Segatron: Segment-aware Transformer for Language Modeling and Understanding

Anonymous submission

Abstract

Transformers are powerful for sequence modeling and have the potential of learning long-term dependency. 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 the Transformer-XL, which is a popular Transformer model based on the relative position encoding with memory extension. Our proposed method outperforms the Transformer-XL base model and large model on the Wiki103 dataset over 1.5 and 1.2 perplexities, respectively, which is comparable to the state-of-the-art result. We further pre-trained our model on the masked language modeling task in BERT but without any affiliated tasks. Experimental results show that our pre-trained model can outperform the original BERT model on various NLP tasks.

1 Introduction

Language modeling (LM) is a traditional sequence modeling task which requires to learn the long-term dependency for the next token prediction base on the previous context. Large neural LM trained on a massive amount of text data has shown great potential on transfer learning and achieved state-of-the-art results in various natural language processing tasks. Compared with traditional word embedding such as Skip-Gram (Mikolov et al., 2013) and Glove (Pennington et al., 2014), pre-trained LM can learn contextual representation and can be fine-tuned as the text encoder for downstream tasks, such as OpenAI GPT (Radford, 2018), BERT (Devlin et al., 2018), XLNET (Yang et al., 2019b) and BART (Lewis et al., 2019). Therefore, pre-trained LM has emerged as a convenient technique in natural language processing.

Most of these pre-trained models use a multi-layer Transformer (Vaswani et al., 2017) and are pre-trained with self-supervised tasks such as language modeling (LM) or masked language modeling (MLM). Besides, state-of-the-art language models (Dai et al., 2019; Baevski and Auli, 2019; Rae et al., 2020) are also based on the Transformer network.

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, in the LM scenery, the input length can range from 512 tokens to 1024 tokens and come from different sentences and paragraphs. Although the token position embedding can help the transformer be aware of the token position by assigning unique index for each token, the token position in a sentence, sentence position in a paragraph, and paragraph position in a document are all implicit. Such segmentation information is essential for language modeling and understanding, which can help to encode better contextual representations.

Hence, we propose a novel segment-aware Transformer (Segatron), which encodes paragraph index in a document, sentence index in a paragraph, token index in a sentence all together during pre-training and fine-tuning stages. We first verify the proposed method with relative position encoding on the language modeling task. By applying the segment-aware mechanism onto Transformer-XL (Dai et al., 2019), our base model trained with the WikiText-103 (Merity et al., 2017) dataset outperforms the

Transformer-XL base by 1.5 perplexities. For the large model, we outperform Transformer-XL large by 1.2 perplexities and achieve the same result as the state-of-the-art model (Rae et al., 2020). We also pre-train the Segatron with MLM target in the same settings with BERT but without the next sentence prediction (NSP) or other affiliated tasks. According to the experimental results, our pre-trained model, SegaBERT, outperforms BERT on both general language understanding and machine reading comprehension: 1.17 average score improvement on GLUE (Wang et al., 2019a), 1.14/1.54 exact match/F1 score improvement on SQUAD v1.1 (Rajpurkar et al., 2016), and 1.24/1.80 exact match/F1 score improvement on SQUAD v2.0 (Rajpurkar et al., 2018).

2 Related Work

Language modeling is a traditional natural language processing task which requires to model the long-term dependency for predicting the next token based on the context.

Most of the recent advances in language modeling are base 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 the 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 prolongs the length of the context. Although longer context has been proven to be helpful for language modeling in these works, how to generate better context representation 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., 2018) are two representative models trained with the auto-regressive language modeling task and the masked language modeling task, respectively. Besides, BERT 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 the NSP to the sentence order prediction (SOP). According to their experiments, the SOP is more challenging than NSP. MLM together with other downstream tasks can benefit from replacing NSP with SOP. Concurrently to ALBERT, Wang et al. (2019b) incorporated a word level objective (restore the order of shuffled words) and sentence level objective (predict whether the second sentence is the next, previous or random one to the first sentence) into BERT. These two tasks can provide additional structural information for BERT.

All these powerful pre-trained models encode input tokens with token position embeddings, 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. However, the input length of the translation or parsing is much shorter than the language modeling, whose inputs usually range from 512 to 1024 and come from different sentences and paragraphs. The motivation for modeling such a long context during pre-training is to learn better contextual representations. But simply assign 0-512 or 0-1024 token position embeddings is not enough for LM to learn the inner relationship among these tokens. Bai et al. (2020) propose to incorporate the segmentation information with paragraph separating tokens, which improves the LM generator in the context of story generation. In this work, we try to encode segmentation information into the Transformer with the segment-aware position encoding approach.

3 Model

3.1 Segment-aware Transformer-XL

Transformer-XL is a memory augmented transformer with relative position encoding. The relative position information is encoded as follows:

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

where $\mathbf{A}^{rel}_{i,j}$ is the self-attention score between query i and key j. $\mathbf{E}^T_{x_i}$ is the input representation of query i. $\mathbf{R}_{i-j}$ is the relative position vector. The other symbols are all learnable variables and detailed in (Dai et al., 2019).

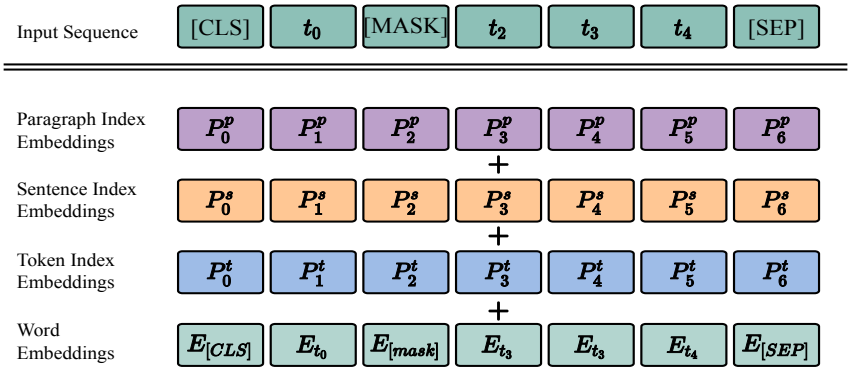


Figure 1: Input Representation of SegaBERT

According to the position encoding implementation of Transformer-XL, the sine and cosine functions are used:

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

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

To introduce paragraph and sentence segmentation to the relative position encoding, the new position vector is defined as:

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

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 vectors of token, sentence, and paragraph. These vectors are defined in Eq. 2 and the dimension of them equals to 1/3 dimension of $\mathbf{R_{I,J}}$.

To equip the recurrence mechanism of Transformer-XL with the segment-aware relative position encoding, the paragraph position, sentence position and 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.

3.2 Segment-aware BERT

Our SegaBERT is based on the BERT architecture, which is a multi-layer transformer-based bidirectional masked language model. For a long sequence of text, SegaBERT leverages the segment information, such as the sentence position and paragraph position information, to learn a better contextual representation for each token. The original BERT uses a learned position embedding to encode the tokens' position information. Instead of using the global token indexing, we introduce three types of embeddings: Token Index Embedding, Sentence Index Embedding, and Paragraph Index Embedding, as shown in Figure 1. Thus, the global token position is uniquely determined by three parts in SegaBERT: the token index within a sentence, the sentence index within a paragraph, and the paragraph index within a long document.

Input Representation. Input $\mathbf{X}$ is a sequence of tokens, which can be one or more sentences or paragraphs. Similar to the input representation used in BERT, the representation x_t for the 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 P_t^p. Two special tokens `[CLS]` and `[SEP]` are added into the text sequence before the first token and after the last token. Following BERT, the text is tokenized into subwords with WordPiece and the maximum sequence length is 512.

Encoder Architecture. The multi-layer bidirectional Transformer encoder is used to encode the contextual representation for the inputs. With L-layer Transformer, the last layer hidden vector $\mathbf{H}_t^L$ of token t is used as the contextualized representation. With the rich segmentation information present in the input representation, the encoder has a better contextualization ability.

Training Objective. Following BERT, we use the masked LM as our training objective. The other training objective, next sentence prediction, is not used in our model.

Training Setup. For the SegaBERT-base, we pretrain the model with English Wikipedia. For the

Model	#Param.	PPL
LSTM+Neural cache (Grave et al., 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
SegaTransformer-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
SegaTransformer-XL large, M=384	257M	**17.1**

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

SegaBERT-large, English Wikipedia and BookCorpus (wikibooks) are used. The text is preprocessed following BERT and tokenized into sub-tokens with WordPiece. For each document of Wikipedia, we firstly split that 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, 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. i-th sub-token will have its Token Index Embedding $\mathbf{P}_i^t$. The maximum paragraph index, sentence index, and token index are 50, 100, and 256, respectively.

We conduct our experiments based on two model sizes: SegaBERT-base and SegaBERT-large:

- **SegaBERT-base**: L=12, H=768, A=12
- **SegaBERT-large**: L=24, H=1024, A=24

Here, we use L to denote the number of Transformer layers, H to denote the hidden size and A to denote the number of self-attention heads.

The pre-training is based on 16 Tesla V100 GPU cards. We train 500K steps for the SegaBERT-base and 1M steps for SegaBERT-large. For the optimization, we use Adam with the learning rate of 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.

4 Experiments

In this section, we first show the results of the SegaTransformer-XL on language modeling (WikiText-103). Then, we show the results of SegaBERT finetuned on several downstream tasks: General Language Understanding Evaluation (GLUE) benchmark and extractive question answering (SQUAD v1.1 and SQUAD v2.0).

4.1 WikiText-103

WikiText-103 is the largest available word-level dataset with long-term dependency for language modeling. There are 103M tokens, 28K articles for training. The average length is 3.6K tokens per article. Following Transformer-XL, we train a base model and a large model with WikiText-103.

The base model is a 16 layer transformer with hidden size of 410 and 10 self-attention heads. This model is trained with 64 batch size, 200K steps. On the other hand, the large model is an 18 layer transformer with hidden size of 1024 and 16 attention heads. This model is trained with 128 batch size, 350K steps. The sequence length and memory length during training and testing all equal to 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 while the Transformer-XL is trained in full-precision; the perplexities we report are tested with the same sequence length and memory length of training; the large model training steps of Transformer-XL is 4M according to their implementation.

The 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 for the Transformer-XL base and decreases 1.2 for Transformer-XL large model. We obtain 17.1 perplexities with our large model – comparable with prior state-of-the-art results of Compressive Transformer (Rae et al., 2020),

Task(Metrics)	BASE model(wikipedia 500K steps)				LARGE model(wikibooks 1000K steps)			
	dev		test		dev		test	
	BERT	SegaBERT	BERT	SegaBERT	BERT	SegaBERT	BERT	SegaBERT
CoLA (Matthew Corr.)	**55.0**	54.7	43.5	**50.7**	60.6	**65.3**	60.5	**62.6**
SST-2 (Acc.)	91.3	**92.1**	91.2	**91.5**	93.2	**94.7**	**94.9**	94.8
MRPC (F1)	**92.6**	92.4	88.9	**89.3**	-	92.3	89.3	**89.7**
STS-B (Spearman Corr.)	88.9	**89.0**	83.9	**84.6**	-	90.3	86.5	**88.6**
QQP (F1)	86.5	**87.0**	70.8	**71.4**	-	89.1	72.1	**72.5**
MNLI-m (Acc.)	83.2	**83.8**	82.9	**83.5**	86.6	**87.6**	86.7	**87.9**
MNLI-mm (Acc.)	83.4	**84.1**	82.8	**83.2**	-	87.5	85.9	**87.7**
QNLI (Acc.)	90.4	**91.5**	90.1	**90.8**	92.3	**93.6**	92.7	**94.0**
RTE (Acc.)	68.3	**71.8**	65.4	**68.1**	70.4	**78.3**	70.1	**71.6**
Average	82.2	**82.9**	77.7	**79.2**	-	86.5	82.1	**83.3**

Table 2: The results on GLUE benchmark. All base models are pre-trained by this work. Every result of the dev set is the average score of 4 times finetuning with different random seeds. Scores of BERT large dev are from (Sun et al., 2019) and scores of BERT large test are from (Devlin et al., 2018).

which is based on the Transformer-XL but trained with longer input length and memory length (512) and more complicated memory cache mechanism. Compared with Transformer-XL large, we achieve the same ppl 18.3 with only 172K steps.

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 notice that there is a contemporaneous work RoutingTransformer (Roy et al., 2020), which modifies the self-attention to local and sparse attention with clustering method. However, their word is in progress and codes are not available. We believe our segment-aware method is vertical to their work and can also be introduced to their model.

4.2 GLUE

The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019a) is a collection of resources for evaluating natural language understanding systems. Following Devlin et al. (2018), we evaluate our model over these tasks: linguistic acceptability CoLA (Warstadt et al., 2018), sentiment SST-2 (Socher et al., 2013), paraphrase MRPC (Dolan and Brockett, 2005), textual similarity STS-B (Cer et al., 2017), question paraphrase QQP[1], textual entailment RTE (Bentivogli et al., 2009) and MNLI (Williams et al., 2018), and question entailment QNLI (Wang et al., 2019a). For all tasks, 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, such as sentiment classification (SST-2), the sentence will be assigned Paragraph Index 0 and Sentence Index 0. For sentence pair classification, 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 the grid-search with GLUE dev set for low-resource tasks CoLA, MRPC, RTE, SST-2, and STS-B. Our grid search space is as follows: Batch size: 16, 24, 32; Learning rate: 2e-5, 3e-5, 5e-5; Number of epochs: 3-10. For QQP, MNLI, and QNLI, the default hyper-parameters are 3e-5 learning rate, 256 batch size, and 3 epochs. The other hyper-parameters are the same as in the HuggingFace Transformers library.[2]

As we can see from Table 2, the average GLUE score of our SegaBERT is higher than BERT on both the dev set and test set. Our large model outperforms the original BERT by 1.2 points on average GLUE score and achieves better scores nearly on all tasks. A similar trend can be observed by comparing the SegaBERT base with the BERT base pre-trained in this work. The improvements over these sentence and sentence pair classification tasks show that our segment-aware pre-trained model can generate better sentence representations compared with BERT. These results demonstrate SegaBERT's effectiveness and generalization capability in natural language understanding.

[1] https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

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

System	Dev	
	EM	F1
BERT base (Single)	80.8	88.5
BERT large (Single	84.1	90.9
BERT large (Single+DA)	84.2	91.1
KT-NET	85.2	91.7
StructBERT Large (Single)	85.2	92.0
SegaBERT base (Single)	83.2	90.2
SegaBERT large (Single)	85.3	92.4

Table 3: Evaluation results of SQUAD v1.1.

System	Dev	
	EM	F1
BERT base	72.3	75.6
BERT base (ours)	75.4	78.2
SegaBERT base	76.3	79.2
BERT large	78.7	81.9
BERT large wwm	80.6	83.4
SegaBERT large	81.8	85.2

Table 4: Evaluation results of SQUAD v2.0.

4.3 Reading Comprehension

For the Reading Comprehension task, we fine-tune in the following manner: The question is assigned Paragraph Index 0 and Sentence Index 0. For the context with n paragraphs, Paragraph Index 1 to $n + 1$ are assigned to them accordingly. Within each paragraph, the sentences are indexed from 0. In this dataset, each question only corresponds to one paragraph. Hence, the paragraph index of the context is 1.

This task can benefit from the segment information more than the tasks in GLUE benchmark because for the reading comprehension task, the context is usually longer, spanning several sentences. The segment position embedding can guide the self-attention layer to encode the text better.

We fine-tune our SegaBERT model with SQUAD v1.1 for 4 epochs with 128 batch size and 3e-5 learning rate. As we can see from Table 3, without any data augmentation (DA) or model ensemble, SegaBERT large outperforms BERT large with DA, StructBERT (Wang et al., 2019b) which pre-trains BERT with multiple unsupervised tasks, and KT-NET (Yang et al., 2019a) which uses external knowledge bases on BERT.

Model	PPL
SegaTransformer-XL base	22.47
- paragraph position encoding	22.51
- sentence position encoding	24.07
Transformer-xl base	24.35

Table 5: Ablation over the position encodings using Transformer-XL base architecture.

The finetuning setting of SQUAD v2.0 is the same as SQUAD v1.1. Results of SQUAD v2.0 are shown in Table 4. Although pre-trained with fewer data and steps, our BERT base outperforms the original BERT base on this task. Compared with our BERT base results, the SegaBERT base further improves 0.9 exact match score and 1.0 F1 score. Besides, we can see that the SegaBERT large even outperforms the BERT large with whole word masking: 1.2 exact match scores and 1.8 F1 scores.

Although we only compared the SegaBERT with BERT-based models for extractive question answering tasks, our proposed method is not specified for BERT but the transformer. We believe that the segmentation information can also help other pre-trained models and could be verified in the future.

4.4 Ablation Study

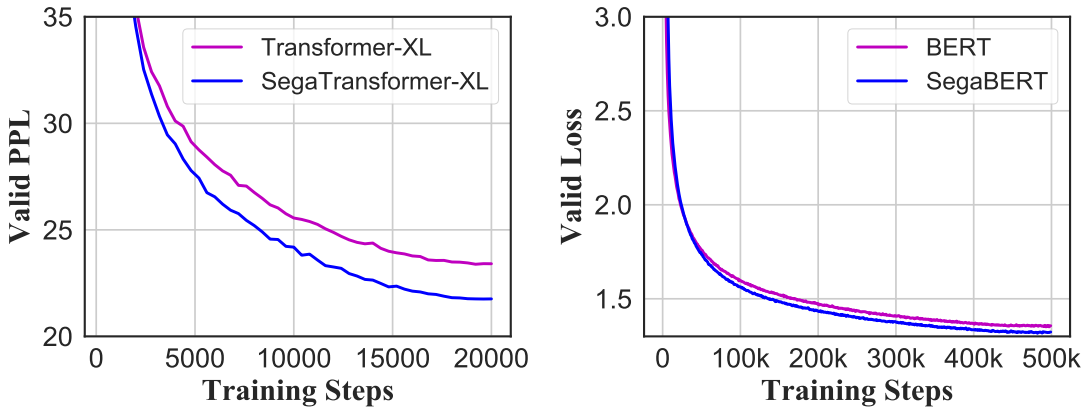
We first conduct an ablation study with SegaTransformer-XL base, to investigate the contributions of the sentence position encoding and the paragraph position encoding, respectively. Experimental results are shown in Table 5. From this table, we can find that the test perplexity decrease from 24.35 to 22.51 without sentence position encoding, from 24.35 to 24.07 without paragraph position encoding. The results show that sentence position encoding is more important than paragraph position encoding for language modeling.

We further conduct another ablation study on token position encoding with BERT base model. The token position in SegaBERT is re-ranged for each sentence and here, we name it as the re-ranged token position encoding. To investigate the contributions of re-ranged token position encoding and segmentation (paragraph and sentence) encoding, we pre-train a base model, BERT with P.S., by adding paragraph encoding and sentence encoding to the original BERT token position encoding.

The results are shown in Table 6. From this table, we can see that the BERT with P.S. still out-

Task(Metrics)	Base model Trained on wikipedia for 500k steps		
	BERT	BERT with P.S.	SegaBERT
CoLA (Matthew Corr.)	55.0	**55.2**	54.7
SST-2 (Acc.)	91.3	**92.1**	**92.1**
MRPC (F1)	**92.6**	92.0	92.4
STS-B (Spearman Corr.)	88.9	**89.2**	89.0
QQP (F1)	86.5	**87.1**	87.0
MNLI-m (Acc.)	83.2	83.2	**83.8**
MNLI-mm (Acc.)	83.4	83.7	**84.1**
QNLI (Acc.)	90.4	**91.5**	**91.5**
RTE (Acc.)	68.7	67.9	**71.8**
GLUE Average	82.2	82.4	**82.9**
SQUAD v1.1 (EM/F1)	81.9/89.4	83.0/**90.3**	**83.2**/90.2

Table 6: Results of base models on dev set of GLUE and SQUAD v1.1. Every result is the average score of 4 runs with different random seeds



(a) Language modeling perplexities

(b) Pre-training losses

Figure 2: Valid perplexities and losses during the training processes of language modeling and pre-training.

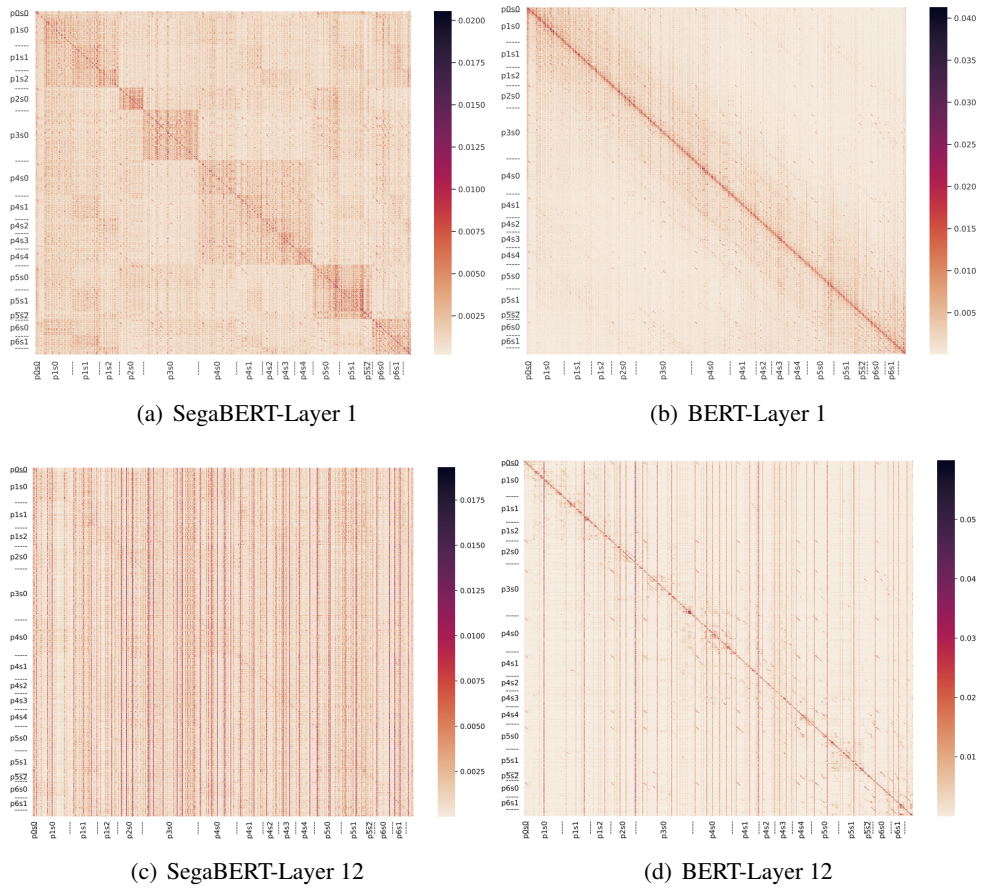
performs the BERT base model on most GLUE tasks. But the average GLUE score is 0.7 lower than SegaBERT, which indicates the re-ranged token position encoding is critical for GLUE tasks. On the other hand, we can observe that BERT with P.S. is comparable with SegaBERT on the SQUAD v1.1, and both of them outperform the BERT base model. This suggests that the segmentation information plays a more important role in machine reading comprehension. These results are reasonable considering the inputs of GLUE tasks are shorter and can benefit from the re-ranged token position encoding, while the inputs of machine reading comprehension task are much longer with richer segmentation information.

4.5 Visualization

In this section, we first plot the valid perplexities of SegaTransformer-XL base and Transformer-XL base during the training process in Figure 2(a). From this figure, we can see that the segment-aware model outperforms the base model and the gap between them becomes larger along the training process. The SegaTransformer-XL at 10K steps approximately matches the performance of Transformer-XL at 20K steps.

We also plot the valid losses of SegaBERT base and BERT base during pre-training in Figure 2(b). At the beginning stage of training, we can see that the validation loss of BERT base is slightly lower than the SegaBERT. After about 30K steps, the SegaBERT begins to outperform BERT steadily. The overall trends of these two figures are similar, which demonstrates our proposed segment-aware method works on both the auto-regressive language modeling and the masked language modeling.

We further visualize the self-attention scores of SegaBERT and BERT base model. Figure 3 shows the average attention scores across different attention heads. By comparing Figure 3(a) with Figure 3(b), we can find that the SegaBERT can cap-



(a) SegaBERT-Layer 1 (b) BERT-Layer 1

(c) SegaBERT-Layer 12 (d) BERT-Layer 12

Figure 3: Self-attention heat maps in the first/last layer of SegaBERT and BERT during encoding the first 512 tokens of a wikipedia article.

ture the context according to the segmentation, for example, token tends to attend more to tokens in its paragraph than tokens in the other paragraphs. A similar trend can be observed in the sentence level but is more prominent in the upper layers, which is shown in Figure 4(a), Figure 5(a), and Figure 7(a) in Appendix A.1. On the other hand, the BERT model without segment-aware 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 3(b).

From Figure 3(c) and Figure 3(d), 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 the semi-fractal like structure can be observed in the first 10 layers of SegaBERT, while the last two layers of SegaBERT are striped structure, which are shown in Figure 3 and Appendix A.1.

These attention behaviors show that our model is prone to attend with segmentation while BERT attends with distance in the shallow layers. In the top layers, both of these two models pay attention to important tokens but our model is more contextual: attends more tokens in the context.

5 Conclusion

In this paper, we propose a novel segment-aware transformer that can encode richer positional information for language modeling. By applying our approach on Transformer-XL and BERT, we train a new language model SegaTransformer-XL and a new pre-trained model SegaBERT, respectively. Our SegaTransformer-XL achieves 17.1 test perplexities on WikiText-103, which is the same score as the SOTA model. On the other hand, our SegaBERT outperforms BERT on GLUE, SQUAD v1.1, and SQUAD v2.0. The experimental results demonstrate that our proposed method works on both relative and absolute position encodings, learnable and non-learnable position embeddings, pre-trained and non-pre-trained language models.

References

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2019. Character-level language modeling with deeper self-attention. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3159–3166. AAAI Press.

Alexei Baevski and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

He Bai, Peng Shi, Jimmy Lin, Luchen Tan, Kun Xiong, Wen Gao, Jie Liu, and Ming Li. 2020. Semantics of the unwritten. *CoRR*, abs/2004.02251.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*. NIST.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity - multilingual and cross-lingual focused evaluation. *CoRR*, abs/1708.00055.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Edouard Grave, Armand Joulin, and Nicolas Usunier. 2017. Improving neural language models with a continuous cache. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ben Krause, Emmanuel Kahembwe, Iain Murray, and Steve Renals. 2018. Dynamic evaluation of neural sequence models. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2771–2780. PMLR.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Alec Radford. 2018. Improving language understanding by generative pre-training.

Jack W. Rae, Chris Dyer, Peter Dayan, and Timothy P. Lillicrap. 2018. Fast parametric learning with activation memorization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4225–4234. PMLR.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2020. Efficient content-based sparse attention with routing transformers. *CoRR*, abs/2003.05997.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2019. ERNIE 2.0: A continual pre-training framework for language understanding. *CoRR*, abs/1907.12412.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Liwei Peng, and Luo Si. 2019b. Structbert: Incorporating language structures into pre-training for deep language understanding. *CoRR*, abs/1908.04577.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019a. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2346–2357. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764.

A Appendix

A.1 Self-attention heat maps

The input article is shown below. After chunked with 512 maximum sequence length limitation, we plot different layer's self attention heat maps in Figure 4, Figure 5, Figure 6, and Figure 7.

Input article:

Japanese destroyer Hatsukaze

The K̈agerō-̈class destroyers were outwardly almost identical to the preceding light cruiser-sized , with improvements made by Japanese naval architects to improve stability and to take advantage of Japan's lead in torpedo technology. They were designed to accompany the Japanese main striking force and in both day and night attacks against the United States Navy as it advanced across the Pacific Ocean, according to Japanese naval strategic projections. Despite being one of the most powerful classes of destroyers in the world at the time of their completion, only one survived the Pacific War.

Ḧatsukaze, built at the Kawasaki Shipbuilding Corporation, was laid down on 3 December 1937, launched on 24 January 1939 and commissioned on 15 February 1940.

At the time of the attack on Pearl Harbor, Ḧatsukaze, was assigned to Destroyer Division 16 (Desdiv 16), and a member of Destroyer Squadron 2 (Desron 2) of the IJN 2nd Fleet, and had deployed from Palau, as part of the escort for the aircraft carrier in the invasion of the southern Philippines and minelayer .

In early 1942, Ḧatsukazeṗarticipated in the invasion of the Netherlands East Indies, escorting the invasion forces for Menado, Kendari and Ambon in January, and the invasion forces for Makassar, Timor and eastern Java in February. On 27-28 February, Ḧatsukazeänd Desron 2 participated in the Battle of the Java Sea, taking part in a torpedo attack on the Allied fleet. During the month of March, Desron 2 was engaged in anti-submarine operations in the Java Sea. At the end of the month, the squadron escorted the Christmas Island invasion force, then returned to Makassar. At the end of April, Ḧatsukazeš̈ailed to Kure Naval Arsenal for maintenance, docking on 3 May.

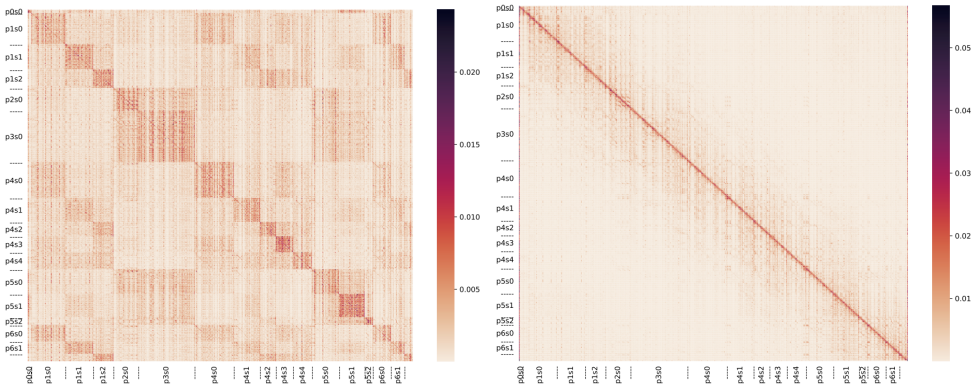
On 21 May 1942, Ḧatsukazeänd Desron 2 steamed from Kure to Saipan, where they rendezvoused with a troop convoy and sailed toward Midway Island. Due to the defeat of the Carrier Striking Force and loss of four fleet carriers in the Battle of Midway, the invasion was called off and the convoy withdrew without seeing combat. Desdiv 16 was ordered back to Kure.

On 14 July, Ḧatsukazeänd Desdiv 16 were reassigned to Desron 10, Third Fleet. On 16 August, Desron 10 departed Kure, escorting a fleet towards Truk. On 24 August, Desron 10 escorted Admiral Nagumo's Striking Force in the Battle of the Eastern Solomons. During September and October, the squadron escorted the fleet patrolling out of Truk north of the Solomon Islands. On 26 October, in the Battle of the Santa Cruz Islands, the squadron escorted the Striking Force, then escorted the damaged carriers and into Truk on 28 October. On 4 November, Desron 10 escorted from Truk to Kure, then engaged in training in the Inland Sea, and then escorted Z̈uikakuf̈rom Truk to the Shortland Islands in January 1943.

On 10 January, while providing cover for a supply-drum transport run to Guadalcanal, Ḧatsukazeässisted in sinking the American PT boats P̈T-43änd P̈T-112.S̈he suffered heavy damage when struck by a torpedo (possibly launched by P̈T-112)ïn the port side; her best speed was 18 knots as she withdrew to Truk, for emergency repairs. Then she sailed to Kure in April for more extensive repairs. In September, Ḧatsukazeänd Desron 10 escorted the battleship from Kure to Truk. In late September and again in late October, Desron 10 escorted the main fleet from Truk to Eniwetok and back again, in response to American carrier airstrikes in the Central Pacific region. Between these two missions, Ḧatsukazeš̈ortied briefly from Truk in early October 1943 to assist the fleet oiler Ḧazakaya,ẅhich had been torpedoed by an American submarine.

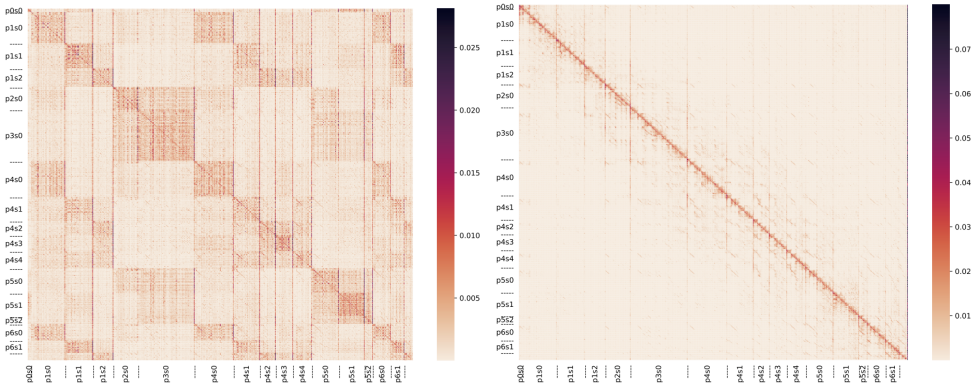
On 2 November 1943, while attacking an Allied task force off Bougainville in the Battle of Empress Augusta Bay, Ḧatsukazec̈ollided with the cruiser . The collision sheared off her bow, leaving her dead in the water. Ḧatsukazeänd the light cruiser were sunk (at position) by Allied destroyer gunfire. Of those on board, 164 were killed, including its commanding officer, Lieutenant Commander Buichi Ashida.

Ḧatsukazeẅas removed from the navy list on 5 January 1944."



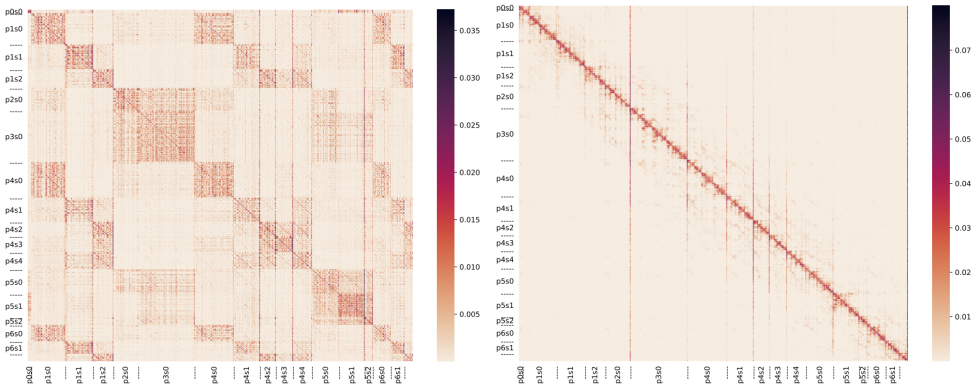
(a) SegaBERT-Layer 3 (b) BERT-Layer 3

Figure 4: Attention heat maps of SegaBERT and BERT base model in the 3rd layer during encoding the first 512 tokens of a wikipedia article.



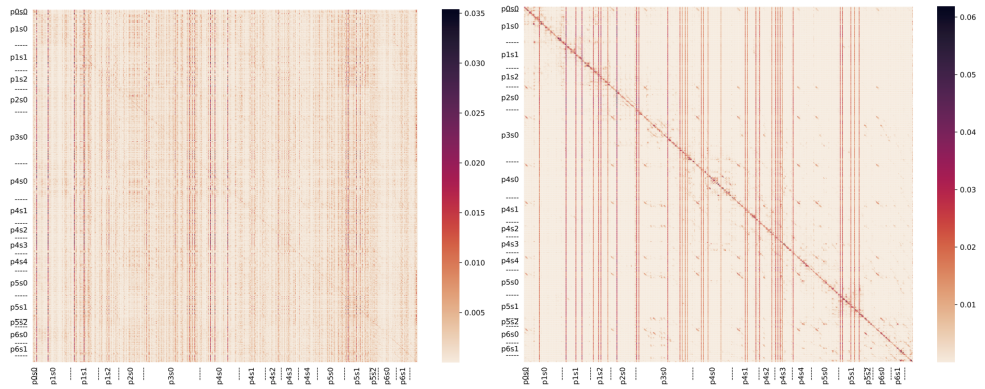
(a) SegaBERT-Layer 6 (b) BERT-Layer 6

Figure 5: Attention heat maps of SegaBERT and BERT base model in the 6th layer during encoding the first 512 tokens of a wikipedia article.



(a) SegaBERT-Layer 9 (b) BERT-Layer 9

Figure 6: Attention heat maps of SegaBERT and BERT base model in the 9th layer during encoding the first 512 tokens of a wikipedia article.



(a) SegaBERT-Layer 11 (b) BERT-Layer 11

Figure 7: Attention heat maps of SegaBERT and BERT base model in the 11th layer during encoding the first 512 tokens of a wikipedia article.