Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 9898169, 1 page
https://doi.org/10.1155/2023/9898169

Retraction

Retracted: Text Mining Based on the Lexicon-Constrained Network in the Context of Big Data

Wireless Communications and Mobile Computing

Received 10 November 2022; Accepted 10 November 2022; Published 21 January 2023

Wireless Communications and Mobile Computing has retracted the article titled "Text Mining Based on the Lexicon-Constrained Network in the Context of Big Data" [1] due to concerns that the peer review process has been compromised.

Following an investigation conducted by the Hindawi Research Integrity team [2], significant concerns were identified with the peer reviewers assigned to this article; the investigation has concluded that the peer review process was compromised. We therefore can no longer trust the peer review process and the article is being retracted with the agreement of the Chief Editor.

The authors disagree with the retraction.

References

[1] B. Wan and M. Sohail, "Text Mining Based on the Lexicon-Constrained Network in the Context of Big Data," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 8703100, 10 pages, 2022.

[2] L. Ferguson, "Advancing Research Integrity Collaboratively and with Vigour," 2022, http://www.hindawi.com/post/advancing-research-integrity-collaboratively-and-vigour/.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8703100, 10 pages
https://doi.org/10.1155/2022/8703100

Research Article

Text Mining Based on the Lexicon-Constrained Network in the Context of Big Data

Boyan Wan[1] **and Mishal Sohail**[2]

[1]*School of Computer Science, National University of Defense Technology, Changsha, China*
[2]*Department of Industrial Engineering, International Ataturk Alatoo University, Bishkek, Kyrgyzstan*

Correspondence should be addressed to Mishal Sohail; dr.mishalsohail@mail.cu.edu.kg

Received 23 December 2021; Revised 3 February 2022; Accepted 15 February 2022; Published 12 March 2022

Academic Editor: Rashid A Saeed

Unstructured textual news data is produced every day; analyzing them using an abstractive summarization algorithm provides advanced analytics to decision-makers. Deep learning network with copy mechanism is finding increasing use in abstractive summarization, because copy mechanism allows sequence-to-sequence models to choose words from the input and put them directly into the output. However, since there is no explicit delimiter in Chinese sentences, most existing models for Chinese abstractive summarization can only perform character copy, resulting in inefficiency. To solve this problem, we propose a lexicon-constrained copying network that models multigranularity in both encoder and decoder. On the source side, words and characters are aggregated into the same input memory using a Transformer-based encoder. On the target side, the decoder can copy either a character or a multicharacter word at each time step, and the decoding process is guided by a word-enhanced search algorithm which facilitates the parallel computation and encourages the model to copy more words. Moreover, we adopt a word selector to integrate keyword information. Experiment results on a Chinese social media dataset show that our model can work standalone or with the word selector. Both forms can outperform previous character-based models and achieve competitive performances.

1. Introduction

In recent years, abstractive summarization [1] has made impressive progress with the development of sequence-to-sequence (seq2seq) framework [2, 3]. This framework is composed by an encoder and a decoder. The encoder processes the source text and extracts the necessary information for the decoder, which then predicts each word in the summary. Thanks to their generative nature, abstractive summaries can include novel expressions never seen in the source text, but at the same time, abstractive summaries are more difficult to produce compared with extractive summaries [4, 5] which are formed by directly selecting a subset of the source text.

It has been also found that seq2seq-based abstractive methods usually struggle to generate out-of-vocabulary (OOV) words or rare words, even if those words can be found in the source text. Copy mechanism [6] can alleviate this problem and meanwhile maintain the expressive power of the seq2seq framework. The idea is to allow the decoder not only to generate a summary from scratch but also copy words from the source text.

Though effective in English text summarization, the copy mechanism remains relatively undeveloped in the summarization of some East Asian languages, e.g., Chinese. Generally speaking, abstractive methods for Chinese text summarization come in two varieties, being word-based and character-based. Since there is no explicit delimiter in Chinese sentence to indicate word boundary, the first step of word-based methods [7] is to perform word segmentation [8, 9]. Actually, in order to avoid the segmentation error and to reduce the size of vocabulary, most of the existing methods are character-based [10–12]. When trying to combine the character-based methods in Chinese with copy mechanism, the original "word copy" degrades to "character copy" which does not guarantee a multicharacter word to be

copied verbatim from the source text [13]. Unfortunately, copying multicharacter words is quite common in Chinese summarization tasks. Take the Large-Scale Chinese Social Media Text Summarization Dataset (LCSTS) [7] as an example; according to Table 1, about 37% of the words in the summaries are copied from the source texts and consist of multiple characters.

Selective read [13] was proposed to handle this problem. It calculates the weighted sum of encoder states corresponding to the last generated character and adds this result to the input of the next decoding step. Selective read can provide location information of the source text for the decoder and help it to perform the consecutive copy. A disadvantage of this approach, however, is that it increases reliance of present computation on partial results before the current step which makes the model more vulnerable to the error accumulation and leads to exposure bias [14] during inference. Another way to make copied content consecutive is through directly copying text spans. Zhou et al. [15] implement span copy operation by equipping the decoder with a module that predicts the start and end positions of the span. Because a longer span can be decomposed to shorter ones, there are actually many different paths to generate the same summary during inference, but their model is optimized by only the longest common span at each time step during training, which exacerbates the discrepancy between two phases. In this work, we propose a novel lexicon-constrained copying network (LCN). The decoder of LCN can copy either a single character or a text span at a time, and we constrain the text span to match a potential multicharacter word. Specifically, given a text and several off-the-shell word segmentators, if a text span is included in any segmentation result of the text, we consider it as a potential word. By doing so, the number of available spans is significantly reduced, making it viable to marginalize over all possible paths during training. Furthermore, during inference, we aggregate all partial paths on the fly producing the same output using a word-enhanced beam search algorithm, which encourages the model to copy multicharacter words and facilitates the parallel computation.

To be in line with the aforementioned decoder, the encoder should be revised to learn the representations of not only characters but also multicharacter words. In the context of neural machine translation, Su et al. [16] first organized characters and multicharacter words in a directed graph named word lattice. Following Xiao et al. [17], we adopt an encoder based on the Transformer [18] to take the word lattice as input and allow each character and word to have its own hidden representation. By taking into account relative positional information when calculating self-attention, our encoder can capture both global and local dependencies among tokens, providing an informative representation of source text for the decoder to make copy decisions.

Although our model is character-based (because only characters are included in the input vocabulary), it can directly utilize word-level prior knowledge, such as keywords. In our setting, keywords refer to words in the source text that have a high probability of inclusion in the summary. Inspired by Gehrmann et al. [19], we adopt a separate word selector based on the large pretrained language model, e.g., BERT [20], to extract keywords. When the decoder intends to copy words from the source text, those selected keywords will be treated as candidates, and other words will be masked out. Experimental results show that our model can achieve better performance when incorporating with the word selector.

Table 1: Percentage of different types of words that occur in the summaries of Chinese text summarization training data.

Word len.	Copied	Generated
1	21.6%	12.3%
2	28.9%	21.8%
⩾3	7.6%	7.7%

2. Related Work

Most existing neural methods to abstractive summarization fall into the sequence to sequence framework. Among them, models based on recurrent neural networks (RNNs) [21–23] are more common than those built on convolutional neural network (CNNs) [1, 24], because the former models can more effectively handle long sequences. Attention [3] is easily integrated with RNNs and CNNs, as it allows the model to focus more on salient parts of the source text [25, 26]. Also, it can serve as a pointer to select words in the source text for copying [6, 13]. In particular, architectures that are constructed entirely of attention, e.g., Transformer [18], can be adopted to capture global dependencies between source text and summary [19].

Prior knowledge has proven helpful for generating informative and readable summaries. Templates that are retrieved from training data can guide summarization process at the sentence level when encoded in conjunction with the source text [27, 28]. Song et al. [29] show that the syntactic structure can help to locate the content that is worth keeping in the summary, such as the main verb. Keywords are commonly used in Chinese text summarization. When the decoder is querying from the source representation, Wang and Ren [30] use the keywords extracted by the unsupervised method to exclude noisy and redundant information. Deng et al. [31] propose a word-based model that not only utilizes keywords in the decoding process but also adds the keywords produced by the generative method into the vocabulary in the hope of alleviating out-of-vocabulary (OOV) problem. Our model is drastically different from the above two models in terms of the way keywords being extracted and encoded.

The most related works are in the field of neural machine translation, in which many researchers resort to the assistance of multigranularity information. On the source side, Su et al. [16] use an RNN-based network to encode the word lattice, an input graph that contains both word and character. Xiao et al. [17] apply the lattice-structured input to the Transformer [18] and generalize the lattice to construct at a subword level. To fully take advantage of multihead attention in the Transformer, Nguyen et al. [32] first partition

input sequence to phrase fragments based on n-gram type and then allow each head to attend to either one certain n-gram type or all different n-gram types at the same time. In addition to n-gram phrases, the multigranularity self-attention proposed by Hao et al. [33] also attends to syntactic phrases obtained from syntactic trees to enhance structure modeling. On the target side, when the decoder produces an UNK symbol which denotes a rare or unknown word, Luong et al. restore it to a natural word using a character-level component. Srinivasan et al. [34] adopt multiple decoders that map the same input into translations at different subword levels and combine all the translations into the final result, trying to improve the flexibility of the model without losing semantic information. While our model and the above models all utilize multigranularity information, our model differs at that we impose a lexical constraint on both encoding and decoding.

3. Model

This section describes our proposed model in detail.

3.1. Notations. Let character sequence $x_{1:I} = \{x_1, \cdots, x_I\}$ be a source text; we can define a text span $x_{i:j}$ that starts with x_i and ends with x_j a potential word if it is contained by any word segmentation result of $x_{1:I}$. Because both characters and words can be regarded as tokens, we include all characters and potential words of the source text in a token sequence $o_{1:M} = \{o_1, \cdots, o_M\}$.

3.2. Input Representation. Given a token $o_m = \{x^1, \cdots, x^l\}$, where l is the token length ($l = 1$ when o_m is a character), we first convert it into a sequence of vectors, using the character embedding $\mathbf{E}^c$. Then, a bidirectional Long Short-Term Memory Network (bi-LSTM) is applied to model the token composition:

$$\mathbf{g}_m = \left[\overleftarrow{\mathrm{LSTM}}\left(\mathbf{E}^c\left(x^1\right)\right), \overrightarrow{\mathrm{LSTM}}\left(\mathbf{E}^c\left(x^l\right)\right)\right], \tag{1}$$

where $\mathbf{g}_m \in \mathbb{R}^d$ denotes the input token representation, which is formed by concatenating the backward state of the beginning character and the forward state of the ending character.

Since the Transformer has no sequential structure, Vaswani et al. [18] proposed positional encoding to explicitly model the order of the sequence. In this work, we assign each token an absolute position which depends on the first character of the token. For example, the absolute position of the word "留学 (studying abroad)" in Figure 1 is the same as that of the character "留 (stay)." By adding the encoding of the absolute position to the token representation, we can get the final input representation $\mathbf{G} = \{\mathbf{g}_1, \cdots, \mathbf{g}_M\}$.

3.3. Encoder. However, absolute position alone cannot precisely reflect the relationship among tokens. Consider again the example in Figure 1; the distance between the word "留学 (studying abroad)" and the character "生 (life)" is 2 according to their absolute positions, but they are actually neighboring tokens in a certain segmentation. To alleviate this problem, Xiao et al. [17] extend the Transformer [18] by taking into account relation types when calculating self-attention. In this work, we adopt relative position as an alternative to relation type. The main idea is that relative position is complementary to absolute position and can guide each token to interact with other tokens in a coherent manner. Given two tokens $x_{a:b}$ and $x_{c:d}$ that correspond to a span of the source text each, the position of $x_{c:d}$ relative to $x_{a:b}$ is determined by both their beginning and ending characters as shown in Table 2. Following Xiao et al. [17], we revise self-attention to integrate relative positional information. Concretely, a self-attention layer consists of h heads, which operate in parallel on a sequence $\mathbf{H} = \{\mathbf{h}_1, \cdots, \mathbf{h}_M\}$ of context vectors with dimension d. After modification, the resulting output **attn** for each attention head is defined as

$$\begin{aligned} e_{ij} &= \frac{\left(\mathbf{W}^q\mathbf{h}_i\right)\left(\mathbf{W}^k\mathbf{h}_j + \mathbf{p}_{ij}^k\right)^T}{\sqrt{d_z}}, \\ a_{ij} &= \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{M}\exp\left(e_{ik}\right)}, \\ \mathbf{attn}_i &= \sum_{j=1}^{M} a_{ij}\left(\mathbf{W}^v\mathbf{h}_j + \mathbf{p}_{ij}^v\right), \end{aligned} \tag{2}$$

where $\mathbf{W}^q$, $\mathbf{W}^k$, and $\mathbf{W}^v \in \mathbb{R}^{d_z \times d}$ are all model parameters, $d_z = d/h$ is the hidden dimension for each head, and $\mathbf{p}_{ij}^k$ and $\mathbf{p}_{ij}^v$ are learned embeddings that encode the position of token t_j relative to token t_i. We concatenate the outputs of all heads to restore their dimension to d and then apply other sublayers (such as feed-forward layer) used in the original Transformer [18] to get the final output of the layer. Several identical self-attention layers are stacked to build our encoder. For the first layer, $\mathbf{H}$ is input representation $\mathbf{G}$. For the subsequent layers, $\mathbf{H}$ is the output of the previous layer.

3.4. Decoder. The encoder proposed by Xiao et al. [17] takes both characters and words as input and thus has the ability to learn multigranularity representations. However, as their decoder is character-based which consumes and outputs only characters, the word representations induced from the encoder cannot receive supervision signal directly from the decoder and remain a subsidiary part of the input memory. To alleviate this problem, we extend the standard Transformer decoder by a lexicon-constrained copying module, which not only allows the decoder to perform multicharacter word copy but also provides auxiliary supervision on word representations. Specifically, at each time step t, we leverage a single-head attention over the input memory $\mathbf{H} = \{\mathbf{h}_1, \cdots, \mathbf{h}_M\}$ and the decoder hidden state $\mathbf{s}_t$ to produce copy distribution $\mathbf{a}_t$ and context vector $\mathbf{c}_t$:

$$e_{tj} = \frac{\left(\mathbf{W}_{\mathrm{copy}}^q\mathbf{s}_t\right)\left(\mathbf{W}_{\mathrm{copy}}^k\mathbf{h}_j\right)^T}{\sqrt{d}}, \tag{3}$$

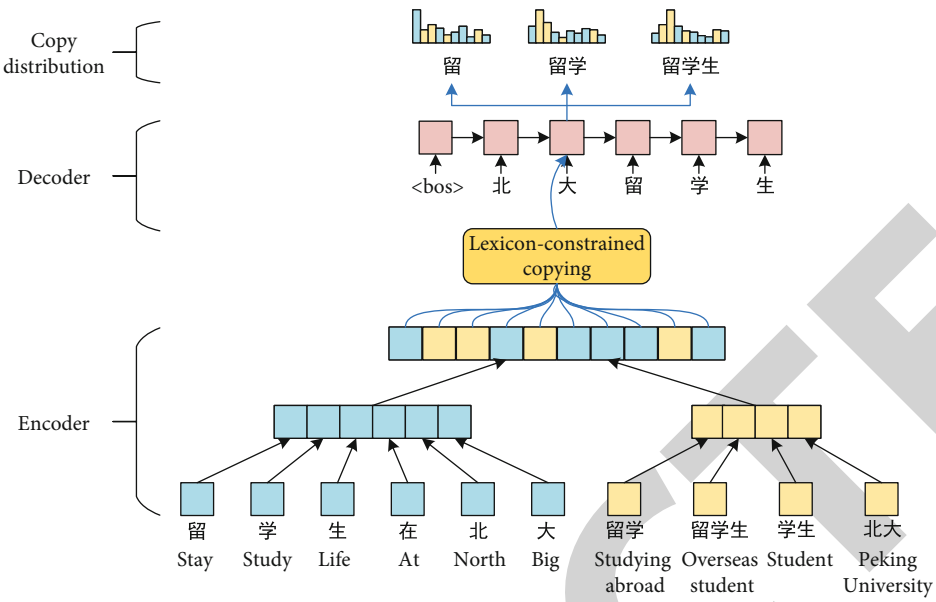

Figure 1: Structure overview of the proposed model.

$$a_{tj} = \frac{\exp\left(e_{tj}\right)}{\sum_{k=1}^{M}\exp\left(e_{tk}\right)}, \tag{4}$$

$$\mathbf{c}_t = \sum_{j=1}^{M} a_{tj}\left(\mathbf{W}_{\text{copy}}^{v}\mathbf{h}_j\right). \tag{5}$$

In addition to the predefined character vocabulary $\mathscr{V}$ and a special token UNK denoting any out-of-vocabulary token, lexicon-constrained copying module expands the output space by two sets $\mathscr{C}$ and $\mathscr{W}$, consisting of characters and multicharacter words that appear in the source text, respectively, so that the probability of emitting any token o at time step t is

$$P(o|\cdot) = \begin{cases} p_{\text{gen}}\text{gen}(o) + p_{\text{copy}}\sum_{i:o_i=o} a_{ti}, & o \in \mathscr{V} \cup \mathscr{C}, \\ p_{\text{copy}}\sum_{i:o_i=o} a_{ti}, & o \in \mathscr{W}, \\ p_{\text{gen}}\text{gen}(\text{UNK}), & \text{otherwise}, \end{cases} \tag{6}$$

where p_{gen} and $p_{\text{copy}} \in [0, 1]$ control the decoder switching between generation mode and copy mode and $\text{gen}(\cdot)$ provide a probability distribution over character vocabulary $\mathscr{V}$:

$$\begin{aligned} p_{\text{gen}} &= \sigma\left(\mathbf{V}_g[\mathbf{s}_t, \mathbf{c}_t] + \mathbf{b}_p\right), \\ p_{\text{copy}} &= 1 - p_{\text{gen}}, \\ \text{gen}(\cdot) &= \text{softmax}\left(\widehat{\mathbf{V}_g}\left(\mathbf{V}_g[\mathbf{s}_t, \mathbf{c}_t] + \mathbf{b}\right) + \widehat{\mathbf{b}_g}\right), \end{aligned} \tag{7}$$

where $\sigma(\cdot)$ is the sigmoid function.

With the introduction of lexicon-constrained copying module, our decoder can predict tokens of variable lengths at each time step and thereby can generate any segmentation of a sentence. Naturally, we expect to evaluate the probability of a summary by marginalizing over all its segmentations. For example, the probability of a summary consisting of only one word "北京 (BeiJing)" can be factorized as

Table 2: Position of $x_{c:d}$ relative to $x_{a:b}$ under different conditions. Constant r is used to limit the relative position from 0 to $2r + 3$, where $2r + 1$ to $2r + 3$ each represent special cases: $x_{c:d}$ includes $x_{a:b}$, $x_{c:d}$ is included in $x_{a:b}$, and $x_{c:d}$ intersects with $x_{a:b}$.

Conditions	Relative position
$d < a$	$\max(0, r - a + d)$
$a = c$ and $b = d$	r
$b < c$	$\min(2r, r + c - b)$
$c \le a \le b < d$ or $c < a \le b \le d$	$2r + 1$
$a \le c \le d < b$ or $a < c \le d \le b$	$2r + 2$
Otherwise	$2r + 3$

$$P(\text{Bei, Jing}) = P(\text{Bei} \mid \epsilon)P(\text{Jing} \mid \text{Bei})P(\epsilon \mid \text{Bei, Jing}) + P(\text{BeiJing} \mid \epsilon)P(\epsilon \mid \text{BeiJing}), \tag{8}$$

where each term corresponds to a segmentation and is the product of conditional probabilities; we use ϵ to denote either the beginning or end of a sentence. Note that the conditional probability here depends on the current segmentation, which means the decoder directly take tokens in a segmentation as input. However, if we feed the decoder with character-level input and reformulate the conditional probability, the above probability can be rewritten as

$$P(\text{Bei, Jing}) = P(\epsilon \mid \text{Bei, Jing})(P(\text{Bei} \mid \epsilon)P(\text{Jing} \mid \text{Bei}) + P(\text{BeiJing} \mid \epsilon)), \tag{9}$$

where we factor out $P(\epsilon \mid \text{Bei, Jing})$, because it is shared by two different segmentations. As can be seen from the above example, the assumption that conditional probability of a token depends only on its preceding character sequence facilitates the reuse of computation and thus makes it feasible to apply dynamic programming. Formally, let character sequence $y_{1:J} = \{y_1, \cdots, y_J\}$ be a summary; its probability can be represented as a recursion:

$$P(y_{1:J}) = \sum_{\substack{o \in \mathcal{V} \cup \mathcal{C} \cup \mathcal{W} \\ o = y_{J-\ell+1:J}}} P(o \mid y_{1:J-\ell}) P(y_{1:J-\ell}). \tag{10}$$

Note that all the above $P(\cdot)$ are inevitably conditioned on the source text, so we omit it for simplicity. We train the model by maximizing $P(y_{1:J})$ for all training pair in the dataset.

During inference, since there is no access to the ground truth, we need a search algorithm which can guide the generation of the summary in a left-to-right fashion. Beam search [35] is the most common search algorithm in seq2seq frameworks, but cannot be used directly in our scenario. To illustrate this, we first define hypothesis as a partial output that consists of tokens. Hypotheses can be further divided into character hypotheses and word hypotheses based on whether their last token is a character or a multicharacter word. For hypotheses within a beam, the standard beam search algorithm updates states for them by feeding their last tokens to the decoder and then generates new hypotheses through suffixing them with a token sampled from the model's distribution. Because our decoder is designed to take only characters as input, multiple decoder steps are required to update the state for a word hypothesis. As a result, it is difficult to conduct batched update for a beam containing both word hypotheses and character hypotheses. To this end, we proposed a novel word-enhanced beam search algorithm, where the beam is also split into two parts: the character beam and the word beam. The word beam is used to update the states for word hypotheses. When their states are fully updated, word hypotheses are placed into the character beam (see lines 5-8 of Algorithm 1). Note that we do not perform generation step for word hypotheses in the word beam, that is to say, with the same length, the more multicharacter words a hypothesis includes, the more generation steps it can skip, which may give it a higher probability.

3.5. Word Selector. We treat keyword selection as a binary classification task on each potential word. To obtain word representations, we first leverage BERT [20], a pretrained language model to produce context-aware representations $\mathbf{x}^c$ for all characters in the source text and then feed them to a bi-LSTM network. Different from Section 3.2 where bi-LSTM is applied to character sequence of each word, here the bi-LSTM takes the whole source character representations as input, in an attempt to build word representation that can reflect how much contextual information the word carries. Given a potential word $x_{a:b}$, where a and b are indexes of characters in the source text, we can calculate its final representation $\mathbf{t}$ as follows:

$$\begin{aligned} \overrightarrow{\mathbf{d}_a} &= \overrightarrow{\text{LSTM}}(\mathbf{x}_a^c), \overrightarrow{\mathbf{d}_b} = \overrightarrow{\text{LSTM}}(\mathbf{x}_b^c), \\ \overleftarrow{\mathbf{d}_a} &= \overleftarrow{\text{LSTM}}(\mathbf{x}_a^c), \overleftarrow{\mathbf{d}_b} = \overleftarrow{\text{LSTM}}(\mathbf{x}_b^c), \\ \mathbf{t} &= \left[\overrightarrow{\mathbf{d}_a}, \overrightarrow{\mathbf{d}_b}, \overleftarrow{\mathbf{d}_a}, \overleftarrow{\mathbf{d}_b}, \overrightarrow{\mathbf{d}_b} - \overrightarrow{\mathbf{d}_a}, \overleftarrow{\mathbf{d}_a} - \overleftarrow{\mathbf{d}_b}\right]. \end{aligned} \tag{11}$$

Then, a linear transformation layer and a sigmoid function can be used sequentially on its final representation to compute the probability of $x_{a:b}$ being selected.

During training, words that appear in both summary and source text are considered as positives, and the rest are negatives. To make sure that decoder can access the entire source character sequence at inference time, in addition to the multicharacter words with the top $-n$ probability, we treat all characters in source text as keywords. Inspired by [19], we utilize keyword information by masking out other words when calculating copy distribution. In particular, we leave e_{tj} in Equation (4) unchanged for keywords and set e_{tj} to zero for the rest of the words.

4. Experiments

4.1. Datasets and Evaluation Metric. We conduct experiments on the Large-Scale Chinese Social Media Text Summarization Dataset (LCSTS) [7], which consists of source texts with no more than 140 characters, along with human-generated summaries. The dataset is divided into three parts; the (source, summary) pairs in PART II and PART III are scored manually from 1 to 5, with higher scores indicating more relevance between the source text and its summary. Following Hu et al. [7], after removing pairs with scores less than 3, PART I, PART II, and PART III are used as training set, verification set, and test set, respectively, with 2.4M pairs in PART I, 8K pairs in PART II, and 0.7K pairs in PART III.

We choose ROUGE score [36] as our evaluation metric, which is widely used for evaluating automatically produced summaries. The metric measure the relevance between a source text and its summary based on their cooccurrence statistics. In particular, ROUGE-1 and ROUGE-2 depend on unigram and bigram overlap, respectively, while ROUGE-L relies on the longest common subsequence.

4.2. Experimental Setup. The character vocabulary is formed by 4000 most frequent characters in the training set. To get all potential words, we use PKUSEG [37], a toolkit for multi-domain Chinese word segmentation. Specifically, there are separate segmentators for four domains, including web, news, medicine, and tourism. We use these segmentators for the source text, and if a text span is included in any of word segmentation results, we regard it as a potential word.

For the lexicon-constrained copying network, we employ six attention layers of 8 heads for both encoder and decoder. Constant r in Table 2 is set to 8. We make character embedding and all hidden vectors the same dimension of 512 and

Input: *model*, *source*, maximum summary length L, beam size k
1: startHyp $\longleftarrow$ getStartHyp(ϵ)
2: $B_c = \{\text{startHyp}\}$, $B_w = \{\}$ and $B_f = \{\}$
3: **for** $t = 0$; $t++$; $t < L$ **do**
4: $n, m = \{\}$
5: $B_c, B_w \longleftarrow$ model.batchedUpdate(B_c, B_w, source)
//Batched update for hypos in both B_c and B_w
6: **for** hyp $\in B_w$ **do**
7: **if** hyp is Updated **then**
8: move hyp into B_c
9: **end if**
10: **end for**
11: Merge hypos of the same character sequence in B_c
12: $n \longleftarrow$ model.generate(B_c)
//Generating new hypotheses and their respective log probabilities from B_c
13: **for** hyp $\in n$ **do**
14: **if** hyp ends with a multicharacter word **then**
15: Move hyp from n into m
16: **end if**
17: **end for**
18: $B_w \longleftarrow k - \text{argmax}_{\text{hyp}\in m}$ hyp.avgLogProb
19: $B_c \longleftarrow k - \text{argmax}_{\text{hyp}\in n}$ hyp.avgLogProb
20: **for** hyp $\in B_c$ **do**
21: **if** hyp ends with ϵ or hyp.len = L **then**
22: Move hyp from n into B_f
23: **end if**
24: **end for**
25: **end for**
26: finalHyp $\longleftarrow \text{argmax}_{\text{hyp}\in B_f}$ hyp.avgLogProb
27: **return** finalHyp

Algorithm 1: Pseudocode for word-enhanced beam search.

Table 3: Results of different models. We use † to indicate that the model utilizes keyword information.

Models	ROUGE-1	ROUGE-2	ROUGE-L
RNN [7]	21.5	8.9	18.6
RNN-Content [7]	29.9	17.4	27.2
COPYNET [13]	34.4	21.6	31.3
superAE [11]	39.2	26.0	36.2
Global Encoding [10]	39.4	26.9	36.5
KESG† [31]	39.4	28.4	35.3
KGWA† [30]	40.9	28.3	38.2
Transformer	38.9	27.4	35.5
CopyTransformer	39.7	28.0	35.8
LCCN	41.7	29.5	38.0
w/o word-enhanced beam search	40.0	28.5	37.1
LCCN+word selector†	**42.3**	**29.8**	**38.4**

Table 4: Results of different approaches to extract keywords.

Models	ROUGE-1	ROUGE-2	ROUGE-L
TFIDF	28.6	13.1	20.7
Encoder of LCCN	42.0	25.6	33.7
Word selector	46.0	28.2	36.1

set the filter size of feed-forward layers to 1024. For word selectors, we use a single-layer Bi-LSTM with a hidden size of 512.

During training, we update the parameters of the lexicon-constrained copying network (LCCN) and word selector by Adam optimizer with $\beta 1 = 0.9$, $\beta 2 = 0.98$, and $\varepsilon = 10^{-9}$. The same learning rate schedule of Vaswani et al. [18] is used to the LCCN, while a fixed learning rate of 0.0003 is set for word selector. The BERT we use in the word selector is pretrained on Chinese corpus by Wolf et al. [38] and we freeze its parameters throughout the training.

During testing, we use a beam size of 10 and take the first 10 multicharacter words predicted by the word selector and all characters in the source text as keywords.

Table 5: Summarization examples. Summaries in the last two blocks are separated by spaces to show the output of LCCN and LCCN+word selector per step. We use keywords to denote the multicharacter words chosen by the word selector.

Source: 9月5日, 陕西咸阳双泉村垃圾站发现一名死亡女婴, 脖子上缠有绳子。近日案件告破, 令人意外的是, 婴儿的父母是17岁左右的在校高中生, 涉嫌勒死婴儿的⊠是孩子父亲。“当时看见孩子生下来心就慌了, 害怕孩子哭⊠便用绳子勒死了”。 On September 5, a dead baby girl with a rope around her neck was found at the Shuangquan Village Garbage Station in Xianyang, Shaanxi Province. Recently, the case was solved. Surprisingly, the parents of the baby were high school students around 17 years old. The father of the baby was suspected of strangulating his baby. “When I saw the baby born, I was in a panic. I was afraid of a baby crying, so I strangled her with a rope.”
Reference: 咸阳两高中生同居生下女婴因害怕孩子哭将其勒死。 Two high school students in Xianyang lived together and gave birth to a baby girl and strangled her for fear of baby crying.
Transformer: 17岁高中生当街勒死亲生父母。 17-year-old high school students strangled their parents in the street.
CopyTransformer: 陕西17岁女婴儿缠绳子勒死。 17-year-old female infant in Shaanxi was strangled with rope.
LCNN: 17岁 高中生 勒 死 婴儿。 17-year-old high school student strangled a infant.
LCNN+word selector: 高中生 因 害怕 勒 死 婴儿。 High school student strangled a infant out of fear. Keywords: 女婴 (baby girl), 婴儿 (infant), 孩子 (baby), 勒死 (strangle), 害怕 (fear), 高中 (high school), 陕西 (Shaanxi), 绳子 (rope), 高中生 (high school students), 父亲 (father)

4.3. Baselines

(i) RNN and RNN-Context are seq2seq baselines provided along with the LCSTS dataset by Hu et al. [7]. Both of them have GRU encoder and GRU decoder, while RNN-Context has an additional attention mechanism

(ii) COPYNET integrates copying mechanism into the seq2seq framework, trying to improve both content-based addressing and location-based addressing

(iii) Supervision with Autoencoder (superAE) uses an autoencoder trained on the summaries to provide auxiliary supervision for the internal representation of seq2seq. Moreover, adversarial learning is adopted to enhance this supervision

(iv) Global Encoding refines source representation with consideration of the global context by using a convolutional gated unit

(v) Keyword and Generated Word Attention (KGWA) exploits relevant keywords and previously generated words to learn accurate source representation and to alleviate the information loss problem

(vi) Keyword Extraction and Summary Generation (KESG) first uses a separate seq2seq model to extract keywords and then utilize keyword information to improve the quality of the summarization

(vii) Transformer and CopyTransformer are our implementations of the Transformer framework in the task of summarization. Copy mechanism is incorporated into CopyTransformer

4.4. Results. Table 3 records the results of our LCCN model and other seq2seq models on the LCSTS dataset. To begin with, we first compare two Transformer baselines. We can see that CopyTransformer outperforms vanilla Transformer by 0.8 ROUGE-1, 0.6 ROUGE-2, and 0.3 ROUGE-L, showing the importance of copy mechanism. The gap between our LCCN and vanilla Transformer is further widened to 1.8 ROUGE-1, 2.1 ROUGE-2, and 2.5 ROUGE-L, which asserts the superiority of lexicon-constrained copying over character-based copying. Compared to other latest models, our LCCN can achieve state-of-the-art performance in terms of ROUGE-1 and ROUGE-2 and is second only to the KGWA in terms of ROUGE-L. When also using keyword information as the KGWA does, LCCN+word selector further improves the performance and overtakes the KGWA by 0.2 ROUGE-L. We also conduct an ablation study by removing the word-enhanced beam search in LCCN, denoted by w/o word-enhanced beam search in Table 3. It shows that word-enhanced beam search can boost the performance of 1.7 ROUGE-1, 1.0 ROUGE-2, and 0.9 ROUGE-L.

4.5. Discussion. Similar to extractive summarization, we can use the top n extracted keywords to form a summary, which then can be used to evaluate the quality of keywords. The first entry of Table 4 shows the performance when the keywords are extracted by TF-IDF [39], a numerical statistic method that relies on the frequency of the word. The second entry shows the performance when we determine keywords based on the source representation learned by the encoder of LCCN. As can be seen from the last entry, word selector outperforms two methods mentioned above by a large margin, indicating the importance of external knowledge brought by the BERT.

Given a source text which describes the criminal case, we show its summaries generated by different models in

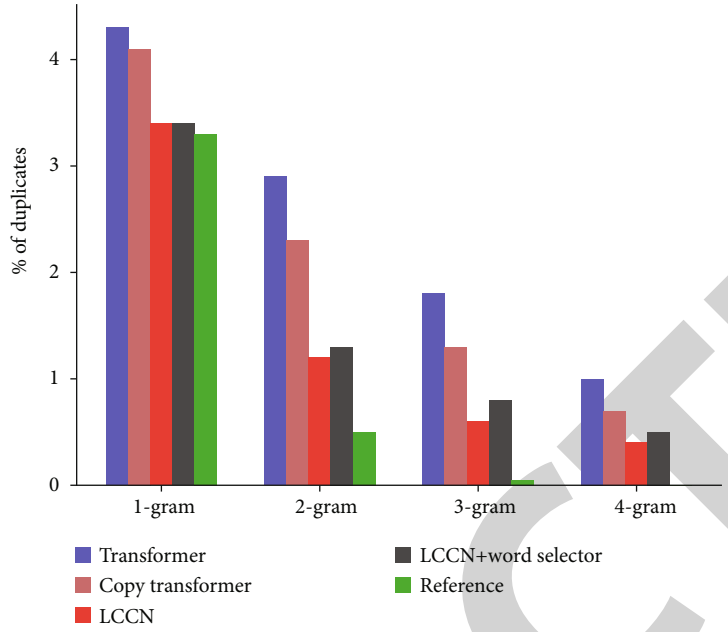

Figure 2: Percentage of the duplicates.

Table 5. It is clear that the suspect of this case is a high school student and the victim is his baby daughter. However, the summary generated by the Transformer mistakes the high school student's parents as victims and claims that the crime took place in the street, which is not mentioned in the source text. The summary of the CopyTransformer also makes a fundamental mistake, resulting the mismatch between the adjective "17岁 (17-year-old)" and the noun "女婴儿 (female infant)". Compared with them, the summary of our LCCN is more faithful to the source text and contains the correct suspect and victim, i.e., "高中生 (high school student)" and "婴儿 (infant)," which are copied from the source text through only two decoder steps. With the help of word selector, our summary can further include the keyword "害怕 (fear)" to indicate the criminal motive.

Compared with character-based models, our LCCN uses fewer steps to output a summary, so it should be able to reduce the possibility of repetition. To prove it, we record the percentage of n-gram duplicates for summaries generated by different models in Table 2. The results as shown in Figure 2 show that our model can indeed alleviate the repetition problem, and we also notice that the repetition rate of the LCCN+word selector is slightly higher than that of LCNN, which may be due to the smaller output space after adding the word selector.

5. Conclusion

In this paper, we propose a novel lexicon-constrained copying network for Chinese summarization. Querying the multigranularity representation learned by our encoder, our decoder can copy either a character or a multicharacter word at each time step. Experiments on the LCSTS dataset show that our model is superior to the Transformer baselines and quite competitive with the latest models. With the help of keyword information provided by the word selector, it can even achieve state-of-the-art performance. In the future, we plan to apply our model to other tasks, such as comment generation, and to other languages, such as English.

Data Availability

The data used to support the findings of this study are included within the article.

Disclosure

A preprint has previously been published [40].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pp. 379–389, Lisbon, Portugal, 2015.

[2] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*, pp. 3104–3112, Montreal, Quebec, Canada, 2014.

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

[4] A. Sharan, H. Imran, and M. L. Joshi, "A trainable document summarizer using Bayesian classifier approach," in *First*

International Conference on Emerging Trends in Engineering and Technology, ICETET '08, pp. 1206–1211, Nagpur, Maharashtra, India, 2008.

[5] H. Saggion and T. Poibeau, "Automatic text summarization: past, present and future," in *Multi-source, Multilingual Information Extraction and Summarization. Theory and Applications of Natural Language Processing*, T. Poibeau, H. Saggion, J. Piskorski, and R. Yangarber, Eds., pp. 3–21, Springer, Berlin, Heidelberg, 2013.

[6] C. Gulcehre, S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio, "Pointing the unknown words," in *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* of *Association for Computational Linguistics*, pp. 140–149, Berlin, Germany, 2016.

[7] B. Hu, Q. Chen, and F. Zhu, "LCSTS: a large scale Chinese short text summarization dataset," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pp. 1967–1972, Lisbon, Portugal, 2015.

[8] H. Zhao and C. Kit, "Integrating unsupervised and supervised word segmentation: the role of goodness measures," *Information Sciences*, vol. 181, no. 1, pp. 163–183, 2011.

[9] D. Cai, H. Zhao, Z. Zhang, Y. Xin, Y. Wu, and F. Huang, "Fast and accurate neural word segmentation for Chinese," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pp. 608–615, Vancouver, Canada, 2017.

[10] J. Lin, X. Sun, S. Ma, and Q. Su, "Global encoding for abstractive summarization," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pp. 163–169, Melbourne, Australia, 2018.

[11] S. Ma, X. Sun, J. Lin, and H. Wang, "Autoencoder as assistant supervisor: improving text representation for Chinese social media text summarization," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pp. 725–731, Melbourne, Australia, 2018.

[12] X. Duan, H. Yu, M. Yin, M. Zhang, W. Luo, and Y. Zhang, "Contrastive attention mechanism for abstractive sentence summarization," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pp. 3042–3051, Hong Kong, China, 2019.

[13] J. Gu, Z. Lu, H. Li, and V. O. K. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*, Berlin, Germany, 2016.

[14] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *4th International Conference on Learning Representations, ICLR 2016*, San Juan, Puerto Rico, 2016.

[15] Q. Zhou, N. Yang, F. Wei, and M. Zhou, "Sequential copying networks," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, pp. 4987–4995, New Orleans, Louisiana, USA, 2018.

[16] J. Su, Z. Tan, D. Xiong, R. Ji, X. Shi, and Y. Liu, "Lattice-based recurrent neural network encoders for neural machine translation," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 3302–3308, San Francisco, California, USA, 2017.

[17] F. Xiao, J. Li, H. Zhao, R. Wang, and K. Chen, "Latticebased transformer encoder for neural machine translation," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pp. 3090–3097, Florence, Italy, 2019.

[18] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pp. 5998–6008, Long Beach, CA, USA, 2017.

[19] S. Gehrmann, Y. Deng, and A. M. Rush, "Bottom-up abstractive summarization," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4098–4109, Brussels, Belgium, 2018.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirectional transformers for language understanding," 2018, https://arxiv.org/abs/1810.04805.

[21] S. Chopra, M. Auli, and A. M. Rush, "Abstractive sentence summarization with attentive recurrent neural networks," in *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 93–98, San Diego California, USA, 2016.

[22] R. Nallapati, B. Zhou, C. dos Santos, Ç. Gulçehre, and B. Xiang, "Abstractive text summarization using sequenceto-sequence RNNs and beyond," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 280–290, Berlin, Germany, 2016.

[23] P. Li, W. Lam, L. Bing, and Z. Wang, "Deep recurrent generative decoder for abstractive text summarization," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*, pp. 2091–2100, Copenhagen, Denmark, 2017.

[24] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. Dauphin, "Convolutional sequence to sequence learning," in *Proceedings of the 34th International Conference on Machine Learning*, pp. 1243–1252, Sydney NSW, Australia, 2017.

[25] A. Çelikyilmaz, A. Bosselut, X. He, and Y. Choi, "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, NAACL-HLT 2018*, pp. 1662–1675, New Orleans, Louisiana, USA, 2018.

[26] A. Cohan, F. Dernoncourt, D. S. Kim et al., "A discourse-aware attention model for abstractive summarization of long documents," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pp. 615–621, New Orleans, Louisiana, USA, 2018.

[27] K. Wang, X. Quan, and R. Wang, "Biset: bi-directional selective encoding with template for abstractive summarization," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pp. 2153–2162, Florence, Italy, 2019.

[28] Z. Cao, W. Li, S. Li, and F. Wei, "Retrieve, rerank and rewrite: soft template based neural summarization," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pp. 152–161, Melbourne, Australia, 2018.

[29] K. Song, L. Zhao, and F. Liu, "Structure-infused copy mechanisms for abstractive summarization," in *Proceedings of the 27th International Conference on Computational Linguistics,*

COLING 2018, pp. 1717–1729, Santa Fe, New Mexico, USA, 2018.

[30] M. Talha, M. Sohail, and H. Hajji, "Analysis of research on amazon AWS cloud computing seller data security," *International Journal of Research in Engineering Innovation*, vol. 4, no. 3, pp. 131–136, 2020.

[31] M. Talha, M. Sohail, R. Tariq, and M. T. Ahmad, "Impact of oil prices, energy consumption and economic growth on the inflation rate in Malaysia," *Cuadernos de Economía*, vol. 44, no. 124, pp. 26–32, 2021.

[32] Q. Wang and J. Ren, "Abstractive summarization with keyword and generated word attention," in *International Joint Conference on Neural Networks, IJCNN 2019 Budapest*, pp. 1–8, Hungary, 2019.

[33] Z. Deng, F. Ma, R. Lan, W. Huang, and X. Luo, "A two-stage Chinese text summarization algorithm using keyword information and adversarial learning," *Neurocomputing*, vol. 425, pp. 117–126, 2021.

[34] P. X. Nguyen and S. R. Joty, "Phrase-based attentions," 2018, https://arxiv.org/abs/1810.03444.

[35] J. Hao, X. Wang, S. Shi, J. Zhang, and Z. Tu, "Multi-granularity self-attention for neural machine translation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pp. 887–897, Hong Kong, China, 2019.

[36] T. Srinivasan, R. Sanabria, and F. Metze, "Multitask learning for different subword segmentations in neural machine translation," 2019, https://arxiv.org/abs/1910.12368.

[37] F. J. Och and H. Ney, "The alignment template approach to statistical machine translation," *Computational Linguistics*, vol. 30, no. 4, pp. 417–449, 2004.

[38] C.-Y. Lin and E. Hovy, "Automatic evaluation of summaries using *n*-gram co-occurrence statistics," in *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 150–157, Edmonton, Canada, 2003.

[39] R. Luo, J. Xu, Y. Zhang, X. Ren, and X. Sun, "Pkuseg: a toolkit for multi-domain Chinese word segmentation," 2019, https://arxiv.org/abs/1906.11455.

[40] B. Wan, Z. Tang, and L. Yang, "Lexicon-constrained copying network for Chinese abstractive summarization," 2020, https://arxiv.org/abs/2010.08197.