We thank all reviewers for their efforts. Below we give detailed responses.

1. "More examples and analysis of attention map" (R1&R3) We provide more examples of the attention map in Figure 1. we also compute the diagonal concentration for the attention map M as quantitative metric. It is define as $C = \frac{\sum_{|i-j|\le 4} M_{i,j}}{\sum_{|i-j|>4} M_{i,j}}$. This indicates how much local dependency that the attention map captures. The result in Table 1 shows that the attention in BERT concentrates more on the local dependency.

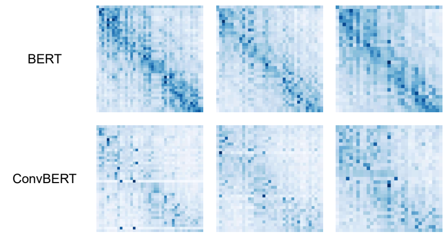

Figure 1: More examples of attention maps.

2. "Inference speed" (R2&R3) We test our mixed-attention block and self-attention baseline from base-sized model on Intel CPU (i7-6900K@3.20GHz). The mixed-attention has lower Flops and is much faster than self-attention, as shown in Table 2. On the other hand, for the code we submitted as the supplementary material, our implementation for mixed-attention on GPU and TPU is not well optimized for the efficiency yet. Thus its acceleration may not be obvious when the input sequence length is short. We will work on further improvement on the low-level implementation.

3. "More experiments on other NLP tasks" (R1&R2&R3) This paper focuses on improving BERT and thus the experiments are conducted in the language pre-training scenario. Thanks for reviewers' suggestions that remind us span based dynamic convolution/mixed attention may be applied for other NLP tasks. Due to limited time for rebuttal, we have not finished tuning the mixed attention model (of similar size as small ConvBERT), but it has shown better performance than transformers (of similar size as small BERT) on language modeling task on WikiText-103, as shown in Table 3. We will explore span-based dynamic convolution/mixed attention on other tasks in the future.

Model	C (diagonal-concentration)
BERT	0.941
ConvBERT	0.608

Table 1: Average concentration on MRPC.

Block	Flops	Speed (ms/sample)
self-attention	26.5G	17.66
mixed-attention	19.3G	**12.94**

Table 2: Inference speed.

Model	Perplexity
Transformer	34.21
Ours	**32.95**

Table 3: Result on WikiText-103.

Model	Modification	Params	GLUE
ConvBERTmedium-small	+BNK,+GL	14M	81.0
ConvBERTmedium-small	+BNK,+GL,+Larger	17M	81.1

Table 4: Ablation study on GLUE dev set.

4. "Actual training time" (R2&R3) We use direct implementation of our proposed algorithm without dedicated low-level engineering acceleration at this stage as done in ELECTRA BERT baseline. Even such, we achieve 2.67× training acceleration (from 8 days to 3 days) on the base-sized model as shown in Table 5.

Model	Training time
ELECTRA-small	12h
ELECTRA-base	192h
CONVBERT-small	12h
CONVBERT-medium-small	18h
CONVBERT-base	72h

Table 5: Training time on TPU v3-8.

5. "Baseline and ablation study" (R1&R2&R4) As suggested by R1, we add a '+BNK,+GL,+Larger' baseline that increases the hidden dimension to 432. Result are shown in Table 4. As expected, increasing the hidden dimension only slightly improves the result (+0.1). As suggested by R2&R4, we add the experiments of only convolution based architecture (also small sized) which performs poorly on downstream tasks and only achieves 64 on GLUE.

6. "Definition of Grouped feed-forward" (R1) We will add the detailed definition in the final version. The grouped feed-forward module is defined as follows

$$M = \Pi_{i=0}^{g}\left[f^i_{\frac{d}{g}\to\frac{m}{g}}\left(H_{[:,\ i-1:i\times\frac{d}{g}]}\right)\right], \quad M' = \mathrm{GeLU}(M), \quad H' = \Pi_{i=0}^{g}\left[f^i_{\frac{m}{g}\to\frac{d}{g}}\left(M'_{[:,\ i-1:i\times\frac{m}{g}]}\right)\right], \qquad (1)$$

where $H, H' \in \mathbb{R}^{n\times d}$, $M, M' \in \mathbb{R}^{n\times m}$, $f_{d_1\to d_2}(\cdot)$ indicates a fully connected layer that transforms dimension d_1 to d_2, g is the group number and Π means concatenation.

6. " Why use point-wise multiplication of $\mathbf{K_s}$ and Q" (R4) Instead of only using K_s, using point-wise multiplication (a bi-linear operator) can merge information between a single token and its nearby tokens. Compared with concatenation that makes the following fc layer occupy more parameters, it saves parameter number.

7. "Comparison with Lite Transformer" (R2) Lite transformer uses two branches of self-attention and dynamic convolution. It is applied for translation, language modeling and abstractive summarization. While our ConvBERT introduces span-based dynamic convolution with self-attention to form mixed-attention and is targeted for pre-training. We have implemented 'Lite Transformer'-like architecture for pre-training. See results in row 4 'Dynamic' of Table 2 in our paper.

8. "Mean and standard error on GLUE result" (R2) The reported result on GLUE development set is the median of 9 runs. We list the mean and standard error of ConvBERT-small model on GLUE in Table 6.

9. "Performance of base-sized model on SQuAD" (R4) As shown in Table 4 in our paper, ConvBERT achieves similar performance on SQuAD dataset with only $1/4$ training cost compared to ELECTRA.

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	Avg.
ConvBERT-small	81.4±0.2	88.3±0.1	90.3±0.1	67.4±0.8	90.2±0.4	86.7±0.6	59.4±1.4	87.8±0.3	81.4±0.5

Table 6: Results on GLUE dev set.