Improving Abstractive Summarization with Iterative Representation

Jinpeng Li[†‡], Chuang Zhang[†*], Xiaojun Chen[†], Yanan Cao[†], Ruipeng Jia[†‡]
[†]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[‡]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*
{lijinpeng, zhangchuang, chenxiaojun, caoyanan, jiaruipeng}@iie.ac.cn

***Abstract*—In the neural abstractive summarization field, comprehensive document representation and summary embellishment are two major challenges. To tackle the above problems, we propose an Iterative Abstractive Summarization (IAS) model through iterating the document and summary representation. Specifically, (1) we design a selective gated strategy to constantly update the input representation in the encoder, which is consistent with the repeated updating of human memory information in human writing. (2) We design an iterative unit to revise the comprehensive representation iteratively for polishing the summary. Moreover, we utilize reinforcement learning to optimize our model for the non-differentiable metric ROUGE, which can alleviate the exposure bias during predicting words effectively. Experiments on the CNN/Daily Mail, Gigaword and DUC-2004 datasets show that the IAS model can generate high-quality summaries with varied length, and outperforms baseline methods significantly in terms of ROUGE and Human metrics.**

***Index Terms*—abstractive summarization, sequence-to-sequence, selective gated, iterative representation, ROUGE**

I. Introduction

Automatic text summarization, which creates condensed summaries of the source documents, is an important task in Natural Language Processing (NLP) fields. It can alleviate the problem of information overload. There are two mainstream methods to handle this task now, extractive methods and abstractive methods. Extractive summarization extracts a summary by copying words, phrases or sentences from source documents. In the past, researchers study the task mainly focus on that. There's a lot of extractive research, including keyphrase [1], feature scores [2], graph ranking [3], sequence labeling [4], dynamic programming [5], classification [6] and so on. The summaries generated by extractive methods usually are inflexible and contain a lot of redundant information. Unlike this, abstractive summarization generates a short version, which may contain a word that does not appear in the original text. Furthermore, abstractive methods have benefited from deep neural network development. Particularly, a large number of previous work has studied the neural sequence-to-sequence framework for abstractive summarization [7]–[10]. In this paper, we mainly focus on the abstractive summarization, since it is closer to the essence of the summary.

Although these approaches have achieved great success, the traditional sequence-to-sequence models are still facing comprehensive document representation and summary embellishment challenges. In human cognitive behaviors, summaries are generated in two stages: reading document repeatedly and polishing summary iteratively. The former can model comprehensive document representation by updating the memory information, and the latter continuously improve the quality of summary by changing words. Conversely, the previous works are inconsistent with human cognitive behaviors. Firstly, the encoder reads the document once to model the document representation. This representation can't consider the information of the decoder and lead to incomplete document representation. Secondly, the decoder just leverages the previously generated words to predict the next word, without considering future words. In addition, they generate the summary only once through the decoder. If a wrong word is generated, the model does not have the ability to modify it. These will lead to a decline in summary quality.

To solve the above problems, we present an Iterative Abstractive Summarization (IAS) model based on the encoder-decoder architecture. The IAS model leverages global information (i.e., the source document and the rough summary) to iteratively review and polish the sequence decoding process, as shown in Figure 1. In the beginning, our IAS model produces a rough summary based on the typical encoder-decoder architecture. Unlike previous work which usually ends here, our model utilizes the information of rough summary and source document to polish the summary iteratively. There are two main modules in the IAS model, selective gated unit and iterative unit. The selective gated unit decides what information representation needs to be updated after each iteration. It reduces the noise of the original document and increases the weight of important words. The iterative unit revises the comprehensive representation through the previous vectors to polish the summary. In addition, sequence-to-sequence model is inevitably to cause the exposure bias problem [11] and the ROUGE metric and cross-entropy loss are not corresponding during the training stage. Therefore, the self-critical sequence training [12] strategy is employed to the IAS model as well as [13], which provides an effective baseline and improves test time consistency.

To evaluate the performance of our model, we use the ROUGE and human evaluation method as evaluation metrics. Experimental results on English long text dataset (CNN/Daily Mail) and English short text dataset (Gigaword) show that the

* Corresponding author: Chuang Zhang, zhangchuang@iie.ac.cn.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

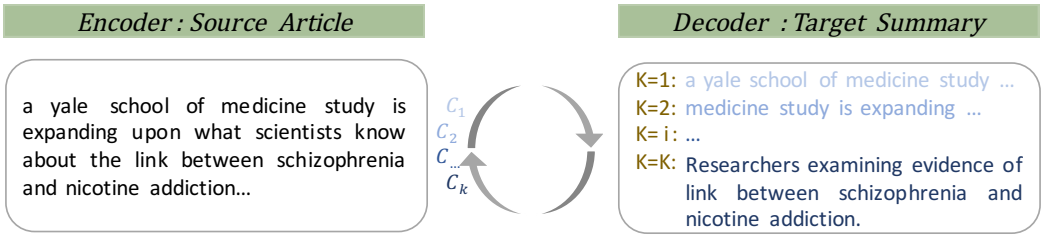


Fig. 1. The Iterative process of our model IAS to produce the target summary from source document.

IAS model outperforms the baselines methods. Moreover, we evaluate our proposed model on the DUC-2004 test set. The experimental results demonstrate the IAS model is very robust and extensible than baselines. The contributions of this paper are listed as follows:

- We propose Iterative Abstractive Summarization (IAS) model to explore the idea of optimizing summary iteratively for abstractive summarization and fit in with human cognitive behaviors.
- We employ a self-critical strategy in IAS model to alleviate the problems of exposure bias as well as the inconsistency between non-differentiable ROUGE and cross-entropy loss.
- Experimental results on CNN/Daily Mail, Gigaword and DUC-2004 datasets demonstrate that the IAS model outperforms baseline including several existing extractive and abstractive methods.

The rest of the paper is structured as follows: Section II introduces the related work about the automatic summarization. Section III describes our proposed method in detail. Section IV presents baselines, datasets, experiments and results. Finally, we conclude this paper and future work in Section V.

II. Related Work

A. Abstractive summarization

In this paper, we deal with the abstractive summarization task. In this field, various models are proposed based on the sequence-to-sequence architecture. Rush et al. [7] first proposed an attention-based abstractive summarization model with encoder-decoder. Based on them, Nallapati et al. [14] proposed to utilize traditional features for improving the encoding ability of the encoder. Besides, they used the Large Vocabulary Trick [15] to relieve the calculation bottleneck of the decoder, and pointer mechanism and hierarchical attention to catch keywords and sentences. Then, See et al. [9] combined pointer network and coverage mechanism into their model to deal with out-of-vocabulary (OOV) words and repetition for improving the performance. Their work can make the model choose to copy a word from the input article or generating word from the fixed vocabulary by itself. Cao et al. [16] proposed a unified model combining the strength of extractive and abstractive summarization, which approach achieves better performance by extracting actual fact descriptions and guiding the sequence-to-sequence. Due to the limitations of the RNN-based model, Gehring et al. [17] proposed a convolutional sequence-to-sequence (ConvS2S) framework, which outperforms RNN-based models in the language modeling and machine translation tasks. Wang et al. [18] used the ConvS2S model to incorporate the topic information and used the policy gradient algorithm to directly optimize the ROUGE score in the summarization task. For good measure, there has been increasing interest in reinforcement learning (RL) to the task of summarization. For example, Paulus et al. [13] combined standard supervised word prediction and reinforcement learning of policy gradient to reduce exposure bias for abstractive summarization. However, these summarization models are inconsistent with human cognitive behaviors in generating summaries. These methods generate summary only once through the encoder and decoder, they lack the process of polishing. In light of this, it is necessary to train a generated model by following the human reading strategy to generate a summary, which is more like a human summary.

B. Human-like Reading Strategy

Recently, Yang et al. [19] proposed an extension model of the encoder-decoder framework called the review network. The review network performs multiple review operations on a hidden state of the encoder based on the attention mechanism. After each review operation, it obtains a fact vector, which is used for the input of the decoder. Although they add review steps on the encoder, it still adopts one-pass decoding. Xia et al. [20] introduced the deliberation networks for sequence generation. A deliberation network has a two-stage decoder. The first stage decoder is used to decode and generate a raw sequence, and the second stage decoder polishes and refines the raw sentence through the process of deliberation. Since the second stage decoder has global information about what the sequence to be generated might be, it can generate a better sequence by looking into future words in the raw sentence. This work has two-pass decoding, but it does not update the encoder information. Another related work is [21], where introduced an extractive summarization model that iteratively polishes the document representation on many passes through the article. We take some inspiration from their work but focus on abstractive summarization task.

III. Methodology

In this section, we introduce the Iterative Abstractive Summarization (IAS) model in detail. We formulate the sum-

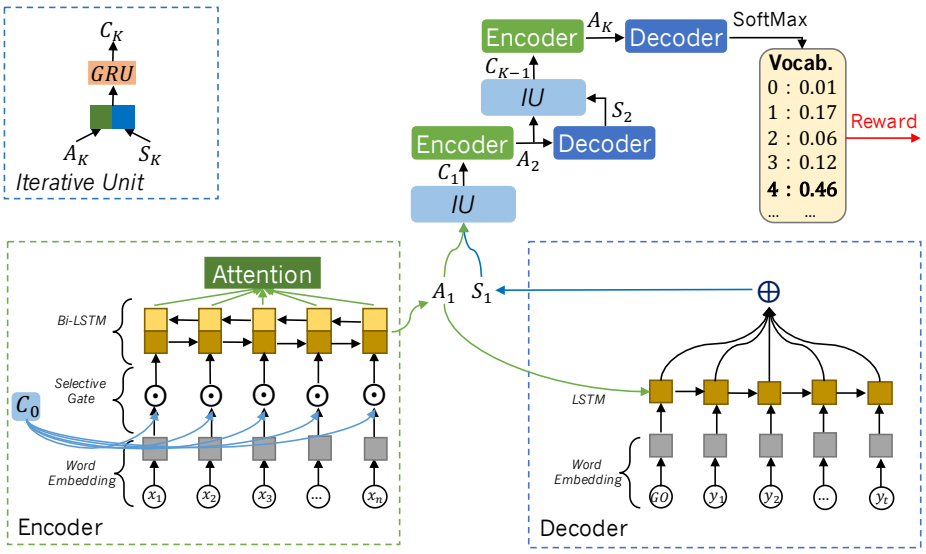


Fig. 2. The IAS Model Architecture. What should be noted is that the attention mechanism and pointer network are not fully connected for clarity. The lower-left and lower-right boxes are encoder and decoder for the first iteration respectively. The upper-left box is the iterative unit (IU). The upper-right corner is the process of iteration.

marization process in Section III(A) and explain the basic infrastructure encoder-decoder in Section III(B). In Section III(C), we describe the selective gated unit. The important iterative unit and reinforcement learning section are shown in Section III(D), Section III(E). The graphical illustration of IAS can be found in Figure 2, IAS consists of two main modules: a selected information module and a polished memory module.

A. Problem Formulation

Each source document $X = (x_1, x_2, ..., x_t, ..., x_n)$ has a standard reference summary $Y = (y_1, y_2, ..., y_t, ..., y_m)$, where n and m are the number of words in the sequence X and Y, respectively. It should be noted that $m << n$. In abstractive method, an input document X is fed to the encoder for a context representation, then the decoder generates a summary $\hat{Y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_t, ..., \hat{y}_M)$, where M denotes the length of the generated summary. In the traditional end-to-end model, the model will stop at the current time step. However, we propose an Iterative Abstractive Summarization (IAS) model to continue polishing the summary, the operation does not stop here. IAS uses an input document to generate a summary $\hat{Y}^1 = (\hat{y}_1^1, \hat{y}_2^1, ..., \hat{y}_t^1, ..., \hat{y}_M^1)$ in first stage. Then we treat the hidden state S^1 of $\hat{Y}^1$ as a rough summary, feed it and source document state A^1 to the iterative unit. The comprehensive representation $\mathcal{C}^1$ produced by the iterative unit will be sent to the next iteration for polishing the summary again, where ^{1} denotes the first time IAS generates summary. After K iterations, we obtain the final summary Y^K, where the value of K depends on the validation set.

B. Basic Framework for Summarization

The words sequence X passes through an embedding layer firstly. Then we employ a bi-directional LSTM to produce the hidden state of the embedded vector $W = (w_{x_1}, w_{x_2}, ..., w_{x_i}, ..., w_{x_n})$, where $w_{x_i} \in \mathbb{R}^D$ is the embedding for word x_i. At time step i, the Bi-LSTM is defined as follows:

$$\overrightarrow{h_i} = \overrightarrow{LSTM}(w_{x_i}, \overrightarrow{h_{i-1}}) \tag{1}$$

$$\overleftarrow{h_i} = \overleftarrow{LSTM}(w_{x_i}, \overleftarrow{h_{i+1}}) \tag{2}$$

$$h_i = [\overrightarrow{h_i}; \overleftarrow{h_i}] \tag{3}$$

where $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ are the forward and the backward hidden state vector at i time step.

In the IAS model, the decoder consists of an embedding layer, a uni-directional LSTM, and a softmax layer. The goal is to generates a summary $\hat{Y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_t, ..., \hat{y}_M)$. At time step t, the previous work generation process of $\hat{y}_t \in \mathbb{R}^V$ is calculated as:

$$a^t = softmax(v^T \tanh(W_h h_t + W_s s_t + b_{attn})) \tag{4}$$

$$c_t = \sum_{i=1}^{n} a_i^t h_i \tag{5}$$

$$s_t = \overrightarrow{LSTM}(w_{y_t}, s_{t-1}, c_{t-1}) \tag{6}$$

$$P_{vocab} = softmax(M^{'}(M[s_t; c_t] + b) + b^{'}) \tag{7}$$

where V is the size of vocabulary, a^t is the attention distribution, c_t is the context vector, s_t is the decoder state and v, W_s, W_h, $M^{'}$, M, b, $b^{'}$, b_{attn} are learnable parameters. We replace w_{x_i} with w'_{x_i} in (1) and (2) by selective gated unit, which is described in the Section 3.3. P_{vocab} is a probability distribution of the vocabulary at time step t. However, there

are multiple iterative decoders in the IAS model. Our P_{vocab} calculation is as follows:

$$O_t = M[s_t^K; c_t] + b \tag{8}$$

$$P_{vocab} = softmax(M^{'} O_t + b^{'}) \tag{9}$$

where K is the number of iteration. The model is trained by minimizing a maximum-likelihood loss. The loss function $\mathcal{L}_{ml}$ for predicted word $\hat{y}_t$ is the negative log likelihood of the target word y_t^* at each time step t:

$$p(\hat{y}_t) = P_{vocab}(\hat{y}_t) \tag{10}$$

$$\mathcal{L}_{ml} = -\frac{1}{T}\sum_{t=1}^{T} log(p(y_t^*)) \tag{11}$$

The common problems of abstractive summarization are the out-of-vocabulary (OOV) generation and multiple "UNK" tokens in the summary. The copy mechanism [22] and pointing [9], [23] are the mainstream solution about these problems. Refer to [9], the generation probability $p_{gen} \in [0, 1]$ for time step t is calculated from the context vector c_t, the decoder hidden state s_t, and the decoder input x_t:

$$p_{gen} = \sigma(W_c^T c_t + W_s^T s_t + W_x^T x_t + b_{gen}) \tag{12}$$

where vectors W_c, W_s, W_x and b_{gen} are learnable parameters. p_{gen} is a soft switch to choose between generating a word from the fixed vocabulary or copying word from the input document sequence. For each time step t, model will give a candidate token $\hat{y}_t$, if $\hat{y}_t$ is an out-of-vocabulary word, then $P_{vocab}(\hat{y}_t) = 0$, if $\hat{y}_t$ does not appear in the source document, then $\sum a_i^t = 0$:

$$p(\hat{y}_t) = p_{gen} P_{vocab}(\hat{y}_t) + (1 - p_{gen}) \sum a_i^t \tag{13}$$

C. The Selective Gated Unit

In the process of writing summaries, human beings will constantly read the original text and update the memory information. The selective gated unit is designed for this purpose. Motivated by the design of GRU, we propose a reasonable alternative solution to combine of iterative information $\mathcal{C}^K$ and the word embedding $W = (w_{x_1}, w_{x_2}, ..., w_{x_i}, ..., w_{x_n})$ in the encoder. We can not determine which word of input is better for the summary, so we let the model learning the weight between two annotations automatically. The selective gated unit needs two inputs, which are the word embedding W and the vector $\mathcal{C}^K$ produced by the iterative unit. The former denotes the word information of the input document. There will inevitably be some noise, so we need to leave important information through the selective gated unit and filter out the noise. For each time step t, the selective gated utilizes them to generate the gated vector g:

$$g = \sigma(W_g \mathcal{C}_k + U_g w_{x_t}) \tag{14}$$

$$w'_{x_t} = g \odot \mathcal{C}_k + (1 - g) \odot w_{x_t} \tag{15}$$

where σ denotes sigmoid activation function, W_g and U_g are learnable weight matrices. This selective gated unit can automatically decide to which extent the information of words should be updated based on the previous iterative summary and grasp more accurate information from the document. After the selective gated unit, we can obtain the new embedded vector $W' = (w'_{x_1}, w'_{x_2}, ..., w'_{x_i}, ..., w'_{x_n})$ as the input of next network layer.

Algorithm 1 Iterative document and Summary Representation

Input: Parallel data X and Y
X denotes the source document
Y is a ground-truth summary
Parameter: Θ
Output: $\mathcal{C}_K$
$\mathcal{C}_K$ denotes the K-th comprehensive vector representation of source document and produced summary

1: Define $K \leftarrow$ Iterative times
2: Define $m \leftarrow$ Target summary length
3: Initialize $\mathcal{C}_0 \leftarrow \mathbf{0}$ as the first comprehensive representation
4: **repeat**
5: Obtain h_i by (3)
6: Let h_i and $\mathcal{C}_K$ fed to selective gated unit
7: Obtain h'_i by (15)
8: $A_K \leftarrow h'_i$
9: **for** $i \leftarrow 1$ to $min(m, maxlength)$ **do**
10: Generate the decoder hidden state $\hat{h}_i^K$
11: **end for**
12: $S_K \leftarrow sum(\hat{h}_i^K)$ as the rough summary representation
13: Obtain $\mathcal{C}_K$ by (16)
14: **until** Iteration condition K reached
15: **return** $\mathcal{C}_K$

D. The Iterative Unit

At training and inference time, in order to adapt the model to the human reading strategy, we propose an iterative representation algorithm, as shown in Algorithm 1. For each input parallel data X and Y, we first initialize several variables: (1) K is defined as the iterative times. (2) m is defined as the target summary length. (3) $\mathcal{C}_0$ is initialized as a zero matrix to record the first representation. Next, we obtain a comprehensive representation through an iterative process (Lines 4-14). After each iteration, we obtain the source document hidden state A^K from the encoder (Lines 5-8) and the summary representation S^K from the decoder (Lines 9-12). These are two vectors that are fed to the iterative unit for polishing the comprehensive representation with the new information (Lines 13). In our proposed model, GRU is employed as our iterative unit (IU) to update the comprehensive representation:

$$\mathcal{C}_K = GRU(M^K[A_K; S_K] + b^K) \tag{16}$$

where M^K and b^K vectors are learnable parameters. During each iteration, the IU merges information according to the preliminary summary predicted by the decoder and the source content of the encoder to improve the quality of summaries. And, the traditional sequence-to-sequence model can only see the previous $(1, t-1)$ words when decoding the current time (t)

TABLE I
DATA STATISTICS OF DATASET

Datasets	CNN/Daily Mail			Gigaword			DUC2004
	Train	Valid	Test	Train	Valid	Test	Test
Doc(Ours)	287,227	13,368	11,490	3,803,957	189,651	1951	500
Ave Doc Len(word)	790.33	768.93	777.88	31.35	31.32	29.68	35.49
Ave Ref Len(sen)	3.68	4.00	3.88	1.00	1.00	1.00	1.00
Ave Ref Len(word)	55.16	61.43	58.31	8.22	8.31	8.79	10.41
Ave Ref Number	1	1	1	1	1	1	4

Doc means the number of documents. Ave Doc Len(word) means the average number of words in the documents. Ave Ref Len(sen) and Ave Ref Len(word) mean the average number of sentences and words in the summaries. Ave Ref Number means the number of summary / document.

word, it loses a more holistic view of summary. IAS effectively avoids this problem through IU, which can consider all the information in the decoder, and help the decoder generate a high-quality summary.

E. The Reinforcement Learning

During the training stage, we use the teacher forcing algorithm [24] to train the model, which is a method for quickly and efficiently training of recurrent neural network models by using the ground-truth word as input. But this will lead to the accumulation of errors in the testing stage and generate a poor summary. This problem called exposure bias [11]. Besides, the model is optimized by the maximum-likelihood objective that can help the model to generate the same summaries as references. Unfortunately, the evaluation metric is the non-differentiable ROUGE. We cannot directly optimize the metric by backpropagation. Therefore, we employ the self-critical sequence training [12] to maximize the non-differentiable ROUGE metric. For the reinforcement learning stage, the IAS model generates two output sequences $\hat{y}$ and y^s. $\hat{y}$ is the baseline output, which is obtained by selecting the words that maximize the output probability distribution. y^s is the sampled sequence, which is generated by sampling from the output probability distribution at each time step. The reward function $r(.)$ is set to the ROUGE-L scores in our paper. We minimize the reinforcement learning loss function as follow:

$$\mathcal{L}_{rl} = \frac{1}{T}(r(\hat{y}) - r(y^s)) \sum_{t=1}^{T} \log(p(y_t^s)) \tag{17}$$

Following [13], we train our model by using the mixture of (11) and (17) for a more fluent summary. The mixed learning objective function as follow:

$$\mathcal{L}_{loss} = \lambda \mathcal{L}_{rl} + (1 - \lambda)\mathcal{L}_{ml} \tag{18}$$

where the λ is a hyper-parameter that measures the ratio of $\mathcal{L}_{rl}$ and $\mathcal{L}_{ml}$, it is tuned on the validation set.

IV. EXPERIMENT

A. Datasets

We use the CNN/Daily Mail, Gigaword and DUC-2004[1] datasets to evaluate the performance of different summarization models as Table I.

- CNN/Daily Mail: To evaluate IAS model performance in a long text, we choose the CNN/Daily Mail dataset that is widely in automatic summarization. The dataset made by Hermann et al. [25], they collected about one million pieces of news data from CNN and Daily Mail as a machine reading comprehension corpus. Each document in this dataset is paired with a manually written summary of multiple sentences. We preprocess the original data and get a non-anonymous version of the summarization dataset, which consists of 287,227 training pairs, 13,368 validation pairs, 11,490 test pairs.
- Gigaword: We use the annotated English Gigaword dataset [26], [27] to evaluate IAS model performance in a short text. The Gigaword dataset is preprocessed identically to [7], consisting of 3,803,957 training pairs, 189,651 verification pairs, and 1,951 test pairs.
- DUC-2004: The test-only DUC-2004 dataset consists of 500 test pairs. It has one different from the other two is that each source text in DUC-2004 is paired with four human-generated reference summaries. We average the scores of these four summaries, which makes the evaluation more objective.

B. Evaluation Metrics

In this work, we use the *ROUGE metrics* [28] and *Human metrics* to evaluate the performance of summarization models under different influential factors. The ROUGE is the automatic evaluation method and based on the n-grams. Therefore, it lacks the consideration of semantic information. The ROUGE-1, ROUGE-2 and ROUGE-L F_1 scores are computed by the pyrouge package[2]. Due to the limitations of the ROUGE evaluation, we also conducted Human metrics by selecting 100 examples randomly in the test set. For getting a more fair and objective result, all generated summaries are re-capitalized and de-tokenized. We invite human evaluators with excellent English literacy skills to evaluate each summary. Human metrics include three important indicators. Conciseness, which reflects the extent of redundancy in the summary. Informativity, which reflects the extent of the summary contained important information from the original document. Readability, which reflects the extent of fluency in the summary. The score range of each indicator is from 0 to 5, and 5 indicates the best score.

[1]http://duc.nist.gov/

[2]https://pypi.org/project/pyrouge/

TABLE II
PERFORMANCE OF VARIOUS MODELS ON CNN/DAILY MAIL TEST SET WITH ROUGE F_1(%) SCORES AND HUMAN EVALUATION SCORES

Model	ROUGE			Human		
	RG-1	RG-2	RG-L	Conciseness	Informativity	Readability
TextRank	35.65	12.59	31.65	-	-	-
SummaRuNNer	38.66	16.11	34.77	3.36	3.31	3.68
Lead-3	40.31	17.73	36.51	-	-	-
Seq2Seq-att.(our impl.)	31.31	11.82	28.87	1.65	2.13	1.78
Pointer-gen.(our impl.)	36.23	15.45	33.06	3.23	3.14	3.35
-IAS w/o RL	36.95	15.67	33.58	3.31	3.29	3.53
-IAS	37.65	16.03	34.18	3.39	3.25	3.58
Pointer-gen.+cov.(our impl.)	39.10	16.95	35.93	3.45	3.14	3.55
-IAS w/o RL	39.27	17.15	36.18	3.51	**3.49**	3.83
-IAS	**39.70**	**17.34**	**36.56**	**3.59**	3.45	**3.98**
Ground-truth	-	-	-	3.65	3.43	4.32

All the scores calculated by pyrouge package have a 95% confidence interval in the official ROUGE script. Baseline models are described in Section IV(D).

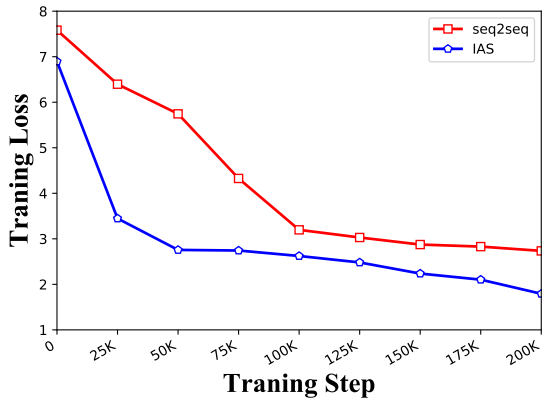


Fig. 3. Training loss of models on the CNN/Daily Mail dataset.

C. Experimental Details

In our work, we limit the vocabulary to 50,000. The source documents and the summaries share the same vocabularies, and the embedding layers of encoder and decoder share the same parameters. We set the word embedding size to 256, which is randomly initialized. The encoder is a single-layer bidirectional LSTM, the decoder is a single-layer unidirectional LSTM, and the hidden layers are set to 256 dimensions in basic model. The batch size is 32. We utilize the Adam optimization method in the training stage. For the hyper-parameters of Adam optimizer, we set the learning rate $r = 10^{-3}$, two momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. We tune the hyper-parameters on the validation sets for better performance. Additionally, we investigate the convergence rate of loss and report the results in Figure 3. We observe that our approach can fit the training sets better and its convergence process is significantly faster than baseline.

D. Baselines

We compare our proposed model with the following extractive and abstractive summarization models for evaluating the performance of our model.

- Lead-3: The model chooses the first three sentences as summary, which is a strong baseline with high ROUGE score, because the writing habit of English is to put the central idea at the beginning of the sentence.
- TextRank: This model is an unsupervised graph-based sentence ranking algorithm [29], which builds the text into a topology map.
- SummaRuNNer: This is a recurrent neural network model with an interpretable advantage for extractive summarization [30].
- ABS/ABS+: This is the first work about the abstractive summarization, it is consist of neural attention Seq2Seq model with an attention-based encoder and a neural network language model decoder [7].
- Seq2Seq-att.: The encoder-decoder is the same as IAS model with $K = 0$, it consists of encoder (a single-layer bi-LSTM) and decoder (a single-layer unidirectional LSTM) structure with attention calculated as [31].
- LVT2K: LVT2K is short for words-lvt2k-1sent, which is trained only on the first sentence from the source document with the large vocabulary trick [8].
- Luong-NMT: [32] implemented the work of [33] uses a two layers LSTM as the encoder-decoder with 500 hidden units per layer.
- Pointer-gen.(+cov.): An abstractive summarization model with pointer network proposed by [9]. We implement two versions of this baseline (with or without the coverage mechanism).

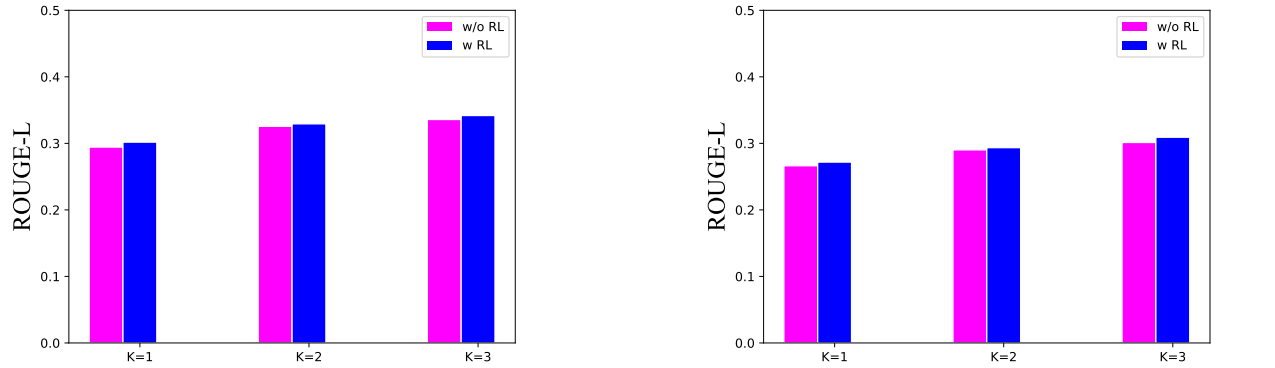
E. Experiment Analysis

CNN/Daily Mail: The results of the IAS model and baselines with full-length ROUGE-1, ROUGE-2 and ROUGE-L F_1 scores on the CNN/Daily Mail are given in Table II. We compare our model with the abstractive (Seq2Seq-att.,

TABLE III
PERFORMANCE OF VARIOUS MODELS ON GIGAWORD TEST SET WITH ROUGE F_1(%) SCORES AND DUC-2004 WITH ROUGE RECALL SCORES

Model	Gigaword			DUC-2004		
	RG-1	RG-2	RG-L	RG-1	RG-2	RG-L
ABS *	29.55	11.32	26.64	26.55	7.06	22.05
ABS+*	29.78	11.89	26.97	28.18	8.49	23.81
LVT2K*	32.67	15.59	30.64	28.35	9.46	24.59
Luong-NMT*	33.10	14.45	30.71	28.55	8.79	24.43
IAS	**33.27**	**15.26**	**30.92**	**29.08**	**10.71**	**26.35**

All the scores calculated by pyrouge package have a 95% confidence interval in the official ROUGE script. The results with $*$ mark are taken from the corresponding papers.



(a) The ROUGE-L scores on CNN/Daily Mail of IAS models

(b) The ROUGE-L scores on Gigaword of IAS models

Fig. 4. The performance of IAS models (only Pointer-gen.) with different iterations K on datasets.

Pointer-gen.) and extractive (TextRank, SummaRuNNer, Lead-3) models. The experimental results demonstrate that with the help iterative unit and selective gated unit, our IAS model (Line 7) achieves significant improvements upon the baseline (Line 5, Pointer-gen.) with +1.42 in terms of ROUGE-1, +0.58 in terms of ROUGE-2, and +1.12 in terms of ROUGE-L. In the reinforcement learning setting, our IAS model still achieves an advantage in all metrics over the baseline. Besides, we employ a coverage mechanism to the IAS model for testing the extension of iterative sequence-to-sequence structure. As shown in Table II, our IAS model with pointer and coverage again receives higher scores than the Pointer-Gen.+cov. from [9] and other baselines. Although the source document of CNN/Daily Mail is very long, the IAS model can generate a high-quality summary. This experiment also shows that iterative unit is scalable for improving the performance of traditional sequence-to-sequence models.

Human evaluation results also are shown in Table II, it demonstrates that our model performs better than baselines. We provide summaries (model-generated by models and ground-truth by humans) for each example to our evaluators. The scores for each example are carefully checked and averaged to get the final result. The three scores of IAS are the closest to the reference score. Even the IAS w/o RL model obtains the highest score on Informativity, it captures more important original information by an iterative process. There is evidence to suggest that the summaries of IAS close to the human-like summary. The IAS model achieves the best performance on Conciseness and Readability, it is good evidence that the generated summary is more streamlined and fluent with the help of reinforcement learning.

Gigaword: Table III shows full-length ROUGE-1, ROUGE-2 and ROUGE-L F_1 scores on Gigaword dataset. We compare the F_1 scores of our IAS model (only in Pointer-gen.) with other baselines (ABS, ABS+, LVT2K, Luong-NMT). Experiments on the Gigaword show that the IAS model not only achieves a good performance in the long text, but also in the short text. Therefore, the idea of an iterative encoder-decoder is meaningful and can be extended to many other methods, which is very important for later research.

DUC-2004: The DUC-2004 dataset is an evaluation-only dataset. Its distribution is similar to Gigaword dataset, we evaluate the performance on the DUC-2004 with the model trained on the Gigaword. The ROUGE-1, ROUGE-2 and ROUGE-L recall scores in this experiment are given in Table III. The results show that the IAS model achieves the best scores than ABS/ABS+, LVT2K and Luong-NMT models. This not only shows that our proposed model is more suitable for abstractive summarization, but also shows that it has a good generalization ability.

Iteration Number: In order to evaluate whether the number of iterations K is helpful for abstractive summarization, we test all the IAS models (only in Pointer-gen.) with different K for a fair comparison environment. Considering the cost of training time, we experiment with k values from 1 to 3 for a higher performance ratio. As shown in Figure 4, it illustrates the relationship between the ROUGE score of IAS and the number of iterations on two English datasets. The model with reinforcement learning has achieved better performance on all datasets, indicating that RL is effective to train model and reduce bias. Figure 4(a) and Figure 4(b) illustrate that the ROUGE scores increase with the increase of the iteration number K increases on CNN/Daily Mail and Gigaword test dataset respectively. We can clearly find that the performance is the best when $K = 3$ (Limited K in [1,3]). The future work needs us to explore a global optimal K and let the model automatically select the optimal number of iterations. This proves that iteration is effective for abstractive summarization and can be quickly extended to other methods.

V. Conclusion

In this paper, we propose a new framework named *iterative abstractive summarization* (IAS) with reinforcement learning to generate a high-quality summary, which achieves a human-like level by learning human reading strategy. In order to learn comprehensive document representation and summary embellishment, we employ a selective gated unit and an iterative unit. We evaluate the IAS model on CNN/Daily Mail, Gigaword and DUC-2004 datasets. Experimental results show that our approach achieves significant improvements over baselines on both extractive and abstractive summarization models. Moreover, the performance of the IAS model achieves satisfactory scores on three datasets. In the future, we aim to evaluate our model on the Chinese dataset and explore the optimal iteration number automatically. In addition, we will research the more adaptive evaluation metric for automatic summarization, which has great significance to the development of automatic summarization.

Acknowledgment

We would like to thank all the reviewers for their insightful and valuable suggestions. This research is supported by the National Key Research and Development Program of China (2016YFB0801003).

References

[1] G. Rabby, S. Azad, M. Mahmud, and K. Zamli, "Teket: a tree-based unsupervised keyphrase extraction technique," *Cognitive Computation*, 03 2020.

[2] H. P. Luhn, "The automatic creation of literature abstracts," *IBM Journal of Research and Development*, vol. 2, no. 2, pp. 159–165, 1958.

[3] L. A. Leiva, "Responsive text summarization," *Information Processing Letters*, vol. 130, pp. 52–57, 2018.

[4] D. Shen, J.-T. Sun, H. Li, Q. Yang, and Z. Chen, "Document summarization using conditional random fields," in *Proc. of IJCAI*, vol. 7, 2007, pp. 2862–2867.

[5] D. Gillick, K. Riedhammer, B. Favre, and D. Hakkani-Tur, "A global optimization framework for meeting summarization," in *Proc. of ICASSP*. IEEE, 2009, pp. 4769–4772.

[6] A. P. Louis, "A bayesian method to incorporate background knowledge during automatic text summarization," in *Proc. of ACL*, 2014, pp. 333–338.

[7] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proc. of EMNLP*, 2015, pp. 379–389.

[8] R. Nallapati, B. Zhou, C. Gulcehre, B. Xiang *et al.*, "Abstractive text summarization using sequence-to-sequence rnns and beyond," in *Proc. of SIGNLL*, 2016, pp. 280–290.

[9] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proc. of ACL*, 2017, pp. 1073–1083.

[10] J. Li, C. Zhang, X. Chen, Y. Cao, P. Liao, and P. Zhang, "Abstractive text summarization with multi-head attention," in *Proc. of IJCNN*, 2019, pp. 1–8.

[11] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. of ICLR(Poster)*, 2016.

[12] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, "Self-critical sequence training for image captioning," in *Proc. of CVPR*, 2017, pp. 7008–7024.

[13] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," in *Proc. of ICLR(Poster)*, 2017.

[14] R. Nallapati, B. Zhou, C. N. dos Santos, Ç. Gülçehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence rnns and beyond," in *Proc. of SIGNLL*, 2016, pp. 280–290.

[15] S. Jean, K. Cho, R. Memisevic, and Y. Bengio, "On using very large target vocabulary for neural machine translation," in *Proc. of ACL*, 2015, pp. 1–10.

[16] Z. Cao, F. Wei, W. Li, and S. Li, "Faithful to the original: Fact aware neural abstractive summarization," in *Proc. of AAAI*, pp. 4784–4791.

[17] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proc. of ICML*, 2017, pp. 1243–1252.

[18] L. Wang, J. Yao, Y. Tao, L. Zhong, W. Liu, and Q. Du, "A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization," in *Proc. of IJCAI*, 2018, pp. 4453–4460.

[19] Z. Yang, Y. Yuan, Y. Wu, W. W. Cohen, and R. Salakhutdinov, "Review networks for caption generation," in *Proc. of NIPS*, 2016, pp. 2361–2369.

[20] Y. Xia, F. Tian, L. Wu, J. Lin, T. Qin, N. Yu, and T.-Y. Liu, "Deliberation networks: Sequence generation beyond one-pass decoding," in *Proc. of NIPS*, 2017, pp. 1784–1794.

[21] X. Chen, S. Gao, C. Tao, Y. Song, D. Zhao, and R. Yan, "Iterative document representation learning towards summarization with polishing," in *Proc. of EMNLP*, 2018, pp. 4088–4097.

[22] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proc. of ACL*, 2016, pp. 1631—1640.

[23] C. Gulcehre, S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio, "Pointing the unknown words," in *Proc. of ACL*, pp. 140—-149.

[24] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural computation*, vol. 1, no. 2, pp. 270–280, 1989.

[25] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proc. of NIPS*, 2015, pp. 1693–1701.

[26] D. Graff, J. Kong, K. Chen, and K. Maeda, "English gigaword," *Linguistic Data Consortium, Philadelphia*, vol. 4, no. 1, p. 34, 2003.

[27] C. Napoles, M. Gormley, and B. Van Durme, "Annotated gigaword," in *Proc. of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, 2012, pp. 95–100.

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

[29] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proc. of EMNLP*, 2004, pp. 404–411.

[30] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents." in *Proc. of AAAI*, 2017, pp. 3075–3081.

[31] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. of ICLR*, 2015.

[32] S. Chopra, M. Auli, and A. M. Rush, "Abstractive sentence summarization with attentive recurrent neural networks," in *Proc. of NAACL-HLT*, 2016, pp. 93–98.

[33] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. of EMNLP*, 2015, pp. 1412–1421.