Abstractive Multi-Document Summarization via Joint Learning with Single-Document Summarization

Hanqi Jin, Xiaojun Wan
Center for Data Science, Peking University
Wangxuan Institute of Computer Technology, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{jinhanqi,wanxiaojun}@pku.edu.cn

Abstract

Single-document and multi-document summarizations are very closely related in both task definition and solution method. In this work, we propose to improve neural abstractive multi-document summarization by jointly learning an abstractive single-document summarizer. We build a unified model for single-document and multi-document summarizations by fully sharing the encoder and decoder and utilizing a decoding controller to aggregate the decoder's outputs for multiple input documents. We evaluate our model on two multi-document summarization datasets: Multi-News and DUC-04. Experimental results show the efficacy of our approach, and it can substantially outperform several strong baselines. We also verify the helpfulness of single-document summarization to abstractive multi-document summarization task.

1 Introduction

Document summarization aims at producing a fluent, condensed summary for the given document or document set. It involves identifying important information and filtering out redundant information from input sources. While single-document summarization takes a single source document as input, multi-document summarization requires producing a summary from a cluster of thematically related documents. There are two primary methodologies for document summarization: *extractive* and *abstractive*. Extractive methods directly select important sentences from the original documents, which are relatively simple but face the drawbacks of information redundancy and incoherence between sentences. Abstractive methods enable generating new words, phrases, and sentences, which are able to generate better summaries with higher readability and conciseness. In this paper, we focus on abstractive document summarization.

Empowered by large parallel datasets automatically harvested from online news websites, sequence-to-sequence learning has shown promising results on abstractive single-document summarization (See et al., 2017; Paulus et al., 2018; Tan et al., 2017; Çelikyilmaz et al., 2018). Compared with single-document summarization, annotated multi-document summarization datasets are often scarce. Several works have explored adapting the neural encoder-decoder model trained for single-document summarization to multi-document summarization. Zhang et al. (2018) add a document set encoder to extend the neural abstractive model trained on large scale single-document summarization corpus to the multi-document summarization task. Lebanoff et al. (2018) incorporate the maximal marginal relevance method into a neural encoder-decoder model trained for single-document summarization to address the information redundancy for multi-document summarization.

Single-document and multi-document summarizations are very closely related in both task definition and solution method (Wan, 2010). Both tasks need to deal with document-level input, identify the important content of documents, and paraphrase the important information to generate the summary, while the main difference is that multi-document summarization involves summarizing multiple input documents. Since the two tasks are closely related, it is promising to learn for two summarization tasks jointly. Compared with single-document summarization, multi-document summarization needs to handle multiple input documents. A simple method is to concatenate multiple documents into a long flat text and treat it as a long sequence-to-sequence task. However, it blurs the boundaries between documents and loses the hierarchy within the document cluster. It is natural to regard multi-document summarization as a two-stage process

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 2545–2554

of summarizing every single document and then merging multiple summaries. Nevertheless, this process is quite trivial, and it is difficult to utilize multi-document summarization corpus to train the single-document summarization model. Furthermore, the synthesis of multiple summaries involves eliminating redundant parts and organizing related paragraphs or sentences, which are also challenges to be solved.

In this work, we propose a joint learning approach to improve neural abstractive multi-document summarization by using single-document summarization corpus to address these issues. Our approach first uses a shared document encoder to encode each document in the document set, then uses a shared decoder to predict the word probabilities for each document, and finally applies a decoding controller to aggregate all output probabilities from the summary decoder to make the final prediction at each decoding step. The shared encoder and decoder are jointly trained on the single document summarization data. In this way, we can unify single-document and multi-document summarizations into one architecture simultaneously, and make better use of single-document and multi-document corpora, so that both tasks can benefit from joint learning, especially for the multi-document summarization task.

We evaluate our approach on the benchmark multi-document summarization datasets, Multi-News and DUC-04, and it brings substantial improvements over several strong baselines for multi-document summarization. We leverage CNN/DailyMail, a single-document summarization dataset, to perform joint learning with Multi-News. We also test the performance on CNN/DailyMail test set, and joint learning also brings certain performance improvement for the single-document summarization baselines.

In summary, we make the following contributions in this paper:

- To the best of our knowledge, we are the first to explore joint learning for neural abstractive single-document and multi-document summarizations.
- We propose a unified model by fully sharing encoder and decoder and utilizing a decoding controller to aggregate the decoder's outputs for multiple input documents.
- Experimental results show that our approach substantially outperforms several strong baselines, and single document summarization is verified to be very helpful to neural abstractive multi-document summarization. Our code is publicly available at https://github.com/zhongxia96/MDS-and-SDS.

2 Related Work

2.1 Multi-Document Summarization

The methods for multi-document summarization can generally be categorized to extractive and abstractive. The extractive methods produce a summary by extracting and merging sentences from the input documents, while the abstractive methods generate a summary using arbitrary words and expressions based on the understanding of the documents. Due to the lack of available training data, most previous multi-document summarization methods were extractive (Erkan and Radev, 2004; Christensen et al., 2013; Yasunaga et al., 2017). Recently, two multi-document summarization datasets have been proposed, one for very long input, aimed at generating Wikipedia (Liu et al., 2018) and another dedicated to generating a comprehensive summary of multiple real-time news (Fabbri et al., 2019). Several works have begun to explore abstractive multi-document summarization. Liu et al. (2018) concatenated multiple source documents into a long flat text and modeled multi-document summarization as a long sequence-to-sequence task. Liu and Lapata (2019) represented cross-document relationships via an attention mechanism that allows sharing information as opposed to simply concatenating text spans and processing them as a flat sequence. Fabbri et al. (2019) incorporated MMR into a hierarchical pointer-generator network to address the information redundancy in multi-document summarization. The above works were all trained and tested on multi-document summarization corpus.

2.2 Adaptation Method from Single to Multi-Document Summarization

Since the neural abstractive models have achieved promising results on single-document summarization (See et al., 2017; Paulus et al., 2018; Gehrmann et al., 2018; Çelikyilmaz et al., 2018), some works trained abstractive summarization models on a single document dataset and adjusted the model to adapt the multi-document summarization task. Zhang et al. (2018) added a document set en-

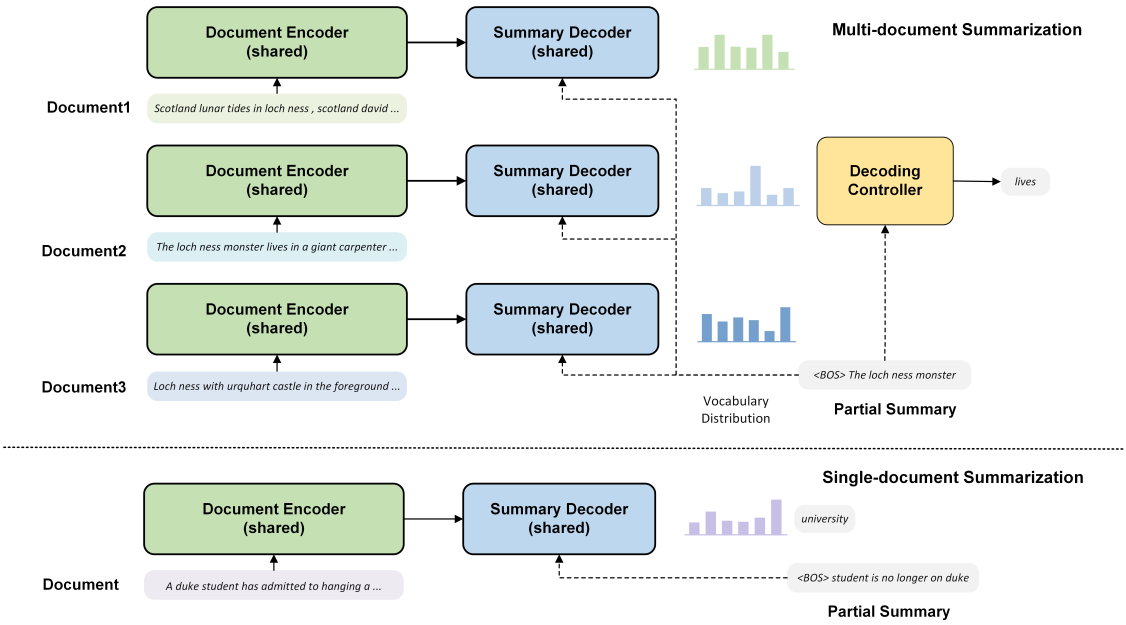

Figure 1: The overview of our model.

coder into the single document summarization framework and tuned the pre-trained model on the multi-document summarization dataset. Lebanoff et al. (2018) combined an extractive summarization algorithm (MMR) for sentence extraction to reweigh the original sentence importance distribution learned in the abstractive single document summarization model. In this work, we propose to jointly learn for two summarization tasks and build a unified model. It utilizes a shared encoder-decoder to summarize a document and use a decoding controller to aggregate all decoders' outputs. Compared with the above adaptation methods, our method can make better use of multi-document and single-document corpora and improve the effectiveness of single-document summarization at the same time.

3 Methodology

3.1 Overview and Notations

Multi-document summarization takes a document cluster $D = \{D_1, D_2, \ldots, D_I\}$ as the input, and produces the summary Y, where I is the number of documents. Each document $D_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,N_i})$ is a sequence of N_i words, and $Y = (y_1, y_2, \ldots, y_M)$ is a sequence of M words. Compared with multi-document summarization, single-document summarization has only one input document. In order to unify the symbols, single-document summarization is regarded as a special input case of $I = 1$.

As illustrated in Figure 1, our model consists of a document encoder, a summary decoder, and a decoding controller. Different documents in multi-document summarization share document encoder and summary decoder. Single-document summarization also shares document encoder and summary decoder with multi-document summarization. A decoding controller is applied to aggregate the outputs of the summary decoder for multiple input documents.

The shared document encoder reads each input document D_i and builds the contextual-level representations C_i.

$$C_i = encoder(D_i) \quad (1)$$

In each decoding step t, the shared summary decoder produces the vocabulary distribution of the next word given previously (predicted) words and each input document D_i.

$$P_i^t = decoder\left(C_i, y_{1:t-1}\right) \quad (2)$$

Note that for multi-document summarization, the same sequence of previous words $y_{1:t-1}$ (i.e., partial summary) is used for decoding for every document of the multiple inputs.

Since single-document summarization only summarizes one input document, the summary decoder can make the final prediction based on the output vocabulary distribution. While for multi-document summarization, a decoding controller is applied to aggregate multiple vocabulary distributions from

the summary decoder for multiple input documents.

$$P_f^t = \sum_{i=1}^{I} P_i^t z_i^t \tag{3}$$

Here z_i^t is the importance weight for each of the multiple vocabulary distributions in the t-th step.

The following sections will introduce the document encoder, the summary decoder, and the decoding controller, respectively.

3.2 Document Encoder

Document encoder reads an input document D_i and constructs its contextual-level representation. For multi-document summarization, multiple input documents can be processed in parallel. This part is the same as Transformer encoder (Vaswani et al., 2017), and we will give a brief introduction. The document encoder is composed of a stack of L identical layers. Each layer has two sub-layers, where the first sub-layer is a multi-head self-attention mechanism, and the second sub-layer is a position-wise fully connected feed-forward network. A residual connection (He et al., 2016) is employed around each of the two sub-layers, followed by layer normalization (Ba et al., 2016).

Tokens of each input document are first represented by word embeddings. Let $e_{i,j}$ denote the embedding assigned to word $x_{i,j}$. Since the Transformer is a non-recurrent model, we need to add the "positional embedding" p_j to the word embedding to indicate the position of the word in the document, and the input representation can be obtained by simply adding these two representations: $w_{i,j} = e_{i,j} + p_j$. We take $\{w_{i,1}, w_{i,2}, \ldots, w_{i,N_i}\}$ as the input to the document encoder. For convenience, we denote the input of the first layer as h^0 and the output of l-th layer as h^l. The multi-head self-attention sub-layer takes the output of the previous layer as the input to construct contextual-level representation, while the FFN sub-layer is used to transform the representation further.

$$\begin{aligned} \tilde{h} &= \text{LayerNorm}(h^{l-1} + \text{MHAtt}(h^{l-1}, h^{l-1})) \\ h^l &= \text{LayerNorm}(\tilde{h} + \text{FFN}(\tilde{h})) \end{aligned} \tag{4}$$

The final output h^L is fed to the summary decoder, and it is also fed to the decoding controller for multi-document summarization. For convenience, we denote the output for the document D_i as C_i.

3.3 Summary Decoder

In each decoding step, the summary decoder takes the decoded subsequences $(y_1, y_2, \cdots, y_{t-1})$ as the input, and predicts the probability distribution of generating the next word for each input document D_i. Similar to the document encoder, the summary decoder is also a stack of L identical layers. The layer consists of three sub-layers: masked multi-head self-attention mechanism, multi-head cross-attention mechanism over the output of the encoder stack, and position-wise feed-forward network.

We also need to add "positional embedding" to the word embedding in the same way as the document encoder. Let d^l denote the output of the l-th layer in the summary decoder, and the input for the first layer as d^0. The masked multi-head self-attention sub-layer is used for encoding the information of the decoded subsequences. The output of the self-attention is fed to the cross-attention sub-layer and feed-forward network. The cross-attention sub-layer performs multi-head attention over the output C_i of the document encoder.

$$\begin{aligned} \tilde{d} &= \text{LayerNorm}\left(d^{l-1} + \text{MHAtt}(d^{l-1}, d^{l-1})\right) \\ g &= \text{LayerNorm}\left(\tilde{d} + \text{MHAtt}(\tilde{d}, C_i)\right) \\ d^l &= \text{LayerNorm}\left(g + \text{FFN}(g)\right) \end{aligned} \tag{5}$$

Let U_i^t denote the output of the L-th layer for document D_i at position t.

The output U_i^t is passed through a softmax layer to calculate the generation distribution of next word over the target vocabulary.

$$\hat{P}_i^t = \text{softmax}\left(U_i^t W_g + b_g\right) \tag{6}$$

where $W_g \in \mathbb{R}^{d_{model} \times d_{vocab}}, b_g \in \mathbb{R}^{d_{vocab}}$ and d_{vocab} is the size of target vocabulary. To tackle the problem of out-of-vocabulary (OOV) words, we compute the copy attention ε_i^t between U_i^t and the input representations C_i to allow copying words from the source text, and obtain the copy distribution (Gu et al., 2016).

$$\begin{aligned} \varepsilon_i^t &= \text{softmax}(U_i^t C_i^\top) \\ \tilde{P}_i^t &= \sum_{j=1}^{N_i} \varepsilon_{i,j}^t o_{i,j} \end{aligned} \tag{7}$$

where $o_{i,j}$ is the one-hot indicator vector for $w_{i,j}$.

The generation probability $\eta_i^t \in [0, 1]$ is calculated from the decoder output U_i^t.

$$\eta_i^t = \sigma\left(U_i^t W_\eta + b_\eta\right) \tag{8}$$

where $W_\eta \in \mathbb{R}^{d_{model} \times 1}, b_\eta \in \mathbb{R}^1$. The overall distribution for document D_i is given by combining the two distributions with η_i^t.

$$P_i^t = \eta_i^t * \hat{P}_i^t + (1 - \eta_i^t) * \tilde{P}_i^t \tag{9}$$

3.4 Decoding Controller

Multi-document summarization requires producing a summary for a cluster of thematically related documents. While the summary decoder has predicted the vocabulary distribution for each input document, the decoding controller aggregates multiple vocabulary distributions to predict the final vocabulary distribution for multi-document summarization. Figure 2 shows a example. To better aggregate multiple vocabulary distributions, the controller needs to grasp the theme of the document cluster. We first use an attention pooling over the document encoder outputs to obtain corresponding document representation, and adopt a bidirectional LSTM (Hochreiter and Schmidhuber, 1997; Schuster and Paliwal, 1997) to encode multiple document representations in the document cluster. Then, we take the output of the bidirectional LSTM as the initial state of another unidirectional LSTM, which will be used to calculate the weights that the next word comes from each document.

Attention Pooling The attention pooling operation is used over the contextual-level representations $C_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,N_i})$ to obtain a fixed-length representation $\hat{c}_i$ for document D_i. We first transform the input vector $c_{i,j}$ into attention score $a_{i,j}$ and value vector $v_{i,j}$. Then we calculate a probability distribution $\hat{a}_i$ over words within the document D_i based on attention scores.

$$\begin{aligned} a_{i,j} &= c_{i,j} W_a \\ v_{i,j} &= c_{i,j} W_v \\ \hat{a}_{i,j} &= \frac{\exp(a_{i,j})}{\sum_{j=1}^{n} \exp(a_{i,j})} \end{aligned} \tag{10}$$

where $W_a \in \mathbb{R}^{d_{model} \times 1}$ and $W_v \in \mathbb{R}^{d_{model} \times d_{model}}$. Finally, we get the document vector $\hat{c}_i$ by weighing the value vectors.

$$\hat{c}_i = \sum_{j=1}^{n} \hat{a}_{i,j} v_{i,j} \tag{11}$$

A bidirectional LSTM is adopted to further encode document representations $\{\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_I\}$. The forward LSTM reads the document context representations from left to right and gets a sequence of hidden states $\left(\overrightarrow{f}_1, \overrightarrow{f}_2, \ldots, \overrightarrow{f}_I\right)$. The backward LSTM reads the document context representations reversely, from right to left, and results in another sequence of hidden states $\left(\overleftarrow{f}_1, \overleftarrow{f}_2, \ldots, \overleftarrow{f}_I\right)$. We add the last forward hidden state $\overrightarrow{f}_I$ and backward hidden state $\overleftarrow{f}_1$ as the output r of the bidirectional LSTM.

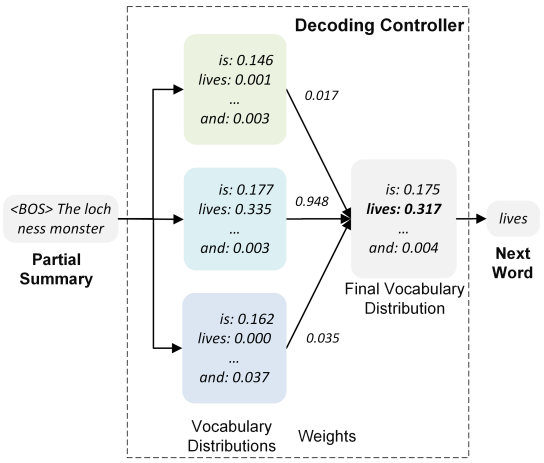

Figure 2: The decoding controller weighs the multiple output distributions to predict the next word. If simply averaging the vocabulary distributions, we will get the word "is". And we can get the correct word "lives" by calculating and using the weights through the decoding controller.

$$r = \overleftarrow{f}_1 + \overrightarrow{f}_I \tag{12}$$

The output r is used as the initial state of another unidirectional LSTM. In the decoding step t, the unidirectional LSTM takes the previous word y_{t-1} as input and produces the new state s_t.

$$s_t = \text{LSTM}(s_{t-1}, y_{t-1}) \tag{13}$$

We calculate the weights z^t using s_t and decoder outputs $U^t = \{U_1^t, U_2^t, \cdots, U_I^t\}$:

$$z^t = \text{softmax}\left(U^t W_z s_t^\top\right) \tag{14}$$

where $W_z \in \mathbb{R}^{d_{model} \times d_{model}}$.

The final vocabulary distribution for multi-document summary generation is the interpolation of all output distributions.

$$P_f^t = \sum_{i=1}^{I} P_i^t z_i^t \tag{15}$$

3.5 Objective Function

We jointly learn the single-document and multi-document summarizations in a unified model. Our

goal is to maximize the probability of output summary Y given a single document S or a document set D. We use $\mathcal{T}_s$ to denote the single-document training set and $\mathcal{T}_m$ to denote the multi-document training set. We calculate negative logarithm likelihood function for single-document and multi-document summarizations, respectively.

$$\begin{aligned} L_s &= -\frac{1}{|\mathcal{T}_s|} \sum_{(S,Y)\in\mathcal{T}_s} \log P(Y|S) \\ L_m &= -\frac{1}{|\mathcal{T}_m|} \sum_{(D,Y)\in\mathcal{T}_m} \log P(Y|D) \end{aligned} \tag{16}$$

For simplicity, we optimize the sum of the above losses.

4 Experiment

4.1 Datasets

We conduct experiments on a latest released Multi-News dataset (Fabbri et al., 2019) and a standard DUC multi-document summarization dataset (Over et al., 2007). The Multi-News dataset contains 44,972 documents-summary pairs for training, 5,622 for development, and 5,622 for test. The number of source documents per summary ranges from 2 to 10. DUC-03 and DUC-04 contain 30 and 50 topics, respectively. Each topic has 10 documents paired with 4 different human-written references. CNN/Dailymail (Hermann et al., 2015; Nallapati et al., 2016) is a large scale single document summarization dataset, which contains 287,226 document-summary pairs for training, 13,368 for development and 11,490 for test.

4.2 Implementation Details

We train the model on the Multi-News and CNN/DailyMail datasets. Considering that different datasets have different expression characteristics, we set different BOS for each dataset in the decoding phase. We take the DUC-04 as the test set, and DUC-03 is used for tuning the model when evaluating on DUC-04 dataset. We set our model parameters based on preliminary experiments on the Multi-News and CNN/DailyMail development set. We prune the vocabulary to 50k and use the word in source text with maximum weights in copy attention to replacing the unknown word to solve the OOVs problem. We set the dimension of word embeddings and hidden units d_{model} to 512, feed-forward units to 1024. We set 4 heads for multi-head self-attention, masked multi-head self-attention, and multi-head cross-attention. The number of layers L is set to 6. We set dropout rate to 0.1 and use Adam optimizer with an initial learning rate $\alpha = 0.0001$, momentum $\beta_1 = 0.9$, $\beta_2 = 0.999$ and weight decay $\epsilon = 10^{-5}$. The learning rate is halved if the valid loss on the development set increases for two consecutive epochs. We use a mini-batch size of 10. Beam search with a beam size of 5 is used for decoding.

4.3 Metrics and Baselines

We use ROUGE (Lin, 2004) to evaluate the generated summary in our experiments. Following previous work, we report ROUGE F1[1] on Multi-News and DUC-04 datasets. We compare our model with several typical baselines and several baselines proposed in the latest years.

PGN (See et al., 2017) is an RNN based model with an attention mechanism and allows the system to copy words from the source text via pointing for abstractive summarization. **CopyTransformer** (Gehrmann et al., 2018) augments Transformer with one of the attention heads chosen randomly as the copy distribution. **Hi-MAP** (Fabbri et al., 2019) expands the pointer-generator network model into a hierarchical network and integrates an MMR module to calculate sentence-level scores. The above baselines are trained on the Multi-News corpus, and have been compared and reported in Fabbri et al. (2019), which releases the Multi-News dataset. We directly report the results of the above methods from this paper. **PG-MMR** (Lebanoff et al., 2018) combines MMR with the abstractive model trained on CNN/DailyMail corpus to generate the summary from multi-document inputs, which requires no multi-document summarization training corpus. **SDS-to-MDS** (Zhang et al., 2018) is an approach to extend the neural abstractive model trained on CNN/DailyMail dataset to the multi-document summarization task, which leverages multi-document summarization corpus to tune the pre-trained single-document summarization model. It originally conducts experiments on the DUC datasets, and we also reproduce their method on the Multi-News dataset. Besides, we implement **CopyTransformer*** to jointly learn single-document and multi-document summarizations, and train it on the CNN/DailyMail and Multi-News corpora. It concatenates the multiple

[1]The ROUGE evaluation option: -c 95 -2 4 -U -r 1000 -n 4 -w 1.2 -a

Model	R-1	R-2	R-SU4
LexRank (Erkan and Radev, 2004)	38.27	12.70	13.20
TextRank (Mihalcea and Tarau, 2004)	38.44	13.10	13.50
MMR(Carbonell and Goldstein, 1998)	38.77	11.98	12.91
PGN (See et al., 2017)	41.85	12.91	16.46
CopyTransformer(Gehrmann et al., 2018)	43.57	14.03	17.37
Hi-MAP(Fabbri et al., 2019)	43.47	14.89	17.41
SDS-to-MDS(Zhang et al., 2018)	44.74	15.93	19.44
CopyTransformer*	45.03	16.35	19.59
Ours	**46.26**	**17.02**	**20.46**

Table 1: ROUGE F1 evaluation results on the Multi-News test set.

Model	R-1	R-2	R-SU4
LexRank (Erkan and Radev, 2004)	35.56	7.87	11.86
TextRank (Mihalcea and Tarau, 2004)	33.16	6.13	10.16
MMR(Carbonell and Goldstein, 1998)	30.14	4.55	8.16
PGN (See et al., 2017)	31.43	6.03	10.01
CopyTransformer(Gehrmann et al., 2018)	28.54	6.38	7.22
PG-MMR(Lebanoff et al., 2018)	36.42	**9.36**	**13.23**
Hi-MAP(Fabbri et al., 2019)	35.78	8.90	11.43
SDS-to-MDS(Zhang et al., 2018)	36.7	7.83	12.4
CopyTransformer*	36.48	8.22	12.29
Ours	**37.24**	8.60	12.67

Table 2: ROUGE F1 evaluation results on the DUC-04 dataset.

input documents into a long flat text, and treats multi-document summarization as a long single-document summarization task. The best hyperparameter configuration is chosen for each model.

4.4 Automatic Evaluation

Following previous work, we report ROUGE-1 (unigram), ROUGE-2 (bigram) and ROUGE-SU4 (skip bigrams with a maximum distance of 4 words) scores as the metrics for automatic evaluation (Lin and Hovy, 2003). In Table 1, we report the results on the Multi-News, and our proposed model outperforms various baseline models. **CopyTransformer** performs much better than **PGN** and achieves 1.72 points improvement on the ROUGE-1 F1, which demonstrates the superiority of the Transformer architecture. The methods of leveraging single-document corpus (i.e., **SDS-to-MDS**, **CopyTransformer***, and ours) perform much better than that of only training on multi-document corpus (i.e., **PGN**, **CopyTransformer**, and **Hi-MAP**). Our model gains an improvement of 1.52 points compared with **SDS-to-MDS**, 1.23 points compared with **CopyTransformer*** on ROUGE-1 F1, which verifies the effectiveness of the proposed architecture for the multi-document summarization task.

In Table 2, we report the results on the DUC-04 test set. Our model achieves scores of 37.24, 8.60 and 12.67 on three ROUGE metrics, respectively. **PG-MMR** and **Hi-MAP** obtain the higher score on ROUGE-2 or ROUGE-SU4 F1, while they employ the MMR technique to avoid the redundancy further. Our proposed model achieves the best performances on ROUGE-1 F1 among all compared models. It indicates our proposed model has a good transferability between different datasets.

4.5 Human Evaluation

To further evaluate the quality of the generated summaries, we carry out a human evaluation. We focus on three aspects: **fluency**, **informativeness**, and **non-redundancy**. The fluency indicator focuses on whether the summary is well-formed and grammatical. The informativeness indicator can reflect whether the summary covers salient points from the input documents. The non-redundancy indicator measures whether the summary contains repeated information. We sample 50 instances from the Multi-News test set and employ five graduate students to rate each summary. Each human judgment evaluates all outputs of different systems for the same sample. Three human judgments are obtained for every sample, and the final scores are averaged across different judges.

Results are presented in Table 3. We can see that our model performs much better than all baselines. The Spearman correlation coefficients between annotators are high, which guarantees the validity of the human evaluation. In the fluency indicator, our model achieves a high score of 3.5, which is higher than 3.42 of **CopyTransfromer*** and 3.3 of **SDS-to-MDS**, indicating that our model can reduce the grammatical errors and improve the readability of the summary. In the informativeness indicator, our model is higher than **CopyTransfromer*** by 0.16 and **SDS-to-MDS** by 0.2, which indicates that our model can effectively capture the salient information. In the non-redundancy indicator, our model also outperforms all baselines. It indicates our proposed method can better avoid repeating information of the generated summary.

4.6 Ablation Study

We perform the ablation study to investigate the influence of joint learning with single-document summarization and the effectiveness of the decoding controller. First, we train the model only on the Multi-News dataset to verify the helpfulness

Model	Fluency	Informativeness	Non-redundancy
CopyTransformer(Gehrmann et al., 2018)	3.1	3.08	2.94
Hi-MAP(Fabbri et al., 2019)	2.98	2.94	3.02
SDS-to-MDS(Zhang et al., 2018)	3.3	3.22	3.18
CopyTransformer*	3.42	3.26	3.24
Ours	**3.5**	**3.42**	**3.36**
Spearman	0.732	0.715	0.698

Table 3: Human evaluation. The ratings are on a Liert scale of 1(worst) to 5(best).

Model	R-1	R-2	R-SU4
Ours	46.26	17.02	20.46
w/o joint learning	44.64	16.14	19.06
w/o decoding controller	44.94	16.07	19.11

Table 4: Results of ablation study on the Multi-News test set.

Model	R-1	R-2	R-L
Lead-3	40.34	17.70	36.57
PGN (See et al., 2017)	39.53	17.28	36.38
CopyTransformer	40.68	18.26	37.38
CopyTransformer*	41.39	18.58	38.03
Ours	**41.7**	**18.86**	**38.36**

Table 5: ROUGE F1 evaluation results on the CNN/DailyMail test set.

of single-document summarization to abstractive multi-document summarization task. Then we replace the decoding controller with a fixed weight vector $z = [1/I, \cdots, 1/I]$ by simply averaging the vocabulary distributions from the summary decoder to verify the effectiveness of the decoding controller.

Table 4 presents the results. We find that the ROUGE-1 F1 score drops by 1.62 and the ROUGE-2 F1 score drops by 0.88 when training the model only on the Multi-News dataset. It indicates joint learning with single-document summarization is beneficial to the multi-document summarization. ROUGE-1 F1 score drops by 1.32 and ROUGE-2 F1 score drops by 0.95 after the decoding controller is removed, which shows that the decoding controller can effectively aggregate the outputs of the summary decoder for multiple input documents.

4.7 Discussion

Performance on Single-Document Summarization In Table 5, we report the results on CNN/DailyMail test set. **CopyTransformer*** outperforms **CopyTransformer** by 0.71 points on ROUGE-1 F1, which indicates joint learning can also improve the performance for single-document summarization. Compared with the **CopyTransformer***, our method gains an improvement of 0.31 points on ROUGE-1 F1, which indicates our method can make better use of multi-document corpus to improve the performance for single-document summarization.

Performance against the Document Number of Inputs Different document number of inputs may affect the summarization performance, so we further test our model and strong baseline

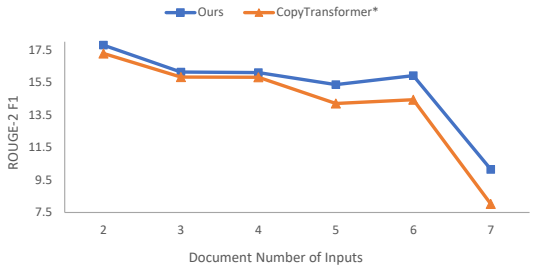

Figure 3: ROUGE-2 F1 score on different document number of inputs for CopyTransformer* baseline and our model on Multi-News test set.

CopyTransformer* with respect to different document number of inputs on the Multi-News test set. The document number of inputs in the test sets ranges from 2 to 7. In Figure 3, we can see that the performances of both models drop when the number of input documents increases. The performance curve of our model always appears on the top of that of **CopyTransformer***, and our model can get better results in the case of more documents than **CopyTransformer***.

5 Conclusion and Future Work

In this paper, we propose a joint learning approach to improve neural abstractive multi-document summarization by using single-document summarization dataset. Specifically, we use the shared document encoder and summary decoder to process each document in the document set, and apply a decoding controller to aggregates all output probabilities from the summary decoder for multi-document summarization. The shared encoder and decoder are jointly trained on the single document sum-

marization dataset. Experimental results show that our approach substantially outperforms several strong multi-document summarization baselines and achieves state-of-the-art or very competitive performances on Multi-News and DUC-04 datasets.

In the future, we will incorporate BERT or other pre-trained language models into our model to further improve the performance.

Acknowledgments

This work was supported by National Natural Science Foundation of China (61772036), Beijing Academy of Artificial Intelligence (BAAI) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Jaime G. Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*, pages 335–336. ACM.

Asli Çelikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 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)*, pages 1662–1675. Association for Computational Linguistics.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 1163–1173. The Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.*, 22:457–479.

Alexander Richard Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. 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*, pages 1074–1084. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 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*, pages 4098–4109. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 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*, pages 1693–1701.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Logan Lebanoff, Kaiqiang Song, and Fei Liu. 2018. Adapting the neural encoder-decoder framework from single to multi-document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4131–4141. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chin-Yew Lin and Eduard H. Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics.

Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Yang Liu and Mirella Lapata. 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*, pages 5070–5081. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 404–411. ACL.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 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*, pages 280–290. ACL.

Paul Over, Hoa Dang, and Donna Harman. 2007. DUC in context. *Inf. Process. Manage.*, 43(6).

Romain Paulus, Caiming Xiong, and Richard Socher. 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.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45(11):2673–2681.

Abigail See, Peter J. Liu, and Christopher D. Manning. 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*, pages 1073–1083. Association for Computational Linguistics.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 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*, pages 1171–1181. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 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*, pages 5998–6008.

Xiaojun Wan. 2010. Towards a unified approach to simultaneous single-document and multi-document summarizations. In *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*, pages 1137–1145. Tsinghua University Press.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir R. Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 452–462. Association for Computational Linguistics.

Jianmin Zhang, Jiwei Tan, and Xiaojun Wan. 2018. Towards a neural network approach to abstractive multi-document summarization. *CoRR*, abs/1804.09010.