of service member in iraq" than that of the attn-baseline.

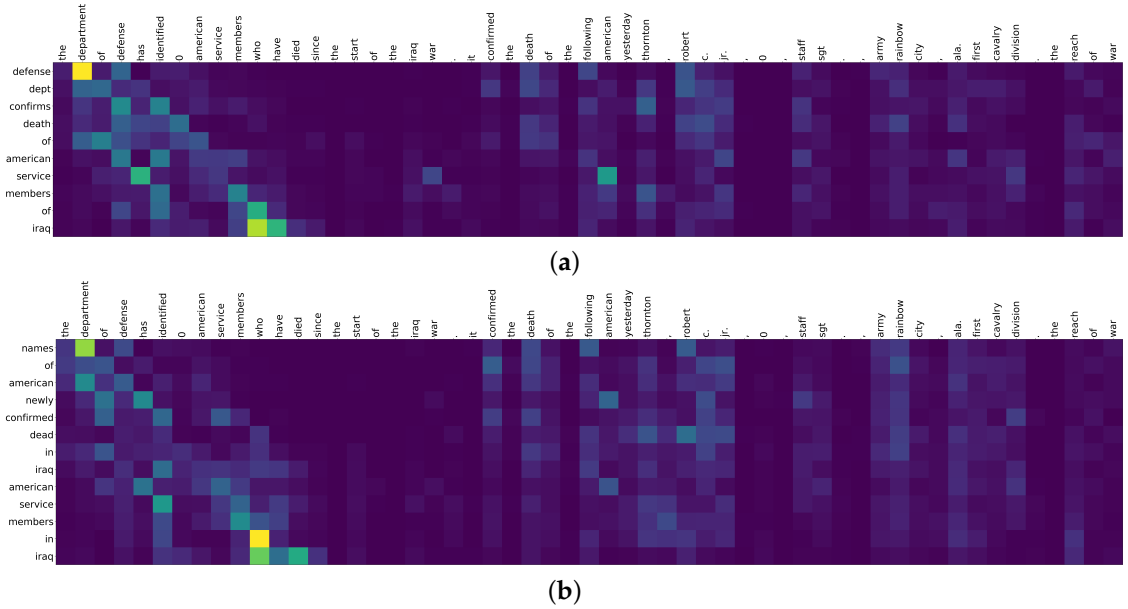

Figure 4. The attention heatmaps of the proposed model and the vanilla attention-based sequence-to-sequence model between a source text and a generated summary. (**a**) A summary by the proposed model; (**b**) a summary by Attn-Baseline.

6. Conclusions

In this paper, we have proposed a novel model to generate a summary from a long text that reflects a topic category into the summary. Topic category is a key information of texts and thus human beings regard it critical when summarizing a text. Therefore, an automatic summarizer that considers a topic category in summarizing a text would generate a summary closer to a human-generated one. In order to apply a topic category to summarizing a text, the proposed model adopts a class activation map (CAM) as topical information of the text, since the CAM generated by a CNN with global average pooling represents the word weights to the topic category. Therefore, a weighted sum of the CAM and the original attention score has been proposed as a new attention score to reflect both the traditional and the topical word importance into generating a summary. As a result, the decoder of

the proposed model focuses more on the words related to the topic category, and reflects the category into the summary.

According to the experimental results on the NYT corpus, the proposed model achieves higher scores in ROUGE-1, ROUGE-2 and ROUGE-L than its two baselines that are a vanilla sequence-to-sequence model (seq-baseline) and a vanilla attention-based sequence-to-sequence model (attn-baseline). These outcomes imply that reflecting topical information into the attention score helps generate a summary that follows the source text closely. In addition, it is also shown through the heatmaps of the proposed model and the attn-baseline that the proposed model generates topic-related words in its summary by focusing more on the topic-related words of a source text, which leads to generation of a summary closer to the human-generated one than the attn-baseline.

One thing to note is that the performance improvement by the proposed model over the baselines in ROUGE-1 is greater than those in ROUGE-2 and ROUGE-L. This is because the CAM pays attention mostly to the topic-related words, rather than to the surrounding words. However, if a topical attention stresses not only on topic-related words but also on their surrounding words, the quality of the generated summary would be improved. Thus, we will study how to reflect the neighbor words of topic-related words as well as the topic-related words into generating a summary.

Recent studies on deep learning for natural language processing result in various models which show higher performance than LSTM-based sequence-to-sequence models such as the transformer [34]. The transformer-based models use a multi-head attention which differentiates them from the legacy attention of LSTM-based sequence-to-sequence models. The multi-head attention interprets the same sentence from different viewpoints by multiple heads and calculate the attention for each head. Therefore, if CAM is applied to the attentions of all heads, every head would have a similar or identical viewpoint due to the topic category and the advantage of multiple heads would disappear. Thus, we will design, as our future work, a transformer-based model that has a head for the categorical viewpoint.

Author Contributions: conceptualization, S.-E.K. and S.-B.P.; methodology, S.-E.K. and N.K.; software, S.-E.K. and N.K.; validation, N.K.; formal analysis, S.-E.K. and N.K.; resources, N.K.; data curation, N.K.; writing–original draft preparation, S.-E.K.; writing–review and editing, S.-B.P.; supervision, S.-B.P.; project administration, S.-B.P.; All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a grant from Kyung Hee University in 2018 (KHU-20182220) and Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2013-0-00109, WiseKB: Big Data based self-evolving knowledge base and reasoning platform).

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

References

1. Cheng, J.; Lapata, M. Neural Summarization by Extracting Sentences and Words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 484–494.
2. Nallapati, R.; Zhou, B.; dos Santos, C.; Gulcehre, C.; Xiang, B. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In Proceedings of the 20th Special Interest Group on Natural Language Learning Conference on Computational Natural Language Learning, Berlin, Germany, 11–12 August 2016; pp. 280–290.
3. Yasunaga, M.; Zhang, R.; Meelu, K.; Pareek, A.; Srinivasan, K.; Radev, D. Graph-based Neural Multi-Document Summarization. In Proceedings of the 21st Conference on Computational Natural Language Learning, Vancouver, BC, Canada, 3–4 August 2017; pp. 452–462.
4. Narayan, S.; Cohen, S.; Lapata, M. Ranking Sentences for Extractive Summarization with Reinforcement Learning. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, New Orleans, LA, USA, 1–6 June 2018; pp. 1747–1759.

5. See, A.; Liu, P.J.; Manning, C. 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, 30–4 July 2017; pp. 1073–1083.
6. Paulus, R.; Xiong, C.; Socher, R. A Deep Reinforced Model for Abstractive Summarization. *arXiv* **2017**, arXiv:1705.04304.
7. Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; Dauphin, Y. Convolutional Sequence to Sequence Learning. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1243–1252.
8. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
9. Rush, A.; Chopra, S.; Weston, J. A Neural Attention Model for Abstractive Sentence Summarization. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 379–389.
10. Nallapati, R.; Zhai, F.; Zhou, B. Summarunner: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In Proceedings of the 31th American Association for Artificial Intelligence Conference on Artificial Intelligence, San Francisco, United States, 4-9 February 2017; pp. 3075–3081.
11. Allamanis, M.; Peng, H.; Sutton, C. A Convolutional Attention Network for Extreme Summarization of Source Code. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 2091–2100.
12. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. In Proceedings of the Institute of Electrical and Electronics Engineers Conference on Computer Vision and Pattern Recognition, Las Vegas, NA, USA, 27–30 June 2016; pp. 2921–2929.
13. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
14. Zhou, Q.; Yang, N.; Wei, F.; Huang, S.; Zhou, M.; Zhao, T. Neural Document Summarization by Jointly Learning to Score and Select Sentences. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 654–663.
15. Erkan, G.; Radev, D. Lexrank: Graph-based Lexical Centrality as Salience in Text Summarization. *J. Artif. Intell. Res.* **2004**, *22*, 457–479.
16. Sutskever, I.; Vinyals, O.; Le, Q. Sequence to Sequence Learning with Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, Quebec, Canada, 8–13 December 2014; pp. 3104–3112.
17. Cao, Z.; Wei, F.; Dong, L.; Li, S.; Zhou, M. Ranking with Recursive Neural Networks and Its Application to Multi-Document Summarization. In Proceedings of the 29th American Association for Artificial Intelligence Conference on Artificial Intelligence, Austin TX, USA, 25–30 January 2015; pp. 2153–2159.
18. Chopra, S.; Auli, M.; Rush, A. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics, San Diego, CA, USA, 13–15 June 2016; pp. 93–98.
19. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer Networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC Canada, 7–12 December 2015; pp. 2692–2700.
20. 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; pp. 76–85.
21. Krishna, K.; Srinivasan, B. Generating Topic-Oriented Summaries Using Neural Attention. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, New Orleans, Louisiana, USA, 1–6 June 2018; pp. 1697–1705.
22. Wang, L.; Yao, J.; Tao, Y.; Zhong, L.; Liu, W.; Du, Q. A Reinforced Topic-Aware Convolutional Sequence-to-Sequence model for Abstractive Text Summarization. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 4453–4460.

23. Li, X.; Shen, Y.; Du, L.; Xiong, C. Exploiting Novelty, Coverage and Balance for Topic-Focused Multi-Document Summarization. In Proceedings of the 19th Association for Computing Machinery International Conference on Information and Knowledge Management, Toronto, ON, Canada, 26–30 October 2010; pp. 1765–1768.
24. Narayan, S.; Cohen, S.; Lapata, M. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1797–1807.
25. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
26. Oquab, M.; Bottou, L.; Laptev, I.; Sivic, J. Is Object Localization for Free? Weakly Supervised Learning with Convolutional Neural Networks. In Proceedings of the Institute of Electrical and Electronics Engineers Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 685–694.
27. Sandhaus, E. The New York Times Annotated Corpus. *Linguist. Data Consort. Phila.* **2008**, *6*, e26752.
28. Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
29. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In Proceedings of the Neural Information Processing Systems 2014 Workshop on Deep Learning, Montreal, Quebec, Canada, 12–13, December 2014.
30. Williams, R.; Zipser, D. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Comput.* **1989**, *1*, 270–280. [CrossRef]
31. 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.
32. Lin, C.; Hovy, E. Manual and Automatic Evaluation of Summaries. In Proceedings of the Association for Computational Linguistics-02 Workshop on Automatic Summarization-Volume 4, Philadelphia, PA, USA, 11–12 July 2002; pp. 45–51.
33. 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, Melbourne, Australia, 15–20 July 2018; pp. 1662–1675.
34. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.