Pretraining-Based Natural Language Generation for Text Summarization

Li Mingzhe

March 29, 2019

Motivation

1. Apply the BERT into text generation tasks.
2. Previous methods use left-context-only decoder and not utilize the pre-trained contextualized language models on the decoder side.
3. Context encoders such as ELMo, GPT, and BERT are pretrained on a huge unlabeled corpus and can generate better contextualized token embeddings, thus the approaches built on top of them can achieve better performance.

Background

1. Text summarization

   attentive sequence-to-sequence framework: consider only one direction context in the decoding process and may generate unnatural sequences.

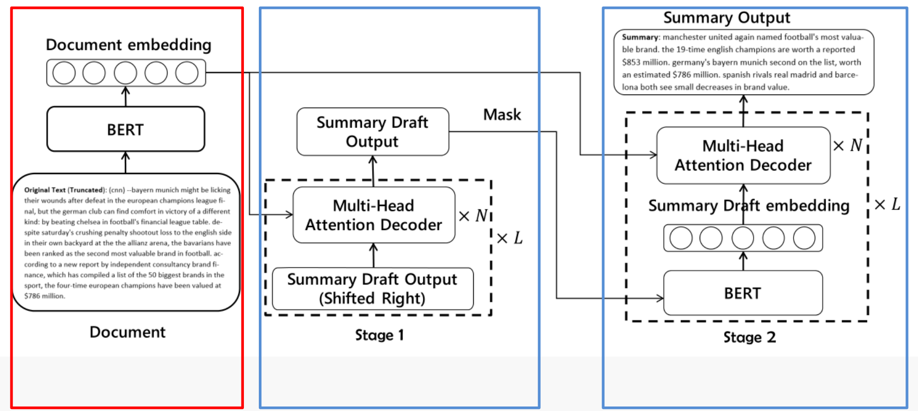
2. Bi-Directional Pre-Trained Context Encoders

   BERT mismatch problem: They model token-level representations and pre-train on both direction. But in decoder, they only use left context.

Our work

1. Propose a natural language generation model based on BERT.
2. Design a two-stage decoder process. In this architecture, our model can generate each word of the summary considering both sides' context information.
3. Conduct experiments on the benchmark datasets CNN/Daily Mail and New York Times and receive improved effect.

Model



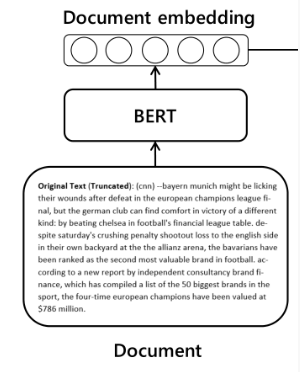
encoder

draft decoder

refine decoder

Summary Draft Generation

Encoder



H = BERT ($x_1, \dots, x_m$)

Input document

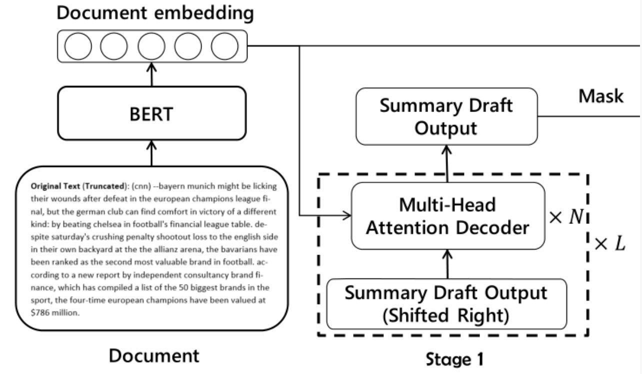
X = { $x_1, \dots, x_m$ }

Summary Draft Generation

Decoder

$$P_t^{vocab}(w) = f_{dec}(q_{<t}, H)$$

$$L_{dec} = \sum_{i=1}^{|a|} -logP(a_i = y_i^* | a_{<i}, H)$$



Soft alignments between summary and document

Map previous output $\{ y_1, \ldots, y_{t-1} \}$ into embedding vectors $\{ q_1, \ldots, q_{t-1} \}$

Summary Draft Generation

Copy mechanism

At decoder time step t, we first calculate the attention probability distribution over source document X using the last layer decoder output of Transformer o_t and the encoder output h_j.

$$u_t^j = o_t W_c h_j$$

$$\alpha_t^j = \frac{\exp u_t^j}{\sum_{k=1}^{N} \exp u_t^k}$$

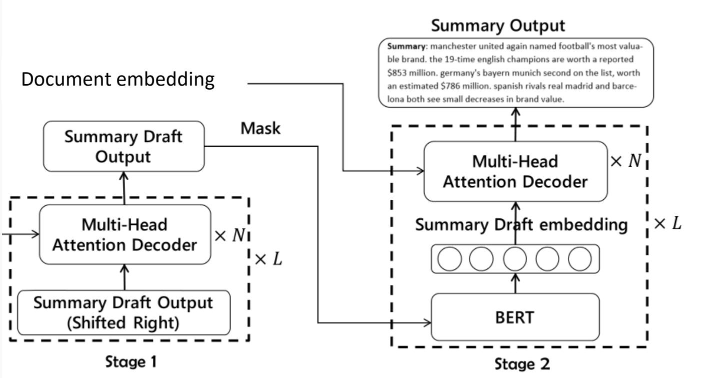
We then calculate copying gate:

$$g_t = sigmoid(W_g \cdot [o_t, h] + b_g)$$

=> $P_t(w) = (1 - g_t) P_t^{vocab}(\mathrm{w}) + g_t \sum_{i:w_i=w} \alpha_t^i$

Summary Refine Process

$$L_{refine} = \sum_{i=1}^{|y|} -logP(y_i = y_i^* | a_{\neq i}, H)$$



Model

Mixed objective

Add a discrete objective to the model, and optimize it by introducing the policy gradient method.

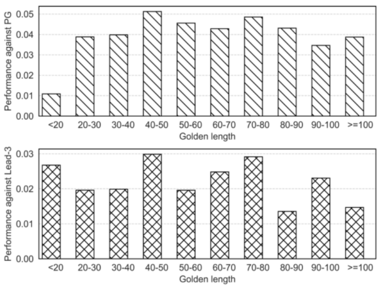
$$L_{dec}^{rl} = R(a^s) \cdot \left[-\log\left(P(a^s|x)\right)\right]$$

$$\hat{L}_{dec} = \gamma * L_{dec}^{rl} + (1-\gamma) * L_{dec}$$

=> $L_{model} = \hat{L}_{dec} + \hat{L}_{refine}$

Result and analysis

Model	ROUGE-1	ROUGE-2	ROUGE-L	R-AVG
Extractive				
lead-3 [See *et al.*, 2017]	40.34	17.70	36.57	31.54
SummmaRuNNer [Nallapati *et al.*, 2017]	39.60	16.20	35.30	30.37
Refresh [Narayan *et al.*, 2018]	40.00	18.20	36.60	31.60
DeepChannel [Shi *et al.*, 2018]	41.50	17.77	37.62	32.30
rnn-ext + RL [Chen and Bansal, 2018]	41.47	18.72	37.76	32.65
MASK-LM^{global} [Chang *et al.*, 2019]	41.60	19.10	37.60	32.77
NeuSUM [Zhou *et al.*, 2018]	41.59	19.01	37.98	32.86
Abstractive				
PointerGenerator+Coverage [See *et al.*, 2017]	39.53	17.28	36.38	31.06
ML+RL+intra-attn [Paulus *et al.*, 2018]	39.87	15.82	36.90	30.87
inconsistency loss[Hsu *et al.*, 2018]	40.68	17.97	37.13	31.93
Bottom-Up Summarization [Gehrmann *et al.*, 2018]	41.22	18.68	38.34	32.75
DCA [Celikyilmaz *et al.*, 2018]	41.69	19.47	37.92	33.11
Ours				
One-Stage	39.50	17.87	36.65	31.34
Two-Stage	41.38	19.34	38.37	33.03
Two-Stage + RL	**41.71**	**19.49**	**38.79**	**33.33**



Model	R-1	R-2
First sentences	28.6	17.3
First k words	35.7	21.6
Full [Durrett *et al.*, 2016]	42.2	24.9
ML+RL+intra-attn [Paulus *et al.*, 2018]	42.94	26.02
Two-Stage + RL (Ours)	**45.33**	**26.53**

Abstractive Summarization of *Reddit* Posts with Multi-level Memory Networks

Li Mingzhe

March 29, 2019

Motivation

1. Key sentences usually locate at the beginning of the text and favorable summary candidates are already inside the text in nearly exact forms.
2. The multi-level memory network is motivated by that when human understand a document, she does not remember it as a single whole document but ties together several levels of abstraction (e.g. word-level, sentence-level, paragraph-level and document-level).

Our work

1. Newly collect a large-scale abstractive summarization dataset named *Reddit TIFU*.
2. Propose a novel model named multi-level memory networks (MMN) to leverage memory networks for the abstractive summarization.
3. With quantitative evaluation and user studies via AMT, we show that our model outperforms state-of-the-art abstractive summarization methods on both *Reddit TIFU* and the Newsroom abstractive subset.

Reddit TIFU Dataset

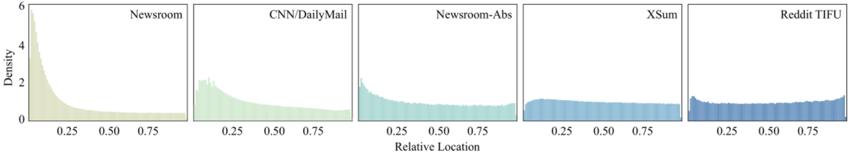
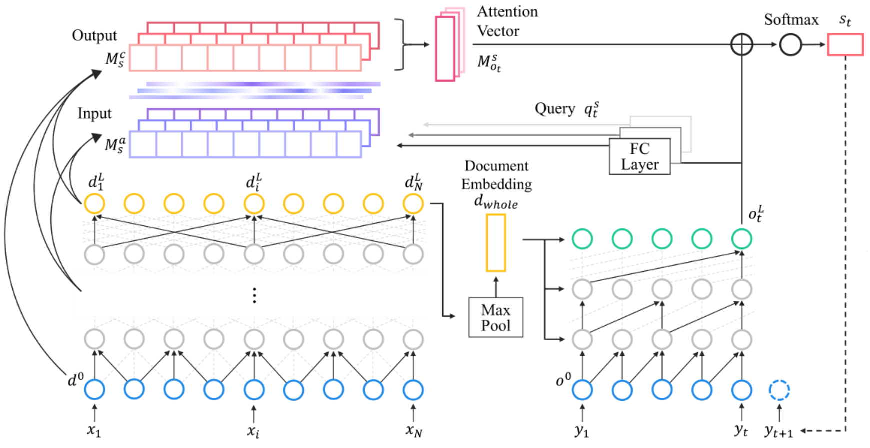


Figure 2: Relative locations of bigrams of gold summary in the source text across different datasets.

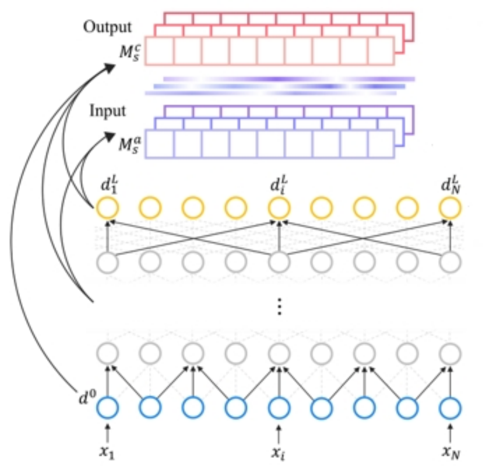
	PG			Lead			Ext-Oracle			PG/Lead	PG/Oracle
Dataset	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	Ratio (R-L)	Ratio (R-L)
CNN/DM (Nallapati et al., 2016)	36.4	15.7	33.4	39.6	17.7	36.2	54.7	30.4	50.8	0.92x	0.66x
NY Times (Sandhaus, 2008)	44.3	27.4	40.4	31.9	15.9	23.8	52.1	31.6	46.7	1.70x	0.87x
Newsroom (Grusky et al., 2018)	26.0	13.3	22.4	30.5	21.3	28.4	41.4	24.2	39.4	0.79x	0.57x
Newsroom-Abs (Grusky et al., 2018)	**14.7**	**2.2**	**10.3**	13.7	2.4	11.2	29.7	10.5	27.2	0.92x	0.38x
XSum (Narayan et al., 2018a)	29.7	9.2	23.2	16.3	1.6	12.0	29.8	8.8	22.7	1.93x	1.02x
TIFU-short	18.3	6.5	17.9	**3.4**	**0.0**	**3.3**	**8.0**	**0.0**	**7.7**	**5.42x**	**2.32x**
TIFU-long	19.0	3.7	15.1	**2.8**	**0.0**	**2.7**	**6.8**	**0.0**	**6.6**	**5.59x**	**2.29x**

Model-Multi-level Memory Networks (MMN)



(a) Multi-level Memory (b) Sequence Decoder

Construction of Multi-level Memory

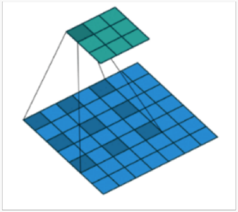


$$M_s^c = d^{m(s)+d^0}$$

$$M_s^a = d^{m(s)}$$

Dilated convolutions

$$F(x,s) = \sum_{i=1}^{k} w(i) * x_{s+d\cdot\left(i-\left\lfloor\frac{k}{2}\right\rfloor\right)} + b$$



$$d_i^0 = W_{emb} x_i$$

Model

Normalized Gated Tanh Units

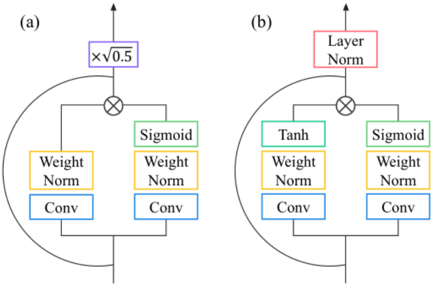
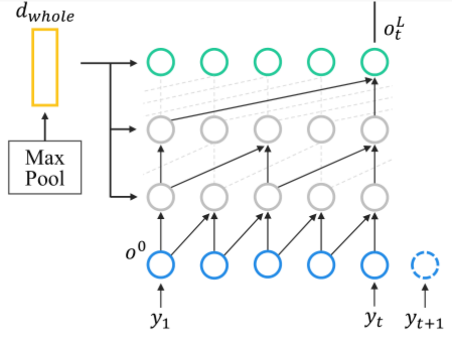


Figure 4: Comparison between (a) the gated linear unit (Gehring et al., 2017) and (b) the proposed normalized gated tanh unit.

$$GTU(d^l) = \tanh(F_f^l(d^l)) \circ \sigma(F_g^l(d^l))$$
$$d^{l+1} = LayerNorm(d^l + GTU(d^l))$$

State-Based Sequence Generation

$$d_{whole} = maxpool([d_1^L; \ldots; d_N^L])$$

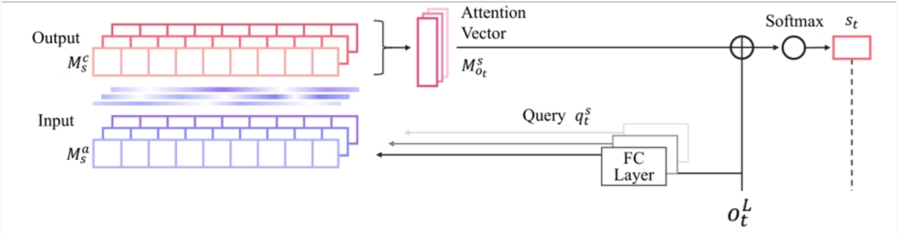


$$h_{f/g}^l = \hat{F}_{f/g}^l(o_{1:t}^l + W_{f/g}^l d_{whole})$$

$$h_a^l = \tanh(h_f^l) \circ \sigma(h_g^l)$$

$$o_{1:t}^{l+1} = LayerNorm(o_{1:t}^l + h_a^l)$$

State-Based Sequence Generation



$$q_t^s = \tanh\left(W_q^s o_t^L + b_q^s\right)$$

$$M_{o_t}^s = softmax\left(\frac{q_t^s (M_s^a)^T}{\sqrt{d_k}}\right) M_s^c$$

$$s_t = softmax\left(W_o\left[M_{o_t}^1; \ldots; M_{o_t}^S; \ o_t^L\right]\right)$$

$$y_{t+1} = argmax(s_t)$$

Training

To solve over-fitting

Using label smoothing:

$$p\left(y_{GT,t}\right) = 1 - \varepsilon, p(y') = \varepsilon/V$$

Loss function:

$$L = -\sum logp_{\theta}(y|x) - D_{KL}(u||p_{\theta}(y|x))$$

Result and analysis

TIFU-short				
Methods	PPL	R-1	R-2	R-L
seq2seq-att	46.2	18.3	6.4	17.8
PG (See et al., 2017)	40.9	18.3	6.5	17.9
SEASS (Zhou et al., 2017)	62.6	18.5	6.4	18.0
DRGD (Li et al., 2017)	69.2	14.6	3.3	14.2
Lead-1	n/a	3.4	0.0	3.3
Best-Match	n/a	8.0	0.0	7.7
MMN	32.1	**20.2**	**7.4**	**19.8**
MMN-NoDilated	**31.8**	19.5	6.8	19.1
MMN-NoMulti	34.4	19.0	6.1	18.5
MMN-NoNGTU	40.8	18.6	5.6	18.1
TIFU-long				
seq2seq-att	180.6	17.3	3.1	14.0
PG (See et al., 2017)	175.3	16.4	3.0	13.5
SEASS (Zhou et al., 2017)	387.0	17.5	2.9	13.9
DRGD (Li et al., 2017)	176.6	16.8	2.0	13.6
Lead-1	n/a	2.8	0.0	2.7
Best-Match	n/a	6.8	0.0	6.6
MMN	**114.1**	**19.0**	**3.7**	**15.1**
MMN-NoDilated	124.2	17.6	3.4	14.1
MMN-NoMulti	124.5	14.0	1.5	11.8
MMN-NoNGTU	235.4	14.0	2.6	12.1

Newsroom-Abs			
Methods	R-1	R-2	R-L
Lead-3 (Grusky et al., 2018)	13.7	2.4	11.2
TextRank (Barrios et al., 2016)	13.5	1.9	10.5
seq2seq-att	6.2	1.1	5.7
PG	14.7	2.2	11.4
MMN	**16.1**	**3.2**	**13.6**

	TIFU-short			TIFU-long		
vs. Baselines	Win	Lose	Tie	Win	Lose	Tie
seq2seq-att	**43.0**	28.3	28.7	**32.0**	24.0	44.0
PG	**38.7**	28.0	33.3	**42.3**	33.3	24.3
SEASS	**35.7**	28.0	36.3	**47.0**	37.3	15.7
DRGD	**46.7**	17.3	15.0	**61.0**	23.0	16.0
Gold	27.0	**58.0**	15.0	22.3	**73.7**	4.0

Thanks for listening!