A APPENDIX

A.1 PRETRAINING

A.1.1 PRETRAINING DATASET

For ERNIE-SPARSE pretraining, we use Wikipedia (English Wikipedia dump[2]; 12GB), BookCorpus (Zhu et al., 2015) (4.6GB), Realnews (Zellers et al., 2019) (7.4GB) and Stories (Trinh & Le, 2018) (11GB). For pretraining, we also sample 5% training data as the validation set to monitor the training process. Table 8 shows statistics of the pretraining data.

Source	Tokens	Avg doc len
Wikipedia	3.0B	515
BookCorpus	1.2B	23K
Realnews	1.8B	3.0K
Stories	2.7B	8.7K

Table 8: Pretraining data statistics

Parameter	**ERNIE-SPARSE**
α of L_{SAOR}	0
learning rate	$3e{-}5$
batch size	256
weight decay	0.1
warmup steps	10k
total steps	1m
max seq length	4096
embedding dim	768
#head	12
#layer	12
activation layer	gelu
dropout	0.1
attn dropout	0.1

Table 9: Hyperparameters for the ERNIE-SPARSE for Pretraining

A.1.2 PRETRAINING HYPERPARAMETERS

We split any document longer than 4096 into multiple documents and we joined multiple documents that were much smaller than 4096. During the pre-training phase, we only use mask language model for training tasks. Specifically, we mask 15% of tokens in these four datasets, and train ERNIE-SPARSE to predict the mask. We warm start ERNIE-SPARSE from RoBERTa's checkpoint. The hyperparameters for these ERNIE-SPARSE are given in Table 9. We use a learning rate warmup over the first 10,000 steps, and polynomial decay of the learning rate. Notably, attention weight in HST are shared with sparse attention.

A.2 TASKS

To evaluate ERNIE-SPARSE, we chose three benchmarks, including LRA, and text classification, as well as question answering. The latter two need to follow the pretraining and finetuning paradigm. Table 10 lists the data distribution, task type, evaluation metric of each dataset.

A.2.1 HYPERPARAMETERS FOR LRA

Table 11 gives the detail list of hyperparameters used to get results shown in Table 1.

A.2.2 HYPERPARAMETERS FOR CLASSIFICATION AND QA

Table 12 gives the detail list of hyperparameters used to get results shown in Table 2 and Table 4.

Corpus	Task	Split	#Sample	Length in percentile				#Label	Metrics
				50%	90%	95%	max		
Long Range Arena (LRA)									
ListOps	Logical Reasoning	Train	96k	954	1646	1800	1999	10	Acc
		Dev	2k	960	1648	1813	1999		
		Test	2k	947	1657	1803	1999		
Text	Sentiment Classification	Train	25k	979	2615	3431	13704	2	Acc
		Dev	25k	962	2543	3333	12988		
Retrieval	Retrieval	Train	147k	7648	13467	20495	72885	2	Acc
		Dev	18k	7665	13359	19928	72885		
		Test	17k	7702	15955	22427	50012		
Image	Category Classification	Train	45k	-	-	-	1024	10	Acc
		Dev	5k	-	-	-	1024		
		Test	10k	-	-	-	1024		
PathFinder	Image Reasoning	Train	160k	-	-	-	1024	2	Acc
		Dev	20k	-	-	-	1024		
		Test	20k	-	-	-	1024		
Text Classification									
Arxiv	Category Classification	Train	33k	14733	34209	43951	1121751	11	Micro F1
		Test	3.3k	14710	32417	40965	850540		
IMDB	Sentiment Classification	Train	25k	215	569	748	3084	2	Micro F1
		Test	25k	212	550	724	2778		
Hyperpartisan	News Classification	Train	516	536	1517	1990	5560	2	Micro F1
		Dev	65	520	1535	1971	2637		
		Test	65	637	1771	1990	5560		
Question Answering									
TriviaQA	Question Answering	Train	110k	4576	5027	5166	10091	Span	Macro F1 & EM
		Dev	14k	4577	5026	5169	10210		
WikiHop	Question Answering	Train	43k	1313	3001	3685	19747	Candidates	Acc
		Dev	5.1k	1413	3184	3871	17004		

Table 10: Downstream tasks statistics. Samples of tasks Image and PathFinder are all 32×32 images.

Hyperparameter	ListOps	Text	Retrieval	Image	Pathfinder
Hyperparameters for HST and SAOR					
HST pooling	$\{$ MIN, MEAN, MAX$\}$				
α of $\mathcal{L}_{SAOR}$	$\{$ 0.5, 5, 10$\}$				
#roll tokens of $\mathcal{L}_{SAOR}$	$\{$ 2, 8, 16 $\}$				
Fixed hyperparameters provided by LRA (Tay et al., 2021)					
learning rate	5e-2	5e-2	5e-2	5e-4	1e-3
batch size	32	32	32	256	512
weight decay	1e-1	1e-1	1e-1	0	0
warmup	1000	8000	8000	175	312
max seq length	2000	1000	4000	1024	1024
embedding dim	512	256	128	32	64
#head	8	4	4	1	2
#layer	4	4	4	1	4
Q/K/V dim	512	256	128	32	32
MLP dim	1024	1024	512	64	64
dropout	0.1	0.1	0.1	0.3	0.2
attn dropout	0.1	0.1	0.1	0.2	0.1
lr decay	root square	root square	root square	cosine	cosine

Table 11: The upper part is the hyperparameter related to ERNIE-SPARSE, while the lower part is the fixed hyperparameter provided by LRA and cannot be changed.

A.3 COMPLEXITY ANALYSIS

In this section, we analyze the complexity of ERNIE-SPARSE. For attention module in ERNIE-SPARSE, it can be analyzed from sparse attention and hierarchical sparse attention modules. For the former, ERNIE-SPARSE's sparse attention pattern is the closest to BigBird with $O(N)$ time complexity (Zaheer et al., 2020) with in practice the maximum length is set as $N = 4096$ with the 3 blocks, each of size $w = 64$, and resulting in 4096×64 computations (note that we ignore the global

[2]https://dumps.wikimedia.org/enwiki/

Hyperparameter	Arxiv	IMDB	Hyperpartisan	WikiHop	TriviaQA
HST pooling	mean	mean	mean	mean	mean
α of $\mathcal{L}_{SAOR}$	10	0.1	0	10	3
#roll tokens of $\mathcal{L}_{SAOR}$	8	8	0	8	8
learning rate	6e-5	1e-5	3e-5	3e-5	3e-5
batch size	48	64	16	48	32
epoch	10	20	20	30	10
warmup	10%	10%	10%	200	10%
max seq len	4096	2048	1024	4096	4096
#global token			128		
local window size			192		
#random token			192		
Optimizer			Adam		

Table 12: Hyperparameters of classification and question answering tasks.

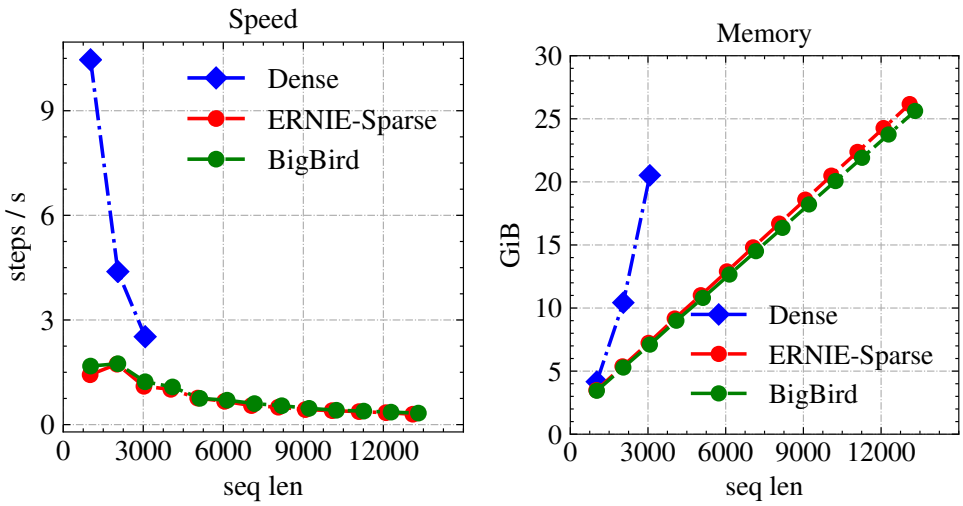

Figure 4: Runtime and memory of full self-attention, ERNIE-SPARSE and BigBird (Zaheer et al., 2020).

attention calculations here for simplicity). For the hierarchical part, because only representative tokens are extracted out for attention, so the real calculations is 64×64, which is far less than the former computation complexity, where $64 = (N \bmod w) = (4096 \bmod 64)$ is the number of blocks. So the increased time complexity of hierarchical sparse attention can be ignored. At the same time, we also carried out a practical test on V100 32GB GPU. We tested ERNIE-SPARSE and dense attention transformer and BigBird, the result is shown in Figure 4. ERNIE-SPARSE 's memory usage scales linearly with the sequence length, unlike the full self-attention mechanism that runs out of memory for long sequences on current GPU. ERNIE-SPARSE and BigBird are almost on par. The speed test data shows that the performance is consistent. The difference between ERNIE Sparse and BigBird is negligible in memory test. Given that ERNIE-SPARSE outperformed BigBird on the test set as mentioned in Section 4, the performance of ERNIE-SPARSE was remarkable.