A Shortcut-Stacked Document Encoder for Extractive Text Summarization

Peng Yan[1,2], Linjing Li[1], Daniel Zeng[1,2]
[1] The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
{yanpeng2017, linjing.li, dajun.zeng}@ia.ac.cn

***Abstract*—While doing summarization, human needs to understand the whole document, rather than separately understanding each sentence in the document. However, inter-sentence features within one document are not adequately modeled by previous neural network-based models that almost use only one layer recurrent neural network as document encoder. To learn high quality context-aware representation, we propose a shortcut-stacked document encoder for extractive summarization. We use multiple stacked bidirectional long short-term memory (LSTM) layers and add shortcut connections between LSTM layers to increase representation capacity. The shortcut-stacked document encoder is built on a temporal convolutional neural network-based sentence encoder to capture the hierarchical structure of the document. Then sentence representations encoded by document encoder are fed to a sentence selection classifier for summary extraction. Experiments on the well-known CNN/Daily Mail dataset show that the proposed model outperforms several recently proposed strong baselines, including both extractive and abstractive neural network-based models. Furthermore, the ablation analysis and position analysis also demonstrate the effectiveness of the proposed shortcut-stacked document encoder.**

I. Introduction

Text summarization aims to condense a document as a shorter version through automatically distilling the most important information from it, which is still a highly challenging task for machines and one of the most important technologies in natural language processing. Text summarization has been applied to almost all domains of the Internet, as the problem of information overload has been becoming more and more serious in the Internet era. Search engines provide summary snippets for better browse experience. Social media use automatic summarization as an intermediate technology for content recommendation, while e-commerce websites use auto-generated summaries for product highlights [1].

The summarization approaches can be mainly divided into two categories: extractive and abstractive. Extractive approaches summarize a document by selecting and assembling the most important parts of it, while abstractive methods generate novel words or phrases that may be not included in the source text. Fig. 1 shows an example of extractive summarization in the CNN/Daily Mail dataset. Early summarization approaches are heavily based on human-engineered features, such as word frequency [2], sentence position and length [3]. Recently, with the development of sequence-to-sequence learning in many text generation tasks, such as machine translation [4] and question answering [5], sequence-to-sequence models are also applied to summarization tasks, both for abstractive methods [6], [7] and extractive methods [8], [9]. Neural network-based summarization models can learn in an end-to-end manner, thus avoiding sophisticated feature engineering in early methods. Most of neural extractive approaches follow the encoder-extractor framework. The encoder reads the meaning of a document and outputs a list of continuous-space representations corresponding to each sentence within the document. Then the extractor selects the salient sentences based on these sentence representations.

However, one important issue is that summarization needs the comprehensive understanding of the whole document, while the neural network-based model is good at processing information at single-sentence level. And in previous works [6], [8], the most common architecture of the document encoder for neural text summarization is only one layer of variants

Article: (1)The women of the University of southern California tennis team capped off an undefeated conference season on Thursday by winning the pac-12 championship. (2)The second-ranked girls defeated the women of the university of California - Los Angeles by a score of 4 - 3 for the win. (3)Then, in celebrating their big victory, they broke the trophy. (4)The USC women's tennis team won the pac-12 championship on Thursday. (5) No doubt contributing to the excitement was the fact that USC was trailing UCLA early on, and were initially down 3 - 0 before winning the final four matches. (6)USC now finishes the season 21 - 2 while UCLA is 18 - 4. (7)Both teams will now compete in the pac-12 championships next Thursday, where the players compete individually. (8)While celebrating their win over UCLA, they smashed and broke the trophy.
Human writing summary: The USC women's tennis team won the pac-12 championship on Thursday. The girls defeated the women of UCLA 4 - 3. While celebrating, they smashed and broke the trophy.
Extractive summary: (4) (2) (8)
(4)The USC women's tennis team won the pac-12 championship on Thursday. (2)The second-ranked girls defeated the women of the university of California - Los Angeles by a score of 4 - 3 for the win. (8)While celebrating their win over UCLA, they smashed and broke the trophy.

Fig. 1: An example of extractive summarization in the CNN/Daily Mail dataset. In the example, extractive summary is generated by selecting three sentences from the source article and contains most of the content in the human writing summary (underlined part).

of recurrent neural networks (RNN), such as long short-term memory (LSTM) [10] and gated recurrent unit (GRU) [11]. This document encoder with simple structure is relatively hard to capture abundant inter-sentence information and fails to produce comprehensive document-level representations.

In this paper, we propose a shortcut-stacked document encoder to obtain better document understanding and representation. Our proposed document encoder has multiple bidirectional LSTM layers with shortcut connections (feeding all previous layers outputs to each layer), thus providing extra source of information to guide the summary extraction. We build our shortcut-stacked document encoder on a temporal convolutional neural network (CNN) based sentence encoder, and feed document encoder's outputs to summary extractor, which is a sentence selection classifier. The contributions of this paper are two folds:

- First, we go beyond the conventional encoder-extractor framework for neural extractive summarization to present a novel model based on shortcut-stacked document encoder, which is, to the best of our knowledge, first used for the extractive summarization task.
- Second, our experiments on the CNN/Daily Mail dataset reveal that our model outperforms both extractive and abstractive baseline models. Also, we investigate the effectiveness of the shortcut-stacked document encoder by the ablation analysis and position analysis.

II. Related Work

A. Text Summarization Task

Early text summarization researches mostly focus on extractive approaches, including greedy approaches [12], hidden Markov models [13], graph based approaches [14] and integer linear programming [15]. Early methods mostly use human-engineered features, because most summarization datasets available such as DUC corpora are not large enough to train deep neural networks. This situation has not been changed until [16] proposes a new corpus based on news stories from CNN and Daily Mail and human generated summaries. This large-sized corpus attracts neural network-based approaches for abstractive summarization at first. These neural abstractive models frame text summarization as a sequence-to-sequence problem [7], [16]. To produce fluent abstractive summaries, some improvement mechanisms are added to the original sequence-to-sequence models, including hierarchical attention [16], graph-based attention [17], the copy mechanism [7], [18], coverage [7], [19], and reinforcement learning-based metric optimization [20].

With the development of neural abstractive models, neural network methods have also been used for extractive summarization [9]. Neural extractive approaches conceptualize extractive summarization as a sequence labeling task, which needs sentence-level extraction labels, but most summarization datasets only include document and summary pairs. In order to overcome the labels absence obstacle to extractive summarization, [9] creates a rule-based method by maximizing the ROUGE score [21] to obtain sentence extraction labels. Then the research of neural extractive summarization has become more and more active. And there are many latest works on neural summarization, for example, [22] improves the memorization capability of summarization model by adding a closed-book decoder. And a neural summarization model is proposed by [23] that enables users to specify a desired length, style or entities that they have a preference in order to control the shape of the final summaries. And in [24], researchers propose a novel summarization model trained by reinforcement learning with multi-reward functions to improve the saliency and directed logical entailment aspects of a good summary.

The most similar work to our model is the one proposed by [8]. They use a hierarchical recurrent neural network-based encoder and a binary classifier for extractive summarization on the CNN/Daily Mail corpus. To capture the hierarchical structure of the documents, they use two recurrent neural network networks to encode text content at sentence level and document level respectively. Different from their method, our model uses a temporal convolutional neural network as the sentence encoder for ease of optimization. And it is worth mentioning that they do not use stacked RNN structure, while we use our novel multiple stacked bidirectional LSTM layers with shortcut connections as the document encoder. And our extraction classifier is a simplification version of theirs without using any human-engineered features, such as absolute and relative position features.

B. Shortcut Structure

The shortcut structure that we use in this work is widely applied to many neural network models for different tasks, such as residual CNN for computer vision [25], highway networks for RNN in speech processing [26], and shortcut connections in hierarchical multitasking learning [27]. A similar shortcut-stacked RNN model is also used for natural language inference tasks [28]. In their work, shortcut-stacked RNN is used as sentence-level encoder that encodes each word within one sentence. Different from their model, our shortcut-stacked bidirectional LSTM layers serve as a document-level encoder that encodes each sentence within one document.

III. Model

A. Problem Statement

A document d can be considered as a sequence of n sentences $(s_1, s_2, \cdots, s_n)$, extractive summarization generates a summary by selecting m sentences $(m < n)$ from d. In this paper, we treat extractive summarization as a sequence of binary classification problems. For a sentence s_j in document d, we predict a summary label $y_j \in \{0, 1\}$, $y_j = 1$ if s_j belongs to the final summary, otherwise $y_j = 0$. Let D be the

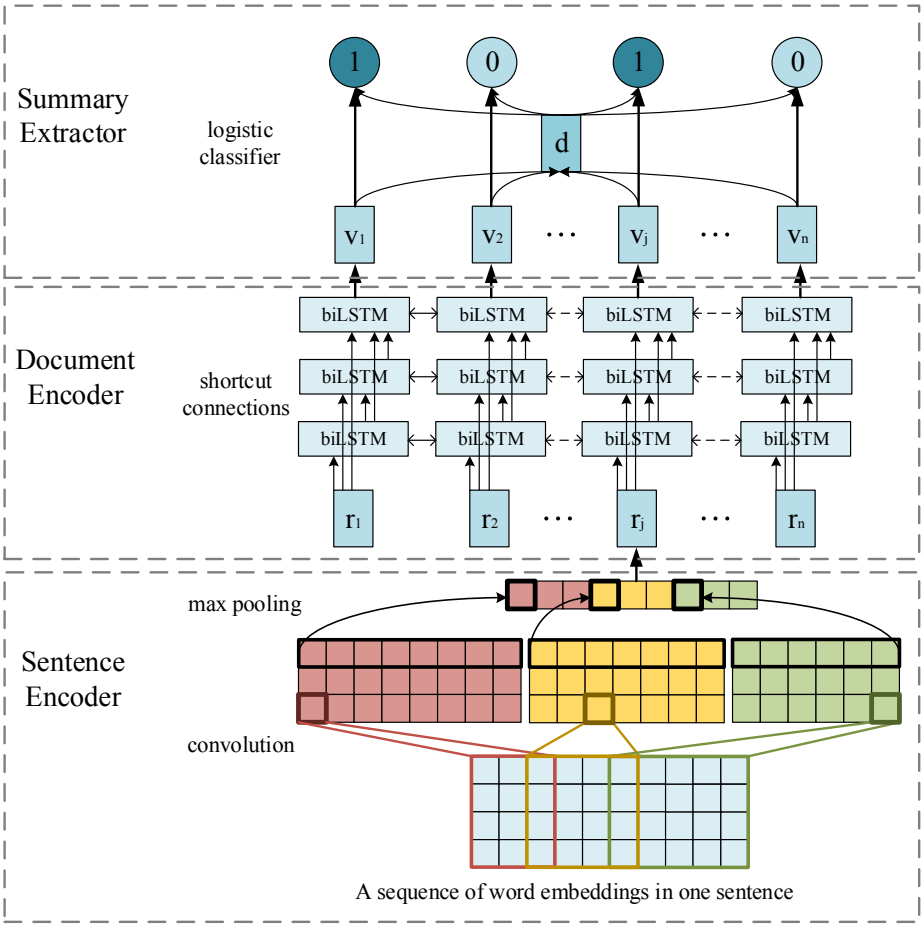

Fig. 2: The overall architecture of the proposed model. The convolutional encoder computes representation r_j for each sentence (Bottom). The shortcut-stacked document encoder computes context-aware sentence representation v_j (Middle). The summary extractor determines the sentence-summary membership, and the final summary is consisted of sentences with label of 1 in their original order in the document (Top).

set of all training documents, the objective is to minimize the cross-entropy loss of all observed labels in the training phase

$$L(\Theta) = \sum_{i=1}^{N} \sum_{j=1}^{n_i} \left(y_j^i \log \left(Prob\left(y_j^i = 1 | D; \Theta \right) \right) + \left(1 - y_j^i \right) \log \left(1 - Prob\left(y_j^i = 1 | D; \Theta \right) \right) \right), \tag{1}$$

where Θ is the parameter vector, N is the number of documents in the training set, n_i is the number of sentences in the i^{th} document, and y_j^i is the binary summary label for the j^{th} sentence in the i^{th} document. $Prob\left(\right)$ represents probability and is implemented as sigmoid function's output. The specific calculation of $Prob\left(y_j^i = 1 | D; \Theta \right)$ is described in details in section III.E.

B. Overall Architecture

Our model follows the encoder-extractor framework as shown in Fig. 2. The role of the encoder is to capture the meaning representation of a document. The whole encoder works in a hierarchical fashion, which reflects the hierarchical structure of documents, i.e., a document consists of a sequence of sentences, while each sentence is composed of a sequence of words. We first get representation vectors at the sentence-level using sentence encoder modeled by a temporal convolutional neural network. Next, we obtain representation vectors of sentences at the document-level using a novel shortcut-stacked document encoder that recursively composes sentences.

Finally, the encoder outputs each sentence's context-aware representation and feeds them into the summary extractor, which is a logistic binary classifier. The summary extractor first learns a document representation, then determines whether a sentence belongs to the final summary or not based on the content of the sentence and the salience with respect to the document. Each module will be explained in detail in the following sections.

C. Sentence Encoder

We use a temporal convolutional neural network to encode and represent each sentence of a document. A similar architecture is proved to be effective in sentence-level classification

tasks such as sentiment analysis [29]. Given a sentence s consisting of a sequence of p words $(w_1, w_2, \cdots, w_p)$, each word is converted to an e-dimensional vector by a pre-trained word embedding matrix W_{emb} learned by word2vec [30]. Thus, a sentence can be represented by a dense matrix $W_{sen} \in R^{p \times e}$. Then the matrix W_{sen} will be fed through multiple one-dimension convolution filters ($K \in R^{e \times c}$) with different window size c and then ReLU activation units

$$f_{i,K}^h = relu(W_{h:h+c+1} \otimes K + b), \tag{2}$$

where $\otimes$ is the Hadamard Product followed by a sum over all elements, $f_{i,K}^h$ denotes the h^{th} element of the i^{th} feature from filter K and b is the bias. We perform max-over-time pooling to capture the temporal dependencies on nearby words and obtain the i^{th} feature of sentence representation $F_{i,K}$ from filter K

$$F_{i,K} = \max_h f_{i,K}^h. \tag{3}$$

Finally, the outputs $F_{i,K}$ from all filters with different window sizes are concatenated together to form the representation r_j for the j^{th} sentence in the document. An example of sentence encoder is illustrated in Fig. 2 (bottom). In the example, a sentence of ten words with 5-dimension word embedding is fed to three filters with different window sizes [3, 4, 5]. The sentence representation has nine dimensions in total, obtained by concatenating features from three filters, each of which has three dimensions. In actual implementation, we use a list of filter window sizes [3, 4, 5] with 100 hidden units for each window size.

D. Document Encoder

After getting each sentence representation r_j , we need a document encoder to encode inter-sentence features at the document level, which are important for text summarization. To increase the representation capacity, we use multiple stacked bidirectional long short-term memory (biLSTM) layers with shortcut connections as the document encoder. Let biLSTM^i represents the i^{th} biLSTM layer among multiple biLSTM layers, which is defined as

$$h_t^i = \text{biLSTM}^i(x_t^i, t), \forall t \in [1, 2, \ldots, n], \tag{4}$$

where h_t^i is the output of the i^{th} biLSTM at time t over input $(x_1^i, x_2^i, \cdots, x_n^i)$. As for typical multiple stacked biLSTM layers, the output sequence of the previous biLSTM layer is the input of the next biLSTM layer. However, different from the typical structure, our shortcut-stacked biLSTM layers are added shortcut connections that feed each layer's output to all next layers, as shown in the Fig. 2 (middle). The shortcut connections can alleviate the gradient vanishing problem during training and enable more document information to flow across different layers to get better context-aware representations. Thus the input of each biLSTM layer in our shortcut-stacked document encoder is the concatenation of all the previous layers' outputs and the original sentence vector sequence. Let $(r_1, r_2, \cdots, r_n)$ represent sentence representations obtained by the sentence encoder, the input of the i^{th} biLSTM layer at time t is

$$x_t^1 = r_t, \tag{5}$$

$$x_t^i = [r_t, h_t^{i-1}, x_t^{i-2}, \cdots, h_t^1](i > 1), \tag{6}$$

where [] represents vector concatenation. From the output of the last layer, we can get the final document-level context-aware sentence vectors $(v_1, v_2, \cdots, v_n)$ for the given document: $d = (s_1, s_2, \cdots, s_n)$. In actual implementation, we experiment with 3 biLSTM layers with 512, 1024, 2048 dimensions each.

E. Summary Extractor

We formulate summary extraction as a sequence of binary classification problems on the context-aware sentence vectors $(v_1, v_2, \cdots, v_n)$. Our summary extractor is a logistic binary classifier that determines whether a sentence belongs to the summary or not. We first learn a document representation $\hat{d}$ by

$$\hat{d} = \tanh(W_d \frac{1}{n} \sum_{j=1}^{n} v_j + b_d), \tag{7}$$

where n denotes the number of sentences in the given document d. W_d and b_d are trainable parameters. Then we compute the probability of the j^{th} sentence belonging to the summary by

$$Prob(y_j = 1 | v_j, \hat{d}) = \sigma(W_s v_j + v_j^\top \hat{d} + b_s)), \tag{8}$$

where W_s and b_s are trainable parameters. $W_s v_j$ reflects the information content of the j^{th} sentence in the document, while $v_j^\top \hat{d}$ captures the salience of the sentence with respect to the document. In the test phase, the model computes the probability of summary membership for each sentence sequentially, then picks the top-3 sentences and concatenates them as the original order in the document.

IV. Experimental Setup

A. Datasets

The proposed model is evaluated on the well-known CNN/Daily Mail dataset, which is originally built for question answering task [5] and modified for text summarization by [16]. Compared to previous Gigaword dataset [6], CNN/Daily Mail dataset has more long documents and summaries. Note that we use a non-anonymized, original-text version of this dataset [7] and the standard split of the dataset for training and test. Table I shows the statistics of the CNN/Daily Mail dataset.

TABLE I: The statistics of the CNN/Daily Mail dataset

	Training	Dev	Test
#(documents)	287,227	13,368	11,490
#(sentences) per document	31.58	26.72	27.05
#(words) per document	791.36	769.26	778.24
#(sentences) per summary	3.79	4.11	3.88
#(words) per summary	55.17	61.43	58.31

However, the CNN/Daily Mail dataset only contains document-summary pairs without extraction labels for each sentence. Hence, we design a rule-based method to generate a proxy label for sentences in a given document. Similar to the method in [8], our rule-based method maximizes the Rouge-L F1 score [21] via a greedy strategy. To be specific, we find the most similar sentence s_{j_t} for each ground-truth summary sentence g_t and filter it through a threshold δ to generate the label of j^{th} sentence y_j

$$j_t = \arg\max_j(\text{Rouge-L}_{F1}(s_j, g_t)), \tag{9}$$

$$y_{j_t} = \begin{cases} 1 & if(\text{Rouge-L}_{F1}(s_{j_t}, g_t) > \delta) \\ 0 & otherwise \end{cases}. \tag{10}$$

In our experiment, δ is set to 0.2. Given these proxy training labels, the CNN/Daily Mail dataset can be used for extractive summarization.

B. Baselines

We compare our approach with the following baselines:

- **LEAD3**: The widely used baseline by extracting the first three sentences as the summary.
- **TextRank**: An unsupervised method based on weighted graph proposed by [31]. We use the implementation in Gensim [32].
- **Pointer**: A state-of-the-art abstractive summarization method proposed by [7], which applies copying and coverage mechanisms.
- **Controllable**: A neural summarization model that enables users to specify a desired length, style or entities that they have a preference in order to control the shape of the final summaries [23].
- **SummaRuNNer**: An extractive summarization method proposed by [8], which considers both text features and sentence absolute and relative position features.
- **REFRESH**: A state-of-the-art reinforced extractive summarization model proposed by [33].
- **ROUGESal+Ent RL**: A novel reinforcement learning model trained with multi-reward functions to improve the saliency and directed logical entailment aspects of a good summary [24].

C. Hyper-parameter Settings

We train a word2vec [30] of 64 dimensions on the CNN/Daily Mail dataset as the word embedding initialization. The word embedding is still updated during training. The vocabulary size is set to 30000. For the convolutional sentence encoder, we use a list of filter window sizes [3, 4, 5] with 100 hidden units for each window size. For the document encoder, we experiment with 3 biLSTM layers with 512, 1024, 2048 dimensions each. All biLSTM parameters are randomly initialized over a uniform distribution within [-0.05, 0.05]. In the training phase, the batch size is 32. Adam optimizer [34] is used with learning rate 0.0001. We apply gradient clipping [35] using 2-norm of 2.0. We use early-stopping regularization technique (with patience equals 8 and number of steps for checkpoint and validation equals 1000) and halve the learning rate whenever validation loss increases. To save the memory space during training, we set the maximum sentence length to 100 words and the maximum number of sentences per document to 60. All hyper-parameters are tuned on the validation set of CNN/Daily Mail dataset.

D. Evaluation

For the automatic evaluation, we use the ROUGE metrics [21], which is based on the comparison of n-grams between the generated summary and the human written reference. We evaluate generated summaries by standard Rouge-1, Rouge-2, and Rouge-L on full length F1.

V. Results and Analysis

A. Overall Performance

As shown in Table II , our model achieves 37.15 Rouge-L F1 score on the CNN/Daily Mail dataset and outperforms two unsupervised baselines (LEAD3 and TextRank [31]) by a wide margin. And our model achieves statistically significant improvements over all the abstractive baseline models (Pointer [7], Controllable [23]), as given by the 95% confidence interval in the official ROUGE script. Also, our proposed model outperforms previously published neural extractive models (SummaRuNNer [8], REFRESH [33], and ROUGESal+Ent RL [24]) in terms of Rouge-1 and Rouge-L F1 scores. However, SummaRuNNer is trained and tested on the anonymized version of CNN/Daily Mail dataset, so the result of SummaRuNNer is not strictly comparable to our proposed model's result on the non-anonymized version dataset. We thus include result of LEAD3* on the anonymized version dataset as a reference in Table II . Our model exceeds LEAD3 by +0.67 Rouge-L F1 points, while SummaRuNNer is worse than LEAD3* by -0.20 points in terms of Rouge-L. Therefore, we can conclude that our model without any manually selected features, is better than SummaRuNNer

TABLE II: Results of different models on the CNN/Daily Mail Test set using Full length ROUGE F1 evaluation (%). Results with * mark are trained and evaluated on the anonymized dataset, others are on the non-anonymized version dataset. All our ROUGE scores have a 95% confidence interval of at most 0.20 as reported by the official ROUGE script.

Models	Rouge-1 F1	Rouge-2 F1	Rouge-L F1
Unsupervised models			
LEAD3	40.27	17.72	36.48
LEAD3*	39.2	15.7	35.5
TextRank	40.20	17.56	36.44
Abstractive models			
Controllable	38.68	15.40	35.47
Pointer	39.53	17.28	36.38
Extractive models			
SummaRuNNer*	39.6	16.2	35.3
REFRESH	40.0	**18.2**	36.6
ROUGESal+Ent RL	40.43	18.00	37.10
Our Model	**40.64**	18.12	**37.15**

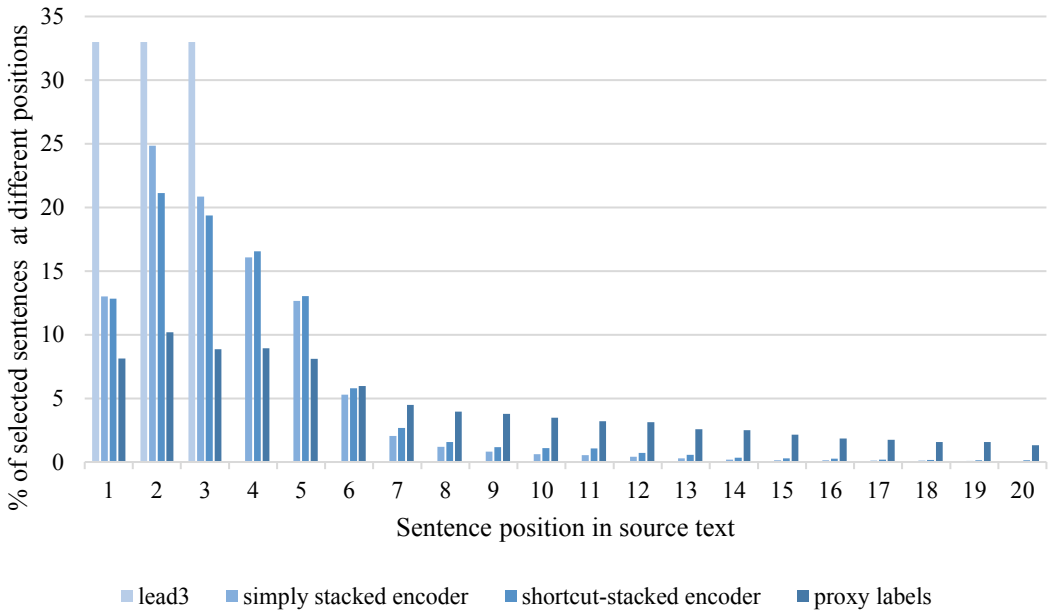

Fig. 3: Selected sentences' position distribution of four methods (including lead3 baseline, the baseline with simply-stacked encoder, our model with shortcut-stacked encoder and proxy labels) on the CNN/Daily Mail Test set. The Y-axis means the proportion of selected sentences in the generated summaries at position i (X-axis) in source text.

baseline that uses human-engineered features, such as absolute and relative position features. And compared to the state-of-the-art extractive model ROUGESal+Ent RL [24], our model performs significantly better without using any reinforcement learning optimization method.

As for computation time, the whole training phase terminated by the early-stopping regularization technique is approximately 2.5 hours on one TITAN X GPU. In testing phase, generating all the 11490 summaries for the CNN/Daily Mail Test set requires only 570.5 seconds. Around 20 summaries per second can be generated by our extractive model, which can fully meet real-time requirement in practical application.

B. Ablation Analysis

We now investigate the effectiveness of the shortcut-stacked document encoder in our overall model. In the ablation analysis, we show the performance changes for different number of biLSTM layers and with/without shortcut connections. We first show the ablation results based on the CNN/Daily Mail Development set in Table III. Note that the dimension size of a biLSTM layer is referring to the dimension of the hidden state for both the forward and backward LSTM. As shown in Table III, the model based on the shortcut-stacked three-layer document encoder achieves the best performance on CNN/Daily Mail Development set.

Then we show results of different document encoders on CNN/Daily Mail Test set in Table IV. As shown, compared to the model with single-layer biLSTM document encoder, the model with the simply stacked three-layer document encoder improves ROUGE scores very little(+0.06 for Rouge-1 F1, +0.03 for Rouge-2 F1, +0.06 for Rouge-L F1). However, compared to the single-layer biLSTM document encoder, our model with shortcut-stacked three-layer document encoder achieves a substantial improvement on ROUGE scores (+0.27 for Rouge-1 F1, +0.25 for Rouge-2 F1, +0.31 for Rouge-L F1), which is around 5-10 times than simply stacked three-layer encoder's improvement.

TABLE III: Results of the ablation analysis on the CNN/Daily Mail Development set. The first two models are two baselines, and the last model is our proposed model.

Models			Performance		
#layers	LSTM dims	shortcut	R-1	R-2	R-L
1	1024	No	41.25	18.43	37.76
3	256 512 1024	No	41.41	18.52	37.87
3	256 512 1024	Yes	**41.62**	**18.77**	**38.15**

C. Position Analysis

Fig. 3 shows the selected sentences' position distributions of four methods (including lead3 baseline, the baseline with simply stacked encoder, our model with shortcut-stacked en-

TABLE IV: Results of the ablation analysis on the CNN/Daily Mail Test set. The first two models are two baselines, and the last one is our proposed model.

Models			Performance		
#layers	LSTM dims	shortcut	R-1	R-2	R-L
1	1024	No	40.37	17.87	36.84
3	256 512 1024	No	40.43	17.90	36.90
3	256 512 1024	Yes	**40.64**	**18.12**	**37.15**

TABLE V: Network configuration of simply stacked encoder and shortcut-stacked encoder in the position analysis.

model	#layers	LSTM dims	shortcut
simply stacked encoder	3	256 512 1024	No
shortcut-stacked encoder	3	256 512 1024	Yes

coder and proxy labels) on the CNN/Daily Mail Test set. The proxy labels is generated by rule-based method as described in section IV.A and network configuration of simply stacked encoder and shortcut-stacked encoder is described in Table V.

As shown in Fig. 3, selected sentences' position distribution of our model with shortcut-stacked encoder is much closer to proxy labels than simply stacked encoder baseline and the lead3 baseline. The simple lead3 baseline selects 100% leading three sentences (sentence 1 to 3) due to its definition, and according to the statistics, about 58.7% sentences selected by simply stacked encoder baseline are in leading three sentences. Compared to the simply stacked encoder and lead3 baseline, our model with shortcut-stacked encoder selects less leading sentences (about 53.3% leading three sentences), which is closer to the proxy labels with no more than 30% leading three sentences. Also, our model selects more tailing sentences than simply stacked encoder baseline and lead3 baseline. For example, in the range of sentence 11 to 20, lead3 baseline does not select any sentence and simply stacked encoder baseline barely extracts any sentences (about 2.09% in total), but our model extracts nearly twice as many sentences (about 3.89% in total) in this range. Therefore, as for the selected sentences' position distribution, our model with shortcut-stacked encoder is much closer to proxy labels than simply stacked encoder baseline, demonstrating the effectiveness of our shortcut-stacked document encoder.

VI. Conclusion

In this paper, we go beyond the encoder-extractor framework to propose a novel shortcut-stacked document encoder in a neural network architecture for extractive summarization. The shortcut connections between biLSTM layers can help document encoder capture better inter-sentence features and produce context-aware representations, thus providing extra source of information to guide the summary extraction. ROUGE evaluation results show that our proposed model surpasses previous state-of-the-art models on the CNN/Daily Mail dataset. In future work, we will continue to evaluate the effectiveness of our shortcut-stacked document encoder on other document-level NLP tasks.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China under Grant 2016QY02D0305 and 2017YFC0820105, the National Natural Science Foundation of China under Grants 71702181, 71621002, as well as the Key Research Program of the Chinese Academy of Sciences under Grant ZDRW-XH-2017-3.

References

[1] C. Khatri, G. Singh, and N. Parikh, "Abstractive and extractive text summarization using document context vector and recurrent neural networks," *CoRR*, vol. abs/1807.08000, 2018.

[2] A. Nenkova, L. Vanderwende, and K. McKeown, "A compositional context sensitive multi-document summarizer: Exploring the factors that influence summarization," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '06. New York, NY, USA: ACM, 2006, pp. 573–580.

[3] D. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Çelebi, S. Dimitrov, E. Drabek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, and Z. Zhang, "Mead - a platform for multidocument multilingual text summarization," in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*. European Language Resources Association (ELRA), 2004.

[4] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 3104–3112.

[5] K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 1693–1701.

[6] 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*. Association for Computational Linguistics, 2015, pp. 379–389.

[7] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1073–1083.

[8] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *AAAI*, 2017.

[9] X. Zhang, M. Lapata, F. Wei, and M. Zhou, "Neural latent extractive document summarization," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 779–784.

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

[11] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1724–1734.

[12] J. G. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Research and Development in Information Retrieval*, 1998, pp. 335–336. [Online]. Available: citeseer.ist.psu.edu/carbonell98use.html

[13] J. M. Conroy and D. P. O'Leary, "Text summarization via hidden markov models," in *SIGIR*, 2001.

[14] G. Erkan and D. R. Radev, "Lexrank: Graph-based lexical centrality as salience in text summarization," *J. Artif. Intell. Res.*, vol. 22, pp. 457–479, 2004.

[15] K. Woodsend and M. Lapata, "Automatic generation of story highlights," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 565–574.

[16] R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence rnns and beyond," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 2016, pp. 280–290.

[17] J. Tan, X. Wan, and J. Xiao, "Abstractive document summarization with a graph-based attentional neural model," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1171–1181.

[18] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 1631–1640.

[19] Q. Chen, X. Zhu, Z. Ling, S. Wei, and H. Jiang, "Distraction-based neural networks for modeling documents," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, pp. 2754–2760.

[20] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," in *International Conference on Learning Representations*, 2018.

[21] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, 2004.

[22] Y. Jiang and M. Bansal, "Closed-book training to improve summarization encoder memory," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 4067–4077.

[23] A. Fan, D. Grangier, and M. Auli, "Controllable abstractive summarization," in *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*. Association for Computational Linguistics, 2018, pp. 45–54.

[24] R. Pasunuru and M. Bansal, "Multi-reward reinforced summarization with saliency and entailment," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018, pp. 646–653.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[26] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, and J. Glass, "Highway long short-term memory rnns for distant speech recognition." IEEE - Institute of Electrical and Electronics Engineers, April 2016.

[27] K. Hashimoto, c. xiong, Y. Tsuruoka, and R. Socher, "A joint many-task model: Growing a neural network for multiple nlp tasks," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017, pp. 1923–1933.

[28] Y. Nie and M. Bansal, "Shortcut-stacked sentence encoders for multi-domain inference," in *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP*. Association for Computational Linguistics, 2017, pp. 41–45.

[29] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1746–1751.

[30] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. USA: Curran Associates Inc., 2013, pp. 3111–3119.

[31] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

[32] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[33] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking sentences for extractive summarization with reinforcement learning," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 1747–1759.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." *CoRR*, vol. abs/1412.6980, 2014.

[35] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13. JMLR.org, 2013, pp. 1310–1318.