Friendly Topic Assistant for Transformer Based Abstractive Summarization

Zhengjue Wang*[1], Zhibin Duan*[1], Hao Zhang†[1], Chaojie Wang[1], Long Tian[1], Bo Chen†[1], Mingyuan Zhou[2]
[1]National Laboratory of Radar Signal Processing, Xidian University, Xi'an, China
[2]McCombs School of Business The University of Texas at Austin, Austin, TX 78712, USA
{zhengjuewang, zhanghao_xidian}@163.com
bchen@mail.xidian.edu.cn, Mingyuan.Zhou@mccombs.utexas.edu

Abstract

Abstractive document summarization is a comprehensive task including document understanding and summary generation, in which area Transformer-based models have achieved the state-of-the-art performance. Compared with Transformers, topic models are better at learning explicit document semantics, and hence could be integrated into Transformers to further boost their performance. To this end, we rearrange and explore the semantics learned by a topic model, and then propose a topic assistant (TA) including three modules. TA is compatible with various Transformer-based models and user-friendly since i) TA is a plug-and-play model that does not break any structure of the original Transformer network, making users easily fine-tune Transformer+TA based on a well pre-trained model; ii) TA only introduces a small number of extra parameters. Experimental results on three datasets demonstrate that TA is able to improve the performance of several Transformer-based models.

1 Introduction

Automatic summarization, requiring both document understanding and text generation, is a comprehensive task in natural language processing (NLP). Extractive approaches (Wong et al., 2008; Liu, 2019; Zhang et al., 2019c) identify and then concatenate the most representative sentences as a summary. By contrast, abstractive summarization (See et al., 2017; Narayan et al., 2018) is more challenging, aiming to generate a summary via rephrasing and introducing new concepts/words. Our work focuses on abstractive summarization, for which sequence-to-sequence (S2S) models are widely studied.

Recently, equipped with the attention mechanism (Vaswani et al., 2017), some Transformer-based language models (Subramanian et al., 2019; Zhang et al., 2019b; Dong et al., 2019; Liu and Lapata, 2019; Lewis et al., 2019; Raffel et al., 2019) are built with an encoder-decoder structure. These models benefit from pre-training on large-scale corpus, and then are fine-tuned to adapt to the summarization task. As a result, the encoder with bidirectional self-attention (SA) extracts document-token features, the decoder with left-to-right SA generates the summary, and the cross attention (CA) bridges the document and summary tokens.

Though achieving appealing performances, these Transformer-based models are better at exploring the relationships among local tokens than the document global semantics. Further, due to the limited position index during pre-training, most Transformer-based models have a maximum capacity of input tokens. Thus, they often truncate the length of a document to satisfy the length limitation of the encoder, which may lose some important semantics, especially for long documents.

Global semantics are important to summarization (Narayan et al., 2018; Ailem et al., 2019; Liu et al., 2019), since one need to comprehend the entire content before generate summaries. Compared with language models, topic models tell global semantics more explicitly. Basically, topic models, such as LDA (Blei et al., 2003) and PFA (Zhou et al., 2012), represent each document as a bag-of-word (BOW) vector and then factor the count vector as a product of topics and topic proportions, as shown in Fig. 1. Topics are global variables, describing the distributions over all tokens in the vocabulary. Topic proportions are local (document-specific) features, describing the weights of corresponding topics in each document. Therefore, topic models explore the *word co-occurrence patterns*, *i.e.*, semantics. However, no Transformer-based

* Equal contribution. † Corresponding author.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 485–497,
November 16–20, 2020. ©2020 Association for Computational Linguistics

model considers these explicit semantics.

In this paper, we rearrange and further explore the semantics of the topic model and develop a friendly topic assistant (TA) for Transformer-based abstractive summarization models. By introducing only a small number of parameters into the fine-tuning stage, TA is a flexible plug-and-play model, consisting of three modules:

- **Semantic-informed attention (SIA):** It is often observed that the learned attentive patterns of many heads are not as reasonable as we expect (Clark et al., 2019; Michel et al., 2019). This motivates us to employ the semantic "*distribution over topics*" as a token representation to construct an explicit semantic-similarity matrix among tokens, which is further used as the attention weights of a newly added head.
- **Topic embedding with masked attention (TEMA):** Since a topic is a distribution over tokens in the vocabulary, we use the mixture of token embeddings to represent the corresponding topic embedding. Thus, topics with large proportions for a document can be considered as extra input tokens of the decoder. Further, a topic describes a co-occurrence pattern of tokens with similar semantics, that is more likely to help the decoder to generate new tokens or concepts not included in the current document. To prevent the topic features affected by the summary-token features via attention, we perform masked attention in the decoder.
- **Document-related modulation (DRM):** Conditional biasing is an efficient way to integrate conditions into the network with a small number of extra parameters (Dumoulin et al., 2018). The topic-proportion vector is a low-dimensional document representation, conditioned on which we infer a document-related bias to modulate some hidden layers of the decoder.

TA does not break any structure of the original Transformer network, and hence is able to be jointly learned with a pre-trained model during the fine-tuning stage. Besides, SIA, TEMA, and DRM are cooperated with some basic Transformer modules, such as embedding and multi-head attention. Therefore, we can plug an arbitrary combination of these three modules into various Transformer-based models.

2 Related work

2.1 Transformer-based models for document summarization

Pre-training and fine-tuning have attracted much attention in Transformer-based models for various NLP tasks. Equipped with pre-trained Bert encoder (Devlin et al., 2019), Liu (2019); Liu and Lapata (2019) propose the BertSUM for both extractive and abstractive tasks; Zhang et al. (2019c) propose a hierarchical Bert model for extractive summarization, where the low-level and high-level Berts are built for sentence and document understanding, respectively.

Although the above methods achieve better performance than LSTM-based models, their Bert encoder pre-trained for document understanding may not well match the decoder trained from scratch for the summary generation (Rothe et al., 2019; Yang et al., 2019). To consider document understanding and generation in a unified framework, some S2S pre-training models are proposed for general purpose, such as MASS (Song et al., 2019), UniLM (Dong et al., 2019), T5 (Raffel et al., 2019), and BART (Lewis et al., 2019), which are further fine-tuned for downstream tasks, summarization included. Aiming at designing a pre-training objective tailored for abstractive text summarization, Zhang et al. (2019b) propose the PEGASUS that achieves the state-of-the-art performance.

2.2 S2S models combined with Topic models

To complement global semantics for S2S models that often focus on sequential information, topic models (Blei et al., 2003; Zhou et al., 2012) are considered to be combined with S2S models. Zhang et al. (2016) represent each word as a distribution over topics, and construct a topic-informed RNN model for neural machine translation. Based on the RNN-based pointer-generator network (See et al., 2017), Ailem et al. (2019) develop a topic augmented decoder that generates a summary conditioned on both the input document and the latent topics of the document. They find that the latent topics reveal more global semantic information that can be used to bias the decoder to generate words. With similar considerations, Narayan et al. (2018) propose another topic-conditioned S2S model under the CNN framework.

Although these models have demonstrated the advantages of combining S2S learning with topic models, integrating topic information into

Transformer-based summarization models is still an underexplored research area.

3 Background

TA is a friendly plug-and-play model that is compatible with many transformer-based summarization models. To illustrate TA without loss of generality, we choose the BertSUM (Liu and Lapata, 2019) as an example Transformer-based model, and PFA (Zhou et al., 2012) as an example topic model.

3.1 BertSUM: a Transformer-based summarization model

Given a data pair $\{\boldsymbol{x}, \boldsymbol{y}\}$, where the document $\boldsymbol{x}$ has N_1 tokens and the summary $\boldsymbol{y}$ has N_2 tokens ($N_2 < N_1$), BertSUM maximizes the following likelihood

$$\prod_{j=1}^{N_2} p(\boldsymbol{y}_j | \{\boldsymbol{x}_i\}_{i=1}^{N_1}, \boldsymbol{y}_{i<j}), \tag{1}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ denote the i-th token in document and summary, respectively.

BertSUM adopts an encoder-decoder architecture, as shown in Fig. 1. The encoder is a pre-trained twelve-layer Bert (Devlin et al., 2019), each layer mainly including a bidirectional SA and a fully-connected network (FNN). The encoder outputs the document-token features $\mathbf{H} \in \mathbb{R}^{N_1 \times d_{\text{model}}}$ at the top layer, where d_{model} is the output dimension of each module (*e.g.*, embedding, SA, CA, and FNN) in Transformer. The decoder is a randomly-initialized six-layer Transformer decoder (Vaswani et al., 2017), each layer mainly including SA, CA, and FNN. Due to the auto-regressive nature of the summary generation in (1), the decoder performs left-to-right SA. The CA forces the summary-token features to attend over all features in $\mathbf{H}$.

3.2 Poisson factor analysis (PFA)

Topic models are good at capturing global semantics of texts (Zhang et al., 2019a; Wang et al., 2020). PFA (Zhou et al., 2012) is a typical topic model inferred by Gibbs sampling or variational autoencoder (Zhang et al., 2018). Specifically, representing document $\boldsymbol{x}$ as a BOW vector $\boldsymbol{b} \in \mathbb{Z}^V$, where $\mathbb{Z} = \{0, 1, \cdots\}$ and V is the vocabulary size, PFA models $\boldsymbol{b}$ under the Poisson likelihood as

$$\boldsymbol{b} \sim \text{Poisson}\left(\boldsymbol{\Phi\theta}\right), \boldsymbol{\theta} \sim \text{Gamma}\left(\boldsymbol{r}, 1\right). \tag{2}$$

In (2), the k-th column of $\boldsymbol{\Phi} \in \mathbb{R}_+^{V \times K}$, denoted as $\boldsymbol{\phi}_k \in \mathbb{R}_+^V$, represents the k-th topic, which is a distribution over all tokens in the vocabulary. For this purpose, PFA applies a Dirichlet prior on $\boldsymbol{\phi}_k$ as $\boldsymbol{\phi}_k \sim \text{Dirichlet}\left(\boldsymbol{\eta}_k\right)$. $\boldsymbol{\theta} \in \mathbb{R}^K$ is the document-specific topic proportion vector (document feature) that represents the strength of the document on each topic. Thus, using the law of total expectation on (2), we have $\mathbb{E}\left[\boldsymbol{b} \,|\, \boldsymbol{\Phi}, \boldsymbol{\theta}\right] = \boldsymbol{\Phi\theta}$, which means that a document can be decomposed as a weighted summation of topics, as illustrated in Fig. 1.

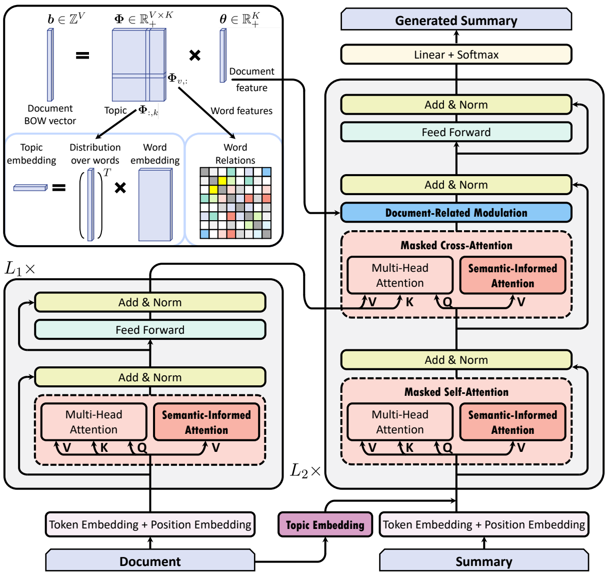


Figure 1: The structure of BertSUM with TA, where the names in bold are our proposed modules in TA.

4 Topic assistant for Transformer

Given a corpus, we train a PFA based on documents. Then, we use the extracted topics and topic proportions to build three plug-and-play modules to help the Transformer fine-tuning, including semantic-informed attention, topic embedding with masked attention, and document-related modulation.

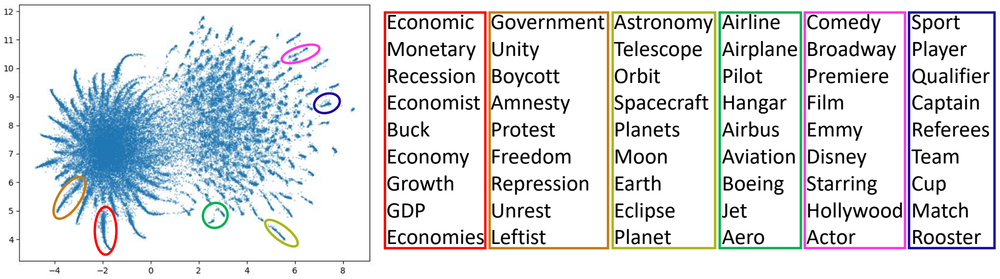
4.1 Semantic-informed attention (SIA)

In Transformer-based models, the multi-head attention explores the relationships among tokens by calculating the token similarities in implicit feature spaces. Specifically, assume we have h heads, thus the attention function $\text{Att}(\cdot)$ in the i-th head is formulated as:

$$\text{head}_i = \text{Att}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = \mathbf{A}_i \mathbf{V}_i, \tag{3}$$

$$\mathbf{A}_i = \text{softmax}\left(\frac{\mathbf{Q}_i {\mathbf{K}_i}^T}{\sqrt{d_k}}\right), i = 1, \cdots, h, \tag{4}$$

where, $\mathbf{A}_i$ is the attention matrix, $\mathbf{Q}_i$, $\mathbf{K}_i$, and $\mathbf{V}_i$ are learnable features, denoting queries, keys, and



Economic	Government	Astronomy	Airline	Comedy	Sport
Monetary	Unity	Telescope	Airplane	Broadway	Player
Recession	Boycott	Orbit	Pilot	Premiere	Qualifier
Economist	Amnesty	Spacecraft	Hangar	Film	Captain
Buck	Protest	Planets	Airbus	Emmy	Referees
Economy	Freedom	Moon	Aviation	Disney	Team
Growth	Repression	Earth	Boeing	Starring	Cup
GDP	Unrest	Eclipse	Jet	Hollywood	Match
Economies	Leftist	Planet	Aero	Actor	Rooster

Figure 2: Distribution of tokens represented by $\boldsymbol{\varphi}_v$ in SIA, learned on CNN/DM using PFA and (5).

values, respectively, d_k (d_v) is the dimension of the queries or keys (values).

However, recent works have illustrated that most attention heads learn simple, and often redundant, positional patterns (Clark et al., 2019; Michel et al., 2019). To improve the representation, some works incorporate external information, such as syntax, into the Transformer-based neural machine translation (Currey and Heafield, 2019; Deguchi et al., 2019). Inspired by their achievements and to focus on our summarization task, we attempts to inject the semantics learned from a topic model into the attention mechanism.

Besides, Raganato et al. (2020) tried to fix the attention matrices of many heads according to token positions, finding that the performance do not drop and is even better in some cases. Motivated by this phenomenon, we introduce an extra head (the $(h+1)$-th head) with a fixed attention matrix to express a semantic-informed attentive pattern.

Recapping $\mathbf{\Phi}$ in (2), each column, $\boldsymbol{\phi}_k$, is a distribution over all tokens, representing a topic. From another view, each row, $\mathbf{\Phi}_{v,:}$, is a token representation, as shown in Fig. 1. With normalization

$$\{\boldsymbol{\varphi}_v\}_{v=1}^{V} = \mathbf{\Phi}_{v,:}/||\mathbf{\Phi}_{v,:}||_1, \tag{5}$$

$\boldsymbol{\varphi}_v$ can be interpreted as a distribution over topics. Thus, we can measure the similarity between tokens using the cosine distance, *i.e.*, $\cos(\boldsymbol{\varphi}_{v_1}, \boldsymbol{\varphi}_{v_2})$, which is an explicit and fixed semantic relation.

In Fig. 2, based on XSum (Narayan et al., 2018), we use UMAP[1] to project $\{\boldsymbol{\varphi}_v\}_{v=1}^{V}$ in (5) into a 2-dimensional space to visualize their relations. We choose six regions, and randomly select 10 example tokens from each region. Clearly, *i)* words with similar meanings are often grouped together, describing a field; *ii)* if two fields are semantically related, their corresponding tokens are closely distributed, such as "*Economic-Government*" and "*Astronomy-Airplane*". These phenomena indicate that $\boldsymbol{\varphi}_v$ in SIA is able to describe the semantic relations among tokens.

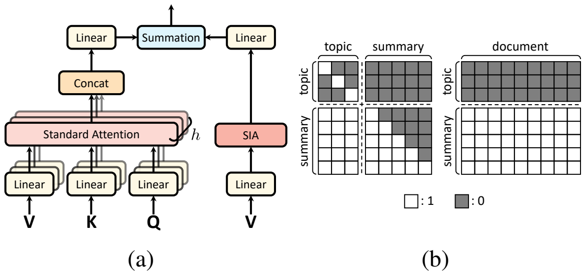


Figure 3: (a) TA for multi-head attention; (b) Mask matrices in decoder SA (left) and decoder CA (right).

To sum up, we consider SIA as an extra head (the $(h+1)$-th head) in every attention layer, as shown in Fig. 3(a), with its attention matrix formally stated as

$$\mathbf{A}_{h+1} = \mathrm{softmax}\left(\frac{[\cos(\boldsymbol{\varphi}_{v_1}, \boldsymbol{\varphi}_{v_2})]}{\sqrt{d_k}}\right), \tag{6}$$

$$\begin{cases} v_1, v_2 : \text{document-token indexes, in encoder SA} \\ v_1, v_2 : \text{summary-token indexes, in decoder SA} \\ \left.\begin{array}{l} v_1 : \text{summary-token index} \\ v_2 : \text{document-token index} \end{array}\right\} \text{in decoder CA} \end{cases}$$

where, $[\cdot]$ denotes a matrix.

Then, the output of multi-head attention is obtained by:

$$\begin{aligned} &\mathrm{Concat}(\mathrm{head}_1, \cdots, \mathrm{head}_h, \mathrm{head}_{h+1})\mathbf{W}^a, \\ =&\mathrm{Concat}(\mathrm{head}_1, \cdots, \mathrm{head}_h)\mathbf{W}^a_{\mathrm{ori}} \\ &+ \mathrm{head}_{h+1}\mathbf{W}^a_{\mathrm{add}} \end{aligned} \tag{7}$$

where, $\mathbf{W}^a \in \mathbb{R}^{(h+1)d_v \times d_{\mathrm{model}}}$ is rearranged as two parameter matrices $\mathbf{W}^a_{ori} \in \mathbb{R}^{hd_v \times d_{\mathrm{model}}}$ and $\mathbf{W}^a_{add} \in \mathbb{R}^{d_v \times d_{\mathrm{model}}}$. Clearly, $\mathbf{W}^a_{add}$ encapsulates the parameters brought by the SIA.

[1] https://umap-learn.readthedocs.io/

4.2 Topic embedding with masked attention (TEMA)

Given a corpus, the topic model is able to learn global topics $\mathbf{\Phi}$. For a specific document $\boldsymbol{x}$, the corresponding topic proportion vector $\boldsymbol{\theta}$ illustrates the importance degree of every topic. Therefore, those important topics represent the major or pertinent semantics of the document, which is expected to help the decoder to generate a summary.

For this purpose, we perform topic embedding so that the Transformer-based models can understand such topic representation. Recapping $\mathbf{\Phi}$ in (2), each column (topic), $\boldsymbol{\phi}_k$, is a distribution over all tokens in the vocabulary. Thus, we consider each topic embedding as a mixture of all token embeddings, as shown in Fig. 1, formally stated as:

$$\mathbf{E}_{\text{topic}} = \mathbf{\Phi}^T \mathbf{E}_{\text{token}} \tag{8}$$

where, $\mathbf{E}_{\text{topic}} \in \mathbb{R}^{K \times d_{\text{model}}}$ and $\mathbf{E}_{\text{token}} \in \mathbb{R}^{V \times d_{\text{model}}}$ are the topic and token embedding matrices, respectively. Clearly, topics and tokens lie in the same embedding space, making it possible to measure the relationships between document-topics and summary-tokens via attention.

Specifically, we choose the top-n topics according to $\boldsymbol{\theta}$, and consider these n topic embeddings as extra decoder inputs to guide the generation. We expect that the attention mechanism could fuse the topic information into the generation. Meanwhile, we should prevent the topic features polluted by the summary-token features via attention. Therefore, we build two kinds of masks for the SA and CA in decoder, as shown in Fig. 3(b).

As discussed before, a topic describes a co-occurrence pattern of all tokens. Moreover, recalling (8), each topic embedding vector can be interpreted as a semantic clustering center of all token embedding vectors, surrounded by tokens with similar semantics, as shown in Fig. 5(a) later. Using topics as inputs, the decoder is more likely to generate some recapitulative or new concepts that do not appear in the current document.

4.3 Document-related modulation (DRM)

Feature biasing is an efficient way to integrate conditions (Dumoulin et al., 2018). Subramani et al. (2019) introduced a sentence-specific bias into a pre-trained language model, showing superior performance on out-of-sample reconstruction.

As shown in (2), the topic proportion vector $\boldsymbol{\theta}$ is a latent representation of document $\boldsymbol{x}$, which can be considered as a conditioning information to fine-tune the Transformer-based models. To this end, we leverage $\boldsymbol{\theta}$ to infer a bias to modulate one hidden layer in every decoder layer. Specifically, in the l-th decoder layer, we infer a global feature bias via:

$$\boldsymbol{z}^{(l)} = \boldsymbol{\theta}^T \mathbf{W}_b^{(l)} \in \mathbb{R}^{d_{\text{model}}}. \tag{9}$$

where, $\mathbf{W}_b^{(l)} \in \mathbb{R}^{K \times d_{\text{model}}}$ is a parameter matrix in DRM. The bias vector $\boldsymbol{z}^{(l)}$ is then added to every position of the output of the CA block (before add and norm), as shown in Fig. 1.

4.4 Properties of TA

TA has three attractive properties, making it friendly to practical applications.

Small parameter footprint TA introduces three modules for the original Transformer encoder-decoder architecture: SIA, TEMA, and DRM. Among them, TEMA needs no extra parameters while SIA and DRM only introduce a small number of parameters compared with the original models, detailed illustrated in Table 8. Therefore, TA can be applied in many Transformer encoder-decoder structures without adding too much memory footprint or sacrificing the learning and test speed.

Plug-and-play The pipeline of pre-training and then fine-tuning has been widely accepted in NLP community, especially for transformer-based models. There are mainly two reasons. Firstly, many well pre-trained models provide checkpoints for users to fine-tune on their own tasks. Secondly, the Transformer models are getting bigger and bigger (Sanh et al., 2019), resulting in a fact that it is almost impossible to pre-train such a big model on a personal computer. Thus, models with plug-and-play property are attractive to Transformer-based models (Dathathri et al., 2019). Although we introduce TA based on a specific model, BertSUM, TA owns flexible plug-and-play property, since SIA, TEMA, and DRM do not break any structure of the original network. In experiments, shown in Table 5 later, we also demonstrate the effectiveness of TA on other Transformer-based summarization models, such as BART (Lewis et al., 2019), UNILM (Dong et al., 2019), and MASS (Song et al., 2019).

Efficient training The autoencoding variational inference (AVI) (Zhang et al., 2018) makes PFA scalable to big corpus and fast in out-of-sample prediction (calculating document-specific $\boldsymbol{\theta}$). In experiments, we find that the engineering-friendly pipeline training strategy[2] achieves attractive performance.

5 Experiments

5.1 Datasets

We evaluate the effectiveness and efficiency of TA on three benchmark datasets, including the CNN/DailyMail (CNN/DM) (Hermann et al., 2015), the New York Times Annotated Corpus (NYT) (Sandhaus, 2008) and the XSum (Narayan et al., 2018). The summary styles of these datasets varies from highlights, composed of several sentences, to very brief one sentence. Table 1 provides the statistics of these datasets. See more detailed descriptions in Appendix A. We perform data preprocessing following Liu and Lapata (2019).

5.2 Implementation details

Given a dataset, we first train the PFA based on the documents in the training set to obtain $\boldsymbol{\Phi}$, composed of 256 topics. More analysis on the number of topics can be found in Apendix B. For each document, we infer the corresponding $\boldsymbol{\theta}$ using the AVI in Zhang et al. (2018). According to the values in $\boldsymbol{\theta}$, we choose top-5 topics to perform topic embedding in TEMA. We adopts the settings in the original Transformer-based models. Following Liu and Lapata (2019), in the test stage, we use beam search with size 5, select the top-3 checkpoints based on their evaluation loss on the validation set, and report the averaged results on the test set. More detailed settings can be found in Appendix B. Our code is available at `https://github.com/BoChenGroup/TA`.

5.3 Quality evaluation on summarization

We evaluate the quality of the generated summaries using ROUGE scores (Lin, 2004). We report unigram and bigram overlap (ROUGE-1 and ROUGE-2) to assess informativeness, and the longest common subsequence (ROUGE-L) to assess fluency.

5.3.1 TA with BertSUM

We first combine TA with BertSUM on the abstractive summarization task. Given BertSUM checkpoints[3] on CNN/DM and XSum provided by Liu and Lapata (2019), we further fine-tune BertSUM with TA. Since Liu and Lapata (2019) did not provide checkpoints on NYT, we jointly fine-tune BertSUM and TA based on a pre-trained Bert.

Table 1: Statistics of summarization datasets.

Datasets	# docs (train/val/test)	avg. doc length		avg. summary length	
		words	sentences	words	sentences
CNN	90.266/1,220/1,093	760.50	33.98	45.70	3.59
DM	196,961/12,148/10,397	653.33	29.33	54.65	3.86
NYT	96,834/4,000/3,452	800.04	35.55	45.54	2.44
XSum	204,045/11,332/11,334	431.07	19.77	23.26	1.00

Table 2: ROUGE scores on CNN/DM test set, where the results are cited from Liu and Lapata (2019).

Model	R1	R2	RL
PTGEN (See et al., 2017)	36.44	15.66	33.42
PTGEN+Cov (See et al., 2017)	39.53	17.28	36.38
DRM (Paulus et al., 2017)	39.87	15.82	36.90
BOTTOMUP (Gehrmann et al., 2018)	41.22	18.68	38.34
DCA (Celikyilmaz et al., 2018)	41.69	19.47	37.92
Transformer (Liu and Lapata, 2019)	40.21	17.76	37.09
BertSUM (Liu and Lapata, 2019)	42.13	19.60	39.18
BertSUM+TA	**43.06**	**20.58**	**39.67**

Table 3: ROUGE scores on XSum test set, where the results are cited from Liu and Lapata (2019).

Model	R1	R2	RL
PTGEN (See et al., 2017)	29.70	9.21	23.24
PTGEN+Cov (See et al., 2017)	28.10	8.02	21.72
TCONVS2S (Narayan et al., 2018)	31.89	11.54	25.75
Transformer (Liu and Lapata, 2019)	29.41	9.77	23.01
BertSUM (Liu and Lapata, 2019)	38.81	16.50	31.27
BertSUM+TA	**39.77**	**17.39**	**32.39**

Table 4: ROUGE scores on NYT test set, where the results are cited from Liu and Lapata (2019).

Model	R1	R2	RL
PTGEN (See et al., 2017)	42.47	25.61	-
PTGEN+Cov (See et al., 2017)	43.71	26.40	-
DRM (Paulus et al., 2017)	42.94	26.02	-
Transformer (Liu and Lapata, 2019)	35.75	17.23	31.41
BertSUM (Liu and Lapata, 2019)	49.02	31.02	45.55
BertSUM+TA	**50.12**	**32.08**	**46.67**

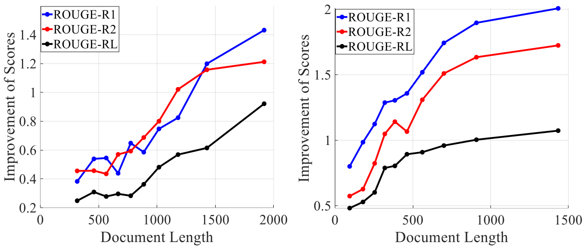
ROUGE scores on CNN/DM, XSum and NYT are given in Tables 2, 3, and 4, respectively. Methods in the first group are LSTM-based or CNN-based models. Compared with them, the outperformance of BertSUM illustrates that the combination of a pre-trained Bert encoder and a Transformer decoder is a better S2S structure. Though having the same structure as the Transformer, the BertSUM employs a Bert encoder pre-trained on a very large corpus, showing higher scores. Equipped with TA, BertSUM+TA achieves superior performance than the BertSUM, with only a few extra parameters, which will be further illustrated in Table 8.

[2] 1) Pre-train a Transformer; 2) Train PFA to extract $\boldsymbol{\Phi}$ and $\boldsymbol{\theta}$; 3) Fine-tune the Transformer+TA.

[3] `https://github.com/nlpyang/PreSumm`

Table 5: ROUGE scores of TA applied in BART (Lewis et al., 2019), UNILM (Dong et al., 2019) and MASS (Song et al., 2019). Results of the UNILM on XSum are obtained by running the public code. Others are from Zhang et al. (2019b).

Model	**CNN/DM**			**XSum**		
	R1	**R2**	**RL**	**R1**	**R2**	**RL**
MASS	42.12	19.50	39.01	39.75	17.24	31.95
MASS+TA	**43.06**	**19.98**	**39.88**	**41.12**	**18.05**	**32.75**
UNILM	43.33	20.21	40.51	42.63	19.10	33.13
UNILM+TA	**43.87**	**20.78**	**40.65**	**43.70**	**20.01**	**34.56**
BART	44.16	21.28	40.90	45.14	22.27	37.25
BART+TA	**44.47**	**21.39**	**41.32**	**45.76**	**22.68**	**38.03**



(a) CNN/DM (b) XSum

Figure 4: The plot of the improvement of BertSUM+TA over BertSUM as a function of the document length for (a) CNN/DM and (b) Xsum, where the improvement is measured by the amount of increase in the ROUGE scores. The documents in each corpus are equally divided into 10 different groups based on their lengths. Each point of a curve indicates the average ROUGE score in its corresponding group.

5.3.2 TA with some advanced Transformers

As discussed above, TA is a plug-and-play model, that is friendly to many Transformer encoder-decoder architectures. To illustrate it, we plug TA into BART (Lewis et al., 2019), UNILM (Dong et al., 2019), and MASS (Song et al., 2019), which are some advanced language models for document understanding and generation.

Based on their pre-trained models, we jointly fine-tune the model with TA for CNN/DM and XSum, respectively, following their settings in public codes. The ROUGE comparisons[4] are shown in Table 5, while the numbers of new parameters brought by TA are listed in Table 8. It is observed that TA is able to improve different types of Transformer encoder-decoder models for abstractive summarization with a few extra parameters.

5.4 TA with different-length documents

To analyze the effectiveness of TA as the documents have different amounts of tokens, we calculate the improvement of BertSUM+TA over BertSUM in terms of the ROUGE scores, with the results shown in Fig. 4.

Note that, as the number of document-tokens exceeds 512 (the length limitation of the Bert encoder), both BertSUM and BertSUM+TA use the initial 512 document-tokens as the inputs of the encoder. Compared with BertSUM that ignores the subsequent document-tokens, BertSUM+TA is able to reserve these information in some degree, since the topic model extracts global semantics from all tokens in the document. As a result, with the increase of the document length, the improvement produced by TA gets more evident. In other words, the global semantics introduced by TA is indeed helpful to Transformer-based models on the summarization task, especially for long documents.

5.5 Semantic similarity

As mentioned before, TA aims to exploit the semantics provided by the topic model to boost the summariazation performance. To evaluate the semantic similarities between the generated summary and the document (or the gold summary), we propose a new criterion, *Semantic Similarity* (SS). Given a set of topics $\mathbf{\Phi}$ and two pieces of text, $\mathbf{D}_1$ and $\mathbf{D}_2$, we firstly infer the topic proportions of these two pieces of text, *i.e.*, $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. Then, the simantic similarity (between $\mathbf{D}_1$ and $\mathbf{D}_2$) with respect to $\mathbf{\Phi}$ can be measured via the cosine similarity between $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, as:

$$\mathrm{SS}(\mathbf{D}_1, \mathbf{D}_2; \mathbf{\Phi}) = \frac{\boldsymbol{\theta}_1^T \boldsymbol{\theta}_2}{\|\boldsymbol{\theta}_1\| \|\boldsymbol{\theta}_2\|}. \tag{10}$$

In our case, after learning the topics of the documents, we use them to further infer the topic proportions of the document $\boldsymbol{\theta}_d$, topic proportions of the gold summary (ground truth) $\boldsymbol{\theta}_g$, and the topic proportions of the generated summary $\boldsymbol{\theta}_s$. Then, we measure the cosine similarities between $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_d$, $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_g$. The averaged SS scores on CNN/DM and XSum are summarized in Table 6. It can be seen that, with the help of TA, the generated summaries are closer to (have higher similarities to) the document, and also closer to the ground truth in the semantic space.

[4] All these methods did not provide results on NYT.

Table 6: Averaged SS scores between the generated summary and the document (or the gold summary).

Dataset	Method	Sum.-Doc.	Sum.-Gold.
CNN/DM	BertSUM	0.622	0.775
CNN/DM	BerSUM+TA	**0.651**	**0.781**
XSum	BertSUM	0.313	0.727
XSum	BerSUM+TA	**0.336**	**0.757**

Table 7: Ablation studies based on BertSUM.

Model	CNN/DM			XSum		
	R1	R2	RL	R1	R2	RL
BertSUM	42.13	19.60	39.18	38.81	16.50	31.27
BertSUM+SIA	42.48	19.99	39.37	39.06	16.80	31.55
BertSUM+TEMA	42.77	20.12	39.46	39.35	17.01	31.98
BertSUM+DRM	42.66	20.33	39.56	39.33	17.16	32.22
BertSUM+TA	43.06	20.58	39.67	39.77	17.39	32.39

Table 8: The amount of newly added parameters (in millions) and the corresponding percentage relative to the model size of the base model[5].

Model	Newly added parameters
BertSUM+TEMA	0M
BertSUM+DRM	3.16M (1.7%)
BertSUM+SIA	7.09M (3.94%)
BertSUM+TA	10.25M (5.69%)
MASS+TA	14.26M (5.93%)
UNILM+TA	31.50M (9.26%)
BART+TA	38.91M (9.58%)

Table 9: Perplexities on the test summary set.

Model	CNN/DM	XSum	NYT
GPT2	19.72	4.21	26.35
GPT2+TEMA	**16.95**	**3.30**	**21.68**

5.6 Ablation study

TA includes three modules: SIA, TEMA, and DRM. In order to understand the effectiveness of each part, we perform ablation studies by combining each module with the BertSUM.

As shown in Table 7, all these modules are able to promote the summarization performance in different degrees. Specifically, SIA introduces explicit semantic relations between tokens. Though effective, SIA mainly focuses on the local relations as the standard Transformer attention does. Compared with SIA, TEMA and DRM are better at introducing global semantics (topics and topic proportions) into the Transformer-based models, achieving more evident improvements. This illustrates that the global semantics, a special "summary", is useful to the summarization task.

5.7 Model size

We plug SIA, TEMA, DRM, and TA into some base models[5]. The amount and ratio of the newly added parameters are listed in Table 8. Clearly, TA introduces a few parameters, less than 10%. Therefore, TA has a friendly memory footprint to the Transformer models.

5.8 Effectiveness of TEMA

As analyzed in Sections 5.6 and 5.7, TEMA is effective for the summarization and adds surprisingly no extra parameter for the Transformers, which excites our curiosity to further analyze TEMA.

TEMA utilizes topic embeddings as part of the decoder inputs. Assisted by the masked attention, TEMA helps the decoder to perform conditional generation, with topics acting as the conditions. Therefore, we plan to investigate the generative ability of TEMA in a pure decoder architecture. To this end, we plug TEMA into the GPT2 small[6] (Radford et al., 2019) and perform fine-tuning, where we consider each training summary as a training sample.

After fine-tuning, we perform summary generation based on the document-topics only[7]. Fig. 5(a) shows some example topics learned from the documents in CNN/DM and XSum. We consider two kinds of conditional topics: *i)* two randomly selected topics and *ii)* top-three topics of a document. Some generated sentences are provided in Fig. 5(b). Whether the topics are representative of a specific document or not, the decoder is able to generate fluent and meaningful samples, that well match the conditional topics. Moreover, compared with setting-*ii*, the style of the generated sentences under setting-*i* is not similar to that of the news in CNN/DM and XSum. This further illustrates that topics are able to provide new concepts for the generation. Table 9 provides the perplexities on the test set. Clearly, with TEMA, GPT2 achieves lower perplexities.

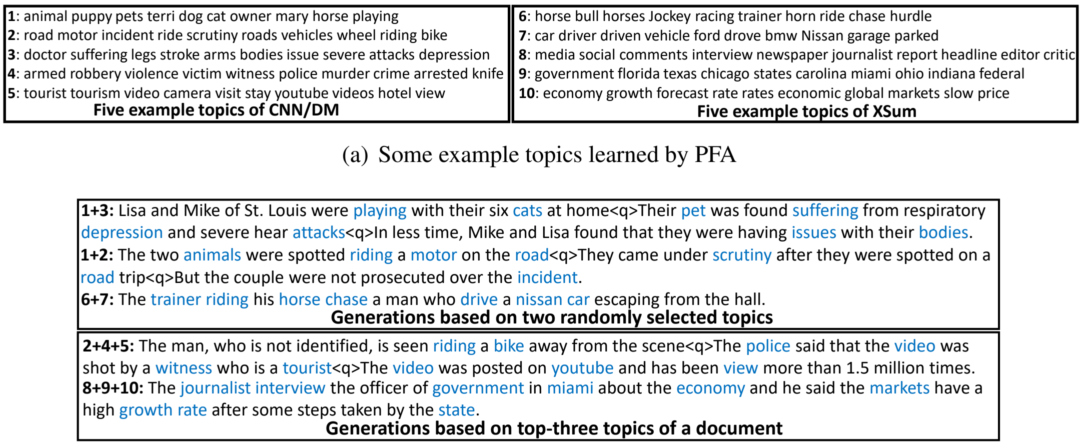
5.9 Generated summary examples

In Fig. 6, we show an example of the generated summaries of BertSUM and BertSUM+TA. It can be seen that BertSUM+TA generates some meaningful and recapitulative words that do not appear

[5] Model sizes of the base models: 180.22M for BertSUM, 240.48M for MASS, 340M for UNILM, and 406M for BART.

[6] Available at `https://github.com/huggingface/transformers`

[7] This model, without a document encoder, is NOT for summarization.



1: animal puppy pets terri dog cat owner mary horse playing
2: road motor incident ride scrutiny roads vehicles wheel riding bike
3: doctor suffering legs stroke arms bodies issue severe attacks depression
4: armed robbery violence victim witness police murder crime arrested knife
5: tourist tourism video camera visit stay youtube videos hotel view
Five example topics of CNN/DM

6: horse bull horses Jockey racing trainer horn ride chase hurdle
7: car driver driven vehicle ford drove bmw Nissan garage parked
8: media social comments interview newspaper journalist report headline editor critic
9: government florida texas chicago states carolina miami ohio indiana federal
10: economy growth forecast rate rates economic global markets slow price
Five example topics of XSum

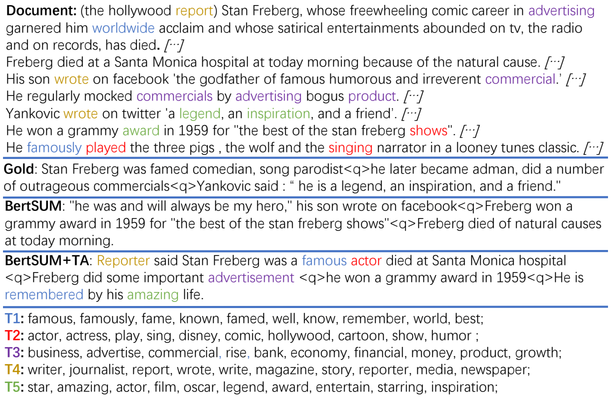
(a) Some example topics learned by PFA

1+3: Lisa and Mike of St. Louis were playing with their six cats at home<q>Their pet was found suffering from respiratory depression and severe hear attacks<q>In less time, Mike and Lisa found that they were having issues with their bodies.
1+2: The two animals were spotted riding a motor on the road<q>They came under scrutiny after they were spotted on a road trip<q>But the couple were not prosecuted over the incident.
6+7: The trainer riding his horse chase a man who drive a nissan car escaping from the hall.
Generations based on two randomly selected topics

2+4+5: The man, who is not identified, is seen riding a bike away from the scene<q>The police said that the video was shot by a witness who is a tourist<q>The video was posted on youtube and has been view more than 1.5 million times.
8+9+10: The journalist interview the officer of government in miami about the economy and he said the markets have a high growth rate after some steps taken by the state.
Generations based on top-three topics of a document

(b) Sentence generation conditioned on topics in (a)

Figure 5: The conditional generation results by pluging TEMA into the GPT2 small.



Document: (the hollywood report) Stan Freberg, whose freewheeling comic career in advertising garnered him worldwide acclaim and whose satirical entertainments abounded on tv, the radio and on records, has died. *[···]*
Freberg died at a Santa Monica hospital at today morning because of the natural cause. *[···]*
His son wrote on facebook 'the godfather of famous humorous and irreverent commercial.' *[···]*
He regularly mocked commercials by advertising bogus product. *[···]*
Yankovic wrote on twitter 'a legend, an inspiration, and a friend'. *[···]*
He won a grammy award in 1959 for "the best of the stan freberg shows". *[···]*
He famously played the three pigs , the wolf and the singing narrator in a looney tunes classic. *[···]*

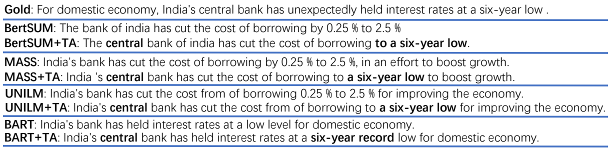
Gold: Stan Freberg was famed comedian, song parodist<q>he later became adman, did a number of outrageous commercials<q>Yankovic said : " he is a legend, an inspiration, and a friend."

BertSUM: "he was and will always be my hero," his son wrote on facebook<q>Freberg won a grammy award in 1959 for "the best of the stan freberg shows"<q>Freberg died of natural causes at today morning.

BertSUM+TA: Reporter said Stan Freberg was a famous actor died at Santa Monica hospital <q>Freberg did some important advertisement <q>he won a grammy award in 1959<q>He is remembered by his amazing life.

T1: famous, famously, fame, known, famed, well, know, remember, world, best;
T2: actor, actress, play, sing, disney, comic, hollywood, cartoon, show, humor ;
T3: business, advertise, commercial, rise, bank, economy, financial, money, product, growth;
T4: writer, journalist, report, wrote, write, magazine, story, reporter, media, newspaper;
T5: star, amazing, actor, film, oscar, legend, award, entertain, starring, inspiration;

Figure 6: Generated summaries of BertSUM and BertSUM+TA for a document in CNN/DM, where we also list the top-five topics of this document.



Gold: For domestic economy, India's central bank has unexpectedly held interest rates at a six-year low .

BertSUM: The bank of india has cut the cost of borrowing by 0.25 % to 2.5 %
BertSUM+TA: The **central** bank of india has cut the cost of borrowing **to a six-year low**.

MASS: India's bank has cut the cost of borrowing by 0.25 % to 2.5 %, in an effort to boost growth.
MASS+TA: India 's **central** bank has cut the cost of borrowing to **a six-year low** to boost growth.

UNILM: India's bank has cut the cost from of borrowing 0.25 % to 2.5 % for improving the economy.
UNILM+TA: India's **central** bank has cut the cost from of borrowing to **a six-year low** for improving the economy.

BART: India's bank has held interest rates at a low level for domestic economy.
BART+TA: India's **central** bank has held interest rates at a **six-year record** low for domestic economy.

Figure 7: Examples of generated summaries on Xsum.

in the document but conveyed by the topics. For example, BertSUM+TA summarizes *shows*, *played*, *singing* as *actor*, and summarizes *worldwide*, *famously* as *remembered* and *famous*. This is due to the fact that a topic describes a co-occurrence pattern of words with similar semantics.

Fig. 7 provides some generated examples as TA combined with different models. It can be seen that some neglected words, such as “central” and “six-year”, are generated with the help of TA. More examples can be found in Appendix C.

6 Conclusion and future work

In this paper, we explore and rearrange semantics of a topic model and then propose a friendly plug-and-play TA for Transformer-based abstractive summarization models. By introducing a small number of parameters, TA is able to further improve the performance of these models, especially under a long-document scenario. In the future, we will study the effectiveness of TA on other NLP tasks, such as the document-level translation, and investigate whether TA is useful for Transformer pre-training.

Acknowledgments

Bo Chen acknowledges the support of the Program for Oversea Talent by Chinese Central Government, the 111 Project (No. B18039), NSFC (61771361), and Shaanxi Innovation Team Project.

References

Melissa Ailem, Bowen Zhang, and Fei Sha. 2019. Topic augmented generator for abstractive summarization. *arXiv preprint arXiv:1908.07026*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Asli Celikyilmaz, 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, Volume 1 (Long Papers)*, pages 1662–1675.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Anna Currey and Kenneth Heafield. 2019. Incorporating source syntax into transformer-based neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 24–33.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: a simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Hiroyuki Deguchi, Akihiro Tamura, and Takashi Ninomiya. 2019. Dependency-based self-attention for transformer nmt. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 239–246.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054.

Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. 2018. Feature-wise transformations. *Distill*, 3(7):e11.

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*, pages 4098–4109.

Karl Moritz Hermann, Tomas Kocisky, 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*, pages 1693–1701.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu. 2019. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. 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)*, pages 3721–3731.

Zhengyuan Liu, Angela Ng, Sheldon Lee, Ai Ti Aw, and Nancy F Chen. 2019. Topic-aware pointer-generator networks for summarizing spoken conversations. *arXiv preprint arXiv:1910.01335*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, pages 14014–14024.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann. 2020. Fixed encoder self-attention patterns in transformer-based machine translation. *arXiv preprint arXiv:2002.10260*.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2019. Leveraging pre-trained checkpoints for sequence generation tasks. *arXiv preprint arXiv:1907.12461*.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

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 (Volume 1: Long Papers)*, pages 1073–1083.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936.

Nishant Subramani, Samuel Bowman, and Kyunghyun Cho. 2019. Can unconditional language models recover arbitrary sentences? In *Advances in Neural Information Processing Systems*, pages 15232–15242.

Sandeep Subramanian, Raymond Li, Jonathan Pilault, and Christopher Pal. 2019. On extractive and abstractive neural document summarization with transformer language models. *arXiv preprint arXiv:1909.03186*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zhengjue Wang, Chaojie Wang, Hao Zhang, Zhibin Duan, Mingyuan Zhou, and Bo Chen. 2020. Learning dynamic hierarchical topic graph with graph convolutional network for document classification. AISTATS.

Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd international conference on computational linguistics (Coling 2008)*, pages 985–992.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Hao Zhang, Bo Chen, Dandan Guo, and Mingyuan Zhou. 2018. Whai: Weibull hybrid autoencoding inference for deep topic modeling. *arXiv preprint arXiv:1803.01328*.

Hao Zhang, Bo Chen, Long Tian, Zhengjue Wang, and Mingyuan Zhou. 2019a. Variational hetero-encoder randomized generative adversarial networks for joint image-text modeling. *arXiv preprint arXiv:1905.08622*.

Jian Zhang, Liangyou Li, Andy Way, and Qun Liu. 2016. Topic-informed neural machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1807–1817.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019c. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

Mingyuan Zhou, Lauren Hannah, David Dunson, and Lawrence Carin. 2012. Beta-negative binomial process and poisson factor analysis. In *Artificial Intelligence and Statistics*, pages 1462–1471.

Appendix

A Data descriptions

In experiments, we evaluate the models on three benchmark summarization datasets. They are the CNN/DailyMail news (CNN/DM) (Hermann et al., 2015), the New York Times Annotated Corpus (NYT) (Sandhaus, 2008) and XSum (Narayan et al., 2018).

CNN/DM CNN/DM consists of news and associated sentence highlights, that is a brief overview composed of a few sentences. Following the standard training/validation/testing splits in Hermann et al. (2015) without anonymizing entities, we perform our experiments. We splits sentences using the Stanford CoreNLP toolkit[8] and pre-process the dataset following Liu (2019).

NYT NYT contains $110,540$ articles with abstractive summaries. Following Liu (2019), we split the dataset into $100,834/9706$ training/test examples based on the date of publication (the test set contains all articles published from January 1, 2007 onward), and use $4,000$ examples from the training set as a validation set. We also follow their filtering procedure, removing documents whose summary has less than 50 tokens (not words), resulting in a filtered test set including $3,452$ examples. We also split the sentences using the Stanford CoreNLP toolkit and perform pre-processing following Liu (2019).

XSum XSum includes $226,711$ news articles, each of which is associated with a one-sentence summary. We use the standard training/validation/testing splits

[8] `https://stanfordnlp.github.io/CoreNLP/`

Table 10: Comparions on the number of topics on the CNN/DM dataset.

Topic Num.	0	32	64	128	256	512
R1	42.13	42.52	42.77	42.91	43.06	43.08
R2	19.60	20.01	20.29	20.40	20.58	20.57
RL	39.18	39.46	39.57	39.62	39.67	39.69

$(204, 045/11, 332/11, 334)$ and follow the pre-processing in Narayan et al. (2018).

To satisfy the maximum capacity of the encoder in the base model, such as 512 for BertSUM, we use truncated document as the encoder input.

B Implementation Details

Topic model We remove stop words[9] to obtain the bag-of-word (BOW) vector for each document, and then use the BOW vectors to infer the topic model. For the PFA, we follow Zhang et al. (2018) to set model parameters.

As mentioned in Zhang et al. (2018), the number of topics is often set as 64, 128, or 256. Thus, we analyzed BertSUM (denoted as 0 topic) and BertSum+TA with different numbers of topics on the CNN/DM dataset, with the results shown in Table 10. It can be seen that a small number of topics are inadequate to express all the semantics, while too many topics are redundant and introduce more learnable parameters. Thus, we set 256 topics in all experiments.

We train PFA in one Nvidia GeForce RTX2080TI GPU. The experiments are performed with mini-batch size 200. We run 30 epochs to train the models on CNN/DM, NYT and Xsum. We use Adam optimizer with *learning-rate* $= 5^{-4}$, *weight-decay* $= 5^{-4}$ to optimize the topic model parameters. The hyper-parameters to update the topics with TLASGR-MCMC are the same with those in Zhang et al. (2018). According to the values of topic proportion in $\boldsymbol{\theta}$, in TEMA, we choose top-5 topics to obtain their corresponding topic embeddings.

Transformer+TA We do not change any setting of the original Transformer models. It should be noted that, to satisfy the maximum capacity of the encoder in the base model, such as 512 for BertSUM, one often use truncated documents as the encoder input. We set the hyper-parameters following the original papers and their public codes, where BertSUM[10] is referred to Liu and Lapata (2019), BART[11] referred to Lewis et al. (2019), UNILM[12] referred to Dong et al. (2019), and MASS[13] referred to Song et al. (2019). We fine-tune all models in four Nvidia GeForce RTX2080 TI GPUs. The experiments are performed with mini-batch size including 200 summary tokens with gradient accumulation every six iterations. Model checkpoints were saved and evaluated on the validation set every 1000 updates. Totally, we update the model $250,000$ times. Following Liu and Lapata (2019), we select the top-3 checkpoints based on their evaluation loss on the validation set, and report the averaged results on the test set. During decoding we used beam search with size 5, and tuned the α for the length penalty between 0.6 and 1 on validation set. It is worth noting that our decoder applies neither a copy nor a coverage mechanism, despite their popularity in abstractive summarization.

C More summary examples

Figs. 8 and 9 show some summary examples.

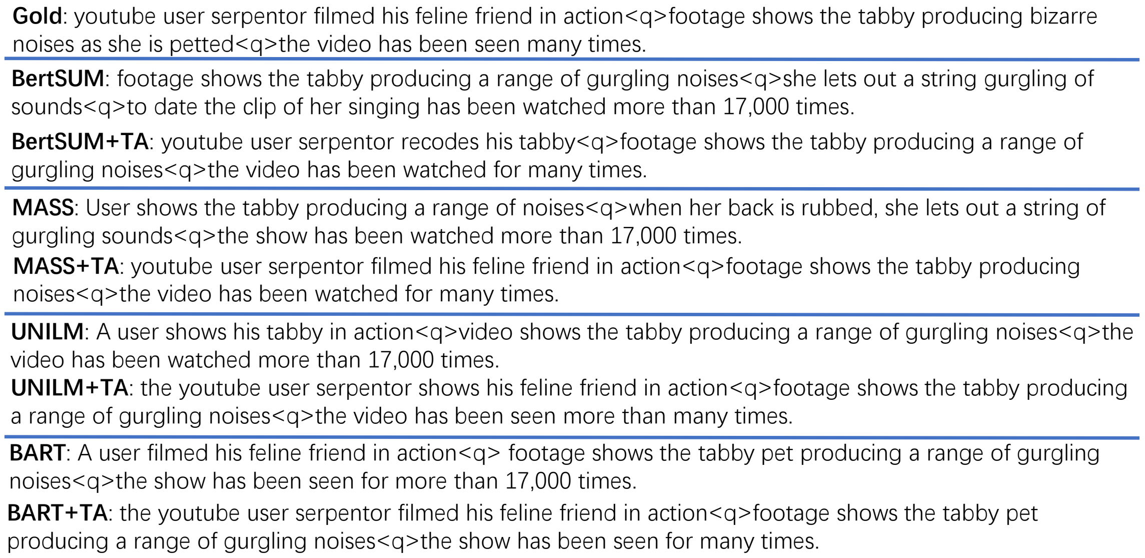
[9] For stop words, we set its semantic representation in SIA in (5) as zero vector.

[10] https://github.com/nlpyang/BertSUM

[11] https://github.com/pytorch/fairseq/tree/master/examples/bart

[12] https://github.com/microsoft/unilm

[13] https://github.com/microsoft/MASS



Gold: youtube user serpentor filmed his feline friend in action<q>footage shows the tabby producing bizarre noises as she is petted<q>the video has been seen many times.

BertSUM: footage shows the tabby producing a range of gurgling noises<q>she lets out a string gurgling of sounds<q>to date the clip of her singing has been watched more than 17,000 times.

BertSUM+TA: youtube user serpentor recodes his tabby<q>footage shows the tabby producing a range of gurgling noises<q>the video has been watched for many times.

MASS: User shows the tabby producing a range of noises<q>when her back is rubbed, she lets out a string of gurgling sounds<q>the show has been watched more than 17,000 times.

MASS+TA: youtube user serpentor filmed his feline friend in action<q>footage shows the tabby producing noises<q>the video has been watched for many times.

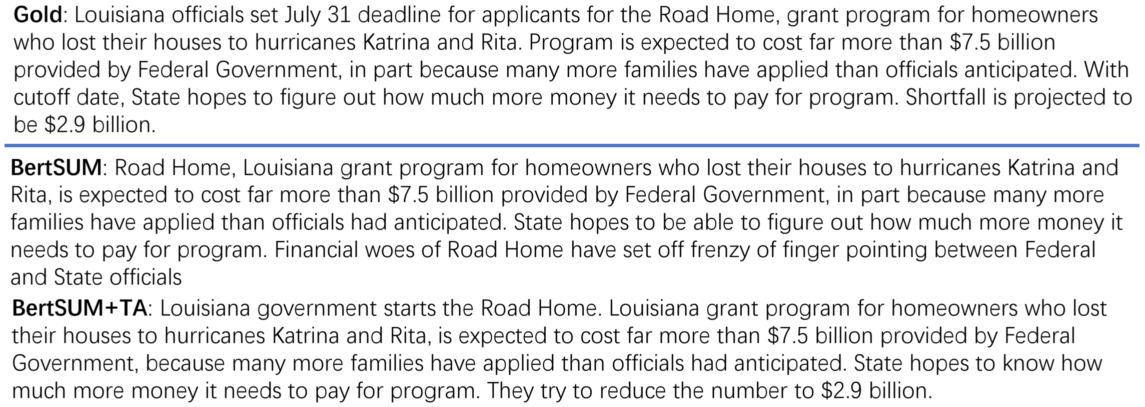
UNILM: A user shows his tabby in action<q>video shows the tabby producing a range of gurgling noises<q>the video has been watched more than 17,000 times.

UNILM+TA: the youtube user serpentor shows his feline friend in action<q>footage shows the tabby producing a range of gurgling noises<q>the video has been seen more than many times.

BART: A user filmed his feline friend in action<q> footage shows the tabby pet producing a range of gurgling noises<q>the show has been seen for more than 17,000 times.

BART+TA: the youtube user serpentor filmed his feline friend in action<q>footage shows the tabby pet producing a range of gurgling noises<q>the show has been seen for many times.

Figure 8: A generated summary example of CNN/DM.



Gold: Louisiana officials set July 31 deadline for applicants for the Road Home, grant program for homeowners who lost their houses to hurricanes Katrina and Rita. Program is expected to cost far more than $7.5 billion provided by Federal Government, in part because many more families have applied than officials anticipated. With cutoff date, State hopes to figure out how much more money it needs to pay for program. Shortfall is projected to be $2.9 billion.

BertSUM: Road Home, Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita, is expected to cost far more than $7.5 billion provided by Federal Government, in part because many more families have applied than officials had anticipated. State hopes to be able to figure out how much more money it needs to pay for program. Financial woes of Road Home have set off frenzy of finger pointing between Federal and State officials

BertSUM+TA: Louisiana government starts the Road Home. Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita, is expected to cost far more than $7.5 billion provided by Federal Government, because many more families have applied than officials had anticipated. State hopes to know how much more money it needs to pay for program. They try to reduce the number to $2.9 billion.

Figure 9: A generated summary example of NYT, where the generation of BertSUM comes from the original paper (Liu and Lapata, 2019).