A Sentence Compression Method Based on "Pre-reading" and Simple Attention Mechanism Post-print

Authors: Lu Zhonglei, Liu Wenfen, Zhou Yanfang, Hu Xuexian, Wang Binyu

Date: 2018-05-20T00:00:00+00:00

Abstract

This research investigates English sentence compression methods and proposes a compression approach based on "pre-reading" and a simple attention mechanism. Within the encoder-decoder framework and built upon the Gated Recurrent Unit (GRU) neural network model, the encoding phase performs two-stage semantic modeling of the source sentence. The first-stage modeling results serve as global information to enhance the second-stage semantic modeling, yielding more comprehensive and accurate semantic encoding vectors. The decoding stage fully accounts for the specific characteristics of deletion-based sentence compression, employing a simple attention (3t-Attention) mechanism that feeds the semantic components in the encoding vector most relevant to the current decoding timestep into the decoder, thereby enhancing both prediction efficiency and accuracy. Experimental results on the Google News sentence compression dataset demonstrate that the proposed compression method surpasses existing publicly reported results. Consequently, the "pre-reading" and simple attention mechanism can effectively improve the precision of English sentence compression.

Full Text

Preamble

Title: An Effective Sentence Compression Method Based on Pre-Reading and Simple Attention Mechanism

Authors: Lu Zhonglei[1], Liu Wenfen[2], Zhou Yanfang[1], Hu Xuexian[1], Wang Binyu[1]
[1]State Key Laboratory of Mathematical Engineering & Advanced Computing, Zhengzhou 450001, China
[2]Guangxi Key Laboratory of Cryptography & Information Security, School of

Computer Science & Information Security, Guilin University of Electronic Technology, Guilin Guangxi 541004, China

Abstract: This paper proposes a method for English sentence compression based on pre-reading and simple attention mechanism. Within the encoder-decoder framework and using Gated Recurrent Unit (GRU) neural networks as the foundation, the original sentence semantics are modeled twice during the encoding stage. The first modeling result serves as global information to enhance the second semantic modeling, yielding a more comprehensive and accurate semantic encoding vector. During the decoding stage, we fully consider the particularity of deletion-based sentence compression and employ a simple 3t-Attention mechanism to input the most relevant semantic portion from the encoding vector to the decoder at each decoding timestep, thereby improving prediction efficiency and accuracy. Experimental results on the Google News sentence compression dataset demonstrate that our proposed compression method outperforms existing published results. Therefore, pre-reading and simple attention mechanism can effectively improve the accuracy of English sentence compression.

Keywords: natural language processing; sentence compression; pre-reading; attention mechanism

0 Introduction

With the rapid growth of online information, people seek to streamline information to save reading time. In recent years, the rapid development of natural language processing technology has enabled computers to participate in this task, with sentence compression being a crucial technique. Also known as sentence reduction, sentence compression aims to algorithmically simulate human text summarization and information extraction capabilities by removing redundant information while preserving core content to automatically generate concise sentences that are grammatically and semantically coherent. This technology is widely used in automatic topic extraction, automatic summarization, web information retrieval, public opinion monitoring, recommendation systems, question-answering systems, and sentiment analysis.

Traditional sentence compression methods obtain compressed sentences by minimizing grammatical error rates or pruning syntactic trees, relying heavily on manually designed rule-based features that require substantial expert knowledge and consume significant human and material resources. Deep learning, with its powerful representation capabilities, offers new technical approaches for sentence compression. Deep learning algorithms are entirely data-driven, automatically extracting features and substantially reducing human effort. Among various deep learning paradigms, sentence compression is a typical sequence prediction task where the original sentence sequence is input to predict the compressed sentence sequence. Such tasks are typically solved using the encoder-decoder

framework, where the encoder converts the input sentence into a dense vector containing semantic information, and the decoder generates keep-or-delete decisions for each word in the original sentence. Fillippova et al. adopted a similar strategy, first applying deep learning models to sentence compression, using a three-layer unidirectional Long Short-Term Memory (LSTM) network stack as encoder-decoder components to achieve results superior to traditional compression systems on large-scale datasets. Tran et al. improved upon Fillippova's model architecture, proposing a bidirectional LSTM model with attention mechanism for sentence compression that achieved good results on small datasets. Additionally, Sigrid et al. incorporated eye-tracking information into sentence compression systems through multi-task prediction to achieve higher accuracy, employing a three-layer unidirectional LSTM encoder-decoder architecture similar to Fillippova's work.

However, current research still suffers from two main limitations. First, most model designs are too simple to fully encode the original sentence semantics into the semantic vector (especially for longer input sequences), resulting in severe information loss and inaccurate decoding. Second, conventional attention mechanisms have high computational complexity and imprecise attention information representation, making them less suitable for sentence compression tasks.

To address these issues, this paper proposes a "pre-reading" mechanism that mimics human sentence compression behavior to achieve more accurate compression. Humans first read through the original sentence to grasp its overall meaning, at which point only partial words can be correctly decided for retention or deletion. They then read word-by-word again, using the global information obtained from the first pass to refine decisions and retain important words. The "pre-reading" mechanism operates similarly by feeding the original sentence sequence twice: the first pass obtains overall semantic information that locally enhances the second semantic representation, increasing semantic weights for words that need to be retained while reducing the influence of redundant words, thereby obtaining a more comprehensive and accurate semantic encoding vector that provides a solid foundation for decoding. Additionally, considering the strict alignment between input and output sequences in sentence compression tasks, we employ a more scientifically simple 3t-Attention mechanism that focuses on the hidden state at the corresponding timestep in the original sentence during each decoding timestep, while also considering the significant influence of immediate neighboring words on the keep-or-delete decision for the current word, thereby improving decoding efficiency and accuracy. The main contributions of this paper are as follows:

a) We pioneer the use of a "pre-reading" mechanism in sentence compression tasks to obtain more comprehensive and accurate semantic encoding vectors.

b) We propose a more scientifically simple 3t-Attention mechanism specifically for sentence compression tasks, reducing computational complexity

while improving decoding efficiency and accuracy.

c) We conduct extensive comparative experiments across multiple models, with results that provide guidance for model and hyperparameter selection.

1.1 GRU Model

Recurrent Neural Network (RNN) is an extension of conventional feedforward neural networks that allows connections within layers and directed cycles, enabling it to handle variable-length input sequences. RNNs have been widely applied in machine translation, automatic question answering, and syntactic parsing. However, in practical training, conventional RNNs suffer from gradient explosion and vanishing problems, performing poorly on long texts. Two improved models emerged: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Chung et al. demonstrated that LSTM and GRU perform comparably on sequence modeling tasks, both preserving important features through "gate" mechanisms to prevent them from being discarded during long-distance propagation. However, since GRU lacks a separate memory cell and has fewer parameters than LSTM, it achieves faster convergence and better iteration performance when data volume is large. Considering these factors, this paper selects GRU as the base model.

GRU addresses the long-term dependency problem in RNN training through update gates and reset gates that adaptively control information flow. Its structure is shown in Figure 1 [Figure 1: see original paper]. The specific working principle is:

$$
\begin{aligned}
z_t &= \sigma(W_{1x}t + U_{1h_{t-1}} + b_1) \\
r_t &= \sigma(W_{2x}t + U_{2h_{t-1}} + b_2) \\
\tilde{h}_t &= \tanh(W_{3x}t + U_3(r_t \odot h_{t-1}) + b_3) \\
h_t &= (1 - z_t)h_{t-1} + z_t\tilde{h}_t
\end{aligned}
$$

where z represents the update gate, r represents the reset gate, x is the input layer, h is the hidden layer, $\tilde{h}$ is the intermediate state corresponding to h. The reset gate r determines whether to discard previous states—when r approaches 0, the previous hidden state h_{t-1} is ignored and the intermediate state $\tilde{h}_t$ is reset to current input information. The update gate z determines whether to update the current hidden state to the new intermediate state $\tilde{h}_t$—when z approaches 1, the previous hidden state h_{t-1} is ignored and the current hidden state is set to the intermediate state $\tilde{h}_t$. The update and reset gates jointly determine the current hidden layer output. U, W, and b are model parameter matrices, and $\odot$ denotes element-wise multiplication.

1.2 Bidirectional GRU Model

Whether RNN, LSTM, or GRU, all only encode and utilize unidirectional semantic information. Furthermore, for timestep t, its hidden layer output contains only information before time t (i.e., preceding context). However, subsequent context is equally important for representing overall semantics. To better represent complete contextual information, building upon the successfully applied Bidirectional RNN (BiRNN) model, we propose using a Bidirectional GRU (BiGRU) model. This model can utilize all available historical and future input information during training to obtain more comprehensive and accurate semantic vectors.

Compared with GRU, BiGRU uses two separate hidden layers to read input bidirectionally: forward and backward. The forward direction reads input in original order (1 to T), while the backward direction reads in reverse order (T to 1). At timestep t, the two hidden layer states are:

$$\vec{h}_t = \mathrm{GRU}(x_t, \vec{h}_{t-1})$$
$$\overleftarrow{h}_t = \mathrm{GRU}(x_t, \overleftarrow{h}_{t+1})$$

BiGRU initializes its states to zero vectors: $\vec{h}_0 = 0$ and $\overleftarrow{h}_{T+1} = 0$. Based on the above equations, we can compute forward hidden states $(\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_T)$ and backward hidden states $(\overleftarrow{h}_1, \overleftarrow{h}_2, \ldots, \overleftarrow{h}_T)$, then combine them through concatenation:

$$h_t = [\vec{h}_t, \overleftarrow{h}_t]$$

Thus, the hidden state h_t simultaneously contains contextual information from both directions, effectively improving the model's memory performance on longer sequences.

1.3 Encoder-Decoder Framework Based on Bidirectional GRU

The encoder-decoder framework represents a new paradigm for natural language processing solutions and is widely used for sequence-to-sequence prediction problems such as machine translation and automatic summarization. Inspired by RNN encoder-decoders, the bidirectional GRU-based encoder-decoder framework consists of two parts: first, a bidirectional GRU model encodes the input sentence sequence to generate a fixed-length dense encoding vector containing

semantic information; second, a GRU model decodes this vector to predict labels for each word sequentially. Therefore, contextual representation (semantic encoding vector) is critical for effective sentence compression in the encoder-decoder framework.

Following Sutskever et al.'s sequence-to-sequence paradigm, we treat this problem by converting "input and output are both word sequences" to "input is a word sequence, output is a 0/1 sequence". The basic idea employs an end-to-end strategy to train the model that maximizes the probability of correct output for input sentences. Specifically, for each training sample (X, Y), we use Stochastic Gradient Descent (SGD) to solve the following optimization problem to learn model parameters θ^*:

$$\theta^* = \arg\max \sum \log p(Y|X;\theta)$$

The sum aggregates prediction losses across all training samples. Using the chain rule to model probability p, we obtain:

$$p(Y|X;\theta) = \prod p(Y_t|Y_1,\ldots,Y_{t-1},X;\theta)$$

No independence assumptions are made here. After obtaining optimal parameters θ^*, we can estimate compression results as:

$$\hat{Y} = \arg\max \log p(Y|X;\theta^*)$$

1.4 Attention-Based GRU Model

Bahdanau et al. proposed that when generating each word during decoding, attention mechanisms can dynamically utilize specific parts of the input sequence relevant to each decoding timestep, achieving source-target alignment and effectively improving prediction accuracy. Since then, attention mechanisms have been widely applied to learn alignments between various modalities, such as speech frame-text alignment in speech recognition and image-text alignment in image caption generation.

The working principle of attention-based GRU decoder is as follows:

$$\begin{aligned}
z_t &= \sigma(W_{1x}t + U_{1h_{t-1}} + V_{1c}t + b_1) \\
r_t &= \sigma(W_{2x}t + U_{2h_{t-1}} + V_{2c}t + b_2) \\
\tilde{h}_t &= \tanh(W_{3x}t + U_3(r_t \odot h_{t-1}) + V_{3c}t + b_3) \\
h_t &= (1 - z_t)h_{t-1} + z_t\tilde{h}_t
\end{aligned}$$

where c_t is the context representation based on attention, dynamically generated according to source-target alignment, and V is the weight matrix for context information. Bahdanau et al. use the weighted sum of all encoder-stage hidden states as the context representation at timestep t:

$$c_t = \sum \alpha_{tj} h_j^E$$

The weight α_{tj} is computed as:

$$r_{tj} = v_\alpha^T \tanh(W_\alpha h_{t-1} + U_\alpha h_j^E)$$
$$\alpha_{tj} = \text{softmax}(r_{tj})$$

We refer to this representation method as conventional attention mechanism.

---

2 Proposed Model

Our method belongs to deletion-based sentence compression, which retains important words while deleting redundant ones. This process can be formulated as a 0/1 labeling problem for each word in the original sentence, where 0 represents deletion and 1 represents retention. The task thus transforms from "input and output are both word sequences" to "input is a word sequence, output is a 0/1 sequence". An example is shown below:

We process this problem based on Sutskever et al.'s sequence-to-sequence paradigm. The basic idea employs an end-to-end training strategy to maximize the probability of correct output for input sentences.

2.1 Pre-Reading Mechanism

For humans, without "pre-reading" the input sentence—that is, without grasping the complete sentence information—it is difficult to obtain correct representations of words or phrases, subsequently affecting sentence compression accuracy. Similarly, while various recurrent neural networks perform well in sequence modeling, the hidden state at timestep t depends only on historical information, and bidirectional models lack direct interaction between forward and backward hidden states, inevitably leading to suboptimal compression effects.

The intuition behind our "pre-reading" mechanism is straightforward. When humans compress sentences, they first read through the original sentence (our "pre-reading") to obtain overall semantics. Based on this pre-reading, they read the sentence again to make keep-or-delete decisions for each word. For computers, this process is implemented by first feeding the original sentence into a neural network to learn its dense distributed representation (semantic vector),

which measures word importance and obtains semantic weights for each word. The original sentence is then fed into the network again, using the first semantic modeling weights to re-extract semantic features, highlighting contributions from high-information words while reducing influence from non-retained words, achieving more task-focused semantic expression.

The general process is illustrated in Figure 2 [Figure 2: see original paper]. Taking forward "pre-reading" as an example (backward pre-reading can be similarly extended), assume the input sequence is $(x_1, x_2, \ldots, x_T)$. We first use standard GRU to read the input sequence:

$$\vec{h}_t^1 = \mathrm{GRU}_1(x_t, \vec{h}_{t-1}^1)$$

where GRU_1 is defined by equations (1)-(4). After obtaining the entire sentence feature vector $\vec{h}_T^1$, we enhance attention semantic information and reduce decision errors through secondary semantic modeling.

The specific architecture is shown in Figure 3 [Figure 3: see original paper]. The output layer is a Softmax classifier that predicts labels for corresponding words or symbols, outputting a 3-dimensional one-hot vector: if retained, the first dimension is 1 (label 1); if deleted, the second dimension is 1 (label 0); if end-of-sentence character, the third dimension is 1 (indicating decoding termination).

2.2 nt-Attention Model

Conventional attention mechanisms suffer from high computational complexity and redundant attention information. In deletion-based sentence compression, where sentence sequences are converted to 0/1 sequences, input and output sequences are equal-length and strictly aligned, making conventional attention mechanisms unnecessary. Tran et al. proposed using the encoder-stage hidden state h_t^E directly as attention information for the corresponding decoder timestep—that is, considering only context information most relevant to the predicted word rather than attending to all sentence components, thereby effectively removing redundant information:

$$h_t = f(x_t, y_{t-1}, h_t^E)$$

However, we argue that immediate neighboring words significantly influence keep-or-delete decisions for the current word. Therefore, we extend t-Attention to 2t-Attention and 3t-Attention for computing weight vector α_t that assists secondary semantic modeling. The specific working principle is shown in Figure 2(b). If the weight α_t for current word x_t approaches 0, the hidden state h_t^2 during secondary encoding carries almost no information from x_t and depends entirely on the previous hidden state h_{t-1}^2. If α_t approaches 1, the structure resembles standard GRU, influenced only by the current word. The secondary modeling rule is:

$$\vec{h}_t^2 = (1-\alpha_t)\vec{h}_{t-1}^2 + \alpha_t \mathrm{GRU}_2(x_t, \vec{h}_{t-1}^2)$$

where α_t is computed as:

$$\alpha_t = \sigma(W\vec{h}_{t-1}^2 + U\vec{h}_T^1 + Vx_t)$$

Matrices W, U, and V are model parameters, and σ is the Sigmoid function. α_t is a weight vector with the same dimension as the hidden state, representing the importance of each dimension in the hidden state. We use a vector rather than a scalar because different dimensions of the hidden state represent different semantic and syntactic features, and a single weight value cannot capture variations in information importance across dimensions.

2.2.1 t-Attention Model We first implement a simple t-Attention model as our baseline, where the hidden state at each encoder timestep directly serves as the attention information for the corresponding decoder timestep:

$$h_t = f(x_t, y_{t-1}, h_t^E)$$

2.2.2 GRU Model with 2t-Attention Taking forward 2t-Attention as an example, when predicting the label for word x_t, we use the combination of hidden states corresponding to x_{t-1} and x_t from the encoding stage as the contextual semantic representation input to the decoder:

$$h_t = f(x_t, y_{t-1}, [h_{t-1}^E, h_t^E])$$

where $[h_{t-1}^E, h_t^E]$ represents the concatenation of hidden states at timesteps $t-1$ and t from the encoding stage. Similarly, backward 2t-Attention can be represented as:

$$h_t = f(x_t, y_{t-1}, [h_t^E, h_{t+1}^E])$$

2.2.3 Bidirectional GRU Model with 3t-Attention To comprehensively consider the influence of neighboring words on the current word, we extend 2t-Attention to 3t-Attention:

$$h_t = f(x_t, y_{t-1}, [\vec{h}_{t-1}^E, \vec{h}_t^E, \overleftarrow{h}_t^E, \overleftarrow{h}_{t+1}^E])$$

Substituting $\mathrm{GRU}_2(x_t, \vec{h}_{t-1}^2)$ into the equation yields:

$$\vec{h}_t^2 = (1-z_t)\vec{h}_{t-1}^2 + z_t\tilde{h}_t^2$$

Further expanding:

$$\vec{h}_t^2 = (1-\alpha_t)\vec{h}_{t-1}^2 + \alpha_t((1-z_t)\vec{h}_{t-1}^2 + z_t\tilde{h}_t^2)$$

$$\vec{h}_t^2 = (1-\alpha_t z_t)\vec{h}_{t-1}^2 + (\alpha_t z_t)\tilde{h}_t^2$$

---

3 Data and Experiments

3.1 Data and Preprocessing

Deep learning models contain numerous parameters and require sufficient training data. To address the scarcity of parallel sentence compression data, Filippova et al. proposed a novel method for automatically generating sentence compression datasets, constructing a large number of "original-compressed" sentence pairs from Google newswire. We conduct experiments and comparative studies based on their publicly available 40,000 sentence pairs. The dataset is divided into three parts: 36,000 pairs for training, 2,000 pairs for validation, and 2,000 pairs for testing.

Before experiments, we preprocess the data. We use the NLTK tokenizer to segment original sentences, then employ the word2vec model for pre-training to obtain 97-dimensional word vectors. Empirical verification shows that pre-training not only accelerates the training process but also yields better models than random initialization; therefore, all our experiments are conducted with pre-trained vectors. We select the top 8,000 most frequent words to form the training vocabulary; words or characters beyond the vocabulary are uniformly represented by "unk" (unknown). A special symbol "eos" (end of sentence) is added at each sentence end as a decoding start indicator. Based on this processing, we construct standard label sequences by marking each word in the sentence: retained words are labeled 1, deleted words 0, and "eos" symbols 2.

3.2 Experimental Setup

In our experiments, the hidden layer units in both encoder and decoder are set to 100. The input is a 100-dimensional vector, where the first 97 dimensions represent the current input word vector. The remaining 3 dimensions differ between encoding and decoding stages: during encoding, they are zero vectors, while during decoding, they represent the one-hot vector of the previous word's ground-truth label (during training) or predicted label (during testing).

Based on Greff et al.'s research on parameter settings and considering our data volume, we initialize the learning rate to 0.001 with a decay rate of 0.9 every 1,000 training steps. We employ early stopping—training terminates when the validation set F1 score fails to increase for 5 epochs—to further prevent overfitting. Specific model parameters are shown in Table 1 .

We select Theano as our experimental framework, with additional environment configuration: Intel Core i7 processor, 16GB RAM, 64-bit Ubuntu 16.04 LTS. Due to substantial computational requirements during training, we add a GTX 1070 GPU accelerator to improve efficiency.

3.3 Experimental Results

Consistent with Tran et al.'s work, we evaluate using F1 score and per-sentence accuracy (Acc), where Acc refers to the proportion of perfectly reproduced compressed sentences. Based on the above settings, we conduct extensive comparative experiments from multiple perspectives including baseline models, attention scope, and pre-reading mechanisms, while also examining the impact of some hyperparameters.

For convenient representation, we define model components as follows: "F" for forward, "B" for backward, "Bi" for bidirectional, "R" for pre-reading, and "tA" for t-Attention. Thus, "BiR-3tA" denotes the "Bidirectional Pre-Reading 3t-Attention" model.

Table 2 compares various baseline models. The first and second rows show models proposed by Fillippova et al. and Tran et al., respectively. Results show that both BiLSTM-tA and BiGRU-tA achieve better results than 3LSTM on the current dataset, with comparable performance, highlighting the enhancement from bidirectional models and attention mechanisms. Below, we use BiGRU-tA as our base model and gradually increase model complexity around two aspects—attention scope and pre-reading mechanism—to analyze various combined models.

Table 3 examines attention scope impact without pre-reading. The second row uses the joint context representation of the current word and its previous word as attention information, while the third row uses the current word and its following word, with the latter performing better. This conclusion aligns with Filippova et al., showing that reverse-order input improves decoding accuracy compared to forward-order input. Results demonstrate that models incorporating neighboring word semantics outperform BiGRU-tA, with the 3t-Attention model that considers both left and right neighbors achieving the best results, indicating that immediate neighboring words significantly influence keep-or-delete decisions. While expanding attention scope, we also examine two methods for representing contextual semantics—concatenation and summation—with concatenation proving superior.

Table 4 compares pre-reading mechanisms (with/without and unidirectional/bidirectional) based on the 3t-Attention model. Results show that models with pre-reading outperform those without, backward pre-reading models outperform forward pre-reading models, and bidirectional pre-reading achieves the best results with an F1 score reaching 0.7936.

Additionally, we further tune some hyperparameters such as word vector dimen-

sion, hidden unit count, and vocabulary size. As shown in Table 5 , with the current data volume, appropriately increasing word vector dimension and hidden unit count improves model performance, while vocabulary size has minimal impact.

In summary, for sentence compression tasks, GRU can replace LSTM. The pre-reading mechanism effectively enhances text modeling capability, while the attention mechanism that fuses left and right neighbor semantics is simple yet effective. The pre-reading mechanism, built upon bidirectional GRU, performs two-stage semantic modeling that simulates human reading behavior, using global information for local adjustment to increase weights of high-information words in semantic representation, thereby improving compression accuracy. The simple attention mechanism leverages direct context representation of the current word, augmented with left and right neighbor semantics while ignoring redundant information, avoiding grammatical error impacts and improving compression efficiency and accuracy. Furthermore, hyperparameter selection affects model performance but remains somewhat empirical.

3.4 Example Analysis

Table 6 shows sentence compression results under different models. Our proposed BiR-3tA model generally produces complete, rich, and grammatically correct compressions. For short input sentences (first two examples), all three models can produce correct compressions; for longer sentences, our model outperforms other compression systems.

The 3LSTM model performs poorly because it contains approximately 1 million parameters, which cannot be optimally tuned using the current training dataset (36,000 sentence pairs). In contrast, our proposed model has fewer parameters and can efficiently capture data distribution patterns and extract frequently occurring features—namely key semantic and syntactic information—while strengthening text semantic modeling. Integrating the simple attention mechanism focuses on context information closely related to the current word, and end-to-end joint training adaptively learns sentence compression decision factors, assigning meaningful weights to each word in the input text that highlight important verbs and nouns while ignoring common words such as prepositions, achieving more precise compression.

References

[1] Chen Jinguang, He Tingting, Li Fang, et al. Chinese sentence pruning based on probability and syntactic analysis [C]// Proceedings of the 5th National Youth Conference on Computational Linguistics, 2010.

[2] Jing H. Sentence reduction for automatic text summarization [C]// Proc of Applied Natural Language Processing Conference. 2000: 310-315.

[3] Jing Xiuli, Zheng Xuewei. Sentence compression method based on Noisy-Channel Model [J]. TV University Journal, 2005 (2): 39-41.

[4] Clarke J, Lapata M. Global inference for sentence compression: An integer linear programming approach [J]. Journal of Artificial Intelligence Research, 2008, 31: 399-429.

[5] Filippova K, Altun Y. Overcoming the lack of parallel data in sentence compression [C]// Proc of EMNLP. 2013: 1481-1491.

[6] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks [C]// Advances in Neural Information Processing Systems. 2014: 3104-3112.

[7] Filippova K, Alfonseca E, Colmenares C A, et al. Sentence compression by deletion with LSTMs [C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2015: 360-368.

[8] Tran N T, Luong V T, Nguyen N L T, et al. Effective attention-based neural architectures for sentence compression with bidirectional long short-term memory [C]// Proc of the 7th Symposium on Information and Communication Technology. New York: ACM Press, 2016: 123-130.

[9] Klerke S, Goldberg Y, Søgaard A. Improving sentence compression by learning to predict gaze [J]. arXiv preprint arXiv: 1604. 03357, 2016.

[10] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model [C]// Proc of Interspeech. 2010, 2: 3.

[11] Luong M T, Pham H, Manning C D. Effective approaches to attention-based neural machine translation [J]. arXiv preprint arXiv: 1508. 04025, 2015.

[12] Ling W, Trancoso I, Dyer C, et al. Character-based neural machine translation [J]. arXiv preprint arXiv: 1511. 04586, 2015.

[13] Wang S, Jiang J. Machine comprehension using match-lstm and answer pointer [J]. arXiv preprint arXiv: 1608. 07905, 2016.

[14] Trischler A, Ye Z, Yuan X, et al. A parallel-hierarchical model for machine comprehension on sparse data [J]. arXiv preprint arXiv: 1603. 08884, 2016.

[15] Legrand J, Collobert R. Joint RNN-based greedy parsing and word composition [J]. arXiv preprint arXiv: 1412. 7028, 2014.

[16] Vinyals O, Kaiser Ł, Koo T, et al. Grammar as a foreign language [C]// Advances in Neural Information Processing Systems. 2015: 2773-2781.

[17] Vinyals O, Toshev A, Bengio S, et al. Show and tell: A neural image caption generator [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. 2015: 3156-3164.

[18] Xu K, Ba J, Kiros R, et al. Show, attend and tell: neural image caption generation with visual attention [C]// Proc of International Conference on Machine

Learning. 2015: 2048-2057.

[19] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult [J]. IEEE Trans on Neural Networks, 1994, 5 (2): 157-166.

[20] Hochreiter S, Schmidhuber J. Long short-term memory [J]. Neural Computation, 1997, 9 (8): 1735-1780.

[21] Cho K, Van Merriënboer B, Bahdanau D, et al. On the properties of neural machine translation: Encoder-decoder approaches [J]. arXiv preprint arXiv: 1409. 1259, 2014.

[22] Chung J, Gulcehre C, Cho K H, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling [J]. arXiv preprint arXiv: 1412. 3555, 2014.

[23] Schuster M, Paliwal K K. Bidirectional recurrent neural networks [J]. IEEE Trans on Signal Processing, 1997, 45 (11): 2673-2681.

[24] Graves A, Jaitly N, Mohamed A. Hybrid speech recognition with deep bidirectional LSTM [C]// Proc of IEEE Workshop on Automatic Speech Recognition and Understanding. 2013: 273-278.

[25] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation [J]. arXiv preprint arXiv: 1406. 1078, 2014.

[26] Luong M T, Pham H, Manning C D. Effective approaches to attention-based neural machine translation [J]. arXiv preprint arXiv: 1508. 04025, 2015.

[27] Cohn T, Hoang C D V, Vymolova E, et al. Incorporating structural alignment biases into an attentional neural translation model [J]. arXiv preprint arXiv: 1601. 01085, 2016.

[28] Rush A M, Chopra S, Weston J. A neural attention model for abstractive sentence summarization [J]. arXiv preprint arXiv: 1509. 00685, 2015.

[29] Hu B, Chen Q, Zhu F. Lcsts: A large scale chinese short text summarization dataset [J]. arXiv preprint arXiv: 1506. 05865, 2015.

[30] Nallapati R, Zhou B, Gulcehre C, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond [J]. arXiv preprint arXiv: 1602. 06023, 2016.

[31] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate [J]. arXiv preprint arXiv: 1409. 0473, 2014.

[32] Chorowski J, Bahdanau D, Cho K, et al. End-to-end continuous speech recognition using attention-based recurrent NN: first results [J]. arXiv preprint arXiv: 1412. 1602, 2014.

[33] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality [C]// Advances in Neural Information Processing Systems. 2013: 3111-3119.

[34] Greff K, Srivastava R K, Koutník J, et al. LSTM: a search space odyssey [J]. IEEE Trans on Neural Networks and Learning Systems, 2016.

[35] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting [J]. Journal of Machine Learning Research, 2014, 15 (1): 1929-1958.

[36] Raskutti G, Wainwright M J, Yu B. Early stopping and non-parametric regression: an optimal data-dependent stopping rule [J]. Journal of Machine Learning Research, 2014, 15 (1): 335-366.

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