Automatic Text Summarization Using Transformers

Siwar Abbes, Sarra Ben Abbès(✉), Rim Hantach, and Philippe Calvez

CSAI Lab ENGIE, Paris, France
{Sarra.Ben-Abbes,Rim.Hantach}@external.engie.com,
Philippe.Calvez1@engie.com

Abstract. Nowadays, we are facing to huge amount of data that makes the task of information analysis quite complex. In this context, automatic text summarization has gained a great deal of success where it is able to extract an efficient short version of documents covering the most important information. In this paper, we propose a new extractive approach for automatic text summarization based on deep learning techniques. This extractive approach can be easily applied on any document independently of its language. Furthermore, by selecting sentences from the document, we guarantee the grammatical and linguistic correctness of summaries. Some experimental results were conducted in order to improve the performance of the proposed approach.

Keywords: Text summarization · Deep learning · Natural language processing

1 Introduction

With the growth of textual information on the web, we are facing to huge amount of data that makes the task of information analysis quite complex. Therefore, natural language processing methods are required to deal with problems related to the tremendous volume of textual data. In this context, text summarization domain is becoming important in the information retrieval domain. In fact, it is very difficult for human beings to manually extract the summaries of a large amount of documents. It has gained a great deal of success where it is able to extract an efficient short version of documents covering the most important information. It has been emerged in different application domains such as news headline generation [15], scientific document abstract generation [23], product review summary [8]. Three types of approaches are proposed [5]: (1) the extractive summarization, aims to select the most important sentences, paragraphs etc., of the original document and concatenates them into a shorther form, (2) the abstractive summarization, aims to create new sentences based on the most useful information from the source document. It is an understanding of the main concepts in a document and then express those concepts in clear natural language, and (3) the hybrid summarization, aims to combine the two models (1) and (2) in order to address its problems and weakness.

B. Villazón-Terrazas et al. (Eds.): KGSWC 2021, CCIS 1459, pp. 308–320, 2021.
https://doi.org/10.1007/978-3-030-91305-2_23

Several efforts have been made to perform text summarization results. However, such methods suffer from defects related to the reliability and the performance of models.

The goal of this work is to propose a new automatic text summarization approach that represents the relevant information of the original document without changing the document intent. In the following paper, we present firstly recent related works. The second section details our proposed approach. In the last section, we highlight some experiments done so far.

2 Related Works

In recent years, text summarization has been widely studied. There are two basic approaches of how to create summaries from document; abstraction and extraction. The main difference between them is how information is extracted from the document and how the summary is generated. Another recent hybrid approaches are proposed.

2.1 Extractive Text Summarization (ETS)

The extractive method is characterized by estimating the relevance of sentences or paragraphs in a document to generate a summary by concatenating the most relevant parts. A single document consists of n sentences $D = \{s_1, s_2, ..., s_n\}$. The i^{th} sentence is denoted as: $s_i = \{w_{i1}, w_{i2}, ..., w_{im}\}$ where w_{ij} is the j^{th} word in s_i. The extractive module learns to pick up a subset of D denoted as

$$\widehat{D} = \{\widehat{s}_1, \widehat{s}_2 ..., \widehat{s}_k, |\widehat{s}_i \in D\}$$

where k sentences are selected. Kageback *et al.* [10] use neural networks to map sentences into vectors and select sentences based on those vectors. Cheng and Lapata [4] select sentences based on an LSTM classifier that predicts a binary label for each sentence. Nallapati et al. [16] adopt a similar approach, SummaRuNNer, a RNN-based model for extractive summarization of documents. It is essentially a two-layer RNN based on sequence classifier.

The drawback here is that it is difficult to optimize the learning due to vanishing gradient problem and that's why, the RNN approach as described does not work particularly well for longer sentences. The other problem plaguing RNNs is the fact that the computation is, by definition, sequential. What does this property entail? A sequential computation cannot be parallelized, since we have to wait for the previous step to finish before we move on to the next one and this lengthens the training time. A common addition to the standard RNN model is to use bidirectional encoders [21], meaning that the input, as well as the input in reverse are encoded into hidden states. Both the forward and the backwards hidden states are then concatenated and fed to the decoder. Bidirectional encoders have been shown to improve performance when encoding longer sequences.

Recently, there has been an attempt at redesigning the RNN-based model by relying more on the attention mechanism introduced by Vaswani et al. [20]. The reason for introducing the attention mechanism was to improve interpretation and enable better generation of longer sequences of text. In [7], the authors introduced a bidirectional attentive encoder-based summarization, where the document encoder has two layers of RNNs; the first layer is based on a self-attentive structure [13] in order to represent a document as a vector and in the second layer, each sentence is concatenated with the document representation returned by the first layer. We can also mention the approach HIBERT (HIerachical Bidirectional Encoder Representations from Transformers) [25] which uses two BERT encoders; a sentence encoder to transform each sentence to a vector and a document encoder to learn sentence representations given their surrounding sentences as context. The main advantage of this approach is the importance of bidirectional language representations: not a shallow concatenation of independently trained left-to-right and right-to-left models. Added to that, the multi-head attention in transformers allows the model to jointly attend to the information from different representation sub-spaces at different positions.

2.2 Abstractive Text Summarization (ATS)

Abstraction is a way of creating summary by rewriting original sentences into shorter ones with preservation of the most important information. This type of summarization builds internal semantic structures and uses NLP techniques to re-phrase the document. Although abstractive summary that is very similar to human produced summary, lack of advancements in NLP and NLU, has hindered its research. Due to the difficulty of automatically generating coherent text, ATS has been considered more complex than the extractive counterpart. In [17], Ramesh Nallapati *et al.* proposed a framework of sequence-to-sequence models based on the attentional Gated Recurrent Unit (GRU) encoder-decoder model. The encoder consists of a bidirectional GRU, while the decoder consists of a uni-directional GRU with the same hidden-state size as that of the encoder, and an attention mechanism over the source-hidden states.

Asli Celikyilmaz *et al.* [2] presented deep communicating agents in an encoder-decoder architecture to address the challenges of representing a long document for abstractive summarization. With deep communicating agents, the task of encoding a long text is divided across multiple collaborating agents, each in charge of a subsection of the input text. These encoders are connected to a single decoder, trained end-to-end using reinforcement learning (RL) to generate a focused and coherent summary.

Similar to this approach, another research [26] was done based on RL for ATS. The authors investigate the effectiveness of another metric of evaluation called BERTScore which is a recently proposed evaluation metric based on n-gram soft-match, as a novel reward function for RL on the abstractive summarization task. They demonstrate its advantage over the most widely-used metric, ROUGE score, via both quantitative evaluation and human evaluation.

The main disadvantage of these approach is that the model describes a sequence of possible events in which the probability of each event depends only on the state attained in the previous event. However, in text summarization, it is preferable to take into consideration not only the left context of the word but also the right one in order to get a better encoding of the input sequence. RL can also lead to an overload of states, which can impact the results.

2.3 Hybrid Methods

Various extractive and abstractive summarization techniques have been investigated but existing approaches are rarely proposed the combination of these two techniques.

In [19], the goal is to generate summaries with varying amounts of reused text by fixing a copy rate as the percentage of summary n-grams appearing in the source text. It uses a Transformer-based architecture to both encode the source text and decode the summary.

In [9] Hsu *et al.* also proposed a unified model where the extractive model consists of a hierarchical bidirectional GRU which selects sentence representations and a classification layer for predicting the relevance of each sentence. The abstractor is based on a bidirectional LSTM to encode the input words and a unidirectional LSTM to decode the summary. In [22], the authors introduced a new approach based on sharing pre-trained BERT decoder: the model first selects sentences by an extractive decoder and then generates summary according to each selected sentence by an abstractive decoder. The main advantage of hybrid methods is the combination of ETS and ATS strengths. However, this type of methods can be time-consuming and costly to create summaries.

Based on the analysis of existing approaches, we decided to conceive two architectures, one based on RNN and another based on Transformers. We choose to work on ETS because by selecting sentences from the summaries, we guarantee the grammatical and linguistic correctness. Moreover, it is easier to apply extractive models on any text document independently of its language. In addition to that, abstractive systems require natural language generation and semantic representation, which are complex and can hardly meet the demands of generating correct facts with proper word relations.

In the next section, we will present the proposed approaches we have worked on which are based on LSTM and Transformer based models.

3 Our Proposed Approaches

The goal of the proposed approaches is to select the most informative sentences, which cover necessary information that is belonged to the gold summary.

Our solutions are based on three modules (see Fig. 1): (i) the first one based on BERT which is designed to learn a contextualized embedding of each sentence in an unsupervised way, (ii) the second one is developed to extract document-level features from the sentence representations. In the first proposition, this

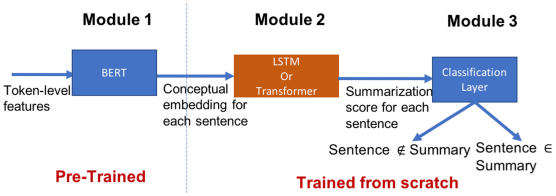

Fig. 1. Proposed architecture.

second module is based on an LSTM model and in the second approach, this module uses Transformers, and (iii) the third module is a classification-layer which, based on the output of the second module, will calculate the probability that a sentence should appear in the final summary or not. In the following sections, we will describe the data preprocessing then we will detail each module in the proposed architecture and justify the choice of models that we have applied.

3.1 Data Preprocessing

To preprocess our data, we tokenized our text documents. Tokenization is the process of converting sentences into separate tokens. In this paper, we used the BERT Word-Piece tokenizer. It means that a word can be broken down into more than one sub-words. This type of tokenization is useful because it deals with out-of-vocabulary words issue. To adapt BERT for text summarization task, we applied some modifications to the original input format of BERT. In the original BERT's configuration [6], the input embeddings are the sum of three kinds of embedding:(i) Token embeddings which represents the meaning of each token, (ii) Segmentation embeddings which are used to differentiate between two sequences, and (iii) Position embeddings which indicates the position of each token in the input. Moreover, special tokens are added to the input sequence. A [CLS] token is used to get features from one sentence or a pair of sentences. The final hidden state corresponding to this token is used as the aggregate sequence representation. To get the representation vector of each sentence, we adopted similar modifications, used in [22], to the input format of BERT. Since we need a symbol for each sentence representation, we should insert the [CLS] token before each sentence. Added to that, we add an interval segment embedding to distinguish multiple sentences within a document. Furthermore, for each sentence we assign a segment embedding SE_0 or SE_1 conditioned on the position of the sentence in the document, is odd or even. As described in [22], our first module is based on 12-layer BERT which will be detailed in the following section.

3.2 First Module: BERT

The first module of our proposed architecture uses BERT model [6] as an embedding module to extract a rich context-based representation of each sentence. BERT stands for Bidirectional Encoder Representations from Transformers.

It is a language model designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As detailled in [6], the transformers in BERT's architecture are linked to capture the bidirectionality. Moreover, BERT is pre-trained on a large corpus of unlabelled text consisting in 33000 million words including the entire Wikipedia and Book Corpus. This pre-training step plays an important role in BERT's success. This can be explained by the fact that training a model on a large text corpus helps to pick up the deeper understandings of how the language works. Furthermore, the BERT model is mostly used in the transfer learning approaches, which consists in adapting the pre-trained model to a specific task which is in our case automatic text summarization. We believe that thanks to its powerful architecture for learning complex features and its pre-training on a large datasets, BERT model can be exploited to build a promising architecture. The output of this first module is a contextualized representation of each sentence which is the [CLS] symbol representation T_i from the top BERT encoder. These sentence-level features will be used as an input to our second module.

3.3 Second Module

The second module describes the two proposed approaches: (1) LSTM-based model, and (2) Transformer-based model. LSTM-based model Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks (RNN), which makes it easier to remember past data in memory. We choose LSTM model because it solves the vanishing gradient issue of RNN. In fact, it preserves gradients over time using dynamic gates that are called memory cells. The hidden state plays an important role in the neural networks memory. It holds information on previous data that the model has seen before. LSTM model [11] contains three gates:

- **Input gate** identifies the values that should be updated and decides what information is relevant to add from the current step.
- **Forget gate** decides what information from the previous hidden state and from the current input should be thrown away or kept.
- **Ouput gate** determines what the next hidden state should be based on the input and the memory of the block.

In order to improve the performance of LSTM-based model, we opt for the Bidirectional LSTM (BiLSTM). BiLSTM [1] has two separate states for forward and backward inputs that are generated by two different LSTMs. In the forward layer, the input is a regular sequence that starts from the beginning of the sentence, while in the backward layer, the input sequence is fed in the opposite order. The idea behind bi-directional network is to capture information of surrounding inputs.

Transformer-Based Model. Our alternative choice for the second module is Transformer-based model. As a second proposed architecture, we applied a

Transformer's encoder instead of an LSTM-based model over the BERT outputs in order to learn summarization-specific features. Recently, there has been an attempt at redesigning the RNN-based model by relying more on the attention mechanism in the context of text summarization. The Transformer, proposed in [20], removed the need of the RNN part by combining the attention mechanism and feedforward layers. Therefore, models based on Transformers reach state-of-the-art results, improving their performance especially on longer sequences of text. The Transformer also proved to train faster than RNN-based models because it allows for more operations running in parallel during training. Moreover, the Transformer was initially applied for the task of machine translation where it outperformed state-of-the art solutions by decreasing training time and increasing translation quality. Thanks to the similarity of the two problems of machine translation and automatic text summarization, it would be reasonable to expect that the Transformer would perform well on. The Transformer's encoder layer [14] has two sub-layers:

1. the first one is a multi-head self-attention mechanism which allows the model to jointly attend to information from different representation sub-spaces at different positions
2. the second one is a fully connected feed-forward network with a single hidden layer which is applied to each position separately and identically. This layer is used to project the attention outputs potentially giving it a richer representation

We employ a residual connection around each of the two sub-layers, followed by layer normalization. The residual connections help the network train, by allowing gradients to flow through the networks directly. The layer normalizations are used to stabilize the network which results in substantially reducing the training time necessary.

We add a positional encoding to the inputs since the Transformer model does not contain recurrence nor convolution. In our case, the positional encoding indicated the position of each sentence in the document.

Multi-Head attention consists of several attention layers running in parallel. Each of these attention layer is a linear transformation of the input representation. It allows the model to associate each input sentence in the document, to other sentences. This helps the model to capture several different aspects of the input and improve its expressive ability. Therefore, the model will be able to generate document-level features which be used as input to the final classification layer.

We will stack the Transformer's encoder several times to further encode the information, where each layer has the opportunity to learn different attention representations therefore potentially boosting the summarization power of the transformer network.

3.4 Third Module: Classification-Layer

This module takes as input the second module's output, denoted by Z_i and adds a linear layer into the outputs of the second module. Then, we compute

the probability of action $a_i \in \{0, 1\}$ to sentence s_i as:

$$p(a_i|Z_i) = \sigma(W_0 Z_i + b_0)$$

where W_0 and b_0 are the model parameters, σ is the sigmoid function. To optimize the extractive module, we use a Binary Cross Entropy Loss:

$$L = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log p(a_i = 1|Z_i) + (1 - y_i) \log(1 - p(a_i = 0|Z_i))$$

where $y_i \in \{0, 1\}$ is the ground-truth label for sentence s_i and N is the number of sentences. When $y_i = 1$, it indicates that sentence s_i should be extracted and be in the summary.

In order to improve the performance of our classification task, we adopted the repetition avoidance technique that was used in [18]. In fact, as illustrated in Fig. 2, while generating the predicted summary, we opt to reduce redundancy as follows: if a candidate sentence, selected to be in the generated summary, has a trigram already existed in the partial summary then it will be skipped. This technique ensures that the predicted summary has not two or more identical set of three successive words. Therefore, we avoid repetition in the produced summaries (based on LSTM or Transformers).

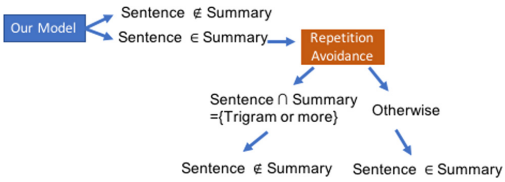

Fig. 2. Repetition avoidance

3.5 Example

An example of the summary generated by our proposed approach. Figure 3 shows that our summary reports the important information of the article and it is close to what is in the gold summary.

The next section will be devoted to the different experiments and the results we got.

4 Experimental Results

Experiments are performed on the CNN/DailyMail dataset [3] that contains news stories in CNN and Daily Mail websites. They are advantageous because

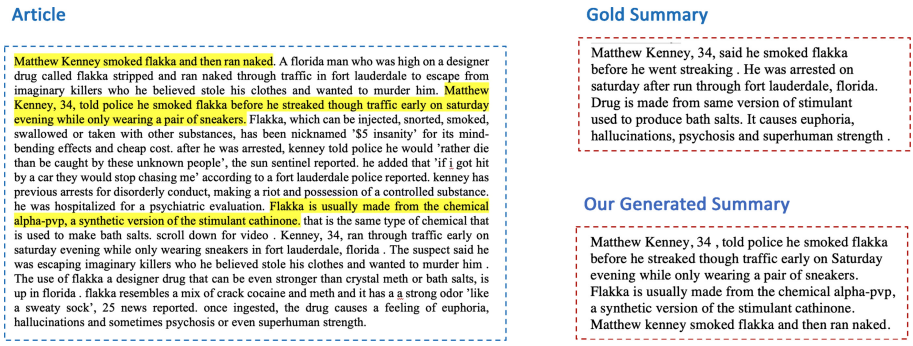

Fig. 3. Example of the summary generated by our proposed approach

each article comes, is paired with a short set of summarized bullet points that represent meaningful highlights. The unique characteristics of this dataset such as long documents, and ordered multi-sentence summaries present interesting challenges. In order to make a fair comparison with recent text summarization approaches, we used the standard splits of [3] for training (287,113 samples), validation (13,368 samples), and testing (11,490 samples).

4.1 Evaluation Measures

As evaluation measures, we opt for the same metrics used to evaluate the state-of-the-art solutions. The most common evaluation metric is ROUGE score. In addition to that, BERTscore is also used to assess the performance of a summarizer model.

ROUGE Scores. One of the automatic evaluation metrics is ROUGE score [12], which is a measure of overlapping n-grams in the generated summary and one or several reference summaries constructed by humans. The most commonly used versions in previous studies are:

- ROUGE-1 (R1) refers to the overlap of uni-gram(each word) between the system and reference summaries
- ROUGE-2 (R2) refers to the overlap of bi-grams between the system and reference summaries
- ROUGE-L (RL): refers to the Longest Common Sub-sequence. It takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams automatically.

For each metric we computed three scores [12]; Precision, Recall and F1-score. ROUGE scores have a major limitation as an evaluation metric. In fact, as ROUGE scores only measure token hard-match, in some cases they will penalize two sentences conveying exactly the same semantic information, but highly reward sentences with completely different semantics yet in similar surface forms.

BERTscore (BT). It is a recently proposed evaluation metric in [24]. Similar to ROUGE score, BERTscore computes a cosine similarity score for each token in the generated summaries with each token in the reference summaries. By computing token similarity using contextualized word embeddings provided by BERT [6], BERTScore successfully incorporates semantic information behind sentences, thus can provide better evaluations for cases where ROUGE score fails to account for meaning-preserving lexical and semantic diversity.

Correlation Between BT and ROUGE. A case study was done in [26] on 2000 cases randomly sampled from the test set of CNN dataset to compare between BERTScore and ROUGE score. The correlation between the two metrics can be explained by the fact that when ROUGE score is maximized, the number of overlapping tokens in reference summaries and predicted summaries are high. In this case, BERTscore is also high as BERTscore between two identical sets of words is 100%. However, having a high BERTscore does not lead to obtain good ROUGE scores as reference and predicted summaries can have the semantic information with completely different words.

4.2 Results

In order to compare our approaches to the existing state-of-the-art, we will take into consideration only the F1-score of each evaluation metric as it is the weighted average of Precision and Recall. The setting used for evaluation was structured as the following: in the first approach, we tested the LSTM then the BiLSTM as a second module. While in the second approach, we stacked the Transformer several times. As mentioned in the previous section, the first and the third modules in both approaches are the same. As we notice in Table 1, the BiLSTM-based model performs better than the LSTM as it takes into consideration both the left and the right context for each input sequence. However, while analyzing the obtained results, we noticed that the Transformer-based approach yields great results compared to the LSTM-based models. This can be explained by the fact

Table 1. Best results of our proposed approaches using F1-score of each evaluation metric.

	R1	R2	RL	BT
First approach				
LSTM	34,24	13,93	30,12	61,85
BiLSTM	36,71	14,62	33,24	**62,34**
Second approach				
1 Transformer	39,72	17,12	36,13	74,14
2 Transformers	40,37	17,51	37,41	86,83
3 Transformers	41,08	18,23	38,92	**86,92**

Table 2. Testing results on the CNN/DailyMail dataset using ROUGE F1.

Model	R1	R2	RL	BT
Extractive models				
Our model	41,08	18,23	**38,92**	**86,92**
Attentive Encoder-based	38.80	12.61	33.85	–
SummaRuNNer	39.60	16.20	35.30	–
HIBERT	42.37	19.95	38.83	–
Abstractive models				–
RNNabs	35.46	13.30	32.65	–
RL	41.69	19.47	37.92	–
RLbertscore	**43.28**	18.69	36.58	62.77
Hybrid models				
Unified model	40.68	17.97	37.13	–
Sharing BERT	41.76	19.31	38.86	–

that, unlike RNNs models, Transformers can handle long sequences with long range of dependencies. In addition, to achieve a higher evaluation scores, the Transformer trains faster and can learn on longer input and output sequences before running out of memory. As illustrated in Table 2, the approach with 3 stacked Transformers shows the best performance so we will compare its scores with the proposed systems mentioned in the second chapter state-of-the-art solutions. Note that our best model is based on 3 Transformers. As illustrated in Table 2, not all systems are evaluated with BERTscore and unfortunately, we did not find an extractive approach that was evaluated with BERTscore because it is a very recent evaluation metric. From the obtained results, our second approach based on three stacks of Transformers outperforms the state-of-the-art result in ROUGE-L and BERTscore. On the one hand, the good score of ROUGE-L shows that our model can generate informative and coherent summaries.

In addition, the highest BERTscore obtained by our approach, can be explained by using an extractive approach as the system summaries, which are semantically very close to the reference summaries. Moreover, we can also notice that pre-trained based summarization approaches [25] and [22] using BERT, have the best performances. We can then conclude that the powerful architecture of BERT helps automatic summarization systems to achieve the highest scores. Furthermore, based on the ROUGE-1 and ROUGE-2 scores, our model is also comparable with most mentioned approaches. In fact, compared to the models [7,16,17] which use RNN-based models, our model achieves better performance. This can be explained by the use of Transformers and its advantages against RNNs models. We can add to that our second approach achieves almost the similar performance of hybrid models [9] and [22] knowing that these systems

combine the pros of extractive and abstractive models. However, our model has an advantage against hybrid approaches considering its fast training without forgetting that these methods are costly to create summaries.

5 Conclusion

In this paper, we propose a new automatic text summarization method. This work focus on extractive model because it will be easier to apply it on any document independently of its language. Different modules have been done: (i) Bert module, (ii) LSTM or Transformer models, and (iii) classification layer. Experimental results shows the ability of our models to extract informative contents and proves that the Transformer-based models outperform LSTM-based models. As future works, our proposed approach can be improved by carrying out more hyper parameter tuning for better performance, which has not been exhaustively performed due to limited time. It is also possible to use a reward function as a linear combination of ROUGE and BERTScore that better approximates human evaluation. Experimentations on different real world datasets will be tested.

References

1. Cai, L., Zhou, S., Yan, X., Yuan, R.: A stacked bilstm neural network based on coattention mechanism for question answering. Comput. Intell. Neurosci. 1–12 (2019)
2. Celikyilmaz, A., Bosselut, A., He, X., Choi, Y.: Deep communicating agents for abstractive summarization. In: Conference of the North American Chapter of the Association for Computational Linguistics, pp. 1662–1675. Louisiana (2018)
3. Chen, D., Bolton, J., Manning, C.D.: A thorough examination of the CNN/Daily Mail reading comprehension task. In: Computational Linguistics Association, pp. 2358–2367. Germany (2016)
4. Cheng, J., Lapata, M.: Neural summarization by extracting sentences and words, pp. 484–494 (2016)
5. Dalal, V., Malik, L.: A survey of extractive and abstractive text summarization techniques. In: International Conference on Emerging Trends in Engineering and Technology, pp. 109–110. USA (2013)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding (2018)
7. Feng, C., Cai, F., Chen, H., de Rijke, M.: Attentive encoder-based extractive text summarization. In: International Conference on Information and Knowledge Management, pp. 1499–1502. USA (2018)
8. Gong, Y., Luo, X., Zhu, K., Ou, W., Li, Z., Duan, L.: Automatic generation of Chinese short product titles for mobile display. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 9460–9465 (2019)
9. Hsu, W.T., Lin, C.K., Lee, M.Y., Min, K., Tang, J., Sun, M.: A unified model for extractive and abstractive summarization using inconsistency loss. In: Computational Linguistics Association, pp. 132–141. Australia (2018)
10. Kågebäck, M., Mogren, O., Tahmasebi, N., Dubhashi, D.: Extractive summarization using continuous vector space models. In: Workshop on Continuous Vector Space Models and their Compositionality, pp. 31–39. Sweden (2014)

11. KNIME: Once Upon A Time ... by LSTM Network (2019). https://www.knime.com/blog/text-generation-with-lstm
12. Lin, C.Y.: ROUGE: a package for automatic evaluation of summaries. In: Text Summarization Branches Out, pp. 74–81. Spain (2004)
13. Lin, Z., et al.: A structured self-attentive sentence embedding. In: International Conference on Learning Representations (2017)
14. Mozer, M.: BERT Does Business: Implementing the BERT Model for Natural Language Processing at Wayfair (2019)
15. Murao, K., et al.: A case study on neural headline generation for editing support. In: Computational Linguistics Association. Minnesota (2019)
16. Nallapati, R., Zhai, F., Zhou, B.: Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In: CoRR (2017)
17. Nallapati, R., Zhou, B., Dos Santos, C., Gulcehre, C., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond. In: Conference on Computational Natural Language Learning, pp. 280–290 (2016)
18. Paulus, R., Xiong, C., Socher, R.: A deep reinforced model for abstractive summarization. In: International Conference on Learning Representations (2018)
19. Song, K., Wang, B., Feng, Z., Liu, R., Liu, F.: Controlling the amount of verbatim copying in abstractive summarization. In: Conference on Artificial Intelligence, pp. 8902–8909 (2020)
20. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
21. Wang, W., Chang, B.: Graph-based dependency parsing with bidirectional LSTM. In: Computational Linguistics Association, pp. 2306–2315. Germany (2016)
22. Wei, R., Huang, H., Gao, Y.: Sharing pre-trained BERT decoder for a hybrid summarization. In: Chinese Computational Linguistics, pp. 169–180 (2019)
23. Yasunaga, M., et al.: ScisummNet: a large annotated corpus and content-impact models for scientific paper summarization with citation networks. In: Conference on Artificial Intelligence (2019)
24. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: evaluating text generation with BERT. In: International Conference on Learning Representations (2020)
25. Zhang, X., Wei, F., Zhou, M.: Hibert: document level pre-training of hierarchical bidirectional transformers for document summarization. In: Computational Linguistics Association, pp. 5059–5069. Italy (2019)
26. hui Zhang, Y., Wang, R., Zhou, Z.: Improving neural abstractive summarization via reinforcement learning with Bertscore. In: Conference on Empirical Methods in Natural Language Processing, pp. 4078–4087. Belgium (2019)