Sparse Transformers

Karol Kaczmarek

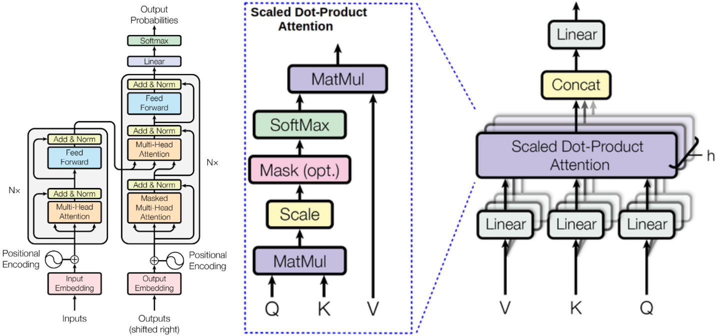
Adam Mickiewicz University
Poznań

Applica.ai
Warsaw

2020

Transformer [1]

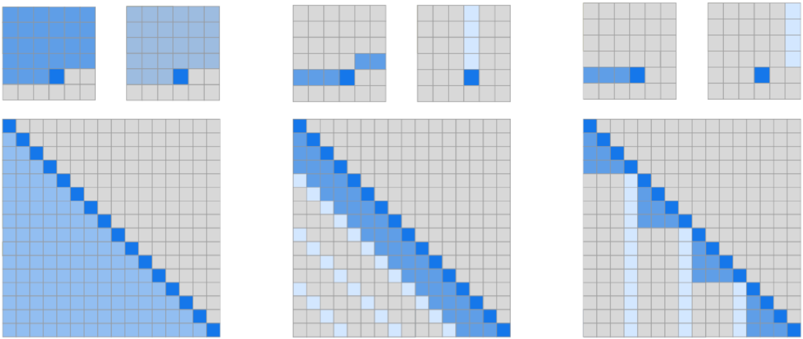
- Transformers are powerful sequence models, but require time and memory that grows quadratically with the sequence length.



Sparse Transformer [2]

- April 2019, OpenAI – code available
- generative model (decoder)
- sparse factorizations of the attention matrix (reduce to $O(n\sqrt[p]{n})$)
- gradient checkpointing
- mixed-precision training
- treat images, text, and audio as a sequence of discrete tokens, typically raw bytes

Factorized attention



(a) Transformer (b) Sparse Transformer (strided) (c) Sparse Transformer (fixed)

Generating images

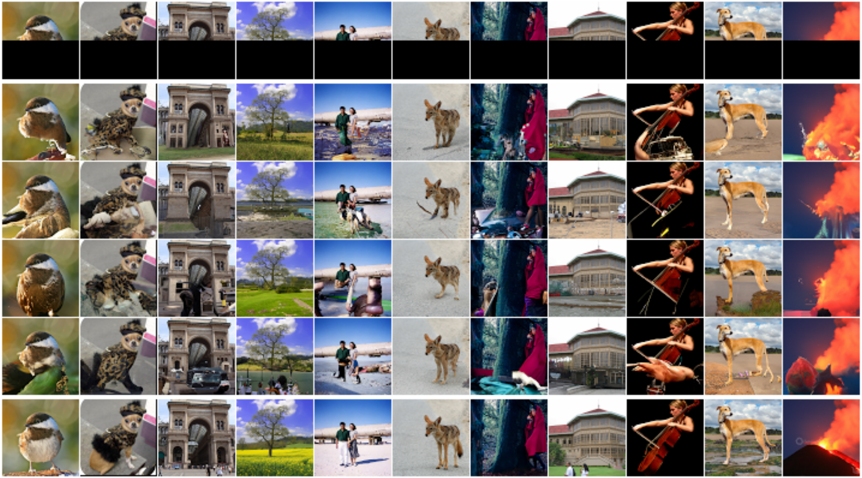
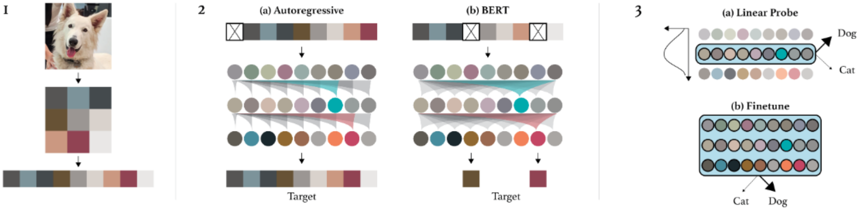
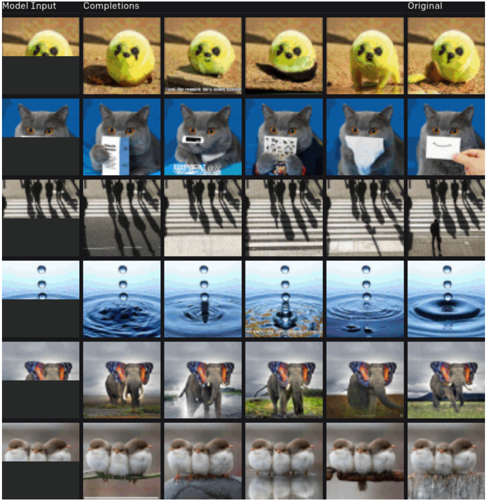


Image GPT (iGPT) [3] - June 2020



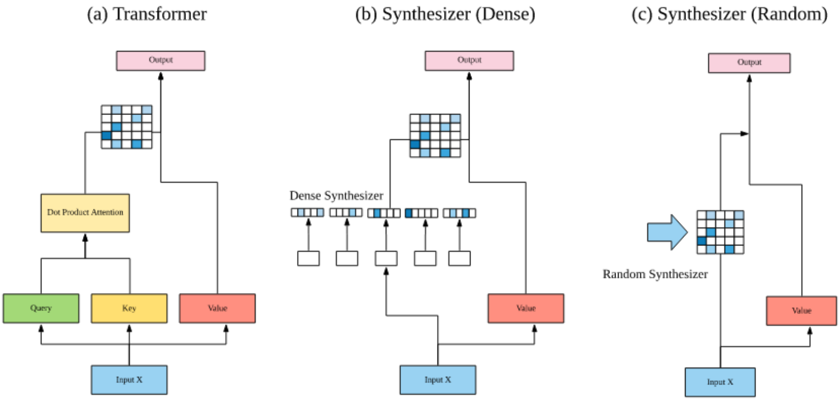
Samples



Synthesizer [4]

- May 2020, Google – source code is not available
- Synthetic Attention - removes the notion of query-key-values in the self-attention module and directly synthesizes the alignment matrix instead (without dot product attention or content-based attention):
  - Dense Synthesizers - learns synthetic attention by conditioning on each input of X and projecting to l dimensions (can be interpreted as learning a token-wise projection to the sequence length l)
  - Random Synthesizers - the attention weights are initialized to random values (the attention weights are not conditioned on any input tokens), learns a task-specific alignment that works well globally across many sample
- use factorized version of synthetic attention

Self-attention pattern



Synthesizer - Factorized Models

- omit the Q and K projections
- Factorized Dense Synthesizer - split into smaller parts of input
- Factorized Random Synthesizer - use low rank matrices

	NMT (BLEU)			LM (PPL)	
Model	# Params	EnDe	EnFr	# Params	LM1B
Transformer [Vaswani et al., 2017]	68M	27.30	38.10	-	-
Transformer (Our run)	68M	27.67	41.57	70M	38.21
Transformer (Control)	73M	27.97	41.83	-	-
Synthesizer (Fixed Random)	61M	23.89	38.31	53M	50.52
Synthesizer (Random)	67M	27.27	41.12	58M	40.60
Synthesizer (Factorized Random)	61M	27.30	41.12	53M	42.40
Synthesizer (Dense)	62M	27.43	41.39	53M	40.88
Synthesizer (Factorized Dense)	61M	27.32	41.57	53M	41.20
Synthesizer (Random + Dense)	67M	27.68	41.21	58M	42.35
Synthesizer (Dense + Vanilla)	74M	27.57	41.38	70M	**37.27**
Synthesizer (Random + Vanilla)	73M	**28.47**	**41.85**	70M	40.05

Table 2: Experimental Results on WMT'14 English-German, WMT'14 English-French Machine Translation tasks and Language Modeling One Billion (LM1B).

T5 methodology - Score

Model	Glue	CoLA	SST	MRPC	STSB	QQP	MNLI	QNLI	RTE
T5 (Base)	83.5	53.1	**92.2**	**92.0/88.7**	89.1/88.9	88.2/91.2	84.7/**85.0**	91.7	76.9
Syn (R)	75.1	41.2	91.2	85.9/79.4	74.0/74.3	85.5/89.0	77.6/78.1	87.6	59.2
Syn (D)	72.0	18.9	89.9	86.4/79.4	75.3/75.5	85.2/88.3	77.4/78.1	86.9	57.4
Syn (D+V)	82.6	48.6	92.4	91.2/87.7	88.9/89.0	88.6/91.5	84.3/84.8	91.7	75.1
Syn (R+V)	**84.1**	**53.3**	**92.2**	91.2/87.7	**89.3/88.9**	**88.6/91.4**	**85.0**/84.6	**92.3**	**81.2**

Table 4: Experimental results (dev scores) on multi-task language understanding (GLUE benchmark) for *small* model and `en-mix` mixture. Note: This task has been co-trained with SuperGLUE.

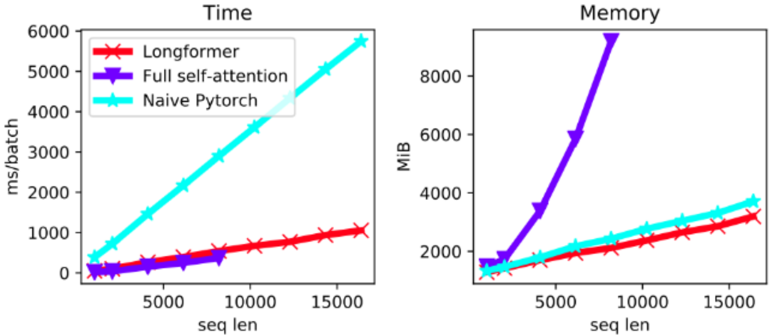
Model	SGlue	BoolQ	CB	CoPA	MultiRC	ReCoRD	RTE	WiC	WSC
T5 (Base)	70.3	78.2	72.1/83.9	59.0	73.1/32.1	**71.1/70.3**	77.3	**65.8**	**80.8**
Syn (R)	61.1	69.5	54.6/73.2	60.0	63.0/15.7	58.4/57.4	67.5	64.4	66.3
Syn (D)	58.5	69.5	51.7/71.4	51.0	66.0/15.8	54.1/53.0	67.5	65.2	58.7
Syn (D+V)	69.7	79.3	74.3/85.7	64.0	73.8/33.7	69.9/69.2	78.7	64.3	68.3
Syn (R+V)	**72.2**	**79.3**	**82.7/91.1**	**64.0**	**74.3/34.9**	70.8/69.9	**82.7**	64.6	75.0

Table 5: Experimental results (dev scores) on multi-task language understanding (SuperGLUE benchmark) for *small* model and `en-mix` mixture. Note: This task has been co-trained with GLUE.

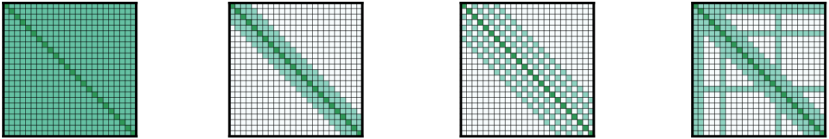
Longformer [5]

- April 2020, Allen Institute for Artificial Intelligence (allenai) – available in transformers library
- windowed local-context self-attention (attention mechanism that scales linearly with sequence length)
- process long sequences without truncating or chunking
- autoregressive character-level language modeling (allowing the model to process sequences of up to 32K characters on modern GPUs)
- replace the full self-attention operation of existing pretrained models (RoBERTa model - MLM objective)
- custom CUDA kernel using TVM (Tensor Virtual Machine)

Memory usage



Self-attention pattern



(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

- Sliding Window - fixed-size window attention surrounding each token, top layers have access to all input locations
- Dilated Sliding Window - window has gaps of size dilation
- Global Attention - few pre-selected input locations (classification = [CLS] token, QA = all question tokens)

AR LM - Score

Model	#Param	Dev	Test
Dataset `text8`			
T12 (Al-Rfou et al., 2018)	44M	-	1.18
Adaptive (Sukhbaatar et al., 2019)	38M	1.05	1.11
BP-Transformer (Ye et al., 2019)	39M	-	1.11
Our Longformer	41M	1.04	**1.10**
Dataset `enwik8`			
T12 (Al-Rfou et al., 2018)	44M	-	1.11
Transformer-XL (Dai et al., 2019)	41M	-	1.06
Reformer (Kitaev et al., 2020)	-	-	1.05
Adaptive (Sukhbaatar et al., 2019)	39M	1.04	1.02
BP-Transformer (Ye et al., 2019)	38M	-	1.02
Our Longformer	41M	1.02	**1.00**

Table 2: *Small* model BPC on `text8` & `enwik8`

Model	#Param	Test BPC
Transformer-XL (18 layers)	88M	1.03
Sparse (Child et al., 2019)	≈100M	0.99
Transformer-XL (24 layers)	277M	0.99
Adaptive (Sukhbaatar et al., 2019)	209M	0.98
Compressive (Rae et al., 2020)	277M	0.97
Our Longformer	102M	0.99

Table 3: Performance of *large* models on `enwik8`

RoBERTa base - Score

- continue pre-training from the RoBERTa - use the sliding window attention with window size of 512 on all layers (uses the same amount of computation as RoBERTa)
- can process sequences up to 4,096 tokens long (8 times longer than BERT - 512 tokens)
- initialize them by copying the 512 position embedding

Wordpieces	WH	TQA	HQA	ON	IMDB	HY
avg.	1,535	6,589	1,316	506	300	705
95th pctl.	3,627	17,126	1,889	1,147	705	1,975

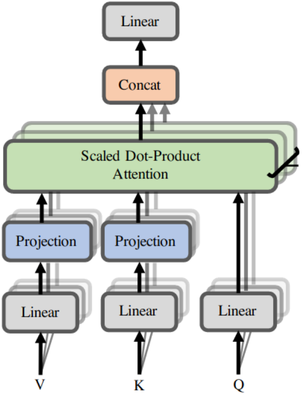
	QA			Coref.	Classification	
Model	WikiHop	TriviaQA	HotpotQA	OntoNotes	IMDB	Hyperpartisan
RoBERTa-base	72.4	74.3	63.5	78.4	95.3	87.4
Longformer-base	**75.0**	**75.2**	**64.4**	**78.6**	**95.7**	**94.8**

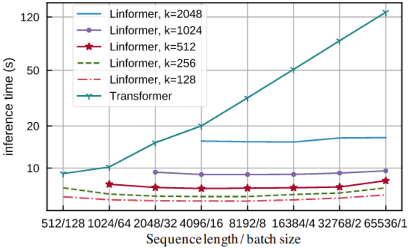
Linformer [6]

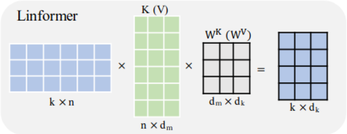
- June 2020, Facebook – code available
- base on RoBERTa model
- self-attention mechanism can be approximated by a low-rank matrix (with different dimension for different layers - smaller dimension for higher layers)
- parameter sharing between projections

Model Architecture	Complexity per Layer	Sequential Operation
Recurrent	$O(n)$	$O(n)$
Transformer, (Vaswani et al., 2017)	$O(n^2)$	$O(1)$
Sparse Tansformer, (Child et al., 2019)	$O(n\sqrt{n})$	$O(1)$
Reformer, (Kitaev et al., 2020)	$O(n\log(n))$	$O(\log(n))$
Linformer	$O(n)$	$O(1)$

Linformer



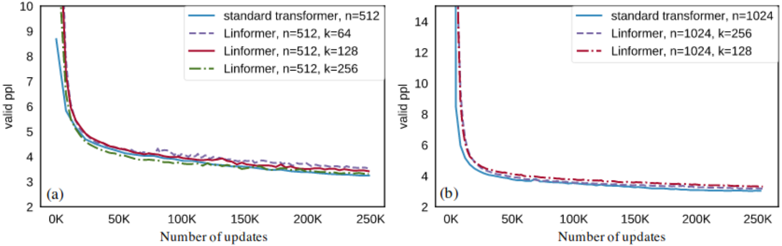




if we can choose a very small projected dimension k, such that $k \ll n$, then we can significantly reduce the memory and space consumption

Score

n	Model	SST-2	IMDB	QNLI	QQP	Average
	Liu et al. (2019), RoBERTa-base	93.1	94.1	90.9	**90.9**	92.25
	Linformer, 128	92.4	94.0	90.4	90.2	91.75
	Linformer, 128, shared kv	**93.4**	93.4	90.3	90.3	91.85
	Linformer, 128, shared kv, layer	93.2	93.8	90.1	90.2	91.83
512	Linformer, 256	93.2	94.0	90.6	90.5	92.08
	Linformer, 256, shared kv	93.3	93.6	90.6	90.6	92.03
	Linformer, 256, shared kv, layer	93.1	94.1	**91.2**	90.8	**92.30**



Inference-time Efficiency Results

length n	projected dimensions k 128	256	512	1024	2048
512	1.5x	1.3x	-	-	-
1024	1.7x	1.6x	1.3x	-	-
2048	2.6x	2.4x	2.1x	1.3x	-
4096	3.4x	3.2x	2.8x	2.2x	1.3x
8192	5.5x	5.0x	4.4x	3.5x	2.1x
16384	8.6x	7.8x	7.0x	5.6x	3.3x
32768	13x	12x	11x	8.8x	5.0x
65536	20x	18x	16x	14x	7.9x

length n	projected dimensions k 128	256	512	1024	2048
512	1.7x	1.5x	-	-	-
1024	3.0x	2.9x	1.8x	-	-
2048	6.1x	5.6x	3.6x	2.0x	-
4096	14x	13x	8.3x	4.3x	2.3x
8192	28x	26x	17x	8.5x	4.5x
16384	56x	48x	32x	16x	8x
32768	56x	48x	36x	18x	16x
65536	60x	52x	40x	20x	18x

Table 3: Inference-time efficiency improvements of the Linformer over the Transformer, across various projected dimensions k and sequence lengths n. Left table shows time saved. Right table shows memory saved.

Other

- Adaptive Attention Span [7] - August 2019, Facebook
- Adaptively Sparse Transformers [8] - June 2020, Facebook
- Reformer [9] - January 2020, Google
- nBRC [10] - June 2020
- GShard [11]- June 30, 2020, Google

Sandwich Transformers [12]

- April 2020
- reordering of the transformer sublayers (the sandwich reordering pattern does not guarantee performance gains across every task)
- test on WikiText-103

Each transformer layer (encoder) consists of a self-attention sublayer (s) followed by a feedforward sublayer (f):

(a) Interleaved Transformer

(b) Sandwich Transformer

WikiText-103 – Randomly generated models with 16 self-attention (s) sublayers and 16 feedforward (f) sub-layers

Model	PPL
fsfsfffsffsfsssffsfssfssssffsffs	20.74
sfssffsffffssssfsfffsfsffsfssssf	20.64
fsffssffssssffsssssffsfssfsfffff	20.33
fsffffffsssfssffsfssffsfsssffsss	20.27
fssffffffsfsssfffssssfffssssffss	19.98
sssfssfsffffssfsfsfsssffsfsfffsf	19.92
fffsfsssfsffsfsffsffsssssffssffs	19.69
fffsffssffsssfssfsssfffffsfsssfs	19.54
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**19.13**
fsffssfssfffssssfffsssffffsfssfs	19.08
sfsffssssffssffffsssffsssfsffsff	18.90
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.83**
sssssssffsffsfsfsffffsfffsfssffs	18.83
sffsfsffsfsssffssfssssssfffffffs	18.77
sssfssffsfssfsffsfffssffsfsffssf	18.68
fffsssssfffsfssssffsfsfsfssffsff	18.64
sfffsssfsfssfsssssfssfffffsfffsf	18.61
ssffssfssssffffffssffsssfsffssff	18.60
fsfsssssfsfsfffffsfffsffssffssss	18.55
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.54**
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.49**
fsfsssssfsfffssfsffsfsfsfsffffss	18.38
sfssffsfsfsffsssssfffsssfffsffsf	18.28
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.25**
sfsfssfsssffsfsfsfsffffssffsfssf	18.19

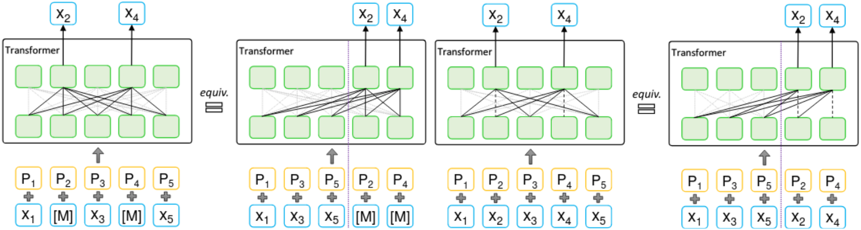
WikiText-103 – Randomly generated models with the same number of parameters

Model	PPL
sfffssfsfsfssffffsfsffsffffff	22.80
sffssfsssssssssssssfsfsssfsffsssfsssfs	21.02
ssssssffsffffssfffffsssfsfsssssssss	20.98
fffffffffsffssffsffssssfsfsssf	20.75
fssfsssffffffssfsssfsfffssssfsfss	20.43
sffsffffffsfsfssfsssfsfsfssfssfs	20.28
sffssffsfffsfsfssssffffffssssff	20.02
fsffsfssffffsfsfffsfffssfffsss	19.93
sffsffssffsfsffsssfsssssfsssfffsss	19.85
ssfffffffssfffssfssffsfsfsffsf	19.82
sfsfsfffsfffssfsfffsffssfsfsfss	19.77
sfsffsssffsffsssfssfffffssssfsssf	19.55
sffsfssfffsffsfssssfsfsffffsfsss	19.49
sffffsffssssfsssfssfffsssfssssfsfs	19.47
fsssffssssssfsfsfsffsffffssfsfssss	19.25
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**19.13**
fssssssfsfsfsfffsfsssfssffssssfsff	18.86
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.83**
ssfsfsssfsssssffsfsfsssfssfsfsssssssf	18.62
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.54**
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.49**
sssfsffsfssfsssffsffffffssfsfff	18.34
sssfsfsffsssfsfffffsfsffffsssff	18.31
sfsfsfsfsfsfsfsfsfsfsfsfsfsfsfsf	**18.25**
ssssssfsssffffsfsfffffffffffsf	18.12

MPNet [13]

- April 2020, Microsoft – code available
- MPNet (masked and permuted language modeling) inherits the advantages of BERT (MLM) and XLNet (PLM) and avoids their limitations:
  - BERT (MLM) - neglects dependency among predicted tokens
  - XLNet (PLM) - does not leverage the full position information of a sentence (does not know the position information of the full sentence during the autoregressive pre-trainin)
- MPNet outperforms MLM and PLM by a large margin, and achieves better results

Unified view of MLM and PLM



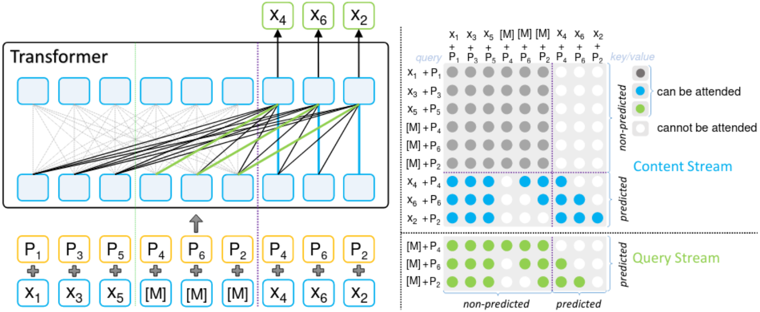
(a) MLM (b) PLM

Example

An example sentence **the task is sentence classification** to illustrate the conditional information of MLM, PLM and MPNet:

Objective	Factorization
MLM (BERT)	$\log P(\text{sentence} \mid \text{the task is [M] [M]}) + \log P(\text{classification} \mid \text{the task is [M] [M]})$
PLM (XLNet)	$\log P(\text{sentence} \mid \text{the task is}) + \log P(\text{classification} \mid \text{the task is sentence})$
MPNet	$\log P(\text{sentence} \mid \text{the task is [M] [M]}) + \log P(\text{classification} \mid \text{the task is sentence [M]})$

MPNet



(a) (b)

Figure 2: (a) The structure of MPNet. (b) The attention mask of MPNet. The light grey lines in (a) represent the bidirectional self-attention in the non-predicted part $(x_{z_{<=c}}, M_{z_{>c}}) = (x_1, x_5, x_3, [M], [M], [M])$, which correspond to the light grey attention mask in (b). The blue and green mask in (b) represent the attention mask in content and query streams in two-stream self-attention, which correspond to the blue, green and black lines in (a). Since some attention masks in content and query stream are overlapped, we use black lines to denote them in (a). Each row in (b) represents the attention mask for a query position and each column represents a key/value position. The predicted part $x_{z_{>c}} = (x_4, x_6, x_2)$ is predicted by the query stream.

Score

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	Avg
Single model on dev set									
BERT (Devlin et al., 2019)	84.5	91.7	91.3	68.6	93.2	87.3	58.9	89.5	83.1
XLNet (Yang et al., 2019)	86.8	91.7	91.4	74.0	94.7	88.2	60.2	89.5	84.5
RoBERTa (Liu et al., 2019a)	87.6	92.8	**91.9**	78.7	94.8	90.2	63.6	**91.2**	86.4
MPNet	**88.5**	**93.3**	**91.9**	**85.2**	**95.4**	**91.5**	**65.0**	90.9	**87.7**
Single model on test set									
BERT (Devlin et al., 2019)	84.6	90.5	89.2	66.4	93.5	84.8	52.1	87.1	79.9
ELECTRA (Clark et al., 2020)	**88.5**	**93.1**	89.5	75.2	**96.0**	88.1	**64.6**	**91.0**	85.8
MPNet	**88.5**	93.0	**89.6**	**80.5**	95.6	**88.2**	64.0	90.7	**86.3**

References I

[1] A. Vaswani and et al., “Attention is all you need,” 2017.

[2] R. Child, S. Gray, A. Radford, and I. Sutskever, “Generating long sequences with sparse transformers,” 2019.

[3] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, P. Dhariwal, D. Luan, and I. Sutskever, “Generative pretraining from pixels,” 2020.

[4] Y. Tay, D. Bahri, D. Metzler, D.-C. Juan, Z. Zhao, and C. Zheng, “Synthesizer: Rethinking self-attention in transformer models,” 2020.

[5] I. Beltagy, M. E. Peters, and A. Cohan, “Longformer: The long-document transformer,” 2020.

[6] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, “Linformer: Self-attention with linear complexity,” 2020.

[7] S. Sukhbaatar, E. Grave, P. Bojanowski, and A. Joulin, “Adaptive attention span in transformers,” 2019.

[8] G. Correia, V. Niculae, and A. Martins, “Adaptively sparse transformers,” 2019.

[9] N. Kitaev, L. Kaiser, and A. Levskaya, “Reformer: The efficient transformer,” 2020.

[10] N. Vecoven, D. Ernst, and G. Drion, “A bio-inspired bistable recurrent cell allows for long-lasting memory,” 2020.

[11] D. Lepikhin, H. Lee, and Y. Xu, “Gshard: Scaling giant models with conditional computation and automatic sharding,” 2020.

[12] O. L. Ofir Press, Noah A. Smith, “Improving transformer models by reordering their sublayers,” 2020.

[13] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, “Mpnet: Masked and permuted pre-training for language understanding,” 2020.