LSG Attention: Extrapolation of pretrained Transformers to long sequences

Charles Condevaux, Sébastien Harispe

LSG Attention: Extrapolation of pretrained Transformers to long sequences

Charles Condevaux[1][0000−0002−0819−9056] and
Sébastien Harispe[2][0000−0001−5630−2743]

[1] CHROME, University of Nîmes, France
charles.condevaux@unimes.fr
[2] EuroMov Digital Health in Motion, Univ Montpellier, IMT Mines Ales, France
sebastien.harispe@mines-ales.fr

Abstract. Transformer models achieve state-of-the-art performance on a wide range of NLP tasks. They however suffer from a prohibitive limitation due to the self-attention mechanism, inducing $O(n^2)$ complexity with regard to sequence length. To answer this limitation we introduce the LSG architecture which relies on Local, Sparse and Global attention. We show that LSG attention is fast, efficient and competitive in classification and summarization tasks on long documents. Interestingly, it can also be used to adapt existing pretrained models to efficiently extrapolate to longer sequences without additional training. Along with the introduction of the LSG attention mechanism, we propose a PyPI package to train new models and adapt existing ones based on this mechanism.

Keywords: Attention mechanism · Long sequences · Extrapolation

1 Introduction

Transformer models [33] are nowadays state-of-the-art in numerous domains, and in particular in NLP where they are used in general language models, and to successfully tackle several specific tasks such as document summarization, machine translation and speech processing to cite a few [13,26]. The cornerstone of Transformer models is the Attention mechanism used to iteratively build complex context-dependent representations of sequence elements, e.g. tokens, by dynamically aggregating prior representations of these elements. Using self-attention, a popular Attention flavour, this is made by computing full attention scores defining how each prior element representation will contribute to building the new representation of an element. Considering a sequence of n elements, the computation of the attention scores is therefore of complexity $O(n^2)$ which is prohibitive dealing with long sequences. Since a large number of models based on full attention have been trained on various datasets and tasks, we are therefore interested in extrapolating those models to long sequences by simply, post training, substituting the full attention trained on shorter input sequences by new attention mechanisms adapted to longer sequences. Common pretrained

models (e.g RoBERTa) are indeed known to underperform when extrapolated to sequences of length exceeding the 512 tokens considered during training. This is due to the nature of the attention mechanism which largely impacts extrapolation capabilities: full attention usually fails to extrapolate, even considering post hoc adaptations, e.g. using a relative positional embedding [30] or duplicating the positional embedding [3]. Defining new attention mechanisms that can efficiently substitute full attention in pretrained models that are not originally capable of handling long sequences would avoid the costs induced by training large language models from scratch. The main contributions of this paper are:

1. LSG (Local Sparse Global) attention, an efficient $O(n)$ approach to approximate self-attention for processing long sequences.[3]

2. Results demonstrating that LSG is fast, efficient and competitive on classification and summarization tasks applied to long documents. It is also shown that LSG can adapt and extrapolate existing pretrained models not based on LSG, with minimal to no additional training.

3. A procedure and a PyPI package to convert existing models and checkpoints (e.g. RoBERTa, DistilBERT, BART) to their LSG variant.[4]

Compared to several contributions aiming at reducing the complexity of self-attention introduced hereafter, a specific focus is given in our work on the extrapolation of existing Transformer models, i.e. reuse, to longer sequences.

2 Related works

Several contributions have been devoted to the optimization of the Attention mechanism. Four categories of approaches can be distinguished in the literature: (i) recurrent models such as Transformers-XL [12] and Compressive Transformers [25] which maintain a memory of past activation at each layer to preserve long-range contextual information; (ii) factorization or kernels aiming at compressing attention score matrices, such as Linformer [34] or Performer [9]; (iii) models based on clustering such as Reformer [21] that dynamically define eligible attention patterns (i.e. where attention may be made); and (iv) models based on fixed or adaptative attention patterns, e.g. Longformer [3] or Big Bird [37].

Recurrent approaches iteratively process the sequence by maintaining a memory to enable long-range dependencies. They generally suffer limitations induced by specific, slow, and difficult to implement forward and back propagation procedures. Alternatively, one of the main line of study for reducing the complexity of Attention is thus to perform sparsity by limiting the number of elements on which new representations will be based, i.e. reducing the number of elements with non-null attention scores. This approach is motivated by the observation of global or data-dependent positional patterns of non-null attention scores depending on the task [7]. The sparsity of attention scores in the traditional Attention mechanism is indeed documented in the literature. It has for instance been shown that in practice, full attention tends to overweight close elements

[3] Checkpoints and datasets are available at https://huggingface.co/ccdv

[4] https://github.com/ccdv-ai/convert_checkpoint_to_lsg

in average, in particular for MLM, machine translation, and seq-to-seq tasks in general [10]. Moreover, according to analyses on the use of multi-head full attention on specific tasks, e.g. machine translation, numerous heads learn similar simple patterns [27]. Such redundant patterns may be hardcoded implementing fixed-positional patterns, eventually in a task-dependent manner.

Two main approaches are discussed in the literature for implementing sparsity: fixed or adaptative patterns based on whether attention scores are computed considering (1) predefined fixed elements based on their location in the sequence, or (2) elements selected from a given procedure. As an example, [35] have shown that fixed $O(n)$ convolutions can perform competitively on machine translation. Longformer proposes an alternative $O(n)$ approach based on sliding and global patterns [3]. In the context of image, audio, and text processing, [7] propose sparse Transformer, an $O(n\sqrt{n})$ model based on sparse factorization of the attention matrix relying on specific 2D factorized attention schemes. Those approaches however prevent the use of task-dependent dynamic patterns. Considering adaptative patterns, [35] also introduced dynamic convolutions as an $O(n)$ complexity substitute to self-attention. Kernels defining the importance of context elements are specified at inference time rather than fixed after training. Another example is Reformer [21], an $O(n \log n)$ approach based on locality-sensitive hashing (LSH) based on random projections.

In a transverse manner, several authors, explicitly or implicitly motivated by the compositional nature of language have studied structured approaches in which subsequences (i.e. blocks) are processed independently and then aggregated. This aims at implementing a local or global dynamic memory for considering close to long-range dependencies. Some approaches use a blockwise approach to reduce the quadratic complexity induced by large sequences in encoder-decoder architectures [4]. Other propose a chunkwise attention in which attention is performed in a blockwise manner adaptively splitting the sequence into small chunks over which soft attention is computed [8]. This idea is also used in Transformer-XL [12]. Masked block self-attention mechanism in which the entire sequence is divided into blocks, to further 1) apply self-attention intra-block for modeling local contexts, to further 2) apply self-attention inter-block for capturing long-range dependencies, as also been proposed [31]. Such an approach enables implementing some forms of connectivity between all positions over several steps without being restricted by full attention limitations. This can also be achieved by factorization techniques, e.g. [7]. More recently authors have proposed global attention mechanisms encoding information related to blocks on which attention is based [1,39,16].

This paper presents LSG (Local, Sparse and Global) attention based on block local attention to capture local context, sparse attention to capture extended context, and global attention to improve information flow. Contrary to prior work mostly focusing on defining new models, the proposed LSG Attention mechanism is model agnostic and aims to facilitate adapting existing (pretrained) models for them to be used on long sequences.

3 LSG: mixing Local, Sparse and Global attentions

LSG assumes (1) that locally, a token needs to capture precise low level information using dense attention, (2) as the context grows, higher level information is sufficient, i.e. a limited number of tokens specifically selected are sufficient. LSG therefore relies on block local attention to capture local context, sparse attention to capture extended context, and global attention to improve information flow.

Local Attention. LSG takes advantage of a block-based processing of the input. The sequence is split into n_b non-overlapping chunks of size b_t. For a given block, each token attends to the tokens inside the block, as well as to those in the previous and next blocks. The local attention window is asymmetrical since a token can connect up to $2 \times b_t - 1$ tokens on the left or on the right.

Sparse Attention. Sparse connections are used to expand the local context by selecting additional tokens. These tokens can be directly selected based on a specific metric or using some computation such as a pooling method. In the proposed approach, each attention head can process different sparse tokens independently. Sparse attention also relies on a block structure where the sparse selection is done inside each block. Five alternative criteria can be used in LSG.

1. *Head-wise strided:* Each attention head attend to a set of tokens following a specific stride defined as the sparsify factor. Figure 1 shows the selection pattern.

2. *Head-wise block strided* selects consecutive tokens, see Figure 2.

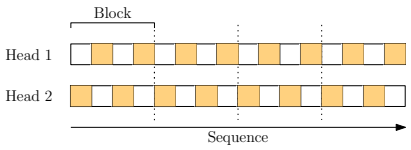

Fig. 1. Head-wise selection (stride 2).

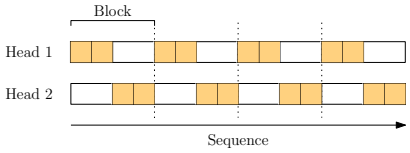

Fig. 2. Block selection (stride 2).

3. *Average pooling:* sparse tokens are computed using average pooling on blocks. For a block of size b_t and a sparsify factor f, pooling is applied to each block with a window of f and a stride of f to produce b_t/f tokens.

4. *Max norm*: selects tokens that are most likely to have high scores. Finding those keys efficiently is difficult in practice so we use a simple and deterministic heuristic selecting inside each block and each head b_t/f tokens with the highest key norm. Indeed, note that for a query and a key $\boldsymbol{q}, \boldsymbol{k} \in \mathbb{R}^d$, $\boldsymbol{q}\boldsymbol{k}^\top = \cos(\theta)\|\boldsymbol{q}\|\|\boldsymbol{k}\|$. If $cos(\theta)$ is positive and $\|\boldsymbol{k}\|$ is high, the key will likely dominate the softmax regardless of the query.

5. *LSH Clustering:* non deterministic approach relying on the LSH algorithm [2]. For each block, b_t/f clusters are built using a single round LSH. To get $c = b_t/f$ hashes and for an input $\boldsymbol{x} \in \mathbb{R}^d$, a random matrix $\boldsymbol{R} \in \mathbb{R}^{d \times c/2}$ is generated, such

that $h(\boldsymbol{x}) = \arg\max([\boldsymbol{x}\boldsymbol{R}; -\boldsymbol{x}\boldsymbol{R}])$ with $[\boldsymbol{a}; \boldsymbol{b}]$ the concatenation of two vectors. Using the key matrix as input, each token inside the block gets a cluster index from $h(\boldsymbol{x})$. Tokens inside a cluster are averaged.

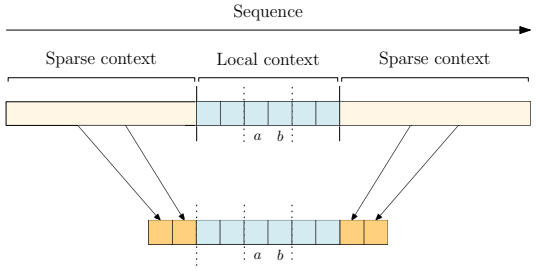

Fig. 3. Local and sparse contexts with a block size of 2 and a sparsity factor of 4. Queries a and b will attend to 6 local keys and 4 sparse keys.

Global Attention. Global tokens improve the flow of information inside the model. They attend to every tokens across the sequence and all tokens attend to them. Rather than picking a subset of tokens, additional tokens are prepended to the sequence and trained using their own embedding matrix (their number is an hyperparameter). When a model is converted to its LSG version, the first global token is initialized as the sum of the [CLS] token and the first position from the positional embedding. The other global tokens are initialized as the sum of [MASK] token and the other positions from the positional embedding. Thus, they can be trained and fine-tuned independently.

Positional Embedding. It is necessary to modify the positional embedding matrix to reuse existing models to process long sequences. In LSG, instead of randomly initializing the new positions, the original matrix is duplicated and concatenated until the desired max sequence length is reached.

4 Experiments

We evaluate LSG in the context of model extrapolation by replacing full attention by LSG attention in various architectures. The official RoBERTa-base checkpoint for classification tasks and BART-base checkpoint for summarization tasks are extrapolated using LSG attention. All metrics are reported for the test set except in the case where only the validation set is available – datasets are all available on the HuggingFace hub. We use a batch size of 32, a linear decaying learning rate, a dropout rate of 0.10 and Adam (0.9, 0.999) optimizer [20] for classification and summarization experiments. An experiment comparing several attention approximations to extrapolate RoBERTa in an MLM task is first discussed; it is used to limit the number of tested alternatives, and therefore reduce the cost of the proposed evaluations. All experiments are conducted on NVIDIA Quadro RTX 8000 48Gb GPUs.

4.1 RoBERTa extrapolation on MLM

A test on a MLM task is performed to question the ability of an attention mechanism to extrapolate a model to longer sequences without additional training. A RoBERTa-base model is here considered and two experiments are conducted. First, the full attention is substituted by different kinds of attention (kernel, factorization, local, fixed pattern) and each model is evaluated on sequences of the same length as those considered during RoBERTa initial training (512 tokens). In the second experiment, their ability to extrapolate to 4096 tokens sequences without additional training is tested (positional embedding duplicated 8 times).

A random sample from Wikipedia + BookCorpus + CC_News is used; BPC and MLM accuracy are in Table 1. RoBERTa's author report a 1.880 BPC loss; we obtain a comparable loss of 1.881 on this random sample.

Only Longformer, Big Bird and LSG obtain competing BPC while processing sequences of the same length as those considered during the original RoBERTa training. Other approaches such as Linformer, Performer or Reformer require additional MLM fine-tuning to leverage an existing checkpoint. It can be seen that RoBERTa fails to extrapolate to longer sequences (+2.454 BPC), which highlights that full attention is not suitable for extrapolation. Longformer and Big Bird are able to perform some form of extrapolation. Therefore, we restrict our comparison to these two approaches in order to limit experimentation costs.

Attention	**512 length**		**4,096 length**	
	BPC	Accuracy	BPC	Accuracy
RoBERTa (full) [23]	1.881	**0.732**	4.335	0.359
Linear Attn. [19]	11.324	0.061	11.474	0.058
Efficient Attn. [32]	21.022	0.102	20.574	0.097
Performer [9]	10.382	0.107	10.556	0.102
Linformer (128 proj.) [34]	22.176	0.098	20.386	0.032
Reformer [21]	17.602	0.003	18.608	0.002
Longformer (512) [3]	1.929	0.726	2.051	0.708
Big Bird (64) [37]	1.881	**0.732**	2.439	0.659
LSG-Norm (128/2) **(block size / sparsity)**	1.919	0.727	2.032	**0.712**
LSG-Stride (128/2)	1.938	0.724	2.046	0.710
LSG-BlockStride (128/2)	1.940	0.724	2.048	0.709
LSG-Pooling (128/2)	1.968	0.720	2.064	0.706
LSG-LSH (128/2)	1.969	0.719	2.065	0.705

Table 1. BPC and MLM accuracy of RoBERTa-base with various Attention.

4.2 Classification Tasks

We compare LSG to Longformer [3] and Big Bird [37], two approaches able to process long sequences with a similar number of parameters. Tests are performed considering sparse attentions with a block size of 128 and a sparsify factor of 4.

Datasets. Standard NLP datasets are used. *IMDb* [24]: binary sentiment analysis classification task from movie reviews. *ArXiv* [17]: set of documents from ArXiv where the objective is to predict a topic from 11 available classes. Because there is no official split, a random one is made of 28K, 2.5K and 2.5K documents for train, validation and test. *Patent* [29]: subset of the Big Patent summarization dataset. The task is redefined as a classification task where the objective is to predict the patent category using the full document (9 classes, random split of 25K, 5K and 5K documents for train, validation and test). Some specific domains are highly dependent on processing long sequences, e.g. legal domain in which sentences tend to be long and complex. To demonstrate the ability of LSG to leverage pretrained models in such cases, the following three datasets are chosen from LexGlue [6], a benchmark focused on legal documents. Tasks where the input is on average significantly longer than 512 tokens have been selected. *Scotus*: given a court opinion, the task is to predict the relevant issue area among 14 choices. *ECtHRa* and *ECtHRb*: the objective is to predict which articles of the European Court of Human Rights (ECHR) have been violated (if any) from case description: multi-label task (10 + 1 labels).

Training setup and architecture. To make a fair comparison between models and architectures, fine-tuning is done with the same learning rate, number of steps and batch size. To show that LSG is compatible with different architectures, the LexGlue tasks are also run with an LSG version of LEGAL-BERT [5].

Results. Micro and Macro F-1 (Table 2) show that LSG outperforms most of the time Longformer and Big Bird models with input sequences up to 4096 tokens long. A major difference lies in the implementation itself since the LSG variant is twice as fast to train on these lengths with no additional memory cost.[5]

	IMDb	**Arxiv**	**Patent**	**Scotus**	**ECtHRa**	**ECtHRb**
Epochs	3	3	3	7	5	5
Learning rate	2e-5	5e-5	2e-5	1e-4	1e-4	1e-4
RoBERTa (512-length)	95.5	87.2/86.8	66.6/61.8	69.4/60.8	62.9/58.2	72.0/65.9
Longformer	95.9	**88.2/87.9**	69.8/63.8	72.9/62.6	68.3/59.7	78.9/72.2
Big Bird ETC	95.4	85.9/85.5	69.4/63.9	69.4/58.2	68.3/60.3	**80.0/70.6**
LSG-Local (256/0)	**96.0**	87.5/87.1	69.9/64.8	**73.3/63.7**	68.8/63.7	79.9/73.4
LSG-Stride (128/4)	95.6	**88.2/87.9**	69.2/64.0	70.5/60.0	69.5/62.3	79.3/71.6
LSG-BlockStride (128/4)	95.7	87.7/87.4	69.6/64.1	72.5/63.1	69.1/58.6	79.5/71.8
LSG-Norm (128/4)	95.7	87.0/86.6	**70.0/64.4**	71.3/60.8	70.1/61.9	79.4/72.1
LSG-Pooling (128/4)	95.9	87.5/87.3	69.4/64.1	72.6/60.9	70.2/61.4	79.0/73.1
LSG-LSH (128/4)	95.8	**88.2/87.9**	69.5/64.2	70.3/54.6	**71.0/60.3**	78.9/71.0
Legal-BERT (512-length)	-	-	-	73.5/60.5	64.2/58.2	73.2/65.9
LSG-Legal-BERT (256/0)	-	-	-	74.5/62.6	71.7/63.9	81.0/75.1

Table 2. Micro/Macro F-1 on classification datasets.

[5] See https://github.com/ccdv-ai/convert_checkpoint_to_lsg for a benchmark.

On Patent, ECtHRa and ECtHRb tasks, the ability to process longer sequences improves significantly the F-measures compared to a vanilla (full attention) RoBERTa model. We also observe that Big Bird model is in general slightly under its counterpart except for the ECtHRb dataset. This probably comes from the random attention mechanism which may require additional training steps. LSG-LSH and Big Bird models are affected by randomness during inference, thus their performance can differ between runs.

Extrapolating LEGAL-BERT with LSG to handle longer sequences improves predictions.The choice of the sparse attention is likely task specific. Using local attention only with a large block size is also a viable option. The role of global tokens is not discussed here since we only use one for all experiments. We show in the next section with summarization tasks the utility of such tokens.

4.3 Summarization Tasks

We evaluate our models on summarization tasks where the input is significantly longer than 1k tokens only. The models have been fine-tuned on each dataset.[6]

Datasets. In both *ArXiv and Pubmed* [11], the goal is to generate an abstract using a document as input. *MultiNews* [14] involves generating human-written summaries from sets of news documents. *MediaSum* [40] consists of using interview transcripts from CNN and NPR media to generate a summary.

Models	Params.
PRIMERA [36]	447M
LED [3]	460M
HAT-BART [28]	471M
Pegasus [38]	577M
Big Bird-Peg. [37]	577M
Hepos [18]	406M
LongT5-Base [15]	220M
LongT5-L	770M
LongT5-XL	3B
Ours, LSG-BART-base (256/0)	145M

Table 3. Parameters count of summarization models.

Training setup and architecture. The BART-base model [22] is converted to its LSG version by replacing the full attention in the encoder part and adding global tokens. The model is then fine-tuned on 4096-length inputs and evaluated. To reduce computational costs, experiments on 16384-length inputs are warm started from the 4096-length experiments.The model is then fine-tuned during a single epoch if necessary using the same training parameters. We propose 3

[6] All summarization experiments are run using a 8e-5 learning rate, a 10% warmup, a length penalty of 2.0 and a beam size of 5 for beam search.

setups for the 16384-length. First we evaluate the model with pure extrapolation from 4096-length (no additional training). In the second setup, we extrapolate by adding 64 global tokens we choose to fine-tune. In the last setup, we extrapolate by adding 64 global tokens and by fine-tuning the full model. Extrapolation is done by concatenating 4 copies of the positional embedding matrix (4×4096).

The tested model - LSG-BART-base - is significantly smaller than common models from the existing literature (Table 3). An input sequence of 16384 tokens can fit on a 32Gb GPU (without attention dropout) during training without a specific memory reduction tool (i.e gradient checkpointing).

Results. LSG-BART is compared to state-of-the-art models by reporting the results from their respective papers. We use ROUGE-1, ROUGE-2 and ROUGE-L evaluation metrics as comparison points.

Models	R1	R2	RL
Pegasus (1K)	45.49	19.90	27.69
Big Bird-Peg. (4K)	46.32	20.65	42.33
HAT-BART (4K)	48.36	21.43	37.00
Hepos-LSH (7.2K)	48.12	21.06	42.72
Hepos-SKN (10.2K)	47.93	20.74	42.58
LongT5-Base (4K)	47.77	22.58	44.38
LongT5-L (16K)	49.98	24.69	46.46
LongT5-XL (16K)	50.23	24.76	46.67
Ours (4K)	47.37	21.74	43.67
Ours (16K)	48.03	22.42	44.32
+ global tuning	48.12	20.46	44.40
+ full tuning	48.32	22.52	44.57

Table 4. ROUGE on PubMed dataset.

Models	R1	R2	RL
TG-MultiSum	47.10	17.55	20.73
PRIMERA (4K)	49.90	21.10	25.9
LongT5-Base (4K)	46.01	17.37	23.50
LongT5-L (4K)	46.99	18.21	24.08
LongT5-L (8K)	47.18	18.44	24.18
LongT5-XL (8K)	48.17	19.43	24.90
Ours (4K)	47.10	18.94	25.22
Ours (16K)	47.30	19.19	25.38
+ global tuning	47.23	19.18	25.29
+ full tuning	47.07	19.04	25.35

Table 5. ROUGE on MultiNews.

Models	R1	R2	RL
Pegasus (1K)	44.70	17.27	25.80
Big Bird-Peg. (4K)	46.63	19.02	41.77
LED (16K)	46.63	19.62	41.83
PRIMERA (4K)	47.58	20.75	42.57
HAT-BART (4K)	46.68	19.07	42.17
Hepos-LSH (7.2K)	48.24	20.26	41.78
Hepos-SKN (10.2K)	47.87	20.00	41.50
LongT5-Base (4K)	44.87	18.54	40.97
LongT5-L (16K)	48.28	21.63	44.11
LongT5-XL (16K)	48.35	21.92	44.27
Ours (4K)	46.65	18.91	42.18
Ours (16K)	47.03	20.19	42.69
+ global tuning	48.08	20.42	43.65
+ full tuning	48.74	20.88	44.23

Table 6. ROUGE on ArXiv dataset.

Models	R1	R2	RL
BART-Large (1K)	35.09	18.05	31.44
T5-large (1K)	30.68	14.88	27.88
LongT5-Base (4K)	35.09	18.35	31.87
LongT5-L (4K)	35.54	19.04	32.20
LongT5-XL (4K)	36.15	19.66	32.80
Ours (4K)	35.16	18.13	32.20
Ours (16K)	35.17	18.13	32.21
+ global tuning	35.22	18.08	32.22
+ full tuning	35.31	18.35	32.47

Table 7. ROUGE on MediaSum.

As shown in Tables 4, 5, 6 and 7, LSG achieves very competitive results by enabling adapting existing pretrained models to longer sequences. On the ArXiv dataset (Table 6), LSG is competitive with every size of the LongT5 model, despite the limited number of model parameters. On the PubMed dataset

(Table 4), LSG also outperforms Pegasus and Big Bird Pegasus, and is close to Hepos models. On the MultiNews dataset (Table 5), LSG is close to the large L and XL LongT5 models. We note that while extrapolation improves metrics, additional fine-tuning has a negative impact in this case. Since this dataset is rather small (45K examples, ∼1.4k steps), fine-tuning a single epoch is not enough for the model to converge properly; longer training is required. On the MediaSum dataset (Table 7), LSG is close to the LongT5-base model again. This dataset has the shortest inputs, thus processing a maximum of 16384 tokens has a marginal impact on performances. These results underline the ability of LSG to efficiently substitute full-attention mechanisms to process long sequences.

The second surprising and important finding is the ability of LSG to improve metrics from 4096 to 16384-length inputs without additional fine-tuning. This is especially true on ArXiv and PubMed datasets which have the longest input sequences. Fine tuning additional global tokens further improves metrics while limiting cost and training time compared to a fully tuned model.

5 Conclusion

We have presented LSG attention, a novel efficient $O(n)$ alternative to the full attention mechanism relying on local, sparse and global attentions. Our results on MLM, classification and summarization tasks show that LSG is a fast and very competitive full attention substitute for pretrained Transformers to efficiently extrapolate to long input sequences. We also proposed an optimized implementation of the LSG attention mechanism on HuggingFace, improving training speed by a factor of 2 without additional memory cost compared to Longformer and Big Bird models. By providing a PyPI package conversion tool to leverage existing models and checkpoints (BERT, RoBERTa, DistilBERT, BART), the proposed approach removes the need of a costly re-training of existing models to handle long sequences.[7]

References

1. Ainslie, J., Ontanon, S., Alberti, C., Cvicek, V., Fisher, Z., Pham, P., Ravula, A., Sanghai, S., Wang, Q., Yang, L.: Etc: Encoding long and structured inputs in transformers. arXiv preprint arXiv:2004.08483 (2020)
2. Andoni, A., Indyk, P., Laarhoven, T., Razenshteyn, I.P., Schmidt, L.: Practical and optimal LSH for angular distance. CoRR **abs/1509.02897** (2015)
3. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer. arXiv:2004.05150 (2020)
4. Britz, D., Guan, M.Y., Luong, M.T.: Efficient attention using a fixed-size memory representation. arXiv preprint arXiv:1707.00110 (2017)
5. Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., Androutsopoulos, I.: LEGAL-BERT: The muppets straight out of law school. In: Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 2898–2904 (Nov 2020)

[7] This work has benefited from LAWBOT (ANR-20-CE38-0013) grant and HPC resources from GENCI-IDRIS (Grant 2023-AD011011309R3).

6. Chalkidis, I., Jana, A., Hartung, D., Bommarito, M., Androutsopoulos, I., Katz, D.M., Aletras, N.: Lexglue: A benchmark dataset for legal language understanding in english. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. Dubln, Ireland (2022)
7. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 (2019)
8. Chiu, C.C., Raffel, C.: Monotonic chunkwise attention. arXiv preprint arXiv:1712.05382 (2017)
9. Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., Belanger, D., Colwell, L., Weller, A.: Rethinking attention with performers. arXiv:2009.14794 (2021)
10. Clark, K., Khandelwal, U., Levy, O., Manning, C.D.: What does bert look at? an analysis of bert's attention. arXiv preprint arXiv:1906.04341 (2019)
11. Cohan, A., Dernoncourt, F., Kim, D.S., Bui, T., Kim, S., Chang, W., Goharian, N.: A discourse-aware attention model for abstractive summarization of long documents. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers) (2018)
12. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q.V., Salakhutdinov, R.: Transformer-xl: Attentive language models beyond a fixed-length context. arXiv preprint arXiv:1901.02860 (2019)
13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
14. Fabbri, A.R., Li, I., She, T., Li, S., Radev, D.R.: Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model (2019)
15. Guo, M., Ainslie, J., Uthus, D.C., Ontañón, S., Ni, J., Sung, Y., Yang, Y.: Longt5: Efficient text-to-text transformer for long sequences. CoRR **abs/2112.07916** (2021)
16. Guo, Q., Qiu, X., Liu, P., Shao, Y., Xue, X., Zhang, Z.: Star-transformer. arXiv preprint arXiv:1902.09113 (2019)
17. He, J., Wang, L., Liu, L., Feng, J., Wu, H.: Long document classification from local word glimpses via recurrent attention learning. IEEE Access **7**, 40707–40718 (2019)
18. Huang, L., Cao, S., Parulian, N., Ji, H., Wang, L.: Efficient attentions for long document summarization. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, Online (Jun 2021)
19. Katharopoulos, A., Vyas, A., Pappas, N., Fleuret, F.: Transformers are rnns: Fast autoregressive transformers with linear attention. CoRR **abs/2006.16236** (2020)
20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014)
21. Kitaev, N., Kaiser, L., Levskaya, A.: Reformer: The efficient transformer. CoRR **abs/2001.04451** (2020)
22. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 7871–7880. Association for Computational Linguistics, Online (Jul 2020)
23. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized BERT pretraining approach. CoRR **abs/1907.11692** (2019)

24. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 142–150. Association for Computational Linguistics (Jun 2011)
25. Rae, J.W., Potapenko, A., Jayakumar, S.M., Lillicrap, T.P.: Compressive transformers for long-range sequence modelling. arXiv preprint arXiv:1911.05507 (2019)
26. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
27. Raganato, A., Scherrer, Y., Tiedemann, J.: Fixed encoder self-attention patterns in transformer-based machine translation. arXiv preprint arXiv:2002.10260 (2020)
28. Rohde, T., Wu, X., Liu, Y.: Hierarchical learning for generation with long source sequences. CoRR **abs/2104.07545** (2021)
29. Sharma, E., Li, C., Wang, L.: BIGPATENT: A large-scale dataset for abstractive and coherent summarization. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 2204–2213. Association for Computational Linguistics, Florence, Italy (Jul 2019)
30. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). pp. 464–468. New Orleans, Louisiana (Jun 2018)
31. Shen, T., Zhou, T., Long, G., Jiang, J., Zhang, C.: Bi-directional block self-attention for fast and memory-efficient sequence modeling. arXiv preprint arXiv:1804.00857 (2018)
32. Shen, Z., Zhang, M., Yi, S., Yan, J., Zhao, H.: Factorized attention: Self-attention with linear complexities. CoRR **abs/1812.01243** (2018)
33. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
34. Wang, S., Li, B.Z., Khabsa, M., Fang, H., Ma, H.: Linformer: Self-attention with linear complexity. CoRR **abs/2006.04768** (2020)
35. Wu, F., Fan, A., Baevski, A., Dauphin, Y.N., Auli, M.: Pay less attention with lightweight and dynamic convolutions. arXiv preprint arXiv:1901.10430 (2019)
36. Xiao, W., Beltagy, I., Carenini, G., Cohan, A.: PRIMERA: Pyramid-based masked sentence pre-training for multi-document summarization. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 5245–5263. Dublin, Ireland (May 2022)
37. Zaheer, M., Guruganesh, G., Dubey, K.A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., et al.: Big bird: Transformers for longer sequences. Advances in Neural Information Processing Systems **33** (2020)
38. Zhang, J., Zhao, Y., Saleh, M., Liu, P.J.: Pegasus: Pre-training with extracted gap-sentences for abstractive summarization (2019)
39. Zhang, X., Wei, F., Zhou, M.: Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. arXiv preprint arXiv:1905.06566 (2019)
40. Zhu, C., Liu, Y., Mei, J., Zeng, M.: Mediasum: A large-scale media interview dataset for dialogue summarization. arXiv preprint arXiv:2103.06410 (2021)