Simple Local Attentions Remain Competitive for Long-Context Tasks

Anonymous ACL submission

Abstract

Many NLP tasks require processing long contexts beyond the length limit of existing pretrained models. To scale these models to longer text sequences, many efficient long-range attention variants have been recently proposed. Despite the abundance of research along this direction, it is difficult to gauge the relative effectiveness of these models in practical use cases, *e.g.*, if we apply these models following the pretrain-and-finetune paradigm. In this work, we aim to conduct a thorough analysis of these emerging models with large-scale and controlled experiments. For each attention variant, we pretrain large-size models using the same long-doc corpus and then finetune these models for real-world long-context tasks. Our findings reveal pitfalls of a widely-used long-range benchmark and show that the other efficient attentions fails to outperforms the simple local-window attention after standard pretraining. Further analysis on local-attention variants suggests that even the commonly used attention-window overlap is not necessary to achieve good downstream results — using disjoint local attentions, we are able to build a simpler and more efficient long-doc QA model that matches the performance of Longformer (Beltagy et al., 2020) with half of its pretraining compute.

1 Introduction

The quadratic complexity of Transformer architectures make it prohibitive to apply large state-of-the-art pretrained models to full-length documents. To efficiently handle longer text while still maintaining the capacity of attention-based models, a long list of efficient attention variants have been proposed and many claim to effectively capture long-range dependencies. Typical paradigms of these architecture innovations involve *learnable sparse attention patterns* (Kitaev et al., 2020; Tay et al., 2020a; Roy et al., 2021), *fixed local patterns* (Beltagy et al., 2020; Ainslie et al., 2020; Zaheer et al., 2020) and *attention matrix approximation methods* (Wang et al., 2020; Choromanski et al., 2021; Xiong et al., 2021). While most of these studies have reported numbers on long sequence inputs, they tend to adopt quite different benchmarks. For instance, Reformer (Kitaev et al., 2020) is tested on the 64k-chunk enwik8 dataset for unidirectional language modeling; Performer (Choromanski et al., 2021) reports masked language modeling (MLM) perplexity on the PG-19 book corpus and protein sequences; Linformer (Wang et al., 2020) reports MLP perplexity with various input length, while most of documents in their pretrain corpus are short documents.[1] The divergence of evaluation protocols make it hard to compare the relative performance of each attention variant and it is also unknown how they perform well in more practical use cases, which typically involve large-scale pretraining and downstream finetuning.

Another line of work such as Longformer (Beltagy et al., 2020) and ETC (Ainslie et al., 2020) conduct experiments on real-world long-context tasks such as long document QA and summarization. These methods only test fixed local attention patterns, *i.e.*, each token can only attend a small set of nearby tokens. To reduce the pretraining cost, these models are all initialized from the RoBERTa (Liu et al., 2019) checkpoint[2] before further long-doc pretraining. While this paradigm is useful to achieve strong downstream performance, it is not ideal for a fair comparison of all available attention mechanisms, due to the fact that some of the models use different parametrization that is incompatible with the vanilla transformer attention.

A recently proposed benchmark (Tay et al., 2021), named long-range arena (LRA), aims to address the devoid of unified evaluation with a

[1] Short documents are concatenated to form long sequences.

[2] By extending the position embeddings and reusing all other parameters.

bundle of long-sequence tasks. However, the text-related tasks in this benchmark are either automatically generated or artificially lengthened by enforcing byte-level inputs, making them rather synthetic. With a fixed byte-level vocabulary and pre-specified model size, all models are trained from scratch with the same epoch limit on each dataset. While the evaluation protocol is consistent across architectures, this setup still deviates from the common paradigm of applying Transformer models, *i.e.*, standard tokenization like BPE or wordpiece, large-scale pretraining followed and task-specific finetuning (Devlin et al., 2019). Thus, an important question yet to be addressed is whether the results on these artificially datasets are indicative for real-world long-context tasks.

In this work, our goal is to better understand the effectiveness of various attention mechanisms through a systematic study on practical long-context tasks. Instead of only relying on language modeling or synthetic tasks, we test each model under the standard pretraining-and-finetuning paradigm. For a fair comparison, we implement these attentions under a unified framework and test them using the same Transformer architecture[3] used by RoBERTa-large. We pretrain all models using a large corpus that contains mostly long documents and then finetune them on tasks like long-document question answering, full document retrieval and classification. Our experiments show the discrepancies between the commonly used LRA benchmark and downstream results (after pretraining). Additionally, our analysis on the best local attention models allows us to further simplify these models and results in a more efficient long-context encoder. More specifically, the key findings of this paper include:

- With proper tuning, we find that all the tested models can achieve similar level of performance on the LRA benchmark while their performance diverges significantly on large-scale pretraining and downstream tasks;
- In our experiments, the other attention paradigms barely outperform the class of simple local attentions on downstream tasks when using similar pretraining compute;
- As a result of our further analysis on the best performing attention variants, we are able to build a long-doc QA model that is on-par with Longformer while being 2x more efficient.

[3]We only modify the attention calculation within the multi-head attention blocks

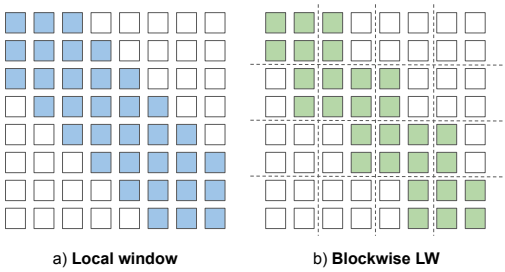

Figure 1: Attention pattern visualization of two types of local attentions: **Left**: Local window attention as in Longformer, with window size 2; **Right**: Blockwise local window attention with block size 2. The rows represent the tokens in the sequence and the columns represent the tokens being attended to.

2 Preliminaries of Tested Attention Variants

We study three classes of efficient attentions:

Fixed local patterns. These methods restrict each token to only attend a local window of tokens. The long-range interactions are achieved by the depth of the model. We consider two variants of these models, the token-wise local window attention (**Local Window**) proposed in Beltagy et al. (2020) where each token attends to the same number of tokens on each side, and a simplified and easy-to-implement blockwise version (**Blockwise LW**) (Zaheer et al., 2020) where each token attends to tokens in the same block and half of the tokens in the left/right blocks. A visualization comparing these two models is shown in Figure 1.

Learnable sparse attention patterns. Instead of relying on the inductive bias of locality, methods like **Reformer** (Kitaev et al., 2020) and **Sinkhorn** Attention (Tay et al., 2020a) allows the model to adaptively select tokens to attend to. Briefly, Reformer uses a learnable hashing function to bucket the sequence and each token only attends to tokens in the same bucket; Sinkhorn uses a learnable sorting function to learn a permutation of the segments and each token will attend to tokens in its own segment and the corresponding segment after permutation.

Kernel-based/Low-rank methods. This class of methods use matrix approximation methods to approximate the full attention function. For sequence length L and the hidden dimension d, **Linformer**

(Wang et al., 2020) simply uses a projection matrix ($L \times k$) to reduce the length of key and value feature matrix, i.e., from $L \times d$ to $k \times d$ ($k \ll L$). **Nyström** (Xiong et al., 2021) attention adopts a classic matrix approximation method which reconstructs the full attention matrix using a sampled sub-matrix. **Performer** (Choromanski et al., 2021) eliminates the need of explicitly calculating the $L \times L$ attention matrix by using a random feature method that can approximate the softmax kernel with only dot-product operations.

Hybrid attention. In additional to these representative methods in each class, our study also includes the more recent **Long-Short** attention (Zhu et al., 2021) which has a similar compression component as in Linformer and combines it with local attentions. Unlike Linformer's compression component that is simply implemented as a standalone projection matrix, **Long-Short** proposes an input-dependent compression layer, which can adaptively reduce the sequence length.

Due to space limit, we refer the readers to the corresponding papers and a recent survey (Tay et al., 2020b) for more thorough model descriptions.

A note on global tokens. For many practical NLP tasks, *e.g.*, classification or entailment, the final layer of the models usually requires a single sequence-level representation as input. For local attention models, it is common practice (Beltagy et al., 2020; Zaheer et al., 2020) to mark a single or a small number of tokens as global tokens and allow these tokens to attend to and be attended by all other tokens. Without incurring much computational cost, these global tokens are important to get better sequence representations and achieve good downstream results. While the mechanism of global tokens has not been used in models with learnable attention patterns, it is straightforward to augment **Reformer** and **Sinkhorn** with global tokens using *gather* operations in standard neural network packages, as their attention scores are still calculated by dot product and softmax operations. Thus, in our experiments, except for the kernel-based/low-rank methods, we augment all other models with global tokens to offset the potential performance gap resulting from this trick.

3 Experiment Setup

We restrict our studies to encoder-only models and leave the analysis of generative models to future work. We begin by implementing a collection of efficient attentions with a unified framework[4], which allow us to plug these models into our pretraining-and-finetuning pipeline in a consistent fashion.

3.1 LRA Experiments

Following recent work on efficient long-range attentions, we take the LRA benchmark as our first set of experiments. As our focus here is on NLP tasks, we consider a subset of LRA tasks with text inputs, *i.e.*, the ListOps, IMDB sentiment analysis and text matching tasks. All tasks are formulated as classification problems: ListOps requires the model to predict the correct output of an expression (10-way classification), sentiment analysis is to predict the positive/negative labels of IMDB reviews and text matching aims to predict citation link between papers. We follow the hyperparameter settings of recent work (Xiong et al., 2021; Zhu et al., 2021). Two-layer Transformer encoders are used across all tasks and enough training updates is allowed to ensure convergence[5]. Note that this is different from the setup proposed in the original LRA benchmark, where different tasks adopt different model sizes. It is observed from recent work that two-layer models with smaller dimensions are sufficient to achieve similar or better results than previously reported results. The final classification layer is added on top of the representations of `[CLS]` tokens which are prepended to each sequence.

3.2 Pretraining and Downstream Tasks

For practical NLP application, large-scale self-supervised training has become an indispensable ingredient to fully unlock the power of Transformer models. In terms of the experiment scale and testing settings, there is a clear gap between LRA's setup and how we apply state-of-the-art Transformer models in practice. For the second set of experiments, we aims to test these models at scale and investigate whether the results on the LRA benchmark are accurate indicators for real-world long-context tasks after standard large-scal pretraining and finetuning.

Pretraining Resource. Following Beltagy et al. (2020), we compile a corpus that contains mostly long documents, including Stories (Trinh and Le,

[4]The code and resources to replicate our study will be released after the review period.

[5]The limit of training updates is arbitrarily set in LRA and various work have reported hugely improved results on the text matching task, simply by running more training steps.

2018), RealNews (Zellers et al., 2019), Books corpus (Zhu et al., 2015) and English Wikipedia. To make the experiments manageable and relevant for standard GPU hardware, we restrict each model's memory usage close to the 16GB threshold when taking 4,096 tokens in each training batch. We control the batch size and training update across all models: we use a batch size of 256 sequences (2^{20} tokens) and pretrain each model using the standard masked language modeling objective for 100k updates. We find that all models' training curves almost stabilize after this amount of training steps. We use 32 A100 GPUs for pretraining and all model runs are finished within around 2 days.

Pretraining Architecture In contrast to Longformer (Beltagy et al., 2020) and Bigbird (Zaheer et al., 2020) where the models are initialized from RoBERTa before pretraining on long documents, we pretrain these models from scratch, as our goal here is to ensure fair comparison and not all architectures can reuse weights from a standard transformer model. In particular, Nyström and Performer do not use the standard dot-product and softmax to compute attention probabilities, making their parameters not compatible with common models like RoBERTa or BERT. Furthermore, other models like Linformer or LongShort introduce additional parameters inside the attention module. In our initial experiments, we observe initializing from the RoBERTa put these models at significant disadvantage compared to other models (e.g., local window attention) that are more compatible with vanilla transformers. Apart from the expanded position embedding matrix and the attention blocks, the architecture hyperparameters are consistent with RoBERTa-large. For both LRA and the large-scale experiments, we adopt the pre layer-normalization trick (Xiong et al., 2020) for feedforward and attention blocks. This usually results in better performance in LRA and turns out to be essential for several models in the pretraining experiments.[6] Additional model-specifc architecture settings and models' average memory usage can be found in the Appendix.

Downstream Datasets and Metrics. We consider practical tasks that naturally involve long documents. We test on extractive QA over long documents, long document classification and document retrieval. For the first two tasks, we use TriviaQA and Hyperpartisan classification respectively, both of which have been used in existing long Transformer work (Beltagy et al., 2020). For full document retrieval, we construct the dataset based on recent open-domain QA work (Lee et al., 2019) that uses passage-level retrievers. We take an existing passage corpus from Karpukhin et al. (2020) and reconstruct the document-level corpus. We consider a document to be positive if it includes the answer passage. We reported token-level answer exact match and F1 for extractive QA and the classification accuracy for Hyperpartisan. For the retrieval task, for the ease of experiments, we reported the mean reciprocal rank on the dev set[7], which has been shown to correlate well with final retrieval metric like answer recall (Oguz et al., 2021). We conduct grid search for all tasks and report the best dev results. Given the small size of the Hyperpartisan dataset, we reported averaged results from 4 random seeds.

Task-specific Architectures for Finetuning. We use standard architectures for the finetuning tasks: for extractive QA, a single-layer MLP span predictor is added on top the output token representations; the classification task uses a binary MLP classifier that takes the `[CLS]` vector as input. For retrieval, we share the query and document encoder using our pretrained models and use dot product of the `[CLS]` vectors as the similarity score. For models that are compatible with global tokens, we use all the question tokens as global tokens in the QA task and use a single global token at the start of the sequences for both classification and retrieval. Except for the Hyperpartisan dataset, the document lengths of the other two datasets usually exceed 4,096 tokens after tokenization. In these cases, we drop the tokens outside the models' position range. We put further implementation details and each task's length statistics in the Appendix.

4 Results and Analysis

4.1 Models Perform Similarly in LRA

We report our reimplemented LRA results in Table 1. While previous work (Tay et al., 2021) have shown clear performance gap between different models, we find that with proper tuning, the results

[6] Linformer and Performer cannot reach reasonable perplexity without pre-layer normalization.

[7] For each question, the ground-truth document will be ranked with all documents (both positive and negative) corresponding to the dev-set questions.

Model	ListOps	Text	Matching	Avg Acc	GFlops
Learnable attention pattern					
Sinkhorn	37.6	63.8	80.4	60.6	0.289
LSH	37.9	62.5	80.5	60.3	0.273
Low-rank/kernel-based approximation					
Linformer	37.7	61.9	78.4	59.3	0.271
Nystrom	37.9	66.1	81.0	61.7	0.256
Performer	37.1	66.1	79.8	61.0	0.205
Hybrid attention					
Long-Short	37.7	65.7	81.6	61.7	0.199
Fixed attention pattern					
Local Window	37.4	65.7	81.6	61.6	0.153
Blockwise LW	37.4	65.6	81.3	61.4	0.146

Table 1: LRA results with our reimplementation.

Model	MLM Train Perplexity
Linformer	4.31
Performer	6.36
Blockwise LW	4.04

Table 2: Training perplexity of our best fixed local attention and other faster attention variants. Each model uses similar GPU memory **and** training time.

of several models could be significantly improved, (*e.g.*, Sinkhorn, Linformer, Reformer, Performer) and *there is no significant performance gap between any of the models* when using similar level of compute (measure by FLOPS). It is worth noting that these improved results are not obtained by increasing the complexity of models (*e.g.*, by using larger bucket size in Sinkhorn), as our implementation either use similar or smaller size models compared to existing work. Also note that while the single global token we added to Sinkhorn and LSH might be essential for some performance gains, it only brings trivial computation overhead.

4.2 Pretraining and Downstream Tasks

We now evaluate these models on practical benchmarks that involve real-world long documents. As shown in Table 3, after we scale up the experiments and control the memory consumption of each models, we see more clear differences between these models than what we observe in LRA. Clearly, fixed local attentions remain to be strong baselines. However, in contrast to LRA, we observe local attentions are significantly better than the other attention variants, for both pretraining perplexity and downstream task results. The only exception in terms of the pretraining perplexity is the hybrid Long-Short attention, which already integrates a local attention component: it achieves better perplexity than fixed local attentions, but the downstream results are at most on par with much simpler models like Blockwise LW. It is worth noting that while we only control the training updates and memory usage in Table 3, the conclusion still holds if we control the training time of each model: We compare the training perplexity of Blockwise LW attention and other faster models with fixed training time in Table 2.

Even though our LRA experiments also study tasks with text inputs, we see clear discrepancies between the two sets of experiments. Apart from models with fixed local attention patterns, improvements on these text LRA tasks often do not transfer to the standard scaled pretraining-finetuning experiments. For instance, while Performer can outperform most of the non-local attention methods on LRA, it performs poorly on both large-scale MLM and downstream long-context tasks. Similarly, while Nyström is significant better than LSH in LRA on average, we observe the opposite trend in Table 3. Among the three tasks, only ListOps is loosely aligned with the MLM perplexity. However, the gaps between each models on this task are still too narrow to be indicative.

Given that large-scale pretraining has become the gold-standard paradigm to build state-of-the-art NLP models. Our findings here call for more

careful and reliable evaluation of lots of existing and emerging long-range attentions. On the other hand, our results also reveal that the local context might still be highly essential even in long context tasks. In the following section, we conduct further analysis on local attention models and attempt to identify the key ingredients of building strong NLP models for downstream long-context tasks.

4.3 Analysis on Local Attentions

As we have seen in §4.2, models that compute exact attention for local contexts around each token achieve better results. Moreover, the Blockwise LW variant performs the best even it does not guarantee a balanced left and right context window for each token. Given these intriguing findings, we aim to investigate the follow questions: *How effective are the long-range mechanism in local attention models?* and *Whether the studied long-context tasks still mostly rely on locality bias?*

Ablation Study. In the Blockwise LW model, there are two mechanisms that enable long-range connections: the global tokens and the attention window overlap, *i.e.*, each token will additionally attend to half the tokens in the neighboring blocks and the receptive field increases with model depth. While both are adopted as common practice in existing work (Zaheer et al., 2020; Beltagy et al., 2020), we study the isolated effect of each component in both pretraining and finetuning experiments. For the non-overlap variant, we increase the block size by a factor of 2 such that the amount of tokens each token attends to remains the same. We show the results in Table 4. Surprisingly, we see different stories in terms of MLM pretraining and downstream tasks. While both mechanisms are useful for achieving lower MLM perplexity, only the global-token mechanism seems important for downstream tasks. Note that in the document retrieval tasks, removing both mechanisms results in slightly better performance. Now the model is only able to use the first block of whole document for retrieval. While this seems to suggest that this task is highly local and involves strong positional bias[8], the gap might be too trivial to be conclusive. Additionally, we only use a single global token for this task, it is likely that assigning more global tokens, *e.g.*, at passage boundaries, could bring additional improvements. Investigating the particular task further is beyond scope of this work. In terms of the effect of attention-window overlap, it is expected that this scheme is crucial for lower perplexity: it not only enables more distant dependencies but also reduces the number of "boundary tokens" which can only attend to one side of the context. However, it is counter-intuitive that the overlapping attention links between neighboring blocks, which adds more long-range information, results in worse downstream performance. Also note that this observation is consistent for all the task we studied. There are two possible implications of this finding: 1) the tested tasks still highly depend on locality bias, *i.e.*, most of important information can be captured solely from the local bias or 2) the overlapping scheme is not effective at capturing the long-range dependency in downstream tasks. To confirm either hypothesis, we conduct another set of experiments with models that have access to different sizes of context.

On Locality Bias. We take the non-overlapping variant and experiment with various block sizes to see whether longer context is actually important to studied tasks. We show the results in Table 5 and the pretraining curves in Figure 2. While the long-range connections brought by the attention overlaps is not helpful for downstream results, we see that increasing the local block sizes does consistently improve both pretraining and downstream performance although the improvement becomes modest beyond block size 256. It is also interesting that the models with smaller blocksizes converge faster at the early stage of pretraining. This suggests a staged pretraining process might be more efficient than directly training from long sequences, which aligns with Press et al. (2021)'s finding on unidirectional LMs. Overall, this set of experiments suggest that increasing model's capabilities to capture longer context is generally helpful for both pretraining and downstream tasks. However, using overlapping attention windows is not an effective way to make use of more context. Thus, we hypothesize the MLM perplexity improvements of overlapping local attentions might mainly come from the reduction of the "boundary" tokens instead of the abilities to capture long-range dependencies. For downstream tasks, the issue of "boundary" tokens is not that essential and the introduction of the overlapping attention windows might disrupt the effective modeling of local context, as the attention module needs to extract both local and distant

[8]The answer context appears in the beginning of the Wikipedia page.

Models	MLM Pretraining PPL ↓	MLM Pretraining k word/sec ↑	Downstream Tasks TriviaQA	Downstream Tasks Doc Retrieval	Downstream Tasks Hyperpartisan
Learnable attention pattern					
Sinkhorn	4.03	11.8	63.3/68.5	80.9	95.0
LSH	3.63	10.0	62.9/67.5	83.6	92.2
Low-rank/kernel-based approximation					
Linformer	4.14	24.6	59.8/65.2	80.3	88.7
Nystrom	3.79	9.5	51.5/57.3	83.1	89.5
Performer	5.58	17.2	24.5/31.9	66.8	94.9
Hybrid attention					
Long-Short	**3.36**	8.4	66.5/71.4	84.5	91.5
Fixed local attention pattern					
Sliding Window	3.47	9.2	65.6/70.7	83.2	**95.3**
Blockwise LW	3.39	13.5	**68.1/72.9**	**85.0**	95.0

Table 3: MLM pretraining and downstream task results.

Model	MLM PPL	TriviaQA	NQ Doc Retrieval	Hyperpartisan
Blockwise LW	**3.39**	68.1/72.9	85.0	95.0
- w/o overlap	3.52	**68.4/73.2**	**86.3**	**96.5**
- w/o overlap & global tokens	3.54	56.5/61.0	85.4	94.6

Table 4: Ablation of the Blockwise LW Model.

Blocksize	Val PPL	TriviaQA Ans F1
64	4.16	68.9
128	3.74	70.7
256	3.52	73.2
512	3.39	73.5

Table 5: Pretraining and long-doc QA results of the non-overlapping blockwise attention.

Blocksize	Speed ↑	Ans EM/F1
Longformer (64k)	6.6k	73.1/77.8
Blockwise LW w/o overlap (64k)	14.8k	**73.2/77.9**

Table 6: Comparing with Longformer with TriviaQA when initializing the models from RoBERTa. Speed is measure by thousand word per second at pretraining.

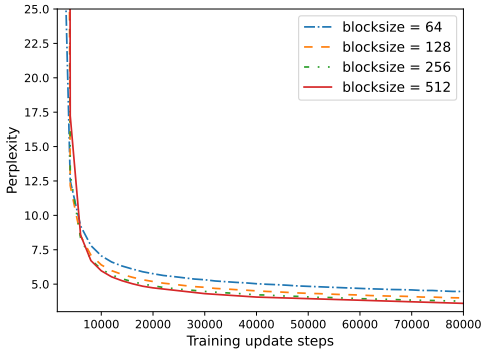

Figure 2: Pretraining curves of the non-overlapping block attentions with various context windows.

information from the same set of tokens.[9]

Initializing from Existing Short Models. While we train all models from scratch for the sake of fair comparison, existing state-of-the-art long context models like Longformer (Beltagy et al., 2020) or BigBird (Zaheer et al., 2020) usually initialize their longer models from an extensively pretrained short model like RoBERTa (Liu et al., 2019). With simple techniques like positional embedding copying, a strong long-context encoder can be initialized without the need of pretraining from scratch. To test our findings from the above analysis in this setting, we follow the same scheme but use the non-overlapping block attention as discussed in §4.3. We compare this model with Longformer (based on Sliding Window attention) as it uses the same long-doc corpus and pretrain-and-finetune pipeline (*e.g.,* packages and

[9] As the depth of the model increase, the tokens' representation will be added information of more distant tokens.

downstream data processing) as our experiments.[10] Same as our setting in §4.2, here we control the batch size and number of training updates: we use a batch size of 64 and train the model for 64k steps. Note that as we drop the attention window overlaps, the model is 2x more efficient than Longformer: Given the same window/block size B and sequence length L, the complexity of the non-overlapping block attention is $L \times B$ compared to Longformer's $2L \times B$. We show the TriviaQA results in Table 6, where the speed is measured by words per second during pretraining. With only half of the pretraining compute, our model with disjoint attention blocks achieves slightly better performance than Longformer. This confirms that our findings about the attention overlap from above section is still valid when the models are not trained from scratch.

5 Related Work

Long-Range Context in Language Models. Various studies have investigated the effective usage of distant context in unidirectional language models. Khandelwal et al. (2018) look into the context usage of LSTM LMs and find that these models are only capable to make full use of the nearby 50 tokens and the longer range context is only roughly captured, *i.e.*, excluding detailed information such as word orders. Similarly, O'Connor and Andreas (2021) study the mid- and long-range context usage in transformer LMs, by manipulating the ordering and lexical information in the text. Their experiments show that while long-range context is usually helpful, most of the usable information is carried by local ordering statistics and non-function words instead of detailed content like sentence orders. These observations provide a possible explanation of our ablation experiments in §4.3 that adding overlaps to attention windows does not yield better downstream results, despite allowing the capture of more long-range interaction. Press et al. (2021) observe diminishing returns as they increase the context length when using sliding windows at inference time. They propose a staged training paradigm which train LMs from smaller context to longer ones. This paradigm can more efficiently use the training compute and achieves lower perplexity compared to directly training with long sequences. Given that models with smaller attention windows converge faster at early training steps (Figure 2), the staged training might also benefit MLM pretraining but further investigation is required to validate whether it can also bring downstream improvements.

Other Long-Range Architectures. Instead of modifying the attention calculation, other work propose to augment transformers with parametric long-term memories. Transformer-XL (Dai et al., 2019) maintains frozen activations of previous tokens in memory and uses them as additional inputs. To handle the shift of positional information of these activations, it also requires a relative position encoding mechanism which brings additional computation cost. The Compressive Transformer (Rae et al., 2020) takes a similar scheme but propose to use compression modules to account for even further memories. Both methods cannot be directly applied to long-context understanding tasks. Under the scheme of kernel-based methods, Katharopoulos et al. (2020); Peng et al. (2021); Schlag et al. (2021) also attempt to linearize the softmax with kernel methods. The core ideas of these methods are similar to Performer and they only differ in the choice of kernel functions. Outside of the transformer families, a recent work (Lei, 2021) proposes to augment recurrent LMs with minimal attention blocks. It is more efficient while achieving stronger LM perplexity compared to Transformer-XL. However, it is still unknown whether this model scales as well as transformer architectures.

6 Conclusion

We present a systematic study of recent proposed efficient attention variants on real-world long-context NLP tasks. In contrast to existing work, we are the first to test these models with a set of unified and large-scale experiments. Our results reveal the gap between a widely used benchmark and practical downstream tasks after conducting large-scale pretraining. Among all the studied attention methods, we find that the simplest local attentions outperform other complex attention paradigms on downstream tasks. We also show that existing local-attention models can be further simplified by removing the attention-window overlap, resulting in faster model that achieves similar or better results. Importantly, our work calls for more careful and practical evaluation protocols while developing long-context NLP models.

[10] Note that while BigBird has a similar overlapping local attention and outperforms Longformer, it uses a larger corpus, more pretraining compute and different finetune pipelines, making a direct comparison difficult.

References

Joshua Ainslie, Santiago Ontañón, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. ETC: encoding long and structured inputs in transformers. In *EMNLP (1)*, pages 268–284. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. abs/2004.05150.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. 2021. Rethinking attention with performers. In *ICLR*. OpenReview.net.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR.

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 284–294, Melbourne, Australia. Association for Computational Linguistics.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *ICLR*. OpenReview.net.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Tao Lei. 2021. When attention meets fast recurrence: Training language models with reduced compute. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7633–7648, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Joe O'Connor and Jacob Andreas. 2021. What context features can transformer language models use? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 851–864, Online. Association for Computational Linguistics.

Barlas Oguz, Kushal Lakhotia, Anchit Gupta, Patrick S. H. Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen-tau Yih, Sonal Gupta, and Yashar Mehdad. 2021. Domain-matched pre-training tasks for dense retrieval. *arXiv*, abs/2107.13602.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. 2021. Random feature attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ofir Press, Noah A. Smith, and Mike Lewis. 2021. Shortformer: Better language modeling using shorter inputs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5493–5505, Online. Association for Computational Linguistics.

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.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based

sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. 2021. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9355–9366. PMLR.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. 2020a. Sparse sinkhorn attention. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 9438–9447. PMLR.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. Long range arena : A benchmark for efficient transformers. In *ICLR*. OpenReview.net.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020b. Efficient transformers: A survey. *arXiv*, abs/2009.06732.

Trieu H. Trinh and Quoc V. Le. 2018. A simple method for commonsense reasoning. abs/1806.02847.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv*, abs/2006.04768.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533. PMLR.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. 2021. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *AAAI*, pages 14138–14148. AAAI Press.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *NeurIPS*, pages 9051–9062.

Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. 2021. Long-short transformer: Efficient transformers for language and vision. *NeurIPS*, abs/2107.02192.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, pages 19–27. IEEE Computer Society.

A Appendix

Downstream Task	Hyperparameter Grid
TriviaQA	learning rate: *1e-5, 3e-5, 5e-6*; warmup ratio: *0%, 10%* of total steps; random seed: *42, 3, 4321*; batch size: *32*; max epochs: *10*
NQ Doc Retrieval	learning rate: *1e-5, 5e-6, 3e-5*; random seed: *42, 3*; batch size: *8*; max epochs: *10*
Hyperpartisan	learning rate: *1e-5, 3e-5*; random seed: *42, 3, 5, 1992*; batch size: *8*; max epochs: *40*

Table 7: Hyperparamters of downstream finetuning.

TriviaQA Average|$P_{95\%}$	**Hyperpartisan** Average|$P_{95\%}$	**NQ doc Retrieval** Average|$P_{95\%}$
769.8|2,067.0	*3,333.9|11,444.3*	*6,732.9|17,493.4*

Table 8: Document length statistics in the tested downstream datasets.

Downstream Task Details. On TriviaQA, there are usually multiple matched spans in the document, we train the model to maximize the marginalized probability of all matched spans. The prediction head in the classification task is defined as a 2-layer MLP with *tanh* activations. For the retrieval task, we follow existing passage retrieval methods and use in-batch documents as negative retrieval targets. The loss is simply a cross-entropy loss defined over the scores of all documents in the batch. All the models are finetuned using the Adam optimizer with linear decays. We conduct grid search for all the tested models. The hyperparameters for all the three tasks are shown in Table 7. In Table 8, we show the average and the 95% percentile of the document lengths in each dataset. As mentioned in the main text, we drop the tokens exceeding 4,096 tokens.

Pretraining Details. Our pretraining pipeline is implemented with fairseq[11]. We control the memory usage of each model by adjusting model-specifc hyperparameters. The details in shown in Table 9. Due to different model designs, we are not able to exactly control the memory consumption. However, the tested local attentions typical requires less GPU memory than all the other models.

[11] https://fairseq.readthedocs.io/en/latest/

Model	Avg Memory Usage (GB)	Architecture Setting
Sinkhorn	14.2	block size: 128
LSH	18.2	num of hash functions: 4; chunk size: 16
Linformer	17.2	compression ratio: 8
Nystrom	16.3	num of landmarks: 256; convolution kernel size: 35;
Performer	14.2	random feature dimension: 256; kernel function: $relu$
Long-Short	16.3	block size: 128; num of landmarks: 32
Sliding Window	15.3	attention window size: 256
Blockwise LW	15.1	block size: 128; overlap: 64
Blockwise LW w/o global toks	14.7	block size: 128
Blockwise LW w/o overlap	13.4	block size: 256
Blockwise LW w/o overlap & global	13.2	block size: 256

Table 9: Model-specific architecture settings and each model's GPU memory usage when feeding in a single sequence of 4,096 tokens.