Postprint: Research on Abstractive Chinese Text Summarization Based on Sequence-to-Sequence Models

Authors: Yu Chuanming, Zhu Xingyu, Gong Yutian, An Lu

Date: 2023-07-26T00:00:00+00:00

Abstract

[Purpose/Significance] To better handle out-of-vocabulary (OOV) words in text summarization tasks, avoid summary repetition, and improve text summarization quality, this study addresses the OOV problem and summary self-repetition problem as research objectives, conducting research on abstractive Chinese text summarization.

[Method/Process] Building upon the sequence-to-sequence (seq2seq) model, we incorporate a pointer-generator mechanism and a coverage mechanism. The pointer-generator copies OOV words into the summary to address the OOV problem, while the coverage mechanism prevents the attention mechanism from repeatedly attending to the same positions to resolve repetition issues. The proposed method is applied to the LCSTS Chinese summarization dataset for experimental validation of model effectiveness.

[Results/Conclusion] Experimental results demonstrate that the ROUGE (recall-oriented understudy for gisting evaluation) scores of summaries generated by this model surpass those of traditional seq2seq models and extractive text summarization models, indicating that the pointer-generator and coverage mechanisms can effectively solve the OOV and summary repetition problems, thereby significantly enhancing text summarization quality.

Full Text

Research on Abstractive Chinese Text Summarization Based on Sequence-to-Sequence Models

Yu Chuanming[1], Zhu Xingyu[1], Gong Yutian[1], An Lu[2]
[1]School of Information and Safety Engineering, Zhongnan University of Eco-

nomics and Law, Wuhan 430073
[2]School of Information Management, Wuhan University, Wuhan 430072

Abstract: [Purpose/Significance] To better handle out-of-vocabulary (OOV) words in text summarization tasks while avoiding summary duplication and improving text summarization quality, this study focuses on solving the OOV problem and self-repetition problem in abstractive Chinese text summarization. [Method/Process] Based on the sequence-to-sequence (seq2seq) model, we incorporate a pointer generator mechanism and a coverage processing mechanism. The pointer generator copies OOV words into the summary to address the OOV problem, while the coverage processing mechanism prevents the attention mechanism from repeatedly focusing on the same position to solve the repetition problem. We apply our method to the LCSTS Chinese summarization dataset to evaluate model effectiveness. [Result/Conclusion] Experimental results show that the ROUGE scores of summaries generated by our model are higher than those of traditional seq2seq models and extractive summarization models, indicating that the pointer generator and coverage mechanism can effectively solve the OOV problem and summary repetition problem, thereby significantly improving text summarization quality.

Keywords: abstractive text summarization; sequence-to-sequence model; attention mechanism; coverage mechanism; pointer generator mechanism

---

With the rapid development of the big data era, textual data such as online news and comments has grown exponentially, making manual summarization face enormous resource and efficiency challenges. How to use machines and programs to automatically summarize texts by eliminating non-critical and redundant information to compress and extract the main information from texts has become a research hotspot.

According to different research tasks, text summarization can be divided into extractive summarization and abstractive summarization. The former directly extracts representative textual elements (including words, phrases, and sentences) from the source text to form a summary, while the latter involves sentence compression and reconstruction, obtaining semantic representations of the source text and using natural language generation techniques to produce summaries. Abstractive methods generate highly abstract summaries based on the semantic information of the text, resulting in outputs that are more semantically similar to the source text and easier for users to understand. In abstractive summarization tasks, the sequence-to-sequence [1] (seq2seq) model is a commonly used approach. This model typically encodes and decodes source documents based on recurrent neural networks (RNN), with summary results obtained through the decoder. However, current seq2seq models still face several challenges. First,

the decoder generates a word at each time step, typically from a fixed vocabulary obtained through probability calculation (e.g., softmax method). Considering computational cost and model training speed, the vocabulary usually does not contain all words in the training set, so some words cannot be used when generating summaries. In the training set, a large number of low-frequency words in the summary are represented as UNK (unknown words), significantly affecting summary readability. Second, seq2seq models typically introduce an attention mechanism during decoding to change the focus of attention, which can easily cause the model to repeatedly focus on the same word at different time steps, resulting in repeated fragments in the generated summary and reducing summary quality.

To address these problems, this paper attempts to apply the pointer generator mechanism to the seq2seq model, conducting empirical research on abstractive Chinese text summarization and testing model effectiveness to provide insights for related research.

---

2 Related Research

2.1 Abstractive Text Summarization

Abstractive text summarization has good coherence and high cohesion, becoming a research hotspot in natural language processing in recent years. Researchers have applied various techniques to abstractive text summarization, including structure-based methods, semantic-based methods, and deep learning-based methods.

2.1.1 Structure-Based Abstractive Text Summarization Structure-based abstractive text summarization methods mainly encode important information of documents through frameworks, templates, trees, and other patterns. For example, H. T. Le et al. [2] performed sentence reduction based on source text sequences, keywords, and syntactic constraints, and used word graphs to complete sentence fusion, ultimately generating abstract summaries that contain complete source document information and are syntactically correct. Zhao Wenjuan et al. [3] filled information related to events into given event templates according to corresponding rules, and verified the effectiveness of this method using the "Germanwings plane crash incident" as an example. Structure-based methods are relatively easy to implement but rely on the discourse structure and form of the source document, having limitations in practical applications.

2.1.2 Semantic-Based Abstractive Text Summarization Semantic-based abstractive text summarization methods mainly identify noun and verb phrases in the source document through natural language processing techniques, use annotation and clustering techniques to determine important

document information, and finally apply the obtained semantic representations to natural language generation systems to generate final summaries. For example, Zhang Han et al. [4] constructed a semantic graph based on source document concepts and their semantic relationships, and used key information in the semantic graph to generate summaries. Results showed that this method could effectively obtain important information from documents, with high accuracy, recall, and F-value for generated summaries. A. Khan et al. [5] used semantic role labeling to identify the semantic structure of sentences, used an improved graph ranking algorithm to rank important graph nodes, and selected the highest-ranked nodes to generate summaries. On the DUC dataset, the ROUGE-1 and ROUGE-2 scores were 0.417 and 0.108 respectively, higher than baseline methods, demonstrating the superior performance of this method. Wang Zhenchao et al. [6] proposed an abstractive summarization method based on document semantic information using events as basic semantic units, clustering events and using events to guide the generation of summary sentences. Semantic-based methods can well capture the semantic information of source documents and effectively improve the semantic relevance between summaries and source documents. Their limitation lies in not using neural networks to automatically learn text features and representations, making the model unable to automatically learn and generate, resulting in low efficiency.

2.1.3 Deep Learning-Based Abstractive Text Summarization Deep learning-based text summarization methods typically treat text summarization as a sequence-to-sequence problem, using the source document as the input sequence and the generated summary as the output sequence. These methods utilize deep networks to more effectively learn text representations and capture important information from source documents. D. Bahdanau et al. [7] first applied the seq2seq model to neural machine translation tasks, using recurrent neural networks to encode source documents into fixed-length vectors and then decode them to generate corresponding translations. Compared with statistical methods, the seq2seq model has better non-linear data processing capabilities but cannot handle longer input sequences well and has poor alignment effects. To further improve its effectiveness, A. M. Rush et al. [8] added an attention mechanism to the encoder-decoder framework, enabling the decoder to focus on different parts of the input at each time step, structurally selecting input subsets to reduce data dimensions while making the model more focused on finding useful information significantly related to input data and current output. Subsequently, researchers improved the model using recurrent decoders [9], hierarchical networks [10], and autoencoders [11], further enhancing model effectiveness. Xie Mingyuan et al. [12] incorporated document category information into abstractive summarization, using convolutional neural networks (CNN) to classify documents and combining text category features with seq2seq to generate summaries, achieving higher ROUGE scores than traditional seq2seq models.

2.2 The seq2seq Model

Since deep learning can effectively reveal and obtain the intrinsic semantic representation of text information, it has achieved better results in abstractive text summarization tasks, making the seq2seq model gradually become mainstream. However, in current research, seq2seq still faces many challenges such as the OOV problem and repeated word problem.

2.2.1 The OOV Problem The seq2seq model typically builds a fixed vocabulary during training, and the decoder samples from this vocabulary to generate words. Considering computational efficiency, researchers usually limit the size of the decoding vocabulary based on word frequency, resulting in some low-frequency words not being decoded and causing the OOV problem. To solve the OOV problem, researchers have proposed methods such as increasing the decoding vocabulary size, reducing vocabulary granularity, and adopting copy mechanisms.

Increasing the decoding vocabulary size is the most direct approach. These methods focus on improving the processing speed of the softmax layer, enabling the vocabulary to maximally include more words, thereby reducing the probability of OOV occurrence. For example, S. Jean et al. [13] used importance sampling to reduce the complexity when calculating the norm related to output word probabilities, thereby improving decoding efficiency. This method can build a vocabulary with a larger size without significantly increasing model complexity. Although the vocabulary is large enough to include all low-frequency words in the training set, it theoretically still cannot cover all words, so the model' s performance on the test set cannot be significantly improved.

Another feasible approach is to theoretically reduce vocabulary granularity. For example, Z. Xie et al. [14] used letters as the basic processing units of the encoder-decoder model and applied it to natural language error correction tasks, achieving the optimal F0.5 value on the CoNLL 2014 Challenge dataset, effectively solving the OOV problem. This type of method changes the model's input and output from word-based units to character- or byte-based units, reducing the occurrence of OOV words and solving the OOV problem to a certain extent, but this method increases the length of sequences processed by the model, thereby increasing training difficulty.

The third method is to adopt a copy mechanism. For example, M. T. Luong et al. [15] used context information to point to the position of OOV words in the source document and copy them into the target sentence. However, this model does not use an attention mechanism, and the model' s pointing to the source document is limited to a specific range, making it unsuitable for more general text generation tasks. J. Gu et al. [16] proposed the CopyNet model, which incorporates the copy mechanism into the seq2seq model, copying OOV words from the source document into the final summary to solve the OOV problem. R. Nallapati et al. [17] configured a switch on the decoder, which is essentially a

sigmoid activation function of a linear layer. When the switch is on, the decoder generates words from the vocabulary in the traditional seq2seq manner; if the switch is off, the decoder points to the corresponding position in the source document and copies the word at that position into the summary. This method is more efficient than the previous two methods and can better solve the OOV problem.

2.2.2 The Repeated Word Problem The seq2seq model typically introduces an attention mechanism during the decoding process to change the focus of attention. The attention mechanism can easily focus on the same word at different time steps, causing the decoder to receive the same input at multiple time steps, resulting in repeated fragments in the final generated summary. The coverage mechanism can effectively solve this repeated word problem. This mechanism was first applied to neural machine translation (NMT) tasks. The typical encoder-decoder framework lacks attention to translated source words, which may lead to over-translation and under-translation problems. Z. Tu et al. [18] added a coverage vector to the NMT model to increase attention to historical attention. After each attention update, the vector is updated using a gated recurrent unit, and the vector is used to adjust future attention distributions.

Against this background, this paper attempts to apply the pointer generator mechanism to the seq2seq model: on the one hand, adding a pointer generator mechanism to the seq2seq model to handle OOV words. When a word in the decoder is an OOV word, the model points to the position of that word in the source document and copies the corresponding word into the final summary to ensure the accuracy of the final summary. Conversely, if the word in the decoder is not an OOV word, the model is similar to the traditional sequence model, and the decoder generates new words from the vocabulary to form the summary, maintaining the abstract generation capability of the seq2seq model. On the other hand, adding a coverage mechanism to the pointer generator network avoids the attention mechanism repeatedly focusing on the same position, thereby reducing repeated words in the summary. On this basis, we conduct empirical research on abstractive Chinese text summarization and test model effectiveness.

---

3 Research Methods

3.1 Research Problem and Related Definitions

The research task of this paper is abstractive Chinese text summarization. Assume the model inputs a sequence X = {x_1, ⋯, x_T} of length T. Discourse generation refers to using sequence X and a certain model to generate a sequence Y = {y_1, ⋯, y_M} of length M, where X is the input sentence sequence, Y is the output sentence sequence, and T and M are the lengths of the input

and output sequences respectively, with T M. The model here consists of two parts: an encoder and a decoder. The encoder inputs sequence X into the encoder at different time steps to obtain its encoding h_t; the decoder inputs this encoding h_t into the decoder to obtain the output sequence Y. The model inputs one word from the source text at a time, converting it into a distributed representation through a word embedding layer.

For a given source document W_i, the model' s goal is to generate a summary sequence composed of words y. From a probability theory perspective, a general seq2seq model selects the word with the highest probability at each time step to form the summary. To simplify notation, the symbols in Table 1 are used in the following model description.

Table 1 Symbol Description - Subscript i: words in the source document and input sequence - Subscript t: a certain time step - h_t, s_t: encoder hidden state sequence and decoder hidden state - a_t: attention distribution at time step t - c_t: context vector - P_{vocab}: probability distribution of all words in the fixed vocabulary - P(y): probability distribution of generating word y

3.2 Model Description

The seq2seq model architecture used in this paper is shown in Figure 1 [Figure 1: see original paper]. It adds a pointer generator mechanism and a coverage processing mechanism to the traditional seq2seq model [19]. At each time step, the model calculates a generation probability to determine whether to copy a word from the source text or generate a word from the vocabulary, using the word distribution in the vocabulary and the attention distribution to obtain the final probability of summary words. The model includes three parts: Encoder, decoder, and attention module (see part A in Figure 1). In this module, the encoder reads the source document as input to obtain encoder hidden states, the decoder generates decoder hidden states based on encoder hidden states, calculates the attention distribution at each time step based on the two hidden states, and obtains the context vector. Pointer generator module (see part C in Figure 1). In this module, on the one hand, the model obtains the vocabulary word distribution and generation probability P_{gen} based on the context vector and decoder hidden state. On the other hand, the model samples from the attention distribution to copy words, with a copy probability of (1 - P_{gen}). The final distribution of target words is obtained based on the two distributions. In Figure 1, the solid part of the final word distribution comes from the attention distribution, and the hollow part comes from the vocabulary word distribution.
Coverage processing module (see part B in Figure 1). The coverage processing module calculates a weighted sum of attention from previous time steps to obtain a coverage vector, which is used as an additional input for calculating the attention distribution. The following sections elaborate on these components.

3.2.1 Encoder, Decoder, and Attention Module (1) Encoder. The encoder consists of a single-layer bidirectional Long Short-Term Memory [20]

(LSTM) network. The encoder reads the input sequence X sequentially, and the hidden state at a certain time step t can be calculated by formula (1):

$$h_t = f(x_t, h_{t-1}) \quad (1)$$

where h_{t-1} represents the encoder hidden state at the previous time step t-1, x_t is the current input, and f() is a non-linear function.

The encoder can transform the input sequence X into a vector through the hidden state sequence. c_t is called the context vector, calculated as in formula (3):

$$c_t = a_t h_t \quad (3)$$

where a_t is the attention distribution at time step t obtained from formula (2), serving as weights for encoder hidden states h_t. The context vector can be seen as a representation of the source sequence information learned at time step t and is the input to the decoder.

(2) Decoder. The decoder consists of a single-layer unidirectional LSTM, generating the target sequence Y based on the context vector c_t and decoder hidden state s_t. As in formula (4), it predicts the probability distribution of words in the vocabulary, and the probability of target words generated by the model is the same:

$$p_{vocab}(y_t|y_{<t}, X) = \text{softmax}(y_{t-1}, s_t, c_t) \quad (4)$$

where y_t and y_{t-1} are target words at time steps t and t-1 respectively, y_{<t} represents all words obtained before time step t, i.e., {y_1, ⋯, y_{t-1}}, and X is the input sequence. The decoder hidden state s_t can be calculated by formula (5):

$$s_t = f(y_{t-1}, s_{t-1}, c_t) \quad (5)$$

The basic seq2seq model can generate words from the vocabulary in any order to obtain the final summary, while the attention mechanism can effectively obtain each word vector in the source text sequence and simultaneously determine vectors more relevant to the output summary, making the model more focused on useful words. At each time step t, the model calculates the current attention distribution according to formula (2):

$$a_t = \text{softmax}(w^T \tanh(w_1 h_t + w_2 s_t + b_1)) \quad (2)$$

In formula (2), w, w_1, w_2, and b_1 are parameters that can be learned through training. h_t is the encoder hidden state obtained from formula (1), and s_t is

the decoder hidden state. The attention distribution can be seen as a probability distribution of words in the source text, telling the decoder where to focus when generating the next word. Words with high probability receive more attention when generating summary words, enabling the model to generate words that better reflect source text information.

The seq2seq model uses its encoder, decoder, and attention modules to calculate the probability distribution of generating each target word from a fixed vocabulary, selecting the word with the highest probability from the vocabulary at each time step to form the summary.

3.2.2 Pointer Generator Module In the seq2seq model, relying solely on the attention mechanism cannot effectively handle OOV words, so a word copying mechanism is added. Inspired by the work of J. Gu et al. [16] and A. See et al. [21], this paper adopts a pointer generator module to complete word copying. In section 3.2.1, the attention distribution a_t and context vector c_t have already been calculated. Based on the obtained context vector c_t, decoder state s_t, and decoder input x_t, the word generation probability p' can be calculated using formula (6). This probability indicates the likelihood that the model generates a word from the vocabulary as part of the summary at each time step:

$$p' = g(w_3 c_t + w_4 s_t + w_5 x_t + b_2) \quad (6)$$

In formula (6), w_3, w_4, w_5, and b_2 are trainable parameters, and g() is a sigmoid activation function. p' is treated as a control switch that can determine whether the model generates from the given vocabulary or copies words from the source document. In fact, this is a judgment of whether words in the source sequence are OOV words. That is, if the target word of the decoder at this time step is an OOV word, the p' value is small, and the model copies words from the source document to generate the summary; otherwise, the model generates new words based on the vocabulary. Based on this, the probability distribution of target word y can be calculated by formula (7):

$$p(y) = p' P_{vocab} + (1 - p') \sum_i a_t^i \quad (7)$$

In formula (7), p' is the word generation probability calculated by formula (6), P_{vocab} is the probability distribution of words in the vocabulary obtained by formula (4), and *i a_t^i represents the sum of attention distributions when y is an OOV word. From formula (7), it can be seen that if y is an OOV word, indicating that the word generated by the decoder at this time step does not appear in the given vocabulary, then the probability* $P\{\text{vocab}\}$ in the vocabulary is 0, meaning that the generated word y comes from the source document and needs to be copied from the source document. Conversely, if y does not appear in the

source document, the cumulative attention distribution value is 0, and the model generates words from the given vocabulary. The ability to effectively handle OOV words and copy them into the final summary is a major advantage of the pointer generator network proposed in this paper, while sequence models based on attention mechanisms are limited by predefined vocabularies and cannot solve the generation problem of OOV words.

3.2.3 Coverage Processing Module This paper uses a coverage processing module to avoid repeated summary fragments. Based on the sum of attention distributions from all previous decoder time steps, a coverage vector is obtained, which can inform the model which words have been attended to in previous time steps, so they do not need repeated attention at the current time step, thus avoiding the generation of repeated words. At t = 0, the coverage vector is a zero vector because at the first time step, all words in the source document have not yet been covered, and the attention distribution being 0 leads to the coverage vector being 0. The coverage vector is an additional input to the attention mechanism, which can be directly added to formula (2) for calculating the attention distribution. This ensures that when using the attention mechanism, the part selected for attention at the current time step is influenced by the parts attended to in previous time steps. Therefore, this can prevent the attention mechanism from repeatedly focusing on the same part at multiple time steps, thereby avoiding the model generating repeated text, which is the core idea of the coverage mechanism.

---

4 Experiments and Analysis

4.1 Dataset

This paper uses the large-scale Chinese short text summarization dataset (LCSTS) constructed by B. Hu et al. [22]. This dataset contains over 2.4 million texts and corresponding author-provided summaries obtained from Sina Weibo, with each text containing no fewer than 80 characters and corresponding summaries between 10 and 30 characters in length. To ensure text quality, researchers collected 50 popular organizational users (with blue "V" verification and over 1 million followers) such as People' s Daily, Economic Observer, and the Ministry of National Defense as seeds, capturing their posted Weibo messages covering politics, economy, military, film, and gaming domains. The original complete dataset consists of three parts: PART I contains 2,400,591 text-summary pairs, while PART II and PART III contain 10,666 and 1,106 text-summary pairs respectively. This paper selects the part with the largest data volume (PART I) for experiments. The Chinese word segmentation tool jieba [23] is used for word segmentation, and the segmented data is processed into binary files, divided into 18 training set data files, 1 validation set data file, and 1 test set data file. Additionally, the fixed vocabulary file for summary

generation in the seq2seq model contains 400,000 words, and the actual words used in experiments can be selected by setting the vocabulary size.

4.2 Evaluation Metrics

To evaluate the quality of summaries generated by different models, this paper uses ROUGE [24] scores as evaluation metrics. This evaluation metric assesses automatically generated summary results based on the overlap of n-gram words between generated summaries and reference summaries (standard summaries), and is an n-gram recall-oriented evaluation method. The basic idea is that experts first generate manual summaries to form a reference summary set (standard summary set), and model-generated summaries are compared with standard summaries. The quality of summaries from different models is evaluated by counting the number of overlapping basic units between them. The more N-gram words (where N can be 1, 2, 3, etc.) that match between the evaluated summary and the standard summary, the higher the ROUGE score, indicating that the model-generated summary is closer to the standard summary and thus of higher quality. This method has become one of the universal metrics for summary evaluation technology [24]. The ROUGE evaluation metric consists of a series of evaluation methods, including ROUGE-N (where N can be 1, 2, 3, etc.) and ROUGE-L. Among them, ROUGE-1 and ROUGE-2 represent the overlap degree of unigrams and bigrams between model-generated summaries and standard summaries, respectively, while ROUGE-L represents the overlap degree based on the longest common subsequence between generated summaries and standard summaries. In the experiments in this paper, ROUGE-1, ROUGE-2, and ROUGE-L are selected to evaluate the quality of automatically generated summaries.

4.3 Comparison Methods

To address the OOV problem and summary fragment repetition problem, this paper adds a pointer generator module and a coverage processing module to the encoder, decoder, and attention module of the seq2seq model. To better study the effectiveness of the pointer generator module and coverage processing module, in the experiments, by setting parameters, we use three abstractive methods for Chinese text summarization experiments: the attention-based seq2seq model (Attention), the seq2seq model with pointer generator module (Attention+PG), and the seq2seq model with both pointer generator module and coverage processing module (Attention+PG+Coverage). The experimental results of the model using pointer generator and coverage processing modules are compared with the experimental results of seq2seq.

At the same time, to further compare the effectiveness of our method relative to extractive summarization methods, we use three typical extractive methods as baselines: the TextRank [25] method, Lead-1-First (extracting the first sentence of the original text), and Lead-1-Last (extracting the last sentence of the original text).

4.4 Parameter Settings

In this experiment, the neural network hidden state is 256-dimensional, word vectors in both source and target sequences are 128-dimensional, and a vocabulary containing 50,000 words is used. The experiment does not pre-train word vectors but learns them from scratch during the training phase, using the Adagrad optimization algorithm with an initial learning rate of 0.15 and an initial accumulator value of 0.1. The Adagrad algorithm differentially assigns learning rates to each parameter adaptively. As the total distance of parameter updates increases, the learning rate slows down. In the testing phase, beam search with a beam size of 4 is used to generate summaries.

4.5 Basic Experimental Results Evaluation

According to the above settings, summarization experiments are conducted on the LCSTS dataset, and the ROUGE scores calculated using the pyrouge package are shown in Table 2 .

Table 2 ROUGE Scores of Different Summarization Methods on the Test Set

Method	ROUGE-1	ROUGE-2	ROUGE-L
Lead-1-First	0.1118	0.0338	0.1038
Lead-1-Last	0.1340	0.0457	0.1234
TextRank	0.1293	0.0399	0.1193
Attention	0.1054	0.0096	0.1014
Attention+PG	0.3083	0.1136	0.2843
Attention+PG+Coverage	0.3487	0.1147	0.3061

From Table 2, it can be seen that among various sequence-to-sequence methods, the model with both coverage mechanism and pointer generator mechanism (Attention+PG+Coverage) achieves the best results on all three metrics, with ROUGE scores of 0.3487, 0.1147, and 0.3061 respectively. The model with only the pointer generator mechanism (Attention+PG) has slightly worse performance than the former (Attention+PG+Coverage), with ROUGE-1, ROUGE-2, and ROUGE-L scores lower by 0.0404, 0.0011, and 0.0218 respectively. The traditional attention-based seq2seq model has the lowest scores on all evaluation metrics, at 0.1054, 0.0096, and 0.1014 respectively, far below the experimental results of the other two seq2seq models (Attention+PG and Attention+PG+Coverage). Among traditional extractive methods, the Lead-1-Last method has the highest scores on ROUGE-1, ROUGE-2, and ROUGE-L, at 0.1340, 0.0457, and 0.1234 respectively, slightly higher than the TextRank and Lead-1-First extractive methods.

Comparing extractive and abstractive methods comprehensively, it can be seen that the model with coverage mechanism and pointer generator mechanism has

better performance on ROUGE-1 and ROUGE-L. Compared with the three extractive methods, the Attention+PG+Coverage model improves ROUGE-1 scores by 0.2369, 0.2147, and 0.2194 respectively, ROUGE-2 scores by 0.0809, 0.0690, and 0.0748 respectively, and ROUGE-L scores by 0.2023, 0.1827, and 0.1131 respectively. Experimental results show that compared with traditional abstractive and extractive models, the model proposed in this paper can effectively improve the effectiveness of Chinese text summarization by combining pointer generator mechanism and coverage mechanism.

Table 3 shows a comparison of summaries generated by different summarization models for the same news article. It can be seen that the summary generated using the traditional seq2seq model (Attention) contains many repeated word fragments, meaning that certain detailed information from the source text is incorrectly generated repeatedly, and these repeated fragments usually consist of words that appear frequently in the training set, while less frequent words (still included in the vocabulary) are often replaced by more common words. For example, in Table 3, the phrase "been cheated" appears three times in the summary generated by this method, significantly reducing summary readability. Additionally, "Shenzhen" in the summary clearly does not match the word "Tianjin" in the source text, and the summary cannot accurately reflect the source text information. By checking the fixed vocabulary generated from the training data, it can be seen that the word "Shenzhen" appears 49,398 times in the training set, while "Tianjin" appears 15,212 times. In comparison, the word "Shenzhen" is more common. Therefore, the baseline method more easily learns the vector representation of "Shenzhen" during training, while the learned vector representation of "Tianjin" is weaker, ultimately making it more likely to generate incorrect common words when generating summaries from the vocabulary. In addition, the summary generated by the baseline method contains multiple [UNK] tokens, indicating that the traditional sequence model cannot generate OOV words not included in the vocabulary, losing important information from the source document and failing to generate summaries containing all information from the source document and with complete semantics, resulting in low-quality summaries.

After adding the pointer generator mechanism (Attention+PG), it can be seen that for the same source document, the summary generated by the pointer generator network replaces the corresponding [UNK] tokens with named entities and other content copied from the source document, making the final summary more readable and almost containing important information from the source document. This shows that using the pointer generator mechanism can effectively handle OOV words. Although the summary generated using the pointer generator mechanism eliminates [UNK] tokens, the generated summary still contains repeated fragments, such as the fragment "Tianjin police cracked down on criminal suspects" appearing repeatedly, causing redundancy in the generated summary.

From Table 3, it can be seen that after adding both pointer generator mechanism

and coverage mechanism (Attention+PG+Coverage), the generated summary is well-formed and contains important information from the source document. The summary results contain no [UNK] tokens that cannot be recognized by the model, while eliminating repeated fragments in the summary. The final summary is more consistent with the reference summary in content and semantics, and more aligned with the summarization results.

Table 3 Comparison of Summaries Generated by Three Abstractive Methods Based on the Same News Article

Source and Method	Summary
Original Text	Recently, Tianjin police cracked a major counterfeit bag case. A physical store had been selling and wholesaling counterfeit bags of brands such as Louis Vuitton, Gucci, Chanel, and Burberry for a long time. Police seized over 7,000 bags of various brands, with an estimated value of over 100 million yuan based on genuine product prices! The suspects involved in selling counterfeit goods have been arrested.
Attention	Tianjin cracked a major counterfeit bag case, LV and Chanel were both counterfeited with a value exceeding 100 million. Shenzhen police cracked a major counterfeiting case and pronounced judgment: [UNK][UNK][UNK][UNK] cheated 1 month cheated 100 million! [UNK][UNK] cheated!! [UNK]!!!!!!!!

Source and Method	Summary
Attention+PG	Tianjin police cracked a major counterfeit bag case police seized over 7,000 worth over 100 million——Currently suspects involved in selling counterfeit goods have been arrested——Tianjin police cracked suspects Tianjin seized over 7,000 brand bags suspects have been arrested. Estimated value over 100 million based on genuine products!
Attention+PG+Coverage	Tianjin police cracked a major counterfeit bag case, seized over 7,000 brand bags of various types, with an estimated value of over 100 million yuan based on genuine products, suspects involved in selling counterfeit goods have been arrested.

4.6 Extended Experimental Results Evaluation

The deep neural network in this paper involves multiple parameters such as vocabulary size and word vector dimension. In the experiments, the word vector dimension in the neural network is set to 128 dimensions, and the vocabulary contains 50,000 words. To further study the impact of different hyperparameters on the quality of summaries generated by the model, we set different vocabulary sizes and word vector dimensions on the Attention+PG+Coverage model for summarization experiments, comparing the ROUGE scores of the model under different vocabulary size and word vector dimension settings to further test the model.

4.6.1 Impact of Vocabulary Size on Experimental Results Table 4 shows the comparison of ROUGE scores obtained by the model proposed in this paper when using different vocabularies in horizontal comparison experiments. From Table 4, it can be seen that as the vocabulary size increases from

20,000 to 80,000, the ROUGE scores of generated summaries show an overall trend of first increasing and then decreasing. When the vocabulary size is 60,000, the model' s ROUGE-1 and ROUGE-L scores are the highest, at 0.3581 and 0.3148 respectively. When the vocabulary contains 70,000 words, the corresponding ROUGE-2 score is the highest, at 0.1178. Specifically, when the vocabulary size is 40,000, the model' s ROUGE scores are lower than when the vocabulary size is 30,000, decreasing by 0.0172, 0.0068, and 0.0184 respectively. Experimental results show that the Attention+PG+Coverage model has different experimental effects under different vocabulary size settings, and vocabulary size has a certain impact on model effectiveness. When the vocabulary is of a specific size (60,000 in this paper), the model has the best effect and generates better quality summaries.

Table 4 Comparison of ROUGE Scores Using Different Vocabulary Sizes

Vocabulary Size	ROUGE-1	ROUGE-2	ROUGE-L
20,000 words	0.3188	0.1054	0.2850
30,000 words	0.3391	0.1157	0.3048
40,000 words	0.3219	0.1089	0.2864
50,000 words	0.3487	0.1147	0.3061
60,000 words	0.3581	0.1156	0.3148
70,000 words	0.3500	0.1178	0.3102
80,000 words	0.3200	0.1088	0.2870

4.6.2 Impact of Vector Dimension on Experimental Results Table 5 shows the comparison of ROUGE scores obtained by the model proposed in this paper when setting different word vector dimensions in horizontal comparison experiments. It can be seen from Table 5 that when the word vector dimension is 128 dimensions, the model has the best effect. Compared with the model using 64-dimensional word vectors, the ROUGE scores are improved by 0.0184, 0.0034, and 0.0123 respectively. Compared with the model using 128-dimensional word vectors, the ROUGE scores are improved by 0.0238, 0.0037, and 0.0158 respectively.

From the experimental results in Table 5, it can be seen that word vector dimension has a certain impact on the effectiveness of the Attention+PG+Coverage model. When the word vector dimension is set to 128 dimensions, the quality of summaries generated by the model is the best.

Table 5 Comparison of ROUGE Scores Using Different Vector Dimensions

Vector Dimension	ROUGE-1	ROUGE-2	ROUGE-L
64-dimensional	0.3303	0.1113	0.2938

Vector Dimension	ROUGE-1	ROUGE-2	ROUGE-L
128-dimensional	0.3487	0.1147	0.3061
256-dimensional	0.3249	0.1110	0.2903

4.7 Discussion

4.7.1 Analysis of Overall Experimental Effectiveness From the overall experimental effectiveness of the model, compared with the traditional attention-based seq2seq model, the Attention+PG+Coverage model proposed in this paper can better handle OOV words and repeated words, thereby effectively improving the effectiveness of abstractive Chinese text summarization.

In handling OOV words, traditional seq2seq models replace them with [UNK] tokens in the final generated summary. These tokens cannot match the content of reference summaries, resulting in lower word matching rates and consequently lower ROUGE scores. Compared with traditional seq2seq models, under the same experimental parameters, the model with the added pointer generator network can significantly improve ROUGE values. This indicates that by pointing to words in the source document and copying them into the summary, the OOV problem in abstractive text summarization can be better solved, thereby improving summary quality.

In handling repeated words, traditional seq2seq models adopt an attention mechanism that repeatedly focuses on the same position in the source document at different time steps, generating the same summary fragments. In most cases, these repeated fragments do not match the reference summary, resulting in lower ROUGE scores. Adding a pointer generator network to the seq2seq model can better handle the OOV problem, but it makes the summary redundant by generating more unnecessary repeated fragments. Adding a coverage mechanism to the pointer generator network can effectively eliminate the word repetition content brought by the pointer generator network, thus achieving better results than the pointer generator network alone.

4.7.2 Analysis of Parameter Setting Impact From the impact of parameter settings on the model, when conducting horizontal comparison experiments on the Attention+PG+Coverage model from the perspectives of vocabulary size and word vector dimension, both vocabulary size and word vector dimension significantly affect the model' s effectiveness on abstractive Chinese text summarization.

Regarding vocabulary size, when the vocabulary is of a specific size (e.g., 60,000), it can include enough high-frequency words while excluding certain low-frequency words, enabling the generation of more source document information as summaries when sampling from the vocabulary. For words not included in the vocabulary, they are copied into the summary through the pointer generator module, thus generating the best quality summaries. When

the vocabulary size is small (e.g., 20,000, 30,000, 40,000, and 50,000), some higher-frequency words in the training set are not included in the vocabulary (compared with the highest-frequency words, relatively higher-frequency words are ignored). When the model samples from the vocabulary, it only includes words with the highest probability in the summary, resulting in summaries containing only a few high-frequency keywords and having lower matching degrees with reference summaries. When the vocabulary size is large (e.g., 70,000 and 80,000), low-frequency words appearing in the training data are also included in the vocabulary. These low-frequency words have weak word vectors in the model learning process, and even if they contain important information from the source document, they are difficult to be selected to form summaries. Therefore, the final summaries tend to lack important information and differ from reference summaries.

Regarding word vector dimension, when setting a specific word vector dimension (e.g., 128 dimensions), the distributed representation can both effectively capture the semantic information of words in the source document and alleviate word vector sparsity. The model effect is far better than results obtained using other vector dimensions, thus generating higher quality summaries. When the word vector dimension is small (e.g., 64 dimensions), the distributed representation learned by the model cannot well capture the meaning of words in the source document, and the model cannot well obtain abstract information from the source document, resulting in gaps between generated summaries and reference summaries. When the word vector dimension is set high (e.g., 256 dimensions), the learned word vectors may become relatively sparse, unable to well represent semantic information and internal connections between words, thus resulting in lower quality summaries based on such vector representations.

4.7.3 Analysis of Model Limitations From the comparison between experimental results and actual generated summaries, summaries generated using the pointer generator network and coverage mechanism are mostly of good quality, showing significant improvement over baseline methods. However, there are a few cases where the final generated summaries are inconsistent with experimental results. For example, in the summary generated by the pointer generator with coverage mechanism, "US study: Obese people are 3 times more likely to experience memory loss, did you know? Did you know?" , there are still unnecessary repeated fragments "Did you know?" . This may be because this fragment appears frequently in the training data and multiple times in the source document, so the model has a higher probability of repeatedly focusing on the same position at different time steps, resulting in repeated content in the summary. The summary generated by the pointer generator network still contains a small number of [UNK] tokens, such as "Man drank 8 liang of liquor and went swimming by the river, [UNK] 10 hours" . The authors analyze that the reason is that there are some words that appear with low frequency in the training set but are still included in the vocabulary (words with frequency ranking in the TOP K), whose vector representations are weak. During model learning, they

cannot be accurately generated from the vocabulary, and simultaneously the model calculates a low word copying probability, so they cannot be copied from the source document and are ultimately replaced by [UNK] tokens.

5 Conclusion

To solve the OOV problem and summary repetition problem in abstractive text summarization, this paper adds a pointer generator module and a coverage processing module to the encoder, decoder, and attention module of the seq2seq model. From the experimental results, on the one hand, the model can copy OOV words from the source document into the final summary, thus effectively solving the OOV problem commonly existing in traditional sequence models and eliminating [UNK] tokens in summaries. On the other hand, the coverage processing module can prevent the model from repeatedly focusing on the same position in the source document at each time step, thereby avoiding the generation of repeated summary fragments. Experimental results show that in abstractive Chinese text summarization tasks, using pointer generator networks and coverage mechanisms can effectively solve the OOV problem and summary repetition problem, thereby significantly improving text summarization quality.

The limitations of this paper are: The experimental dataset is limited to Chinese. In future work, we will use datasets in more languages to study the application of seq2seq models to abstractive text summarization. We only compare with some classic non-seq2seq model methods (including TextRank, Lead-1-First, and Lead-1-Last). In future work, we will compare with more non-sequence models to further verify the effectiveness of the model.

References

[1] SUTSKEVER I, VINYALS O, LE Q V. Sequence to sequence learning with neural networks[C]//Proceedings of 2014 annual conference on neural information processing systems (NIPS). Montreal: Neural Information Processing Systems Foundation, 2014: 3104-3112.

[2] LE H T, LE T M. An approach to abstractive text summarization[C]//Proceedings of 2013 soft computing and pattern recognition (SoCPaR). Hanoi: IEEE, 2013: 371-376.

[3] Zhao Wenjuan, Liu Zhongbao. Research on Chinese framework-based network event extraction and related algorithms[J]. Information Studies: Theory & Application, 2016, 39(10): 112-116.

[4] Zhang Han, Zhao Yuhong. Construction of a medical multi-document summarization extraction model based on semantic graphs[J]. Library and Information Service, 2017, 61(8): 112-119.

[5] KHAN A, SALIM N, FARMAN H, et al. Abstractive text summarization based on improved semantic graph approach[J]. International journal of parallel programming, 2018, 46(1): 1-25.

[6] Wang Zhenchao, Sun Rui, Ji Donghong. Event-guided multi-document generative summarization method[J]. Application Research of Computers, 2017, 34(2): 343-346.

[7] BAHDANAU D, CHO K, BENGIO Y. Neural machine translation by jointly learning to align and translate[EB/OL]. [2017-12-30]. https://arxiv.org/pdf/1409.0473.pdf.

[8] RUSH A M, CHOPRA S, WESTON J. A neural attention model for abstractive sentence summarization[EB/OL]. [2017-12-30]. https://arxiv.org/pdf/1509.00685.pdf.

[9] CHOPRA S, AULI M, RUSH A M. Abstractive sentence summarization with attentive recurrent neural networks[C]//Conference of the North American chapter of the Association for Computational Linguistics. San Diego: Human Language Technologies, 2016: 93-98.

[10] GULCEHRE C, AHN S, NALLAPATI R, et al. Pointing the unknown words[C]//Proceedings of the 54th annual meeting of the Association for Computational Linguistics. Berlin: ACL, 2016: 140-149.

[11] MIAO Y, BLUNSOM P. Language as a latent variable: discrete generative models for sentence compression[C]//Proceedings of the 2016 conference on empirical methods in natural language processing. Austin: EMNLP, 2016: 319-328.

[12] Xie Mingyuan. Text automatic summarization model based on text category[J]. Computer Knowledge and Technology: Academic Exchange, 2018, 14(1): 206-208.

[13] JEAN S, CHO K, MEMISEVIC R, et al. On using very large target vocabulary for neural machine translation[EB/OL]. [2018-02-10]. https://arxiv.org/pdf/1412.2007.pdf.

[14] XIE Z, AVATI A, ARIVAZHAGAN N, et al. Neural language correction with character-based attention[EB/OL]. [2017-12-30]. https://arxiv.org/pdf/1603.09727.pdf.

[15] LUONG M T, SUTSKEVER I, LE Q V, et al. Addressing the rare word problem in neural machine translation[J]. Bulletin of university of agricultural sciences and veterinary medicine cluj-napoca. veterinary medicine, 2014, 27(2): 82-86.

[16] GU J, LU Z, LI H, et al. Incorporating copying mechanism in sequence-to-sequence learning[C]//Proceedings of the 54th annual meeting of the Association for Computational Linguistics. Berlin: ACL, 2016: 1631-1640.

[17] NALLAPATI R, ZHOU B, SANTOS C N D, et al. Abstractive text summarization using sequence-to-sequence RNNs and beyond[C]//Proceedings of the

20th SIGNLL conference on computational natural language learning. Berlin: CoNLL, 2016: 280-290.

[18] TU Z, LU Z, LIU Y, et al. Modeling coverage for neural machine translation[C]//Proceedings of the 54th annual meeting of the Association for Computational Linguistics. Berlin: ACL, 2016: 76-85.

[19] CHO K, MERRIENBOER B V, GULCEHRE C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[EB/OL]. [2018-03-01]. https://arxiv.org/pdf/1406.1078.pdf.

[20] HOCHREITER S, SCHMIDHUBER J. Long Short-Term Memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[21] SEE A, LIU P J, MANNING C D. Get to the point: summarization with pointer-generator networks[C]//Proceedings of the 55th annual meeting of the Association for Computational Linguistics. Vancouver: ACL, 2017: 1073-1083.

[22] HU B, CHEN Q, ZHU F. LCSTS: a large scale Chinese short text summarization dataset[C]//Proceedings of the 2015 conference on empirical methods in natural language processing. Lisbon: EMNLP, 2015: 2667-2671.

[23] SUN J. Chinese word segmentation tool[EB/OL]. [2017-10-20]. https://pypi.python.org/pypi/jieba/.

[24] FLICK C. ROUGE: a package for automatic evaluation of summaries[EB/OL]. [2017-12-30]. http://www.aclweb.org/anthology/W04-1013.

[25] MIHALCEA R, TARAU P. TextRank: bringing order into texts[C]//Proceedings of the 2004 conference on empirical methods in natural language processing. Barcelona: EMNLP, 2004: 404-411.

Author Contributions

Yu Chuanming: Conceptualization, data acquisition, model experimentation, initial draft writing and revision;
Zhu Xingyu: Baseline method and vocabulary size extension experiments, initial draft writing and revision;
Gong Yutian: Dataset preprocessing, vector dimension extension experiments, paper revision;
An Lu: Conceptualization and paper revision.

Note: Figure translations are in progress. See original paper for figures.