Segatron: Segment-Aware Transformer for Language Modeling and Understanding

He Bai,[1] **Peng Shi,**[1] **Jimmy Lin,**[1 2] **Yuqing Xie,**[1] **Luchen Tan,**[2] **Kun Xiong,**[2] **Wen Gao,**[3] **Ming Li**[1 2]

[1]David R. Cheriton School of Computer Science, University of Waterloo
[2] RSVP.ai
[3] School of Electronics Engineering and Computer Science, Peking University
{he.bai, peng.shi, jimmylin, yuqing.xie, mli}@uwaterloo.ca, {lctan, kun}@rsvp.ai, wgao@pku.edu.cn

Abstract

Transformers are powerful for sequence modeling. Nearly all state-of-the-art language models and pre-trained language models are based on the Transformer architecture. However, it distinguishes sequential tokens only with the token position index. We hypothesize that better contextual representations can be generated from the Transformer with richer positional information. To verify this, we propose a segment-aware Transformer (Segatron), by replacing the original token position encoding with a combined position encoding of paragraph, sentence, and token. We first introduce the segment-aware mechanism to Transformer-XL, which is a popular Transformer-based language model with memory extension and relative position encoding. We find that our method can further improve the Transformer-XL base model and large model, achieving 17.1 perplexity on the WikiText-103 dataset. We further investigate the pre-training masked language modeling task with Segatron. Experimental results show that BERT pre-trained with Segatron (SegaBERT) can outperform BERT with vanilla Transformer on various NLP tasks, and outperforms RoBERTa on zero-shot sentence representation learning. Our code is available on GitHub.[1]

Introduction

Language modeling (LM) is a traditional sequence modeling task which requires learning long-distance dependencies for next token prediction based on the previous context. Recently, large neural LMs trained on a massive amount of text data have shown great potential for representation learning and transfer learning, and also achieved state-of-the-art results in various natural language processing tasks.

To the best of our knowledge, state-of-the-art language models (Dai et al. 2019; Baevski and Auli 2019; Rae et al. 2020) and pre-trained language models (Radford 2018; Devlin et al. 2019; Yang et al. 2019; Lan et al. 2020) all use a multi-layer Transformer (Vaswani et al. 2017). The Transformer network was initially used in the seq2seq architecture for machine translation, whose input is usually a sentence. Hence, it is intuitive to distinguish each token with its position index in the input sequence. However, the input length can grow to 1024 or more tokens and come from different sentences and paragraphs for language modeling. Although vanilla position encoding can help the transformer be aware of the token position by assigning a unique index to each token, the token index in a sentence, sentence index in a paragraph, and paragraph index in a document are all implicit. Such segmentation information is essential for language modeling, as tokens in different segments of context hold different significance for next token prediction. If the Transformer model can be aware of the segment position of each context token, we hypothesize that better context representations will be encoded. This statement is not made lightly, as for 3000 years, many languages including ancient Latin, Greek, English, French, and Chinese did not have punctuations or paragraphs. The introduction of sentence and paragraph separators was fundamental, so is indexing them to train Transformers. Although punctuations and paragraph breakers can provide boundary information to some extent, the boundary is not as straightforward as segment position, especially for the dot-product self-attention based Transformer.

Hence, we propose a novel segment-aware Transformer (Segatron), which encodes paragraph index in a document, sentence index in a paragraph, and token index in a sentence all together for the input sequence. We first verify the proposed method with relative position encoding on the language modeling task. By applying the segment-aware mechanism to Transformer-XL (Dai et al. 2019), our base model trained with the WikiText-103 dataset (Merity et al. 2017) outperforms Transformer-XL base by 1.5 points in terms of perplexity. Our large model achieves a perplexity of 17.1, the same score as Compressive Transformer (Rae et al. 2020), which is a more complicated model with longer input context and additional training objectives. We also pre-train masked language models with Transformer (BERT-base) and Segatron (SegaBERT-base) with English Wikipedia for 500K training steps. According to experimental results, SegaBERT can outperform BERT on both general language understanding (GLUE) and machine reading comprehension (SQUAD and RACE) tasks. We further pre-trained a large model SegaBERT-large with the same data used in BERT. Experimental results show that SegaBERT-large can not only outperform BERT-large on all the above tasks, but also outperforms RoBERTa-large on zero-shot Semantic Textual Similarity tasks. These results demonstrate the value of segment encodings in Transformers.

[1]https://github.com/rsvp-ai/segatron_aaai

Model

In this section, we show how to apply our proposed segment-aware Transformer to language modeling. More specifically, we first introduce our Segatron-XL (Segment-aware Transformer-XL) with non-learnable relative position encoding for autoregressive language modeling. Then we introduce our pre-trained Segatron (SegaBERT) with learnable absolute position encoding for masked language modeling (MLM).

Segatron-XL

We first introduce our method in the context of autoregressive language modeling, by replacing the vanilla Transformer index in Transformer-XL (Dai et al. 2019) with Segatron. Transformer-XL is a memory augmented Transformer with relative position encoding:

$$\begin{aligned}\mathbf{A}_{i,j}^{rel} &= \mathbf{E}_{x_i}^T \mathbf{W}_q^T \mathbf{W}_{k,E} \mathbf{E}_{x_j} + \mathbf{E}_{x_i}^T \mathbf{W}_q^T \mathbf{W}_{k,R} \mathbf{R}_{i-j} \\ &+ u^T \mathbf{W}_{k,E} \mathbf{E}_{x_j} + v^T \mathbf{W}_{k,R} \mathbf{R}_{i-j}\end{aligned} \tag{1}$$

where $\mathbf{A}_{i,j}^{rel}$ is the self-attention score between query i and key j. $\mathbf{E}_{x_i}$ and $\mathbf{E}_{x_j}$ are the input representations of query i and key j, respectively. $\mathbf{R}_{i-j}$ is the relative position embedding. $\mathbf{W}_{k,E}$ and $\mathbf{W}_{k,R}$ are transformation matrices for input representation and position embedding, respectively. $\mathbf{u}$ and $\mathbf{v}$ are learnable variables. The position embeddings are non-learnable and defined as:

$$\mathbf{R}_{i-j,k} = \begin{cases} sin(\frac{i-j}{10000^{2k/dim}}) & k < \frac{1}{2}dim \\ cos(\frac{i-j}{10000^{2k/dim}}) & k \geq \frac{1}{2}dim \end{cases} \tag{2}$$

where dim is the dimension size of $\mathbf{R_{i-j}}$, and k is the dimension index.

Our proposed method introduces paragraph and sentence segmentation to the relative position encoding. The new position embeddings $\mathbf{R_{I,J}}$ are defined as:

$$\mathbf{R}_{\mathbf{I},\mathbf{J},k} = \begin{cases} \mathbf{R^t}_{t_i - t_j, k} & k < \frac{1}{3}dim \\ \mathbf{R^s}_{s_i - s_j, k - \frac{1}{3}dim} & \frac{2}{3}dim > k \geq \frac{1}{3}dim \\ \mathbf{R^p}_{p_i - p_j, k - \frac{2}{3}dim} & k \geq \frac{2}{3}dim \end{cases} \tag{3}$$

where $\mathbf{I} = \{t_i, s_i, p_i\}$, $\mathbf{J} = \{t_j, s_j, p_j\}$. t, s, and p are token position index, sentence position index, and paragraph position index, respectively. $\mathbf{R^t}$, $\mathbf{R^s}$, and $\mathbf{R^p}$ are the relative position embeddings of token, sentence, and paragraph. These embeddings are defined in Eq. 2 and the dimensions of each are equal to 1/3 of $\mathbf{R_{I,J}}$. The input representation of our model is shown in Figure 1(a).

To equip the recurrence memory mechanism of Transformer-XL with the segment-aware relative position encoding, the paragraph position, the sentence position, and the token position indexes of the previous segment should also be cached together with the hidden states. Then, the relative position can be calculated by subtracting the cached position indexes from the current position indexes.

Pre-trained Segatron

We will introduce how to pre-train a language model with our proposed Segatron in this section.

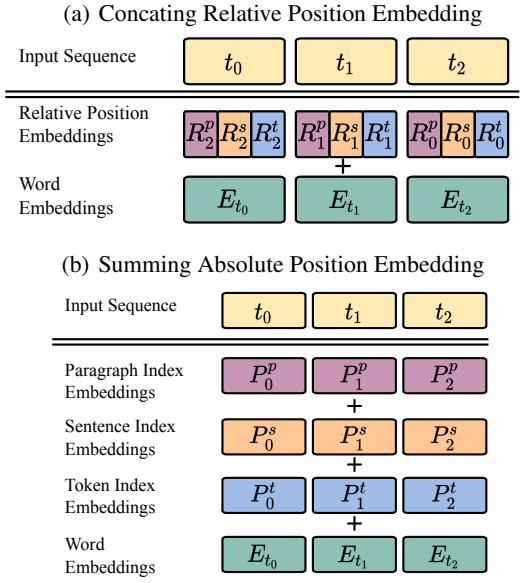


Figure 1: Input representation of Segatron-XL and SegaBERT.

First, pre-training a masked language model in the setting of BERT is a practical choice, as BERT is a popular baseline model and requires less computational resources compared with more recent large models. For example, BERT-large only needs about 10% of the resources of RoBERTa-large (Liu et al. 2019). Hence, in this paper, we first pre-train two base size models: SegaBERT-base$^-$ and BERT-base$^-$ with only English Wikipedia data for 500K training steps, to compare BERT pre-trained with Transformer and Segatron fairly. We then pre-train a large size model SegaBERT-large with Wikibooks dataset and 1M training steps, same as BERT-large.

Input Representation. Input $\mathbf{X}$ of SegaBERT is a sequence of tokens, which can be one or more sentences or paragraphs. The representation x_t for token t is computed by summing the corresponding token embedding $\mathbf{E}_t$, token index embedding $\mathbf{P}_t^t$, sentence index embedding $\mathbf{P}_t^s$, and paragraph index embedding $\mathbf{P}_t^p$, as shown in Figure 1(b). Two special tokens `[CLS]` and `[SEP]` are added to the text sequence before the first token and after the last token, and their paragraph/sentence indexes are the same as their adjacent tokens. Following BERT, the text is tokenized into subwords with WordPiece and the maximum sequence length is 512.

Training Objective. Following BERT, we use the masked LM as our training objective. However, next sentence prediction (NSP) is not used in our model, as our input contains more than two sentences.

Data preparation. For the pre-training corpus we use English Wikipedia and Bookcorpus (Zhu et al. 2015). For each document, we firstly split each into N_p paragraphs, and all the sub-tokens in the i-th paragraph are assigned the same Paragraph Index Embedding $\mathbf{P}_i^p$. The paragraph index starts from 0 for each document. Similarly, each paragraph is further segmented into N_s sentences with NLTK (Bird, Klein,

Model	#Param.	PPL
LSTM+Neural cache (Grave, Joulin, and Usunier 2017)	-	40.8
Hebbian+Cache (Rae et al. 2018)	-	29.9
Transformer-XL base, M=150 (Dai et al. 2019)	151M	24.0
Transformer-XL base, M=150 (ours)	151M	24.4
Segatron-XL base, M=150	151M	**22.5**
Adaptive Input (Baevski and Auli 2019)	247M	18.7
Transformer-XL large, M=384 (Dai et al. 2019)	257M	18.3
Compressive Transformer, M=1024 (Rae et al. 2020)	257M	17.1
Segatron-XL large, M=384	257M	**17.1**

Table 1: Comparison with Transformer-XL and competitive baseline results on WikiText-103.

and Loper 2009), and all the sub-tokens in the i-th sentence are assigned the same Sentence Index Embedding $\mathbf{P}_i^s$. The sentence index starts from 0 for each paragraph. Within each sentence, all the sub-tokens are indexed from 0; the i-th sub-token will have its Token Index Embedding $\mathbf{P}_i^t$.

When building a training example, we randomly (length weighted) sample a document from the corpus and randomly select a sentence in that document as the start sentence. Then, the following sentences are added to that example until the example meets the maximum length limitation (512) or runs out of the selected document. If any position index in that example exceeds the maximum index, all such position indexes will be subtracted by one until they meet the maximum requirements. The maximum position index of paragraph, sentence, and token are 50, 100, and 256, respectively.

Training Setup. Liu et al. (2019) have shown that BERT pre-trained with document input (more than two sentences) without NSP performs better than the original BERT on some tasks. Hence, we not only pre-train a SegaBERT-large, but also pre-train two base models with the same setting for fair comparison. Similar to BERT, the base model is 12 layers, 768 hidden size, and 12 self-attention heads. The large model is 24 layers, 1024 hidden size, and 24 self-attention heads. For optimization, we use Adam with learning rate 1e-4, β_1=0.9, β_2=0.999, with learning rate warm-up over the first 1% of the total steps and with linear decay of the learning rate.

Experiments

In this section, we first conduct autoregressive language modeling experiments with our proposed Segatron and also conduct an ablation study with this task. Then, we show the results of pre-trained SegaBERT on general language understanding tasks, semantic textual similarity tasks, and machine reading comprehension tasks.

Autoregressive Language Modeling

Dataset WikiText-103 is a large word-level dataset with long-distance dependencies for language modeling. This dataset preserves both punctuations and paragraph line breakers, which are essential for our segmentation pre-processing. There are 103M tokens, 28K articles for training. The average length is 3.6K tokens per article.

Model	PPL
Transformer-XL base	24.35
+ paragraph position encoding	24.07
+ sentence position encoding	22.51
Segatron-XL base	22.47

Table 2: Ablation over the position encodings using Transformer-XL base architecture.

Model Configuration Following Transformer-XL, we train a base size model and a large size model. The base model is a 16 layer Transformer with a hidden size of 410 and 10 self-attention heads. This model is trained for 200K steps with a batch size of 64. The large model is an 18 layer Transformer with a hidden size of 1024 and 16 attention heads. This model is trained with 350K steps with a batch size of 128. The sequence length and memory length during training and testing all equal 150 for the base model and 384 for the large model. The main differences between our implementation and Transformer-XL are: we use mixed-precision mode; our input/memory lengths between training and testing are the same; the large model training steps of Transformer-XL are 4M according to their implementation.

Main Results Our results are shown in Table 1. As we can see from this table, the improvement with the segment-aware mechanism is quite impressive: the perplexity decreases 1.5 points for the Transformer-XL base and decreases 1.2 for Transformer-XL large. We also observe that our large model achieves 18.3 PPL with only 172K training steps. We finally obtain a perplexity of 17.1 with our large model – comparable to prior state-of-the-art results of Compressive Transformer (Rae et al. 2020), which is based on Transformer-XL but trained with longer input length and memory length (512) and a more complicated memory cache mechanism.

It is worth noting that we do not list methods with additional training data or dynamic evaluation (Krause et al. 2018) which continues training the model on the test set. We also note that there is a contemporaneous work RoutingTransformer (Roy et al. 2020), which modifies the self-attention to local and sparse attention with a clustering method. How-

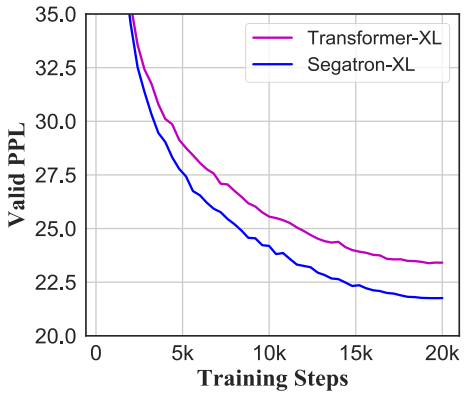


Figure 2: Valid perplexities during the training processes of language modeling.

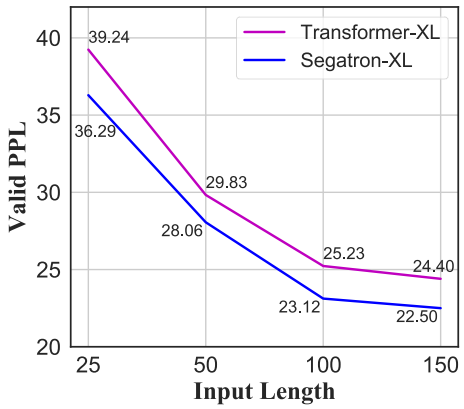


Figure 3: Test perplexities of Segatron-XL and Transformer-XL with different input lengths.

ever, their implementations are not available. We believe our method is orthogonal to their work and can be introduced to their model.

Analysis We plot the valid perplexity of Segatron-XL base and Transformer-XL base during training in Figure 2. From this figure, we can see that the segment-aware model outperforms the base model all the time, and the gap between them becomes larger as training progresses. Segatron-XL at 10K steps approximately matches the performance of Transformer-XL at 20K steps. We then test the effectiveness of Segatron over different input lengths (25, 50, 100, and 150 input tokens) by comparing Transformer-XL and Segatron-XL base models. As we can see from Figure 3, the improvements are consistent and significant. There is no evidence showing our method prefers shorter or longer input.

Ablation Study We finally conduct an ablation study with Segatron-XL base, to investigate the contributions of the sentence position encoding and the paragraph position encoding, respectively. Experimental results are shown in Table 2. From this table, we find that the PPL of Transformer-XL decreases from 24.35 to 24.07/22.51 after adding paragraph/sentence

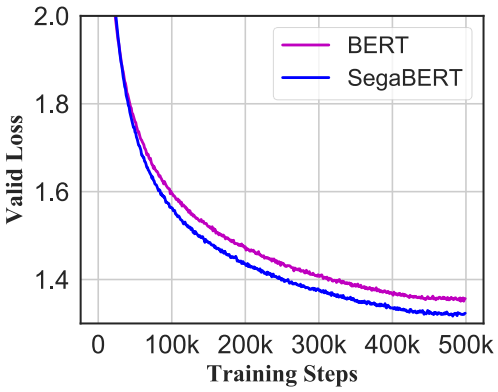


Figure 4: Valid losses during the pre-training.

position encoding, and further decreases to 22.47 by encoding paragraph and sentence positions simultaneously. The results show that both the paragraph position and sentence position can help the Transformer to model language. Sentence position encoding contributes more than paragraph position encoding in our experiments.

Pre-trained Masked Language Model

We first plot the valid losses of BERT-base$^-$ and SegaBERT-base$^-$ during pre-training in Figure 4. The overall trends between Figure 2 and Figure 4 are similar, which demonstrates that our proposed segment-aware method works on both auto-regressive language modeling and masked language modeling. We will detail our experiments with our pre-trained models in the following sections.

General Language Understanding The General Language Understanding Evaluation (GLUE) benchmark (Wang et al. 2019) is a collection of resources for evaluating natural language understanding systems. Following Devlin et al. (2019), we evaluate our model over these tasks: linguistic acceptability CoLA (Warstadt, Singh, and Bowman 2019), sentiment SST-2 (Socher et al. 2013), paraphrase MRPC (Dolan and Brockett 2005), textual similarity STS-B (Cer et al. 2017), question paraphrase QQP, textual entailment RTE (Bentivogli et al. 2009) and MNLI (Williams, Nangia, and Bowman 2018), and question entailment QNLI (Wang et al. 2019). We fine-tune every single task only on its in-domain data without two-stage transfer learning.

On the GLUE benchmark, we conduct the fine-tuning experiments in the following manner: For single-sentence classification tasks, such as sentiment classification (SST-2), the sentence will be assigned Paragraph Index 0 and Sentence Index 0. For sentence pair classification tasks, such as question-answer entailment (QNLI), the first sentence will be assigned Paragraph Index 0 and Sentence Index 0 and the second sentence will be assigned Paragraph Index 1 and Sentence Index 0.

We conduct grid search with the GLUE dev set for small data tasks: CoLA, MRPC, RTE, SST-2, and STS-B. Our grid search space is as follows:

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	AVG
BERT-base$^-$	83.2	90.4	86.5	68.3	91.3	**92.6**	55.0	88.9	82.0
SegaBERT-base$^-$	83.8	91.5	87.0	71.8	92.1	92.4	54.7	89.0	82.8
BERT-large (best of 3)	87.3	93.0	**91.4**	74.0	94.0	88.7	63.7	90.2	85.3
SegaBERT-large	**87.6**	**93.6**	89.1	**78.3**	**94.7**	92.3	**65.3**	**90.3**	**86.4**

Table 3: Fair comparison on GLUE dev. The two base models are pre-trained in the same setting. For large models comparison, we choose the best of 3 BERT-large models: the original BERT, whole word masking BERT, and BERT without NSP task. Results of BERT-large (best of 3) are from Yang et al. (2019).

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	AVG
BERT-base$^-$	82.9	90.1	70.8	65.4	91.2	88.9	43.5	83.9	77.1
SegaBERT-base$^-$	83.5	90.8	71.4	68.1	91.5	89.3	50.7	84.6	78.7
BERT-large	86.7	92.7	72.1	70.1	**94.9**	89.3	60.5	86.5	81.6
SegaBERT-large	**87.9**	**94.0**	**72.5**	**71.6**	94.8	**89.7**	**62.6**	**88.6**	**82.7**

Table 4: Results on GLUE test set. Results of BERT-large are from Devlin et al. (2019).

Model	STS-12	STS-13	STS-14	STS-15	STS-16	STS-B	SICK-R	AVG
S-BERT-large	72.27	78.46	74.90	80.99	76.25	79.23	73.75	76.55
S-BERT-large*	72.39	78.06	**75.26**	81.79	76.35	78.64	73.85	76.62
S-RoBERTa-large	**74.53**	77.00	73.18	81.85	76.82	79.10	**74.29**	76.68
S-SegaBERT-large	74.49	**78.64**	74.88	**83.28**	**77.10**	**79.42**	73.77	**77.37**

Table 5: Zero-shot spearman's rank correlation $\rho \times 100$ between the negative distance of sentence embeddings and the gold labels. STS-B: STS benchmark, SICK-R: SICK relatedness dataset. Results of BERT-large and RoBERTa-large are from Reimers and Gurevych (2019).

- Batch size: 16, 24, 32;
- Learning rate: 2e-5, 3e-5, 5e-5;
- Number of epochs: 3-10.

For QQP, MNLI, and QNLI, we use the default hyper-parameters: 3e-5 learning rate, 256 batch size, and 3 epochs. The other hyper-parameters are the same as in the Hugging-Face Transformers library.[2]

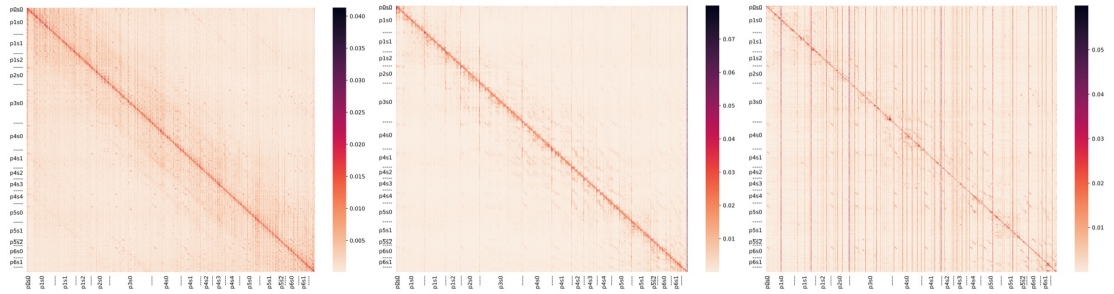
We compare BERT and SegaBERT in a fair setting to decouple the effects of document-level inputs and the removal of NSP. In Table 3, two base models are pre-trained by us and the only difference is the position encoding. We can see that our SegaBERT-base$^-$ outperforms BERT-base$^-$ on most tasks. We also notice that SegaBERT-base$^-$ is lower than BERT-base$^-$ by over 2.5 points on CoLA. However, this gap decreases to 0.1 on the test set, which is shown in Table 4. This is because the size of CoLA is quite small and not as robust as other datasets. Improvements can also be observed easily when comparing SegaBERT-large with the best score of 3 BERT-large models.

These results demonstrate SegaBERT's effectiveness in general natural language understanding. The improvements on these sentence and sentence pair classification tasks show that our segment-aware pre-trained model is better than vanilla Transformer on sentence-level tasks.

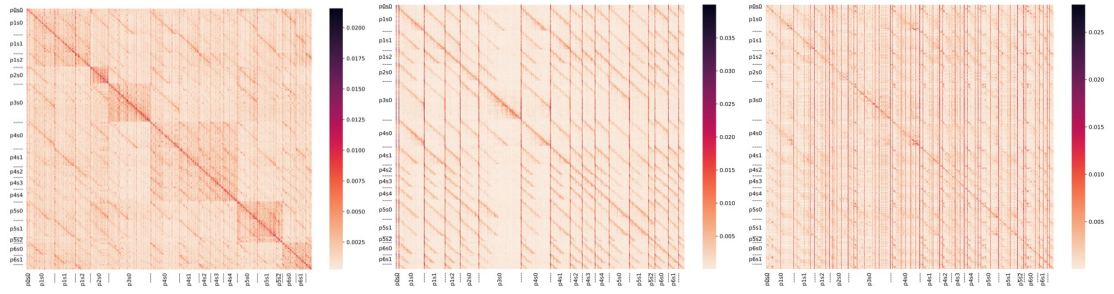
Sentence Representation Learning Since our SegaBERT has shown great potential on sentence-level tasks, in this section, we further investigate whether SegaBERT can generate better sentence representations. Following Sentence-BERT (Reimers and Gurevych 2019), we fine-tune SegaBERT in a siamese structure on the combination of SNLI (Bowman et al. 2015) and MNLI datasets. The fine-tuned model is named S-SegaBERT. We then evaluate the zero-shot performance of S-SegaBERT and other baselines on Semantic Textual Similarity (STS) tasks using the Spearman's rank correlation between the cosine similarity of the sentence embeddings and the gold labels.

In Table 5, the results of S-BERT-large and S-RoBERTa-large are from Reimers and Gurevych (2019). The results of S-BERT-large* are re-implemented by us, which is similar to Sentence-BERT's results. We can see that our SegaBERT achieves the highest average scores on STS tasks, even outperforms RoBERTa, which uses much more training data, larger batch size, and dynamic masking. These results conform with our improvements on GLUE benchmarks, which indicate that a language model pre-trained with Segatron can learn better sentence representations (single sentence encoding) than the original Transformer.

[2] https://github.com/huggingface/transformers



(a) BERT-Layer 1 (b) BERT-Layer 6 (c) BERT-Layer 12



(d) SegaBERT-Layer 1 (e) SegaBERT-Layer 6 (f) SegaBERT-Layer 12

Figure 5: Self-attention heat maps of the first, the sixth, and the last layer of SegaBERT and BERT when encoding the first 512 tokens of a Wikipedia article.

System Model	SQUAD1.1 EM	SQUAD1.1 F1	SQUAD2.0 EM	SQUAD2.0 F1
BERT-base	80.8	88.5	72.3	75.6
BERT-base$^-$	81.9	89.4	75.4	78.2
SegaBERT-base$^-$	83.2	90.2	76.3	79.2
BERT-large	84.1	90.9	78.7	81.9
BERT-large wwm	86.7	92.8	80.6	83.4
SegaBERT-large	86.0	92.6	81.8	85.2

Table 6: Evaluation results on SQUAD v1.1 and v2. Results of BERT-base and BERT-large are from Devlin et al. (2019). Results of BERT-large wwm on SQUAD v1.1 are from BERT's github repository. There are no official results of BERT-large wwm on SQUAD v2 and here we report our fine-tuning results.

Reading Comprehension We finally test our pre-trained model on machine reading comprehension tasks. For these tasks, the question is assigned Paragraph Index 0 and Sentence Index 0. For a context with n paragraphs, Paragraph Index 1 to $n+1$ are assigned to them accordingly. Within each paragraph, the sentences are indexed from 0.

We first fine-tune our SegaBERT model with SQUAD v1.1 (Rajpurkar et al. 2016) for 4 epochs with 128 batch size and 3e-5 learning rate. The fine-tuning setting of SQUAD v2.0 (Rajpurkar, Jia, and Liang 2018) is the same as SQUAD v1.1. Results are shown in Table 6. As we can see from Table 6, our pre-trained SegaBERT-base$^-$ outperforms our pre-trained BERT-base$^-$ on both dataset: 1.3 EM and 0.8 F1 improvements on SQUAD v1.1; 0.9 EM and 1.0 F1 improvements on SQUAD v2. It should be noticed that our pre-trained BERT-base$^-$ outperforms the original BERT-base model, although ours is pre-trained with fewer data and steps. This confirms Liu et al. (2019)'s finding that BERT pre-trained with document-level input can contribute to performance improvements on SQUAD. For large models, as we cannot afford to train a new BERT-large model in the same setting as BERT-base$^-$, we compare our model with BERT-large wwm (with whole word masking), which is a stronger baseline model. We can see that SegaBERT large is slightly lower than BERT-large wwm on SQUAD v1.1 but outperforms it on SQUAD v2 over 1.2 EM and 1.8 F1.

Model	Acc-Dev	Acc-Test
BERT-large	72.7	72.0
SegaBERT-large	74.5	73.8

Table 7: Accuracy on dev and test sets of RACE. Results of BERT-large are from Pan et al. (2019).

We further test our models with RACE (Lai et al. 2017), which is a large-scale reading comprehension dataset with more than 28,000 passages. RACE has significantly longer contexts than SQUAD. Our results are shown in Table 7. The overall trend is similar to SQUAD.

Visualization We further visualize the self-attention scores of BERT-base$^-$ and SegaBERT-base$^-$ in different layers. Figure 5 shows the average attention scores across different attention heads. By comparing Figure 5(d) with Figure 5(a), we find that SegaBERT can capture context according to the segmentation, for example, tokens tend to attend more to tokens in its paragraph than tokens in the other paragraphs. A similar trend can be observed at the sentence level but is more prominent in the shallow layers On the other hand, the BERT model seems to pay more attention to its neighbors: the attention weights of the elements around the main diagonal are larger than other positions in Figure 5(a), and a band-like contour around the main diagonal can be observed in this figure.

From Figure 5(f) and Figure 5(c), we can see the attention structure in the final layer is different from the shallow layers, and SegaBERT pays more attention to its context than BERT. We also notice that a fractal-like structure can be observed in the first 10 layers of SegaBERT, while the last two layers of SegaBERT have a striped structure.

These attention behaviors show that: in the shallow layers, our model is segment-aware while BERT is neighborhood-aware; in the top layers, both of these two models focus on some tokens across the article rather than local neighbors, but our model can capture more contextual tokens.

Related Work

Language modeling is a traditional natural language processing task which requires capturing long-distance dependencies for predicting the next token based on the context.

Most of the recent advances in language modeling are based on the Transformer (Vaswani et al. 2017) decoder architecture. Al-Rfou et al. (2019) demonstrated that self-attention can perform very well on character-level language modeling. Baevski and Auli (2019) proposed adaptive word input representations for the Transformer to assign more capacity to frequent words and reduce the capacity for less frequent words. Dai et al. (2019) proposed Transformer-XL to equip the Transformer with relative position encoding and cached memory for longer context modeling. Rae et al. (2020) extended the Transformer-XL memory segment to fine-grained compressed memory, which further increases the length of the context and obtains a perplexity of 17.1 on WikiText-103.

Although these works prove that longer context can be helpful for the language modeling task, how to generate better context representations with richer positional information has not been investigated.

On the other hand, large neural LMs trained with a massive amount of text have shown great potential on many NLP tasks, benefiting from the dynamic contextual representations learned from language modeling and other self-supervised pre-training tasks. OpenAI GPT (Radford 2018) and BERT (Devlin et al. 2019) are two representative models trained with the auto-regressive language modeling task and the masked language modeling task, respectively. In addition, BERT is also trained with an auxiliary task named next sentence prediction (NSP). ALBERT (Lan et al. 2020) then proposed to share parameters across layers of BERT and replaced NSP with sentence order prediction (SOP). According to their experiments, SOP is more challenging than NSP, and MLM together with other downstream tasks can benefit more from replacing NSP with SOP. Concurrently to ALBERT, Wang et al. (2020) proposed two auxiliary objectives to provide additional structural information for BERT.

All these powerful pre-trained models encode input tokens with token position encoding, which was first proposed by Vaswani et al. (2017) to indicate the position index of the input tokens in the context of machine translation and constituency parsing. After that, Transformer has been extensively applied in machine translation and other sequence generation tasks (Li et al. 2019; Liu and Lapata 2019; Roller et al. 2020). However, the input length of language modeling tasks are much longer than these tasks, and simply assigning 0–512 token position embeddings is not enough for LMs to learn the linguistic relationships among these tokens. Bai et al. (2020) show that incorporating segmentation information with paragraph separating tokens can improve the LM generator (GPT2) in the context of story generation. However, compared with punctuation and paragraph breaker, segment position indexes are more straightforward for dot-product self-attention based Transformers. In this work, we try to encode segmentation information into the Transformer with the segment-aware position encoding approach.

Conclusion

In this paper, we propose a novel segment-aware Transformer that can encode richer positional information for language modeling. By applying our approach to Transformer-XL, we train a new language model, Segatron-XL, that achieves 17.1 test perplexity on WikiText-103. Additionally, we pre-trained BERT with our SegaBERT approach and show that our model outperforms BERT on general language understanding, sentence representation learning, and machine reading comprehension tasks. Furthermore, our SegaBERT-large model outperforms RoBERTa-large on zero-shot STS tasks. These experimental results demonstrate that our proposed method works on both language models with relative position embeddings and pre-trained language models with absolute position embeddings.

Acknowledgments

This work was partially supported by NSERC OGP0046506, the National Key R&D Program of China 2016YFB1000902 and 2018YFB1003202. We would like to thank Wei Zeng and his team in Peng Cheng Laboratory (PCL) for computing resources to support this project.

References

Al-Rfou, R.; Choe, D.; Constant, N.; Guo, M.; and Jones, L. 2019. Character-Level Language Modeling with Deeper Self-Attention. In *AAAI 2019, Hawaii, USA, January 27 - February 1, 2019*, 3159–3166.

Baevski, A.; and Auli, M. 2019. Adaptive Input Representations for Neural Language Modeling. In *ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Bai, H.; Shi, P.; Lin, J.; Tan, L.; Xiong, K.; Gao, W.; Liu, J.; and Li, M. 2020. Semantics of the Unwritten. *Arixv* abs/2004.02251.

Bentivogli, L.; Magnini, B.; Dagan, I.; Dang, H. T.; and Giampiccolo, D. 2009. The Fifth PASCAL Recognizing Textual Entailment Challenge. In *TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*.

Bird, S.; Klein, E.; and Loper, E. 2009. *Natural Language Processing with Python*. O'Reilly.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In *EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, 632–642.

Cer, D. M.; Diab, M. T.; Agirre, E.; Lopez-Gazpio, I.; and Specia, L. 2017. SemEval-2017 Task 1: Semantic Textual Similarity - Multilingual and Cross-lingual Focused Evaluation. *Arxiv* abs/1708.00055.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J. G.; Le, Q. V.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *ACL 2019, Florence, Italy, July 28- August 2, 2019*, 2978–2988.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019*, 4171–4186.

Dolan, W. B.; and Brockett, C. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *IWP@IJCNLP 2005, Jeju Island, Korea, October 2005*.

Grave, E.; Joulin, A.; and Usunier, N. 2017. Improving Neural Language Models with a Continuous Cache. In *ICLR 2017, Toulon, France, April 24-26, 2017*.

Krause, B.; Kahembwe, E.; Murray, I.; and Renals, S. 2018. Dynamic Evaluation of Neural Sequence Models. In *ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80, 2771–2780.

Lai, G.; Xie, Q.; Liu, H.; Yang, Y.; and Hovy, E. H. 2017. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In *EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, 785–794.

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Li, J.; Wang, X.; Yin, D.; and Zong, C. 2019. Attribute-aware Sequence Network for Review Summarization. In *EMNLP-IJCNLP 2019*, 3000–3010. Hong Kong, China.

Liu, Y.; and Lapata, M. 2019. Text Summarization with Pretrained Encoders. In *EMNLP-IJCNLP*, 3721–3731.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *Arixv* abs/1907.11692.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2017. Pointer Sentinel Mixture Models. In *ICLR 2017, Toulon, France, April 24-26, 2017*.

Pan, X.; Sun, K.; Yu, D.; Chen, J.; Ji, H.; Cardie, C.; and Yu, D. 2019. Improving Question Answering with External Knowledge. In *MRQA@EMNLP 2019, Hong Kong, China, November 4, 2019*, 27–37.

Radford, A. 2018. Improving Language Understanding by Generative Pre-Training.

Rae, J. W.; Dyer, C.; Dayan, P.; and Lillicrap, T. P. 2018. Fast Parametric Learning with Activation Memorization. In *ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80, 4225–4234.

Rae, J. W.; Potapenko, A.; Jayakumar, S. M.; Hillier, C.; and Lillicrap, T. P. 2020. Compressive Transformers for Long-Range Sequence Modelling. In *ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Rajpurkar, P.; Jia, R.; and Liang, P. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, 784–789.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, 2383–2392.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, 3980–3990.

Roller, S.; Dinan, E.; Goyal, N.; Ju, D.; Williamson, M.; Liu, Y.; Xu, J.; Ott, M.; Shuster, K.; Smith, E. M.; et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637* .

Roy, A.; Saffar, M.; Vaswani, A.; and Grangier, D. 2020. Efficient Content-Based Sparse Attention with Routing Transformers. *arXiv* abs/2003.05997.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA*, 1631–1642.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NeurIPS 2017, 4-9 December 2017, Long Beach, CA, USA*, 5998–6008.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Wang, W.; Bi, B.; Yan, M.; Wu, C.; Bao, Z.; Peng, L.; and Si, L. 2020. StructBERT: Incorporating Language Structures into Pre-training for Deep Language Understanding. In *ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Warstadt, A.; Singh, A.; and Bowman, S. R. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics* 7: 625–641.

Williams, A.; Nangia, N.; and Bowman, S. R. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1*, 1112–1122.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R.; and Le, Q. V. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, 5754–5764.

Zhu, Y.; Kiros, R.; Zemel, R. S.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In *ICCV 2015, Santiago, Chile, December 7-13, 2015*, 19–27.