


ARTICLE

# Reducing repetition in convolutional abstractive summarization

Yizhu Liu<sup>1</sup>, Xinyue Chen<sup>2</sup>, Xusheng Luo<sup>3</sup> and Kenny Q. Zhu<sup>1,\*</sup>

<sup>1</sup>Shanghai Jiao Tong, Shanghai, China, <sup>2</sup>Carnegie Mellon University, Pennsylvania, USA, and <sup>3</sup>Alibaba Group, Hangzhou, China

\*Corresponding author: Email: [kzhu@cs.sjtu.edu.cn](mailto:kzhu@cs.sjtu.edu.cn)

(Received 9 April 2020; revised 18 September 2021; accepted 20 September 2021; first published online 24 November 2021)

## Abstract

Convolutional sequence to sequence (CNN seq2seq) models have met success in abstractive summarization. However, their outputs often contain repetitive word sequences and logical inconsistencies, limiting the practicality of their application. In this paper, we find the reasons behind the repetition problem in CNN-based abstractive summarization through observing the attention map between the summaries with repetition and their corresponding source documents and mitigate the repetition problem. We propose to reduce the repetition in summaries by attention filter mechanism (ATTF) and sentence-level backtracking decoder (SBD), which dynamically redistributes attention over the input sequence as the output sentences are generated. The ATTF can record previously attended locations in the source document directly and prevent the decoder from attending to these locations. The SBD prevents the decoder from generating similar sentences more than once via backtracking at test. The proposed model outperforms the baselines in terms of ROUGE score, repeatedness, and readability. The results show that this approach generates high-quality summaries with minimal repetition and makes the reading experience better.

**Keywords:** Abstractive summarization; Repetition reduction; Convolutional sequence-to-sequence model; Attention mechanism

## 1. Introduction

Abstractive summarization is the task of creating a short, accurate, informative, and fluent summary from a longer text document. It attempts to reproduce the semantics and topics of original text by paraphrasing. Recently, sequence to sequence models (Rush *et al.* 2015; Chopra *et al.* 2016; Nallapati *et al.* 2016; See *et al.* 2017; Paulus *et al.* 2018) have made great progress on abstractive summarization. A recent study (Bai *et al.* 2018) suggests that, without additional, complicated structures or features, convolutional sequence to sequence (CNN seq2seq) models (Gehring *et al.* 2017; Fan *et al.* 2018; Liu *et al.* 2018) are more effective and can be trained much faster due to their intrinsic parallel nature compared to recurrent neural networks (RNNs). Furthermore, unlike RNN-based models, the convolutional models have more stable gradients because of their back-propagation paths. Self-attention-based model is the basis of many recent state-of-the-art (SOTA) systems, which always need multi-layer self-attention and have greater computational complexity than CNN seq2seq models (Vaswani *et al.* 2018). Thus, we take CNN seq2seq models as the target model to improve on and compare with in this paper.

Unfortunately, just like RNN-based models, CNN-based models also produce summaries with substantial repeated word sequences which impacts the reading efficiency. Table 1 illustrates one test case from the CNN/Daily Mail summarization dataset. In this case, the basic CNN produces

**Table 1.** Summary generated by the basic CNN model. The parts of source document and summary in the same color denote that they are aligned by the CNN

Source Document
<p>... manchester city are rivalling manchester united and arsenal for valenciennes teenage defender dayot upamecano. <b>the 16-year-old almost joined united in the january transfer window</b>, only for him to opt to stay in france for a few more months. centre-back umecano has played for france at u16 and u17 level. monaco, inter milan and paris stgermain had also expressed interest. <b>fourth-placed city face aston villa at the etihad stadium on saturday ....</b></p>
Reference summary
<p>dayot upamecano was close to signing for manchester united in january. the 16-year-old, however, opted to stay in france with valenciennes. centre-back upamecano has played for france at u16 and u17 level. arsenal are also interested in the defender as man city join chase.</p>
Basic CNN model (CNN)
<p><i>manchester city face aston villa at the etihad stadium on saturday.</i>  <b>the 16-year-old almost joined united in the january transfer window.</b>  <i>manchester city face aston villa at the etihad stadium on saturday.</i></p>

two identical sentences (italicized) in the result. Unlike machine translation or paraphrasing in which the output words and input words are almost one-to-one aligned, the output of summarization is “compressed” from the input document. Naturally, every sentence or word sequence in the summary corresponds to one or more places in the source document. If there were two identical word sequences in the summary, they might be looking at and summarizing the same “spots” in the source. This is evident from the attention map for the three sentences generated by CNN, shown in Figure 1. The first and third sentences attend to the same location in the source (red boxes), while the second sentence attends to another separate location in the source (green box). The two attention maps in the red boxes are very similar.

Driven by this intuition, a few efforts have been made on “remembering” what has been focused on before at decoding. For example, Paulus *et al.* (2018) and Fan *et al.* (2018) use intra-temporal attention (Nallapati *et al.* 2016) as well as intra-decoder attention to avoid attending to the same parts in the source by revising attention scores while decoding. See *et al.* (2017) and Gehrmann *et al.* (2018) respectively propose coverage mechanism and coverage penalty, which records the sum of attention distributions of all previously generated words in a different way to track the summarized information. While these approaches discourage repetition to some extent, they do so in an indirect manner. That is, they do not make use of the attention information in the source directly. Consequently, they may still generate repeated phrases, especially in long sentences (shown in the first five sections of Table 2).

In this paper, we propose an attention filter mechanism that directly redistributes the attention from each word in the output summary to the source. It does so by computing the *parts of interest* (POIs) in the source per segment in the summary and then minimizing the attention scores of words in these POIs that have already been attended to by the preceding segments during decoding. POIs are the segments of source document that are attended by the segments in its corresponding summary, such as the green and red segments of the source document in Table 1 and Figure 1. Different segments in summary thus do not attend to the same semantic spots of source, and repetition is reduced. We can get segments in different ways. As shown in Table 3, we compare the segments in different types. The baseline with a sentence as the segment (sentence-level segment) always loses important information in reference summary, such as “silas randall timberlake.” The first sentence in generated summary based on sentence-level segment attends to the second sentence in source. The attention score of the second sentence in source is minimized, and this source sentence is no longer attended. So, the model with sentence-level segment loses the

**Table 2.** Generated summaries of the source in Table 1. The sentences in *italicized* are repetitive sentences. Each summary segment and its attended POI in source document are in the same color

Source Document
<p>... manchester city are rivalling manchester united and arsenal for valenciennes teenage defender dayot upamecano. <i>the 16-year-old almost joined united in the january transfer window, only for him to opt to stay in france for a few more months.</i> centre-back umecano has played for france at u16 and u17 level.</p> <p>monaco, inter milan and paris stgermain had also expressed interest. <i>fourth-placed city face aston villa at the etihad stadium on saturday.</i> ...</p>
<p><b>Intra-temporal attention (ITA)</b></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p>
<p><b>Intra-temporal + Intra-decoder (ITDA)</b></p> <p><i>manchester city are rivalling manchester united and arsenal for valenciennes teenage.</i></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>manchester city are rivalling manchester united and arsenal.</i></p>
<p><b>Coverage model (COV)</b></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>manchester city are rivalling manchester united and arsenal for valenciennes.</i></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p>
<p><b>Coverage penalty (COVP)</b></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>manchester city are rivalling manchester united and arsenal.</i></p>
<p><b>Semantic cohesion loss (SCL)</b></p> <p><i>manchester city are rivalling manchester united and arsenal for defender dayot upamecano.</i></p> <p><i>manchester city are rivalling for valenciennes teenage.</i></p>
<p><b>Diverse Convolutional Seq2Seq Model (DivCNN)</b></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>the 16-year-old has played for france for a few months</i></p>
<p><b>Trigram decoder (TRI)</b></p> <p><i>defender dayot upamecano has played for france at unk and unk level.</i></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p>
<p><b>Ours (Attention Filter + Sentence-level Backtracking decoder)</b></p> <p><i>manchester city face aston villa at the etihad stadium on saturday.</i></p> <p><i>the 16-year-old almost joined united in the january transfer window.</i></p> <p><i>manchester city are rivalling manchester united and arsenal for teenage defender dayot upamecano.</i></p>

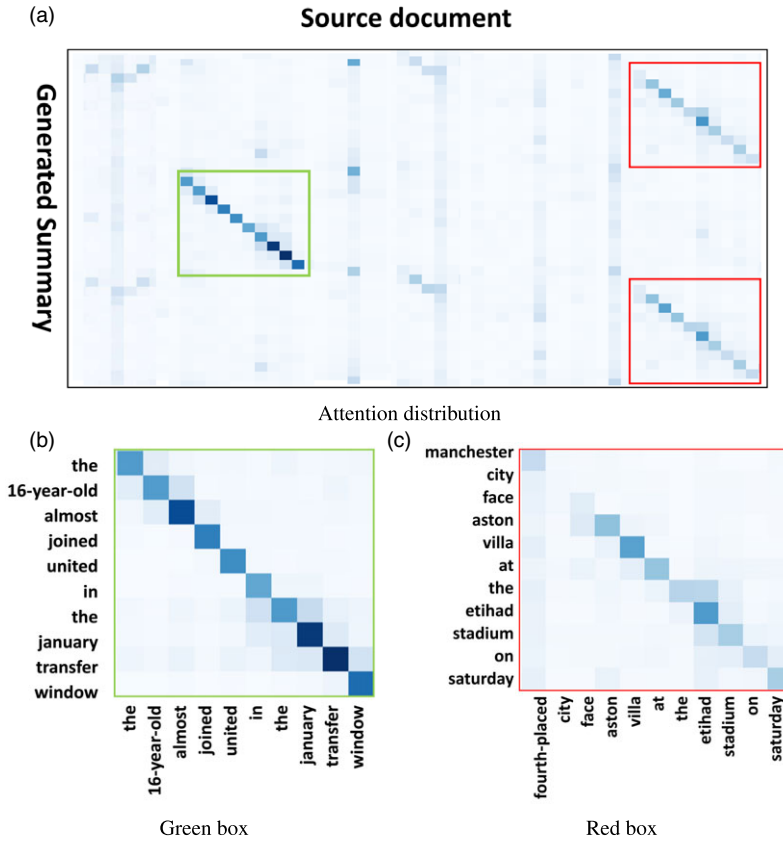


Figure 1. Attention for example on CNN model in Table 1.

important information “silas randall timberlake” during decoding. The baseline with N-gram as a segment (N-gram segment) may bring about grammatical and semantic problems. Suppose that  $N$  equals  $3^a$ , as shown in Table 3, the green part of generated summary-based N-gram segment does not attend to the “the couple announced” in source document. As N-gram cannot be seen as a complete and accurate semantic unit, the decoder of model with N-gram segment attends to the segment in source with inaccurate grammar and semantics. Thus, the generated summary based on N-gram segment has grammatical and semantic errors. We use punctuations to separate the source or target into segments (punctuation-based segments), since punctuations play an important role in written language to organize the grammatical structures and to clarify the meaning of sentences (Briscoe 1996; Kim 2019; Li *et al.* 2019b). It is very simple but effective. In this paper, a segment means a sentence or clause delimited by punctuation, which carries syntactic and semantic information. Specifically, we calculate the attention in terms of segments (larger semantic units than tokens and smaller semantic units than sentences) which intuitively helps with the emphasis of attention and POIs in source. This is different from previous approaches which all do not exactly pinpoint these parts in the source, which we believe is critical in reducing repetition.

Despite the above effort, there are cases, where similar sentences exist in the same source document:

**Example 1.** “the standout fixture in the league on saturday sees leaders chelsea welcome manchester united . . . chelsea midfielder oriol romeu, **currently on loan at stuttgart**, . . . romeu is **currently on a season-long loan at bundesliga side stuttgart.**”

<sup>a</sup>We set  $N$  as 3 because ground-truth summaries almost never contain the same trigram twice.

**Table 3.** The summary generated by attention filter mechanism with different types of segments. The parts of text in different colors denote segments. The segments in the same color attend to the same POI in the source

Source
(1) justin timberlake and jessica biel, welcome to parenthood.
(2) the celebrity couple announced the arrival of their son, silas randall timberlake, . . .
(3) silas was the middle name of timberlake 's maternal grandfather bill bomar, who died in 2012.
(4) the couple announced the pregnancy in january, . . .
(5) it is the first baby for both.
Reference
timberlake and jessica biel welcome son silas randall timberlake.
the couple announced the pregnancy in january.
Sentence-level segment
the couple announced the arrival of their son.
it is the first baby for both.
N-gram segment
the couple announced silas randall timberlake pregnancy in january.
Punctuation-based segment
the couple announced the arrival of their son, silas randall timberlake.
the couple announced the pregnancy in january.
it is the first baby for both.

In this case, even if the decoder attends to different POIs of source document as it produces words, repetition may still result. At different time steps, the decoder may attend to sentences that are similar in different positions. One potential solution to this is semantic cohesion loss (SCL) (Çelikyılmaz *et al.* 2018) which takes the cosine similarity between two consecutively generated sentences as part of the loss function. It may attend to the same POI and generate similar sentences (SCL row in Table 2). The other is Diverse Convolutional Seq2Seq Model (DivCNN) (Li *et al.* 2019a), which introduces determinantal point processes (DPPs) (Kulesza and Taskar 2011) into deep neural network (DNN) attention adjustment. DPPs can generate subsets of input with both high *quality* and high *diversity* (QD-score). For abstractive summarization, DivCNN takes hidden states of DNN as QD-score. DivCNN selects the attention distribution of the subsets of source document with high QD-score first and then adds the selected attention distribution into model loss as a regularization. DivCNN does not directly redistribute the attention, so it may still attend to similar POIs. To improve QD-score, DivCNN tends to attend to scattered subsets of sentences in source document, which leads to semantic incoherence. As shown in Table 2 (DivCNN row), the content about the 16-year-old is inconsistent with source document. Besides, trigram decoder (TRI) (Paulus *et al.* 2018) directly forbids repetition of previously generated trigrams at test time. While this simple but crude method avoids the repeat of any kind completely, it ignores the fact that *some amount of repetition* may exist in natural summaries. On the other hand, the meddling of the sentence generation during beam search causes another problem: it tends to generate sentences that are logically incorrect. In Table 2 (TRI row), the defender dayot didn't really play for France, according to the source. That is, the subject and object are mismatched. As trigram cannot

reflect the complete semantic information, trigram decoder is likely to generate logically incorrect summaries due to the trigram-based meddling of the sentence generation during testing. In order to avoid the logical incorrectness caused by trigram decoder, we introduce a *sentence-level* backtracking decoder (SBD) that prohibits the repeat of the same sentence at test time. Compared with trigram decoder, SBD can avoid repetition and generate more logical summaries. Our summary produced for the example is shown in the last section of Table 2.

Reducing repetition in abstractive summarization provides high-quality summaries for users and improves their reading efficiency. We expect that other natural language generation (NLG) tasks suffered from repetition problem can be enhanced with our approach. Our contributions are summarized as follows:

- (1) We find the reasons behind repetition problem in abstractive summaries generated by CNN seq2seq models through observing the attention map between source documents and summaries.
- (2) We propose an effective approach that redistributes attention scores during training time and prevents repetition by sentence backtracking at runtime to reduce repetition in CNN seq2seq model.
- (3) Our approach outperforms the SOTA repetition reduction approaches on CNN-based models substantially by all evaluation metrics, including ROUGE scores, repeatedness and readability.

Next, we present the basic CNN seq2seq model and our extension, followed by the evaluation of our approach and a discussion of related work.

## 2. Approach

In this section, we describe the model architecture used for our experiments and propose our novel repetition reduction method, which is an extension to the basic model.<sup>b</sup>

In summarization task, the input (source document) and output (summary) are both sequences of words. Suppose the input and output are respectively represented as  $\mathbf{x} = (x_1, x_2, \dots, x_m)$  and  $\mathbf{y} = (y_1, y_2, \dots, y_n)$  ( $m > n$ ), the goal is to maximize the conditional probability  $p(\mathbf{y}|\mathbf{x})$ :

$$p(\mathbf{y}|\mathbf{x}) = \prod_t^n p(y_t | y_1, y_2, \dots, y_{t-1}, \mathbf{x}) \quad (1)$$

Furthermore, we aim to generate summaries that are not only fluent and logically consistent with the source documents, but also with a small amount of repeatedness, which is natural in human-written summaries.

### 2.1. Basic CNN seq2seq model

Our basic model is multi-layer convolutional seq2seq networks (Gehring *et al.* 2017) with attention mechanism<sup>c</sup>, as illustrated in Figure 2.

For CNN seq2seq models, we combine word embeddings and position embeddings to obtain input  $\mathbf{X} = (X_1, \dots, X_m)$  and output  $\mathbf{Y} = (Y_1, \dots, Y_n)$ . We denote  $\mathbf{z}^l = (z_1^l, \dots, z_m^l)$  and  $\mathbf{h}^l = (h_1^l, \dots, h_n^l)$  respectively as convolutional output of the encoder and decoder in the  $l$ -th layer. Each element of the output generated by the decoder network is fed back into the next layer of decoder network. In each layer, GLU (Dauphin *et al.* 2017) and residual connections

<sup>b</sup>All of the data and source code can be downloaded from <https://github.com/YizhuLiu/sumrep>

<sup>c</sup><https://github.com/pytorch/fairseq>

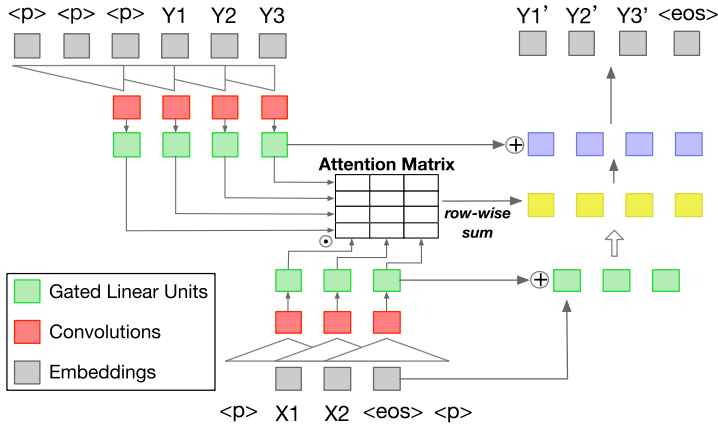


Figure 2. Convolutional seq2seq model.

(He *et al.* 2016) are used respectively as a nonlinear gate and guarantee for sufficient depth of the network:

$$h_i^l = GLU \left( W^l \left[ H_{i-k/2}^{l-1}, \dots, H_{i+k/2}^{l-1} \right] + b_w^l \right) + H_i^{l-1} \tag{2}$$

where  $k$  is kernel width.  $W$  and  $b$  are trainable parameters.

For each decoder layer, the multi-step attention integrates encoder information. We compute decoder state  $d_i^l$  for attention via

$$d_i^l = W_d^l h_i^l + b_d^l + Y_i \tag{3}$$

$$a_{ij}^l = \frac{\exp \left( d_i^l \cdot z_j^u \right)}{\sum_{t=1}^m \exp \left( d_i^l \cdot z_t^u \right)} \tag{4}$$

$$c_i^l = \sum_{j=1}^m a_{ij}^l \left( z_j^u + X_j \right) \tag{5}$$

$$H_i^l = c_i^l + h_i^l \tag{6}$$

where  $d_i^l$  is decoder state,  $z_j^u$  is the encoder state,  $u$  is the last layer of encoder, and  $a_{ij}$  is attention score. The inner product between decoder state and encoder outputs is used to measure the affinity. The conditional input to the current decoder layer is a weighted sum of the encoder states and input representations. We get  $H_i^l$  by adding  $c_i^l$  to  $h_i^l$ , which forms the input for the next decoder layer or the final output.

Finally, we compute the probability distribution for the next word using the top decoder output:

$$p \left( y_{i+1} | y_1, \dots, y_i, \mathbf{x} \right) = \text{softmax} \left( W_o H_i^l + b_o \right) \tag{7}$$

### 2.2. Attention filter mechanism (ATTF)

We propose an ATTF as a novel extension to the basic model, which can record previously attended locations in the source document directly and generate summaries with a natural level of repeatedness. This method aims at relieving the repetition problem caused by decoders attending to the same POI in source document.

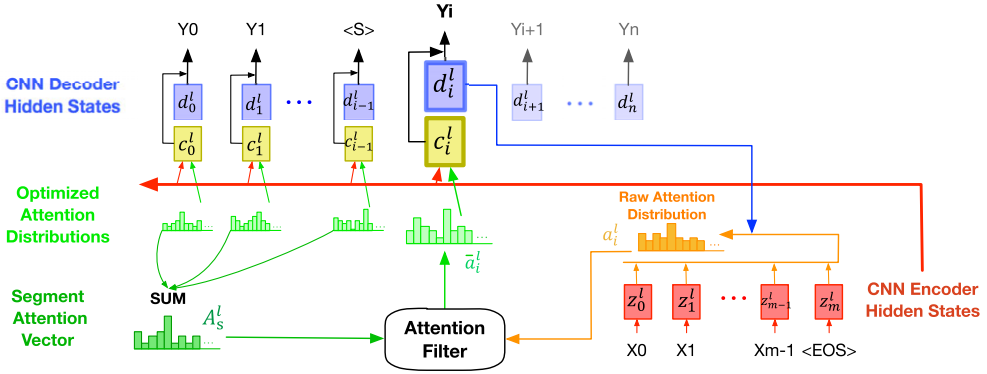


Figure 3. Overview of Attention Filter Mechanism (ATTF).

2.2.1. Notations

In this mechanism, both source document and summary are respectively split into segments by punctuations. For the convenience of description, we take “sections” as the segments in source document and “segments” as the segments in summary. We denote the punctuation marks as <S>.  $\mathbf{u} = (u_0, u_1, \dots, u_M)$  denotes the positions of <S> in source document and  $\mathbf{v} = (v_0, v_1, \dots, v_N)$  denotes the positions of <S> in summary. Both  $u_0$  and  $v_0$  are  $-1$ . Therefore, we can represent source document as  $\mathbf{U} = (U_1, U_2, \dots, U_M)$  and  $\mathbf{V} = (V_1, V_2, \dots, V_N)$  as summary.  $U_i$  is the  $i$ -th section and  $V_i$  is  $i$ -th segment. Both  $U_i$  and  $V_i$  are the sequences of tokens without punctuation tokens.

Let  $D$  denote the number of tokens in the source document.  $a_i^l = (a_{i1}^l, a_{i2}^l, \dots, a_{ij}^l, \dots, a_{iD}^l)$  is a  $D$ -dimensional vector that records the attention scores in  $l$ -th layer of the  $i$ -th token in the summary over tokens in the source document. We define segment attention vector in the  $l$ -th layer as  $A^l = (A_0^l, A_1^l, \dots, A_N^l)$ . For  $s$ -th segment  $V_s$ ,  $A_s^l = (A_{s1}^l, A_{s2}^l, \dots, A_{sD}^l)$  is a vector representing segment attention distribution over tokens in the source document.  $A_{sj}^l$  is the attention score between  $V_s$  and  $j$ -th token in the source document.

2.2.2. Description

To measure the relevance between  $j$ -th token of the source document and the  $s$ -th segment  $V_s$ , we sum up attention scores of each token in the  $s$ -th segment over  $j$ -th token of the source document:

$$A_{sj}^l = \sum_{i=v_{s-1}+1}^{v_s-1} a_{ij}^l \tag{8}$$

We set  $A_0^l$  as a zero vector, because nothing is attended before generating the first segment.

To find the most attended sections of segment  $V_s$ , we sort the elements inside the filter vector,  $A_s^l$ , in descending order, and record the top  $k$  elements’ positions in the source document as:  $\mathbf{p} = (p_1, \dots, p_k)$  where  $k = v_s - v_{s-1} - 1$ . In the other words,  $\mathbf{p}$  records the position of  $k$  words most attended to  $V_s$  in the source document.  $k$  is the same as the number of tokens in the  $s$ -th segment, because each token in a segment is aligned with at least one token in source document, according to the principle of the seq2seq model. Thus, for the  $s$ -th segment, we can find its most attended sections by  $\mathbf{p}$ . We locate the elements at  $\mathbf{p}$  in the source document as well as the sections they belong to. For section  $U_t$ , we take the tokens that belong to  $U_t$  and are located in the positions of  $\mathbf{p}$  as  $P_{U_t}$ . If the size of  $P_{U_t}$  is larger than  $\beta$ , a predefined constant, the section  $U_t$  is a POI of



segment  $V_s$ , which should not be attended to again.  $\mathbb{U}_s$  denotes a set of all such POIs for  $V_s$ .  $\mathbb{U}_0$  is an empty set.

We construct two multi-hot vectors  $g_s$  and  $g'_s$  for each segment  $V_s$ . The dimensions of them are the number of tokens in source document,  $D$ , which are the same as the dimension of  $A_s^l$ . For  $g_s$ , we set elements on the position of tokens belonging to sections in  $\mathbb{U}_s$  to 0, and the elements on other positions to 1.  $g_{sj}$  is the  $j$ -th elements of  $g_s$ . If  $g_{sj} = 0$ , it means that the  $j$ -th token is attended by segment  $V_s$  during the generation of  $V_s$ .  $g'_{sj}$  is  $j$ -th element of  $g'_s$ , which is the flipped version of  $\prod_{q=1}^s g_{qj}$ . In the other words,  $g'_{sj}$  is  $1 - \prod_{q=1}^s g_{qj}$ . If  $\prod_{q=1}^s g_{qj} = 0$  and  $g'_{sj} = 1$ , it means that the  $j$ -th token of the source document has been attended before. The filter on  $a_{ij}^l$  in Equation (4) is given as:

$$\tilde{a}_{ij}^l = a_{ij}^l \prod_{q=1}^s g_{qj} + \min \left( \frac{A_{sj}^l}{v_s - v_{s-1} - 1} \right) g'_{sj} \tag{9}$$

where  $\tilde{a}_{ij}^l$  is the filtered attention score.  $A_{sj}$  is the attention score between  $j$ -th token of the source document and the  $s$ -th segment.  $g_{sj}$  and  $g'_{sj}$  denote whether  $j$ -th token of the source document has been attended. We penalize the attention score of attended tokens in source document. We take the minimum attention score between tokens in source document and summary (i.e.,  $\min_{A_s} \left( \frac{A_{sj}^l}{v_s - v_{s-1} - 1} \right)$ ) as the attention score between the  $i$ -th token in target and the attended tokens in the source. Equation (5) now becomes<sup>d</sup>

$$\tilde{c}_i^l = \sum_{j=1}^m \tilde{a}_{ij}^l (z_j^u + X_j) \tag{10}$$

By using segment-wise attention and revising attention scores of attended POIs directly, our model optimizes the attention distribution between the encoder states and decoder states in such a way that the alignment relationship between source document and summary is enhanced, and noise for attention from encoder outputs is reduced. As shown in Table 4, the segments in the example are separated by punctuation. For the basic CNN model, the second and third sentence repeatedly attend to the fifth segment in source document. After applying ATTF model, the attention scores of the third and fifth segment in source document are penalized during generating words in the third sentence of ATTF. The last sentence of the summary generated by ATTF attends to the seventh segment in source.

The ATTF helps avoid repeatedly attending to the same POIs and therefore avoid repetition in summary generation.

### 2.3. Sentence-level backtracking decoder (SBD)

To tackle repeated sentences or phrases in the source (Example 1), we propose a SBD.

At test time, we prevent the decoder from generating identical or very similar sentences more than once via backtracking. An intuitive solution is to backtrack the generation process to the beginning of the repeated segment and regenerate it by following the second best choice in the beam. We call this simple approach **SBD-b1**. However, this is suboptimal because the parents of the current top  $b$  choices may not include all the top  $b$  choices at the parent level. Here  $b$  is the

<sup>d</sup>After Equation (9), an alternative way to get  $\tilde{c}_i^l$  is to use the re-normalized filtered attention scores  $\tilde{a}_{ij}^l$ . We re-normalize the filtered attention scores by  $\tilde{a}_i^l = \text{softmax}(\tilde{a}_i^l)$  where  $\tilde{a}_i^l = (\tilde{a}_{i1}^l, \tilde{a}_{i2}^l, \dots, \tilde{a}_{iD}^l)$ . Then, Equation (10) becomes  $\tilde{c}_i^l = \sum_{j=1}^m \tilde{a}_{ij}^l (z_j^u + X_j)$ .

**Table 4.** Summary generated by the basic CNN model and ATTF model. The *segments* of summary use the same color as their attended *section* in the source document

Source document	
	(1)justin timberlake and jessica biel, (2)welcome to parenthood. (3)the celebrity couple announced the arrival of their son, (4). . . (5) the couple announced the pregnancy in january, (6). . . (7)it is the first baby for both.
Basic CNN model (CNN)	(1)the couple announced the the arrival of their son. (2)the couple announced the pregnancy in january. (3)the couple announced the pregnancy in january.
ATTF (our)	(1)the couple announced the arrival of their son. (2)the couple announced the pregnancy in january. (3)it is the first baby for both.

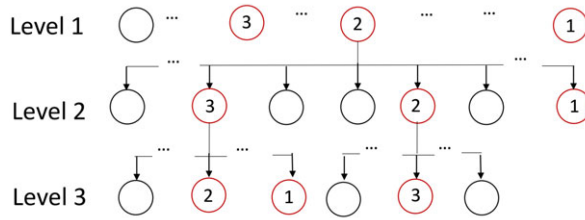
**Table 5.** Summary generated by basic CNN model with different backtracking methods

Source document	
	justin timberlake and jessica biel, welcome to parenthood. the celebrity couple announced the arrival of their son, silas randall timberlake, . . . “ silas was the middle name of timberlake ’s maternal grandfather bill bomar, who died in 2012, . . . the couple announced the pregnancy in january, with an instagram post. it is the first baby for both.
Basic CNN model (CNN)	the couple announced the arrival of their son. the couple announced the pregnancy in january. <i>the couple announced the pregnancy in january. (repeated segment)</i>
CNN+TRI	the couple announced the arrival of their son. <u>silas randall timberlake , who died in 2012.</u>
CNN+SBD-b1	the couple announced the arrival of their son. the couple announced the pregnancy in january. <u>silas was the middle name of timberlake ’s maternal grandfather.</u>
CNN+SBD-b2	the couple announced the arrival of their son. <u>silas randall timberlake , died in 2012.</u>
CNN+SBD	the couple announced the arrival of their son. they announced the pregnancy in january, with an instagram post.

beam size. As shown in Figure 4, suppose that level 3 is the beginning of the repeated segment, the first choices at level 1 and 2 are excluded by beam search.

An alternative approach (SBD-b2) backtracks all the way until the current top *b* choices all share the same prefix token sequence. This means that the current best choices in the beam reach some consensus that the generated prefix summary is good and should be retained. While this algorithm backtracks further and may include better choices, it does not completely solve the problem of SBD-b1. As shown in Figure 4, suppose that level 3 is the beginning of the repeated segment and the second choice in level 1 is the only prefix token sequence of top *b* choices in level 2, the first and third choices at level 1 are excluded by beam search after generating words based on the second choice in level 1.

Our best approach (SBD) backtracks to the beginning of the whole summary and regenerates all the choices in the beam up to the point before the repeated segment. That way, all the best choices are known to the algorithm and we can make an optimal choice after excluding the first word of the previously repeated segment. As shown in Table 5, SBD-b1 and SBD-b2 backtrack



**Figure 4.** Backtracking in Beam Search ( $b = 3$ ). This figure shows the progress of generating summary at test. The circles denote the candidate words (choices) in vocabulary, which are sorted by the probability of being selected in descending order. Each circle at level  $l$  has  $N$  choices at level  $l + 1$ .  $N$  is the number of words in vocabulary. The number in circles is the order of these choices according to the probability. The generation order is from level 1 (top) to level 3 (bottom).

the generator process to “january.” and “son.” respectively. The summaries generated by SBD-b1 and SBD-b2 are incoherent and inconsistent with the source document. Our best approach (SBD) will save the sequence before repeated segment, that is, “*the couple announced the arrival of their son. the couple announced the pregnancy in january.*” and backtrack to the beginning of the summary and regenerate the summary. When the saved sequence appears in the beam, we remove the first word (“the”) in repeated segment from the choices vocabulary. Compared with SBD-b1 and SBD-b2, SBD generates more fluent and coherent summaries.

To determine whether two sentences,  $p$  and  $q$ , are similar, we define a boolean function as:

$$sim(p, q) = \begin{cases} 1 & \text{if } o(p, q) > n \text{ OR } o(p, q) > \frac{1}{2} \cdot l \\ 0 & \text{others} \end{cases} \tag{11}$$

where  $o(p, q)$  denotes the length of the longest common substring (LCS) between  $p$  and  $q$ ,  $l$  is the minimum of the lengths of  $p$  and  $q$ , and  $n$  is a constant.  $sim(p, q) = 1$  means the two sentences are similar.

This method cooperates with ATTF in reducing repetition caused by the noises in dataset. Compared with TRI, SBD does not interrupt the beam search process in the middle of a sentence, hence significantly reducing related grammatical and factual errors. As shown in Table 5, the summary generated by SBD is grammatical and factual. Besides, SBD is capable of producing a more informative summary since it yields more chances to other candidate sentences.

### 3. Evaluation

In this section, we introduce the experimental set-up and analyze the performance of different models.

#### 3.1. Datasets

CNN/Daily Mail (Hermann *et al.* 2015)<sup>e</sup> is a popular summarization dataset, which contains news articles paired with summaries. There are 286,817 training pairs, 13,368 validation pairs, and 11,487 test pairs. Table 1 shows an example pair from training data. We follow See *et al.* (2017) in data preprocessing and use the non-anonymized version, which fills in the blanks with answer named entities.

Also, we tried our model on other two abstractive summarization datasets about news, which are Newsroom (Grusky *et al.* 2018) and DUC 2002<sup>f</sup>. For Newsroom, there are 1,321,995 document–summary pairs, which are divided into training (76%), development (8%), test (8%),

<sup>e</sup><https://cs.nyu.edu/~kcho/DMQA/>

<sup>f</sup><https://www-nlpir.nist.gov/projects/duc/guidelines/2002.html>

---

**Algorithm 1** Calculation of Total Repeatedness

---

**Input:** a sentence set  $s = s_1, s_2, \dots, s_n$ **Output:** Total repeatedness percentage  $p$ 

- 1: Let *total* be the sum of lengths of the sentences in  $s$ .
  - 2:  $n \leftarrow total$
  - 3:  $overlap \leftarrow 0$
  - 4: **while**  $n \geq 3$  **do**
  - 5:   The lengths of LCS between two sentences from  $s$  comprise a length set  $L$ .
  - 6:    $n \leftarrow \max(L)$ .
  - 7:   Find a substring  $b$  with length  $n$  that appears most frequently in  $s$ .
  - 8:   Let  $k$  be the frequency that  $b$  appears in  $s$ .
  - 9:    $overlap \leftarrow overlap + k \cdot n$
  - 10:   Remove every appearance of substring  $b$  from sentences in  $s$ .
  - 11: **end while**
  - 12:  $p \leftarrow overlap/total$
  - 13: **return**  $p$
- 

and unreleased test (8%). At testing, we use 8% released test data. DUC-2002 (DUC) is a test set of document–summary pairs. We use the models trained on CNN/Daily Mail to do the test on DUC and demonstrate the generalization of the models.

**3.2. Model parameters and evaluation metrics**

In the following experiments, we tokenize source documents and targets using the word tokenization method from NLTK (Natural Language Toolkit). The NLTK module is a massive toolkit, aimed at helping with the entire natural language processing (NLP) methodology. All the competing models contain eight convolutional layers in both encoders and decoders, with kernel width of 3. For each convolutional layer, we set the hidden state size to 512 and the embedding size to 256. To alleviate overfitting, we apply a *dropout* ( $p = 0.2$ ) layer to all convolutional and fully connected layers. Similar to Gehring *et al.* (2017), we use Nesterov’s accelerated gradient method (Sutskever *et al.* 2013) with gradient clipping 0.1 (Pascanu *et al.* 2013), momentum 0.99, and initial learning rate 0.2. Training terminates when learning rate  $\leq 10e - 5$ . Beam size  $b = 5$  at test time.

We set the threshold  $\beta$  to 3, because nearly 90% of sections are with length  $\geq 3$ . We set  $n$  (Equation (11)) to 5, since less than 5% of reference summaries have the LCS of less than 5. We use the following evaluation metrics:

- **ROUGE** scores (F1), including ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L(R-L) (Lin 2004). ROUGE-1 and ROUGE-2, respectively, refer to the overlap of unigram (each word) and bigrams between the generated summaries and reference summaries. ROUGE-L denotes Longest Common Subsequence (LCS)-based statistics. ROUGE-2 is the most popular metric for summarization.
- **Repeatedness** (Rep) includes N-gram repeatedness, sentence repeatedness, and total repeatedness, which reflects the effectiveness of different methods on repetition reduction.

– **N-gram repeatedness** is the percentage of repeated N-grams in a summary:

$$Rep_{Ngram} = \frac{n_{Ngram}}{N_{Ngram}} \quad (12)$$

where  $n_{ngram}$  is the number of repeated N-grams and  $N_{ngram}$  is the total number of N-grams in a summary.

- **Sentence repeatedness** is the percentage of repeated sentences in a summary:

$$Rep_{sent} = \frac{n_{sent}}{N_{sent}} \tag{13}$$

where  $n_{sent}$  is the number of repeated sentences and  $N_{sent}$  is the total number of sentences in a summary. For sentence repeatedness, if the sentences contain the same trigram, these sentence are repetitive sentences.<sup>8</sup>

- **Total repeatedness** (Algorithm 1) is a comprehensive score that unifies word-level and sentence-level repeatedness. It is not computed by N-gram repeatedness score and sentence repeatedness score.
- **Repeatedness Correlation** measures how well the total repeatedness scores of summaries generated by each model correlate with total repeatedness scores of reference summaries. The more the correlative generated summary and reference summary are, the better generated summary is. The correlation is evaluated with a set of three metrics, including Pearson correlation ( $r$ ), Spearman rank coefficient ( $\rho$ ), and Kendall’s tau coefficient ( $\tau$ ). Given total repeatedness scores of reference summaries (ref) and their corresponding generated summaries (gen),  $X = score(ref) = (x_1, x_2, \dots, x_n)$  and  $Y = score(gen) = (y_1, y_2, \dots, y_n)$ , we can get paired data  $(X, Y) = (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ .  $n$  is the number of pairs.

- For Pearson correlation ( $r$ ),

$$r = \frac{\sum_{i=1}^n (x_i - \bar{X})(y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{X})^2 \cdot \sum_{i=1}^n (y_i - \bar{Y})^2}} \tag{14}$$

where  $\bar{X}$  and  $\bar{Y}$  are the mean of variables of  $X$  and  $Y$ .

- For Spearman rank coefficient,

$$\rho = \frac{\sum_{i=1}^n (R(x_i) - \overline{R(X)})(R(y_i) - \overline{R(Y)})}{\sqrt{\sum_{i=1}^n (R(x_i) - \overline{R(X)})^2 \cdot \sum_{i=1}^n (R(y_i) - \overline{R(Y)})^2}} \tag{15}$$

where  $R(x_i)$  and  $R(y_i)$  are the rank of  $x_i$  and  $y_i$ .  $\overline{R(X)}$  and  $\overline{R(Y)}$  are the mean rank of  $X$  and  $Y$ .

- For Kendall’s tau coefficient,

$$\tau = \frac{n_c - n_d}{n_c + n_d} = \frac{n_c - n_d}{n(n - 1)/2} \tag{16}$$

where  $n_c$  is the number of *concordant* pairs.  $n_d$  is the number of *discordant* pairs. Any pair of total repeatedness scores  $(x_i, y_i)$  and  $(x_j, y_j)$ , where  $i < j$ . They are said to be *concordant*, if both  $x_i > x_j$  and  $y_i > y_j$ ; or if both  $x_i < x_j$  and  $y_i < y_j$ . They are said to be *discordant*, if  $x_i > x_j$  and  $y_i < y_j$ ; or if  $x_i < x_j$  and  $y_i > y_j$ . If  $x_i = x_j$  or  $y_i = y_j$ , the pair is neither concordant nor discordant.

- **Readability** (Readable) is a human evaluation, which can be used as the supplement to ROUGE. We educate human annotators to assess each summary from four independent perspectives:

<sup>8</sup>Any two sentences in one reference summary almost never contain the same trigram (Paulus *et al.* 2018).

- (1) Informative: How informative the summary is? Is the summary logically consistent with source document?
- (2) Coherent: How coherent (between sentences) the summary is?
- (3) Fluent: How grammatical the sentences of a summary are?
- (4) Factual: Are there any factual errors in the summary?

Readability score will be judged on the following five-point scale: Very Poor (1.0), Poor (2.0), Barely Acceptable (3.0), Good (4.0), and Very Good (5.0). The score reflects the fluency and readability of the summary.

We use *readability* to complement ROUGE scores, since Yao *et al.* (2017) showed that the standard ROUGE scores cannot capture grammatical or factual errors. We randomly sample 300 summaries generated by each model and manually check their readability. Each summary is scored by four judges proficient in English. The Cohen's Kappa coefficient between them is 0.66, indicating agreement. Here we use the average annotation score.

### 3.3. Baselines

Our goal is to evaluate the effectiveness of our repetition reduction technique. We need to select a basic model and implement different repetition reduction methods on top of it. After applying different repetition reduction methods, the basic model should be able to largely reflect the difference in the effectiveness of these repetition reduction methods. The basic model also needs to have higher training efficiency (higher speed and less memory usage) without being limited by computing resources while ensuring the quality of generation.

We choose to implement all existing repetition reduction techniques on top of vanilla CNN seq2seq model, because the vanilla CNN seq2seq model is fast and enjoys the best accuracy among the other vanilla RNN seq2seq models such as RNN seq2seq model and LSTM seq2seq model (Bai *et al.* 2018; Gehring *et al.* 2017). The vanilla CNN seq2seq model and vanilla self-attention-based model have similar feature capture capabilities. With long inputs, the self-attention-based models will have greater computational complexity (Vaswani *et al.* 2018), such as Transformer. As the inputs of summarization are very long, the self-attention-based models always need much more time during training and testing. Besides, the self-attention-based models contain more training parameters, which need more memory usage at training and testing time.

We did not implement the repetition reduction methods on top of the seq2seq models with higher ROUGE scores, because the effectiveness of the repetition reduction is not necessarily reflected in the ROUGE (See *et al.* 2017; Paulus *et al.* 2018; Fan *et al.* 2018). As shown in Table 6, after reducing repetition, the summary becomes better, but the ROUGE score is not improved. Therefore, our evaluation mainly compares the effectiveness of different repetition reduction techniques in terms of all four metrics above. As known, ROUGE is not very good at evaluating abstractive summarization and the room for improvement on the ROUGE scores is very limited. If the repetition reduction methods were applied on top of the models with higher ROUGE scores, the differences in ROUGE scores by these repetition reduction techniques would be indistinguishable and complicate the analysis. Hence, in this work, we construct seven baselines by converting repetition reduction techniques developed on RNN seq2seq models to their counterparts on vanilla CNN seq2seq models, to be fair. The baselines are as follows:

- CNN is the original convolutional seq2seq model (Gehring *et al.* 2017).
- ITA integrates *intra-temporal attention* (Nallapati *et al.* 2016) in CNN seq2seq model, which normalizes attention values using attention history through timestamps.
- ITDA adds *intra-decoder attention* mechanism (Paulus *et al.* 2018) based on ITA, which also normalizes attention values using past decoders states. It is transferred to CNN seq2seq model in Fan *et al.* (2018).

**Table 6.** Example of generated summaries

(a) Source document and reference summary		
<b>Source</b>		
justin timberlake and jessica biel, welcome to parenthood. the celebrity couple announced the arrival of their son, silas randall timberlake, . . . the couple announced the pregnancy in january, . . . it is the first baby for both.		
<b>Reference</b>		
timberlake and jessica biel welcome son silas randall timberlake. the couple announced the pregnancy in january.		
(b) The generated summaries of source in (a) and their ROUGE scores		
Model	Summary	R-2
<b>COV</b>	timberlake and jessica biel announced the pregnancy in january. the couple announced the pregnancy in january.	0.60
<b>ATTF+SBD</b>	the couple announced the arrival of their son, silas randall timberlake. the couple announced the pregnancy in january. it is the first baby for both.	0.52

- **COV** adopts the coverage mechanism (See *et al.* 2017), where repeatedly attending to the same locations is penalized in the form of *coverage loss*.
- **COVP** adds the coverage penalty (Gehrmann *et al.* 2018) to loss function, which increases whenever the decoder repeatedly attends to the same locations of source document.
- **SCL** adds semantic cohesion loss (Çelikyilmaz *et al.* 2018) to loss function. Semantic cohesion loss is the cosine similarity between two consecutive sentences.
- **DivCNN** uses DPPs methods (Micro DPPs and Macro DPPs) to produce attention distribution (Li *et al.* 2019a). DPPs consider both quality and diversity, which helps model attend to different subsequences in source document.
- **TRI** uses *trigram decoder* (Paulus *et al.* 2018) at testing. The generation of repetitive trigrams is banned during beam search.

### 3.4. Results

**Segmentation.** As shown in Table 3, we can get segments from the documents and summaries in three ways: *sentence-level segment*, *N-gram segment*, and *punctuation-based segment*.

Table 7 shows the results of ATTF trained using different segmentation methods. The ATTF trained by punctuation-based segments performs best in terms of all evaluation metrics. The ROUGE scores and repeatedness scores of these three segmentation methods are similar because they all redistribute the attention distribution and avoid attending to the same segments. The difference is only from the different type of segments. As shown in Table 7, the ATTF trained on sentence-level segments achieve a higher readability score than the ATTF trained on N-gram segments, and the ROUGE scores of sentence-level segmentation is lower than N-gram segmentation. The former is because N-gram segments may cause grammatical and semantic problems and the latter is because the model with sentence-level segment may lose the important information, as shown in Table 3.

**Table 7.** ROUGE scores, total repeatedness scores (%), and readability scores of the summaries in CNN/Daily Mail dataset generated by the ATTF model using sentence-level, N-gram, and punctuation-based segments

	R-1	R-2	R-L	Rep	Readable
Sentence-level	35.44	15.25	35.68	3.45	3.86
N-gram	35.52	15.21	35.90	3.36	3.70
Punctuation-based	<b>36.32</b>	<b>15.38</b>	<b>36.09</b>	<b>3.27</b>	<b>4.42</b>

**Table 8.** ROUGE scores on CNN/Daily Mail dataset. In our experiments, \* means that the model applies the repetition reduction methods to the decoder during test

(a) The models without operations at test.				(b) The models with operations at test.			
Model	R-1	R-2	R-L	Model	R-1	R-2	R-L
CNN	34.33	14.25	35.68	SBD-b1*	34.24	14.33	34.75
ITA	34.30	14.20	35.67	SBD-b2*	35.88	14.83	35.15
ITDA	34.62	14.52	35.94	SBD*	37.19	15.45	36.03
COV	35.85	14.81	35.96	TRI*	36.81	15.47	36.00
COVP	34.53	14.41	35.81	ATTF+TRI*	37.33	16.65	36.30
SCL	35.13	14.61	35.93	ATTF+SBD*	<b>37.69</b>	<b>17.02</b>	<b>36.47</b>
DivCNN	35.64	15.01	35.84				
ATTF	<b>36.32</b>	<b>15.38</b>	<b>36.09</b>				

**Accuracy.** As shown in Table 8, our model (ATTF+SBD) outperforms all the baselines in ROUGE scores, indicating we are able to generate more accurate summaries.

Without any special operations at testing, our ATTF model achieves the highest score on ROUGE, showing its effectiveness in improving summary quality. Models with SBD or TRI at testing are more effective than the basic CNN seq2seq model because more information is involved in summary generation as a byproduct of repetition reduction. Compared with its two variants, SBD is a little slower but has higher ROUGE scores, reflecting its advantage due to better choices taken globally. Therefore, we use SBD as our backtracking decoder in the following experiments. The number of explored candidate hypotheses, up to a point of repetition, is less than 30 tokens. The ROUGE score of SBD is higher than TRI on R-1 and R-L, but lower on R-2. The reason is that R-2 and R-L, respectively, evaluate bigram-overlap and longest common sequence between the reference summary and generated summary. This is in line with different techniques in SBD and TRI, the former promoting the diversity of sentences, and the latter promoting that of trigrams. SBD has higher ROUGE scores than ATTF, because the summaries from SBD do not have the repetition caused by attending to similar sentences in source. Unlike ATTF, SBD cannot obtain the ability to attend to different POIs through training. In Table 10, the sentences in SBD are not repetitive but summarized from the same POI. The summaries may lose important information when only using SBD. The readability score of SBD is lower than ATTF in Table 9.

For models that tackle repetition both at training and test time, ATTF+SBD outperforms ATTF+TRI. SBD works in synergy with ATTF, and they together process information with *section/segment* as a unit. ATTF+SBD scores higher ROUGE than the other baselines, demonstrating its power to reduce repetition and generate more accurate summaries. Besides, as shown in Table 6, the quality of a summary cannot be evaluated by ROUGE scores alone. ATTF+SBD



**Table 9.** Repeatedness scores (%) and readability scores on CNN/Daily Mail dataset. The “Gold” denotes reference summaries, which are the most readable. By default, the readability score of reference summaries is judged to be 5.0

(a) The models without operations at test.									
	Gold	CNN	ITA	ITDA	COV	COVP	SCL	DivCNN	ATTF
1-gram	33.79	56.25	54.44	51.18	42.18	52.46	52.23	38.43	<b>34.98</b>
2-gram	2.98	36.55	34.76	30.64	16.77	32.11	34.08	12.62	<b>8.16</b>
3-gram	0.43	32.62	31.10	27.14	12.95	28.59	30.58	10.15	<b>5.11</b>
4-gram	0.12	30.18	28.85	25.04	11.17	26.48	28.34	9.01	<b>4.19</b>
Sent	3.98	49.45	48.34	42.96	14.52	25.52	27.58	8.77	<b>6.69</b>
Total-Rep	0.51	18.86	17.94	15.62	7.77	16.46	17.65	10.37	<b>3.27</b>
Readable	5.0	2.95	3.18	3.46	3.66	3.75	3.70	3.65	<b>4.42</b>

(b) The models with operations at test.					
	Gold	TRI*	SBD*	ATTF+TRI*	ATTF+SBD*
1-gram	33.79	31.91	<b>29.88</b>	32.0	30.83
2-gram	2.98	3.17	<b>2.84</b>	2.94	3.71
3-gram	0.43	<b>0.0</b>	<b>0.40</b>	<b>0.0</b>	<b>0.74</b>
4-gram	0.12	<b>0.0</b>	<b>0.06</b>	<b>0.0</b>	<b>0.13</b>
Sent	3.98	<b>0.0</b>	<b>3.47</b>	<b>0.0</b>	<b>3.44</b>
Total-Rep	0.51	<b>0.0</b>	<b>0.44</b>	<b>0.0</b>	<b>0.80</b>
Readable	5.0	3.62	3.87	3.75	<b>4.57</b>

**Table 10.** Summaries generated from Example 1

<b>Reference:</b> oriol romeu is on a season-long loan at stuttgart from chelsea. the spanish midfielder predicts the scores in saturday’s matches. oriol goes head-to-head with sportmail’s martin keown.
<b>ATTF:</b> chelsea beat manchester united on saturday. <i>oriol romeu is currently on a season-long loan at stuttgart. oriol romeu is currently on a season-long loan at bundesliga side stuttgart.</i>
<b>SBD*:</b> chelsea beat manchester united on saturday. chelsea face manchester united in the premier league.
<b>ATTF+SBD*:</b> chelsea face manchester united in the premier league on saturday. <i>oriol romeu is currently on loan at stuttgart.</i>

obviously produces a better, logically more consistent summary despite a lower ROUGE score. Due to the variable nature of abstractive summarization, ROUGE is not the optimal evaluation metric. Repeatedness and readability score, in our opinion, are important complementary metrics to ROUGE scores.

**Repeatedness.** To demonstrate the effectiveness of ATTF and SBD in reducing repetition, we compare *repeatedness* (Table 9) of generated summaries. Lower repeatedness means the model is more capable of reducing repetition. In Table 9, Gold row shows the repeatedness scores of reference summaries. ATTF achieves the lowest score among all baselines without any operations at test time. As shown in Tables 1 and 2 and Figure 5, baseline models suffer from severe repetition

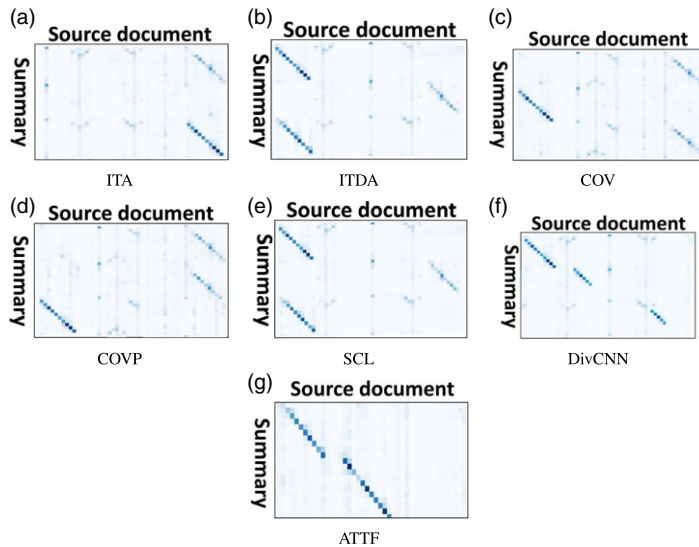


Figure 5. Attention distribution of summaries for the source in Table 1.

problem because they attend to the same POIs of the source document. DivCNN adjusts attention distribution in an indirect manner that adds the attention of the subsets (with high-quality diversity score) selected from source document into the loss. Thus, DivCBNN may still attend to similar but different sentences, resulting in lower ROUGE scores and higher repeatedness. Besides, DivCNN is trained to attend to diversity subsets, which means that the selected subsets are more scattered (as shown in Figure 5) and lead to semantic incoherence. However, ATTF attends to different POIs and generates summaries such as this:

**ATTF:** manchester city are rivalling manchester united and arsenal for defender dayot pamecano. the 16-year-old joined in the january transfer window only for him to opt to stay in france.

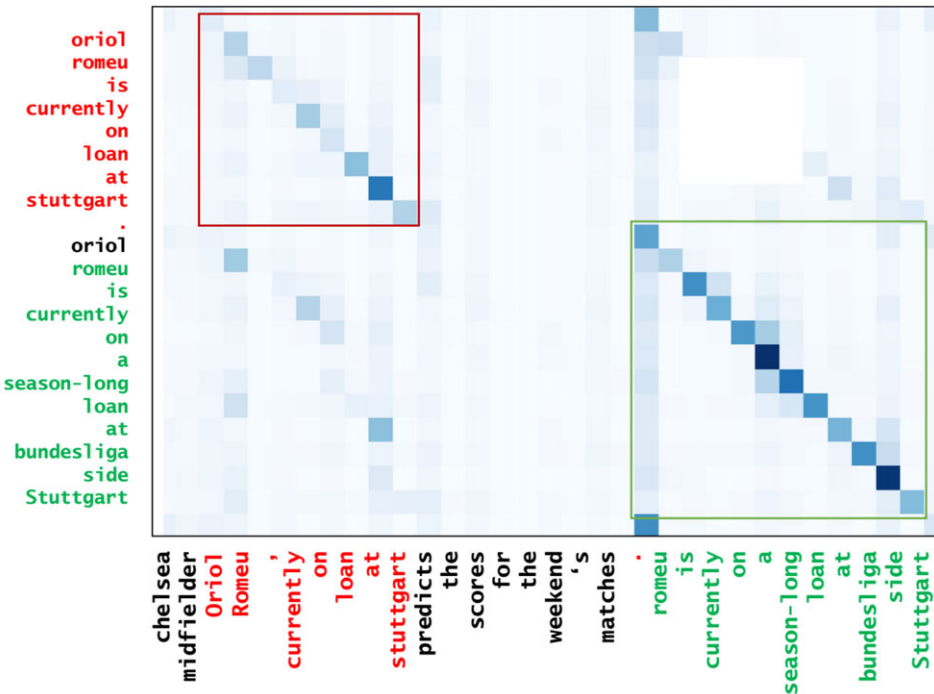
Compared with the Gold standard, ATTF still generates some repetitive sentences due to similar sentences in source such as Example 1. The summary generated by ATTF and its local attention are shown in Table 10 and Figure 6. Also, SBD further reduces the repetition when combined with ATTF.

As shown in Table 9, TRI has the lowest total repeatedness score. It does not generate any repetitive N-grams ( $N > 2$ ) and sentences because TRI prevents the generation of the same trigrams during testing. But as the Gold row shows, reference summaries do have some natural repetition. Therefore, we evaluate the correlation of repeatedness distribution between generated summaries and reference summaries (Table 11(a)). Our proposed models perform best, which indicates that ATTF and SBD are more capable of producing summaries with a natural level of repeatedness. In addition, as shown in Table 11(b), the correlations between the repeatedness and the human readability judgment are about 0.7, which means that the repeatedness score is useful. The repetition in summaries will affect coherence between sentences and the readability of summaries.

**Readability.** As shown in Table 9, the models with ATTF achieve the highest readability score among all baselines, which means ATTF is more readable. As shown in Table 9(b), TRI achieves the best scores on repeatedness, but lower readability scores than other models. Readability is a human evaluation metric considering the logical correctness. (See Section 3.2). As shown in Table 9 and Table 13, the Readable scores of the models with TRI are lower than the models with SBD, which indicates the effectiveness of SBD on logical correctness. Specially, after using TRI, the readability of ATTF+TRI becomes lower than ATTF. This means that the TRI will bring

**Table 11.** Correlation Evaluation ( $r$  = Pearson,  $\rho$  = Spearman and  $\tau$  = Kendall's tau)

(a) Repeatedness correlation between the generated summaries and Gold summaries.				(b) The correlation between the repeatedness of the generated summaries and the human readability judgment.			
	$r$	$\rho$	$\tau$		$r$	$\rho$	$\tau$
ATTF	<b>1.0</b>	<b>1.0</b>	<b>1.0</b>	ATTF	0.78	0.74	0.70
TRI*	1.0	0.89	0.84	SBD*	0.75	0.71	0.68
SBD*	<b>1.0</b>	<b>1.0</b>	<b>1.0</b>	ATTF+SBD*	0.75	0.74	0.69
ATTF+TRI*	1.0	0.89	0.84				
ATTF+SBD*	<b>1.0</b>	<b>1.0</b>	<b>1.0</b>				



**Figure 6.** Attention distribution for ATTF in Table 10.

logical incorrectness to the generated summaries. TRI interrupts the process of sentence generation during beam search through trigrams that cannot reflect the complete grammatical structure and semantic information. TRI is likely to generate summaries with more grammatical and factual errors. SBD forbids the repetition at sentence-level during testing, which considers complete grammatical structure and semantic information. As shown in Table 2 and Table 5, SBD weakens the influence of the meddling of sentence generation during beam search and generates more readable summaries. The higher ROUGE scores show that SBD enhances the performance of CNN and ATTF by reducing the repetitive unreadable sentences.

**Speed.** We compare the speed of our model on CNN/Daily Mail to RNN (See *et al.* 2017), FastRNN (Chen and Bansal 2018) and Transformer-large (Vaswani *et al.* 2017) which used K40. We perform experiments on GTX-1080ti and scale the speed reported for the RNN methods, since GTX-1080ti is twice as fast as K40 (Gehring *et al.* 2017).

**Table 12.** Time and speed of training and testing. The training time of the model with SBDs is empty because SBDs are only used during testing

Model	Train		Test	
	Time (h)	Time (s)	summaries/s	tokens/s
RNN	336.54	21600	0.48	29.60
FastRNN	98.4	3600	2.92	219.60
Transformer	115.44	2094.6	5.65	214.83
CNN	48.3	346.1	30.36	1343.46
SBD-b1*	-	412.8	25.46	1126.38
SBD-b2*	-	843.5	12.16	551.24
SBD*	-	912.8	11.51	493.68
ATTF	108	1332	7.89	349.00
ATTF+SBD*	-	1832.3	5.74	253.77

As shown in Table 12, the CNN is faster than Transformer based on multi-head self-attention mechanism. In terms of computational complexity, Vaswani *et al.* (2017) show that the per-layer complexity of self-attention is  $O(n^2d)$  and the per-layer complexity of CNN is  $O(knd^2)$ .  $n$  is the sequence length,  $d$  is the representation dimension, and  $k$  is the kernel width of convolutions. So the difference between the complexity of Transformer and CNN depends on  $n$ ,  $d$ , and  $k$ . In our experiments, we follow Gehring *et al.* (2017) in the experimental setting for CNN and Vaswani *et al.* (2017) for Transformer. Both experimental settings are standard settings of the vanilla models. As the average sequence length of source documents in our datasets is more than 500, the  $n$  in the complexity of CNN and Transformer is greater than 500. The representation dimension of CNN,  $d_{cnn}$ , is 256. The representation dimension of Transformer,  $d_{trans}$ , is 1024. The kernel width of CNN is 3. Thus, in our experiment, the speed of CNN is faster. The training speed and testing speed of CNN+ATTF are faster than RNN seq2seq model and Transformer since the training of CNN is parallel. The gap of training/testing time between ATTF and Transformer is not much larger, but the memory usage of Transformer is much larger than ATTF. This is because that Transformer has more trainable parameters than ATTF. The training speed and testing speed of FastRNN are faster than RNN because FastRNN is not an end-to-end model. FastRNN is a two-stage framework, which first uses an extractor to extract several salient sentences from source document and then uses an abstractor to summarize each salient sentence. The final summary is the concatenation of these summarized salient sentences. The extractor in FastRNN is a pointer network with a sequence of sentences as input. The encoder of extractor is the same as RNN. The FastRNN adopts RNN as abstractor and trains extractor and abstractor in parallel, which speeds up the encoder and decoder of RNN seq2seq model by shortening the input and output. As an end-to-end model, the input and output of our CNN+ATTF model are sequences of words in complete source document and summary, which are much longer than the input and output of the extractor and abstractor of FastRNN. The training speed of CNN+ATTF is similar to FastRNN as the CNN can be trained in parallel. The testing speed of CNN+ATTF is faster than FastRNN because FastRNN should extract sentences first and then abstract each sentence during test.

The convergence rate of models with ATTF depends on three aspects: *dataset*, *basic model*, and *experimental settings*. For *dataset*, ATTF redistributes the attention distribution between source documents and summaries during decoding, which dynamically searches the attended segment in source by the predicted segments in summary. Thus, the convergence rate of the models with

**Table 13.** ROUGE scores, total repeatedness (Rep), and readability on Newsroom and DUC

Model	Newsroom					DUC				
	R-1	R-2	R-L	Rep	Readable	R-1	R-2	R-L	Rep	Readable
CNN	27.43	18.62	26.77	20.38	2.65	26.02	8.76	21.05	19.32	2.32
TRI*	27.96	19.83	27.01	<b>0.0</b>	2.84	26.75	9.34	22.19	<b>0.0</b>	2.51
SBD*	28.20	20.17	26.94	0.62	3.02	26.86	9.27	22.75	0.43	2.85
ATTF	28.43	20.05	27.32	1.53	3.42	26.94	9.50	22.33	1.72	3.11
ATTF+SBD*	<b>28.93</b>	<b>21.46</b>	<b>27.55</b>	0.58	<b>3.73</b>	<b>27.02</b>	<b>9.56</b>	<b>23.05</b>	0.40	<b>3.5</b>

ATTF depends on the length of source documents and summaries. The ATTF applied on better *basic models* converges faster, because better basic models learn the alignment between source documents and summaries better. The *experimental setting* also impacts the convergence rate of the model with ATTF. At the beginning of training, a large learning rate makes the model converge faster. When most of the samples have been trained, the model converges rapidly by decreasing the learning rate.

As shown in Table 8 and Table 12, SBD is the best SBD. Compared with SBD-b1 and SBD-b2, SBD logs higher ROUGE scores without losing much on speed. ATTF+SBD achieves the best ROUGE scores and its training time and testing time do not increase too much.

**Generalization.** Table 13 shows the generalization of our abstractive system to other two datasets, Newsroom and DUC 2002, where our proposed models achieve better scores than vanilla CNN seq2seq model in terms of ROUGE scores, readability and repeatedness. We use the same settings of  $\beta = 3$  in Section 2.2.2 and  $n = 5$  in Equation (11), because the proportion of segments with length greater than 3 and reference summaries with LCS greater than 5 were about 90%. As shown in Table 13, our proposed models can generalize better on other datasets about news, along with repetition reduction and the improvement of readability. This shows that our proposed models can be generalized well.

**Normalization.** The attention scores of the basic attention mechanism without filtering satisfy the probability distribution. As for the filtered attention scores of ATTF, we penalize the attention scores of the tokens that have been attended to in the source document and keep the attention scores of other tokens in the source document the same. In this way, we can keep the difference of attention scores between the tokens that have not been attended to, which may avoid ignoring some source contents that need to be summarized. After re-normalizing the filtered attention scores, the decoder tends to attend to the tokens of the source document with high filtered attention scores. It can also prevent the attention scores of the tokens that have not been attended to from too small.

As shown in Table 14, the R-2 F1 scores of ATTF and ATTF with re-normalized attention scores (Norm ATTF) are similar over all datasets. ROUGE recall measures how well a generated summary matches its corresponding reference summary by counting the percentage of matched n-grams in the reference. ROUGE precision indicates the percentage of n-grams in the generated summary overlapping with the reference. ATTF is always better than Norm ATTF on R-2 recall, and Norm ATTF is better than ATTF on R-2 precision. As shown in Table 14(b), since the difference of attention scores is not magnified by normalization, the summary generated by ATTF can more comprehensively attend to the information in the source. However, when the attention scores of the tokens that have not been attended to are too small, the decoder may attend to less important information of the source, such as “traffic and highway patrol. . . .” (ATTF row)

**Table 14.** The comparison of ATTF and the re-normalized ATTF (Norm ATTF)

(a) The recall, precision and F1 score of R-2.									
Model	CNN/Daily Mail			Newsroom			DUC		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
-10 ATTF	14.71	16.65	15.38	19.23	23.82	20.05	7.88	11.48	9.50
Norm ATTF	16.18	14.47	15.32	22.94	19.44	19.97	9.82	9.07	9.54

(b) The example of generated summaries.	
<b>Source document</b>	
. . . . . police have been left stunned after coming across a car parked on a busy street covered in hundreds of chili peppers. <u>traffic and highway patrol command shared the bizarre image on their facebook page</u> , showing a silver car on the side of the road laden down with a vast amount of large red chillies. according to several commenters on the post, the incident occurred on bridge st in hornsby on the upper north shore of sydney and <u>is a regular occurrence</u> . . . . .	
<b>Reference summary</b>	
police have come across a car covered in chili peppers. the car was parked on the side of a busy street in hornsby. according to facebook users, it is a regular occurrence.	
ATTF	police have come across a car parked on a busy street covered in hundreds of chili peppers. <u>traffic and highway patrol command shared the image on their facebook page</u> . The incident occurred on bridge st in hornsby and <u>is a regular occurrence</u> .
Norm ATTF	police have come across a car parked on a busy street covered in hundreds of chili peppers. The incident occurred on bridge st in hornsby.

in Table 14(b). The summary generated by Norm ATTF is likely to miss some important information, such as “it is a regular occurrence.”, due to the magnified differences between filtered attention scores.

**Comparison of ATTF and TRI.** In our experiments, the TRI is the basic CNN seq2seq model with trigram decoder during test. For ROUGE scores, as shown in Table 8 and Table 13, ATTF gets lower ROUGE scores than TRI on CNN/Daily Mail and higher ROUGE scores on Newsroom and DUC. For repeatedness scores, as shown in Table 9 and Table 13, the difference between ATTF and TRI in Newsroom is smaller than that between ATTF and TRI in CNN/Daily Mail. As some of the source documents in CNN/Daily Mail have the similar but different segments (as shown in Figure 6), ATTF may attend to such segments and generate summaries with repetition. The summaries with repetition always achieve lower ROUGE scores. The better performance of ATTF on Newsroom and DUC indicates the ATTF is more effective than TRI in the case of the datasets without repetition in source document. As shown in Table 11, the repeatedness correlation scores of ATTF are higher than TRI, which indicates the summaries generated by ATTF are more similar to human-written summaries. Besides, the readability scores of ATTF are better than TRI on all datasets, which means that the summaries generated by the attention-based modification are more fluent and readable than simple trigram blocking.

**Significance Test.** We use significance test to prove that the ROUGE scores in Table 8 are reliable. We take t-test (Loukina *et al.* 2014) as our significance test to measure that the ROUGE scores between our proposed approach (ATTF+SBD) and each baseline are significant or not. As shown in Table 15, all p-values are less than 0.05. The smaller the p-value, the higher the significance. Thus, the difference in the similarity results is significant.

**Table 15.** p-value of significance test between our best proposed model (ATTF+SBD) and baselines on ROUGE scores

Model	CNN	ITA	ITDA	COV	COVP	SCL	DivCNN	TRI
R-1	2.32e-35	6.14e-34	2.76e-32	4.14e-30	2.51e-32	3.11e-32	2.28e-31	5.25e-30
R-2	6.34e-48	2.12e-48	4.52e-44	4.61e-50	3.17e-41	3.29e-44	4.27e-40	1.33e-43
R-L	3.68e-10	5.67e-12	3.94e-12	7.12e-12	2.15e-10	3.43e-15	6.81e-10	3.67e-12

Overall, the summaries generated by sequence-to-sequence models with attention mechanism always contain repetition. Through our observations, there are two reasons for repetition in abstractive summarization. One is that the traditional attention mechanisms attend to the same location in source document at decoding. The other is that the attention mechanism attends to the repetitive sentences in different locations in source document. As shown in Figure 5 and Table 10, our proposed ATTF and SBD effectively mitigate the above two problems. As ATTF deals with incorrect attention distribution between the inputs of encoder and decoder to reduce repetition in generated summaries, the seq2seq models with attention mechanism between encoder and decoder can be improved via ATTF. The SBD can only be used at testing, which is suitable for the models with decoder. Since RNN-based and transformer-based seq2seq models, including attention mechanism between encoder and decoder, always suffer from repetition in generated summaries, we can reasonably deduce that these models will benefit from our proposed ATTF and SBD as well. The higher ROUGE scores (Table 8) of our model mean that the summaries generated by our model are more similar to their corresponding reference summaries. The natural-level repeatedness and higher readability score (Table 9) of our model indicate that our model can produce summaries with higher quality. ATTF is applied to the attention mechanism between encoder and decoder, which impacts the time of decoding at training and testing. SBD only impacts the time of decoding during test. ATTF+SBD takes about the same amount of time for additional models to slow down. For RNNs and transformers, after adding ATTF+SBD, there would be less than six times slowdown (as shown in Table 11, for the vanilla CNN, there is a roughly six times slowdown after adding ATTF+SBD), since RNNs and transformers spend more training time and testing time on encoding than CNN. As a result, our model can improve the reading speed and accuracy of reading comprehension.

#### 4. Related work

In this section, we discuss neural-based abstractive summarization and some previous work on repetition reduction methods in abstractive summarization.

##### 4.1. Neural-based abstractive summarization

Automatic summarization condenses long documents into short summaries while preserving the important information of the documents (Radev *et al.* 2002; Allahyari *et al.* 2017; Shi *et al.* 2021). There are two general approaches to automatic summarization: extractive summarization and abstractive summarization. Extractive summarization selects sentences from the source articles, which can produce grammatically correct sentences (Bokaei *et al.* 2016; Verma and Lee 2017; Naserasadi *et al.* 2019; Zhong *et al.* 2019). Abstractive summarization is a process of *generating* a concise and meaningful summary from the input text, possibly with words or sentences not found in the input text. A good summary should be coherent, non-redundant, and readable (Yao *et al.* 2017). Abstractive Summarization is one of the most challenging and interesting problems in the

field of NLP (Carenini and Cheung 2008; Pallotta *et al.* 2009; Sankarasubramaniam *et al.* 2014; Bing *et al.* 2015; Rush *et al.* 2015; Li *et al.* 2016; Yao *et al.* 2017; Nguyen *et al.* 2019).

Recently, neural-based (encoder–decoder) models (Rush *et al.* 2015; Chopra *et al.* 2016; Nallapati *et al.* 2016; See *et al.* 2017; Paulus *et al.* 2018; Liu and Lapata 2019; Wang *et al.* 2019; Lewis *et al.* 2020; Liu *et al.* 2021) have made some progress for abstractive summarization. Most of them use RNNs with different attention mechanisms (Rush *et al.* 2015; Nallapati *et al.* 2016; See *et al.* 2017; Paulus *et al.* 2018). Rush *et al.* (2015) are the first to apply the neural encoder–decoder architecture to text summarization. See *et al.* (2017) enhance this model with a pointer generator network which allows it to copy relevant words from the source text. RNN models are difficult to train because of the vanishing and exploding gradient problems. Another challenge is that the current hidden state in an RNN is a function of previous hidden states, so RNN cannot be easily parallelized along the time dimension during training and evaluation, and hence training them for long sequences becomes very expensive in computation time and memory footprint.

To alleviate the above challenges, convolutional neural network (CNN) models (Gehring *et al.* 2017; Fan *et al.* 2018; Liu *et al.* 2018; Zhang *et al.* 2019b) are applied into seq2seq models. Gehring *et al.* (2017) propose a CNN seq2seq model equipped with gated linear units (Dauphin *et al.* 2017), residual connections (He *et al.* 2016), and attention mechanism. Liu *et al.* (2018) modify the basic CNN seq2seq model with a summary length input and train a model that produces fluent summaries of desired length. Fan *et al.* (2018) present a controllable CNN seq2seq model to allow users to define high-level attributes of generated summaries, such as source style and length. Zhang *et al.* (2019b) add a hierarchical attention mechanism to CNN seq2seq model. CNN-based models can be parallelized during training and evaluation. The computational complexity of these models is linear with respect to the length of sequences. CNN model has shorter paths between pairs of input and output tokens so that it can propagate gradient signals more efficiently. CNN model enables much faster training and more stable gradients than RNN. Bai *et al.* (2018) showed that CNN is more powerful than RNN for sequence modeling. Therefore, in this work, we choose the vanilla CNN seq2seq model as our base model.

#### 4.2. Repetition reduction for abstractive summarization

Repetition is a persistent problem in the task of neural-based summarization. It is tackled broadly in two directions in recent years.

One direction involves *information selection* or *sentence selection* before generating summaries. Chen and Bansal (2018) propose an extractor–abstractor model, which uses an extractor to select salient sentences or highlights and then employs an abstractor network to rewrite these sentences. Sharma *et al.* (2019) and Bae *et al.* (2019) also use extractor–abstractor model with different data preprocessing methods. All of them can not solve repetition in seq2seq model. Tan *et al.* (2017) and Li *et al.* (2018a); Li *et al.* (2018b) encode sentences using word vectors and predict words from sentence vector in sequential order, whereas CNN-based models are naturally parallelized. While transferring RNN-based model to CNN model, the kernel size and the number of convolutional layers cannot be easily determined when converting between sentences and word vectors. Therefore, we do not compare our models to those models in this paper.

The other direction is to improve the *memory of previously generated words*. Suzuki and Nagata (2017) and Lin *et al.* (2018) deal with word repetition in single-sentence summaries, while we primarily deal with multi-sentence summaries with sentence-level repetition. There is almost no word repetition in multi-sentence summaries. Jiang and Bansal (2018) add a new decoder without attention mechanism. In CNN-based model, attention mechanism is necessary to connect encoder and decoder. Therefore, our model also is not compared with the above models in this paper. The following models can be transferred to CNN seq2seq model and are used as our baselines. See *et al.* (2017) integrates coverage mechanism, which keeps track of what has been summarized, as a feature that helps redistribute the attention scores in an indirect manner, in order to



discourage repetition. Tan *et al.* (2017) use distraction attention (Chen *et al.* 2016), which is identical to coverage mechanism. Gehrmann *et al.* (2018) add coverage penalty to loss function which increases whenever the decoder directs more than 1.0 of total attention toward a word in encoder. This penalty indirectly revises attention distribution and results in the reduction of repetition. Çelikyilmaz *et al.* (2018) uses SCL, which is the cosine similarity between two consecutive sentences, as part of the loss that helps reduce repetition. Li *et al.* (2019a) add DPPs methods into DNN attention adjustment and takes attention distribution of subsets selected from source document by DPPs as the part of loss. Paulus *et al.* (2018) propose intra-temporal attention (Nallapati *et al.* 2016) and intra-decoder attention which dynamically revises the attention distribution while decoding. It also avoids repetition at test time by directly banning the generation of repeated trigrams in beam search. Fan *et al.* (2018) borrows the idea from Paulus *et al.* (2018) and builds a CNN-based model.

Our model deals with the attention in both encoders and decoders. Different from the previous methods, our *attention filter mechanism* does not treat the attention history as a whole data structure but divides it into sections (Figure 3). Previously, the distribution curve of accumulated attention scores for each token in the source document tends to be flat, which means critical information is washed out during decoding. Our method emphasizes previously attended sections so that important information is retained.

Given our observation that repetitive sentences in the source are another cause for repetition in summary, which cannot be directly resolved by manipulating attention values, we introduce *SBD*. Unlike Paulus *et al.* (2018), we do not ban repetitive *trigrams* in test time. Instead, our decoder regenerates a sentence that is similar to previously generated ones. With the two modules, our model is capable of generating summaries with a natural level of repetition while retaining fluency and consistency.

#### 4.3. Pretrained models for summarization

The pretrained transformer language models have success in summarization tasks.

Some of the pretrained summarization models apply pretrained contextual encoders, such as BERT (Devlin *et al.* 2019). BERT proposes a transformer-based masked language model, where some of the tokens of an input sequence are randomly masked, and the goal is to predict these masked tokens with the corrupted sequence as input. Liu and Lapata (2019) introduce a document-level encoder based on BERT which is able to express the semantics of a document and obtain representations for its sentences. Zhong *et al.* (2020) leverage the BERT in a Siamese (Bromley *et al.* 1993) network structure to construct a new encoder for the representation of the source document and reference summary. Zhang *et al.* (2019a) propose a novel HIBERT encoder for document encoding and apply HIBERT to summarization model.

Others are pretrained on sequence-to-sequence (seq2seq) models. UniLM (Dong *et al.* 2019) is a multi-layer transformer network, which utilizes specific self-attention masks based on three language model (i.e., unidirectional, bidirectional, and seq2seq language models) to control what context the prediction conditions on. The seq2seq language model in UniLM attends to bidirectional contexts for source document and left contexts only for summary. For the pretraining seq2seq model, BART (Lewis *et al.* 2020) uses an arbitrary noising function to corrupt input, instead of the masked language model. Then, the corrupted input is reconstructed by training on a transformer seq2seq model. ProphetNet (Qi *et al.* 2020) trains on the transformer seq2seq model and takes future n-gram prediction as self-supervised. PEGASUS (Zhang *et al.* 2020) uses self-supervised objective Gap Sentences Generation to train a transformer seq2seq model. Compared with previous pretrained models, PEGASUS masks sentences rather than smaller continuous text spans. Through fine-tuning the pretrained models or representations on summarization task, the quality of generated summaries can be improved.

The excellent performance of the pretrained summarization models is from large-scale training datasets and heavy network structures, which always brings huge consumption of training

time and memory space. However, the goal of our approach is to reduce repetition in abstractive summarization. The comparison results of the summaries generated by the vanilla models adding different reducing repetition methods can obviously show the effectiveness of different reducing repetition methods. Thus, we take the vanilla model as our basic model and do not compare our proposed approach with the pretrained models.

## 5. Conclusion

Abstractive summarization plays an important part in NLP tasks. Its goal is to generate a short summary which expresses the main ideas of the source document. CNNs have met great success in abstractive summarization. Compared with RNNs, CNNs are more effective and can be trained much faster due to their intrinsic parallel nature and more stable gradients. However, we find that repetition is a persistent problem in the task of CNN seq2seq abstractive summarization.

In this paper, we focus on the repetition problem in abstractive summarization based on CNN seq2seq model with attention mechanism. We analyze two possible reasons behind the repetition problem in abstractive summarization: (1) attending to the same location in source and (2) attending to similar but different sentences in source. In response, we presented two methods to modify existing CNN seq2seq model, that is, a section-aware attention mechanism (ATTF) and a SBD. The ATTF can record previously attended locations in the source document directly and prevent decoder from attending to these locations. The SBD prevents the decoder from generating similar sentences more than once via backtracking at test. The proposed models are able to train a model that produces summaries with natural-level repetition that are fluent and coherent. It means that the summaries generated by our model are more accurate and readable. This can help user quickly get the main information from large of textual data, saving reading time and improving reading efficiency. As some other NLG tasks based on seq2seq model with attention mechanism are orthogonal to our proposed methods, they can also be enhanced with our proposed models. In order to assess the effectiveness of our proposed approaches in repetition reduction, we presented two evaluation metrics: *Repeatedness* and *repeatedness correlation*. Repeatedness measures the repetition rate of N-grams and sentences in summaries. Repeatedness correlation tests how well the repetition of generated summaries correlate with natural-level repetition. We also argue that ROUGE is not a perfect evaluation metric for abstractive summarization. The standard ROUGE scores cannot capture grammatical or factual errors. Thus, we proposed readability score to complement ROUGE scores. Readability is a human evaluation, which measures the fluency and readability of the summary. Our approach outperforms the baselines in all evaluation metrics, including ROUGE scores, repeatedness, repeatedness correlation, and readability.

**Competing Interests.** Yizhu Liu and Xinyue Chen are the students of Shanghai Jiao Tong University. Xusheng Luo is employed at Alibaba Group. Kenny Q. Zhu is employed at Shanghai Jiao Tong University.

## References

- Allahyari, M., Pouriyeh, S.A., Assefi, M., Safaei, S., Trippe, E.D., Gutierrez, J.B., and Kochut, K. J. (2017). Text summarization techniques: A brief survey. *arXiv*, abs/1707.02268.
- Bae, S., Kim, T., Kim, J. and Lee, S. (2019). Summary level training of sentence rewriting for abstractive summarization. *arXiv*, abs/1909.08752.
- Bai, S., Kolter, J.Z. and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv*, abs/1803.01271.
- Bing, L., Li, P., Liao, Y., Lam, W., Guo, W. and Passonneau, R.J. (2015). Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. Association for Computer Linguistics, pp. 1587–1597.

- Bokaei, M.H., Sameti, H. and Liu, Y.** (2016). Extractive summarization of multi-party meetings through discourse segmentation. *Natural Language Engineering* 22(1), 41–72.
- Briscoe, T.** (1996). The syntax and semantics of punctuation and its use in interpretation. In *Proceedings of the Association for Computational Linguistics Workshop on Punctuation*, pp. 1–7.
- Bromley, J., Bentz, J.W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., Säckinger, E. and Shah, R.** (1993). Signature verification using a “siamese” time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence* 7(4), 669–688.
- Carenini, G. and Cheung, J.C.K.** (2008). Extractive vs. NLG-based abstractive summarization of evaluative text: The effect of corpus controversiality. In *INLG 2008 - Proceedings of the Fifth International Natural Language Generation Conference, June 12–14, 2008, Salt Fork, Ohio, USA*. The Association for Computer Linguistics.
- Çelikyilmaz, A., Bosselut, A., He, X. and Choi, Y.** (2018). Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1–6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, pp. 1662–1675.
- Chen, Q., Zhu, X., Ling, Z., Wei, S. and Jiang, H.** (2016). Distraction-based neural networks for modeling document. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9–15 July 2016*. IJCAI/AAAI Press, pp. 2754–2760.
- Chen, Y. and Bansal, M.** (2018). Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15–20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 675–686.
- Chopra, S., Auli, M. and Rush, A. M.** (2016). Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12–17, 2016*. Association for Computational Linguistics, pp. 93–98.
- Dauphin, Y.N., Fan, A., Auli, M. and Grangier, D.** (2017). Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017*. PMLR, pp. 933–941.
- Devlin, J., Chang, M., Lee, K. and Toutanova, K.** (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pp. 4171–4186.
- Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M. and Hon, H.** (2019). Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 December 2019, Vancouver, BC, Canada*, pp. 13042–13054.
- Fan, A., Grangier, D. and Auli, M.** (2018). Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, NMT@ACL 2018, Melbourne, Australia, July 20, 2018*. Association for Computational Linguistics, pp. 45–54.
- Gehring, J., Auli, M., Grangier, D., Yarats, D. and Dauphin, Y. N.** (2017). Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017*. PMLR, pp. 1243–1252.
- Gehrmann, S., Deng, Y., and Rush, A. M.** (2018). Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, pp. 4098–4109.
- Grusky, M., Naaman, M. and Artzi, Y.** (2018). Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1–6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, pp. 708–719.
- He, K., Zhang, X., Ren, S. and Sun, J.** (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016*. IEEE Computer Society, pp. 770–778.
- Hermann, K.M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M. and Blunsom, P.** (2015). Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7–12, 2015, Montreal, Quebec, Canada*, pp. 1693–1701.
- Jiang, Y. and Bansal, M.** (2018). Closed-book training to improve summarization encoder memory. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, pp. 4067–4077.
- Kim, S.** (2019). Deep recurrent neural networks with layer-wise multi-head attentions for punctuation restoration. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12–17, 2019*. IEEE, pp. 7280–7284.

- Kulesza, A. and Taskar, B.** (2011). k-DPPs: Fixed-size determinantal point processes. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*. Omnipress, pp. 1193–1200.
- Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V. and Zettlemoyer, L.** (2020). BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5–10, 2020*. Association for Computational Linguistics, pp. 7871–7880.
- Li, L., Liu, W., Litvak, M., Vanetik, N. and Huang, Z.** (2019a). In conclusion not repetition: Comprehensive abstractive summarization with diversified attention based on determinantal point processes. In *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019*. Association for Computational Linguistics, pp. 822–832.
- Li, W., He, L. and Zhuge, H.** (2016). Abstractive news summarization based on event semantic link network. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*. Association for Computer Linguistics, pp. 236–246.
- Li, W., Xiao, X., Lyu, Y. and Wang, Y.** (2018a). Improving neural abstractive document summarization with explicit information selection modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, pp. 1787–1796.
- Li, W., Xiao, X., Lyu, Y. and Wang, Y.** (2018b). Improving neural abstractive document summarization with structural regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 4078–4087. Association for Computational Linguistics.
- Li, X. L., Wang, D. and Eisner, J.** (2019b). A generative model for punctuation in dependency trees. *Transactions of the Association for Computational Linguistics* 7, 357–373.
- Lin, C.-Y.** (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain.
- Lin, J., Sun, X., Ma, S. and Su, Q.** (2018). Global encoding for abstractive summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*. Association for Computational Linguistics, pp. 163–169.
- Liu, Y., Jia, Q. and Zhu, K. Q.** (2021). Keyword-aware abstractive summarization by extracting set-level intermediate summaries. In *WWW 2021: The Web Conference 2021, Virtual Event/Ljubljana, Slovenia, April 19-23, 2021*. ACM/IW3C2, pp. 3042–3054.
- Liu, Y. and Lapata, M.** (2019). Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 5070–5081.
- Liu, Y., Luo, Z. and Zhu, K. Q.** (2018). Controlling length in abstractive summarization using a convolutional neural network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, pp. 4110–4119.
- Loukina, A., Zechner, K. and Chen, L.** (2014). Automatic evaluation of spoken summaries: the case of language assessment. In *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2014, June 26, 2014, Baltimore, Maryland, USA*. Association for Computer Linguistics, pp. 68–78.
- Nallapati, R., Zhou, B., dos Santos, C. N., Gülçehre, Ç. and Xiang, B.** (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. Association for Computational Linguistics, pp. 280–290.
- Naserasadi, A., Khosravi, H. and Sadeghi, F.** (2019). Extractive multi-document summarization based on textual entailment and sentence compression via knapsack problem.
- Nguyen, M., Cuong, T.V., Nguyen, X.H. and Nguyen, L.** (2019). Web document summarization by exploiting social context with matrix co-factorization. *Information Processing and Management* 56(3), 495–515.
- Pallotta, V., Delmonte, R. and Bristot, A.** (2009). Abstractive summarization of voice communications. In *Human Language Technology. Challenges for Computer Science and Linguistics - 4th Language and Technology Conference, LTC 2009, Poznan, Poland, November 6-8, 2009, Revised Selected Papers*. Springer, pp. 291–302.
- Pascanu, R., Mikolov, T. and Bengio, Y.** (2013). On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. JMLR.org, pp. 1310–1318.
- Paulus, R., Xiong, C. and Socher, R.** (2018). A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. [OpenReview.net](https://openreview.net).
- Qi, W., Yan, Y., Gong, Y., Liu, D., Duan, N., Chen, J., Zhang, R. and Zhou, M.** (2020). Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*. Association for Computational Linguistics, pp. 2401–2410.
- Radev, D. R., Hovy, E. H. and McKeown, K. R.** (2002). Introduction to the special issue on summarization. *Computational Linguistics* 28(4), 399–408.

- Rush, A. M., Chopra, S. and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. Association for Computational Linguistics, pp. 379–389.
- Sankarasubramaniam, Y., Ramanathan, K. and Ghosh, S. (2014). Text summarization using wikipedia. *Information Processing and Management* 50(3), 443–461.
- See, A., Liu, P.J. and Manning, C.D. (2017). Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 1073–1083.
- Sharma, E., Huang, L., Hu, Z. and Wang, L. (2019). An entity-driven framework for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computer Linguistics, pp. 3278–3289.
- Shi, T., Keneshloo, Y., Ramakrishnan, N. and Reddy, C.K. (2021). Neural abstractive text summarization with sequence-to-sequence models. *Transactions on data Science* 2(1), 1:1–1:37.
- Sutskever, I., Martens, J., Dahl, G. E. and Hinton, G. E. (2013). On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. JMLR.org, pp. 1139–1147.
- Suzuki, J. and Nagata, M. (2017). Cutting-off redundant repeating generations for neural abstractive summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*. Association for Computational Linguistics, pp. 291–297.
- Tan, J., Wan, X. and Xiao, J. (2017). Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 1171–1181.
- Vaswani, A., Bengio, S., Brevdo, E., Chollet, F., Gomez, A. N., Gouws, S., Jones, L., Kaiser, L., Kalchbrenner, N., Parmar, N., Sepassi, R., Shazeer, N. and Uszkoreit, J. (2018). Tensor2tensor for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas, AMTA 2018, Boston, MA, USA, March 17-21, 2018 - Volume 1: Research Papers*. Association for Machine Translation in the Americas, pp. 193–199.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 5998–6008.
- Verma, R. M. and Lee, D. (2017). Extractive summarization: Limits, compression, generalized model and heuristics. *Computación y Sistemas* 21(4).
- Wang, K., Quan, X. and Wang, R. (2019). Biset: Bi-directional selective encoding with template for abstractive summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 2153–2162.
- Yao, J., Wan, X. and Xiao, J. (2017). Recent advances in document summarization. *Knowledge and Information Systems* 53(2), 297–336.
- Zhang, J., Zhao, Y., Saleh, M. and Liu, P. J. (2020). PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, pp. 11328–11339.
- Zhang, X., Wei, F. and Zhou, M. (2019a). HIBERT: document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 5059–5069.
- Zhang, Y., Li, D., Wang, Y., Fang, Y. and Xiao, W. (2019b). Abstract text summarization with a convolutional seq2seq model. *Applied Sciences* 9, 1665.
- Zhong, M., Liu, P., Chen, Y., Wang, D., Qiu, X. and Huang, X. (2020). Extractive summarization as text matching. In Jurafsky, D., Chai, J., Schluter, N. and Tetreault, J.R. (eds), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, pp. 6197–6208.
- Zhong, M., Liu, P., Wang, D., Qiu, X. and Huang, X. (2019). Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computer Linguistics, pp. 1049–1058.