Optimized BERT Modifications

Marat Saidov

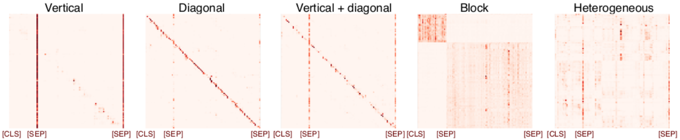
Higher School of Economics

masaidov@edu.hse.ru

April 21, 2020

Motivation (dark secrets of BERT 😎)

BERT heads share a **limited set** of attention patterns. We obtain **model overparameterization**. Main patterns:



What information these attention maps keep?

- Attention to linguistic features. Wrong hypothesis: vertical lines show the strong linguistic features. (most of them are presented in [CLS] and [SEP] tokens);
- Token-to-token attention: noun-pronoun & verb-subject links usually coincide with words positions in the sentence;
- **Disabling self-attention heads**. Achieved $0.1\% - 1.2\%$ performance gain on GLUE benchmark (except Language Inference tasks) $\Rightarrow$ some of the heads hurt the performance;

Factorized embedding parameterization

Hyperparameters: WordPiece embedding size E, hidden layer size H, vocabulary size V ($\approx 10^5$ for WordPieces).

$H \equiv E$ (XLNet, RoBERTa practices).

$H \uparrow \Rightarrow V \times E \uparrow$ (embedding matrix size). Matrix is sparse, unnecessary increase.

Key idea: to untie E from H and break down into 2 matrices.

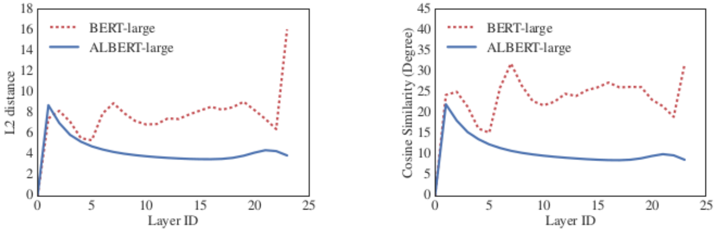
Factorized embedding parameterization

Words: different embedding size is important.
WordPieces: $E = \text{const}$.

We project OHE-representation into a lower dimensional embedding space. If $H \gg E$, significant win in a # embedding parameters:
$O(V \times H) \Rightarrow O(V \times E + E \times H)$

Cross-layer parameter sharing

Authors tried to share FFN parameters, attention parameters, both of them.



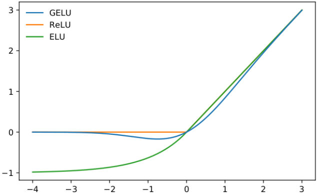
Note: weight-sharing has an effect on stabilizing network parameters.

GELU activations

GELU (**G**aussian **E**rror **L**inear **U**nits) – a probablistic view on a neuron's output.

$$X \sim \mathcal{N}(0,1) \Rightarrow \Phi(x) = P(X \leq x) \Rightarrow m \sim \text{Bernoulli}(\Phi(x))$$

$$\text{GELU}(x) = m \cdot x$$



Sentence Order Prediction (SOP)

NSP: **topic prediction** and **coherence prediction** tasks simultaneously.

Key idea: Coherence is more complicated, topic overlaps parameter values.

SOP: negative example is a flip of positive example segments.

RACE dataset

RACE consists of near $28,000$ passages and near $100,000$ questions generated by human experts (English instructors). It covers news, stories, ads, biography, philosophy etc.

Dataset	RACE-M	RACE-H	RACE
Word Matching	29.4%	11.3%	15.8%
Paraphrasing	14.8%	20.6%	19.2%
Single-Sentence Reasoning	31.3%	34.1%	33.4%
Multi-Sentence Reasoning	22.6%	26.9%	25.8%
Ambiguous/Insufficient	1.8%	7.1%	5.8%

Overall comparison to BERT

ALBERT-xxlarge with 70% BERT parameters achieved significant results for several downstream tasks:

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

Embedding parameterization experiments

Under the non-shared conditions larger embedding space size E gives better performance, by not much. Let $E = 128$:

Model	E	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base not-shared	64	87M	89.9/82.9	80.1/77.8	82.9	91.5	66.7	81.3
	128	89M	89.9/82.8	80.3/77.3	83.7	91.5	67.9	81.7
	256	93M	90.2/83.2	80.3/77.4	84.1	91.9	67.3	81.8
	768	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base all-shared	64	10M	88.7/81.4	77.5/74.8	80.8	89.4	63.5	79.0
	128	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	256	16M	88.8/81.5	79.1/76.3	81.5	90.3	63.4	79.6
	768	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8

Parameter sharing experiments

Another technique: L layers are divided into N groups of size M, sharing is provided inside each group.

$M \downarrow \Rightarrow$ (Performance) $\uparrow$

$M \downarrow \Rightarrow$ (Parameters number) $\uparrow\uparrow$

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base E=768	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7	81.6
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6	79.5
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base E=128	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0	80.1
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6	81.7
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4	80.2
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9	81.6

SOP Experiments

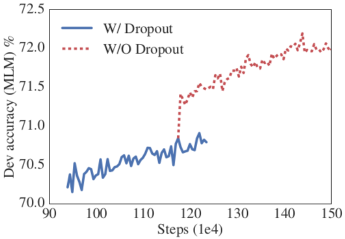
Sentence order prediction task dominates next sentence prediction both in pre-training and fine-tuning.

	Intrinsic Tasks			Downstream Tasks					
SP tasks	MLM	NSP	SOP	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
None	54.9	52.4	53.3	88.6/81.5	78.1/75.3	81.5	89.9	61.7	79.0
NSP	54.5	90.5	52.0	88.4/81.5	77.2/74.6	81.6	**91.1**	62.3	79.2
SOP	54.0	78.9	86.5	**89.3/82.3**	**80.0/77.1**	**82.0**	90.3	**64.0**	**80.1**

Avoiding dropout hypothesis

There is empirical and theoretical evidence showing that a combination of BN and dropout in CNN may have harmful results.
Hypothesis: dropout can hurt transformed-based models performance.
Problem: ALBERT is a very special case of the transformer.

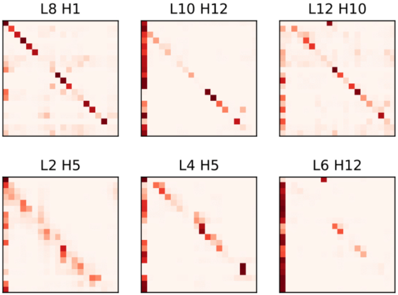
	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
With dropout	94.7/89.2	89.6/86.9	90.0	96.3	85.7	90.4
Without dropout	**94.8/89.5**	**89.9/87.2**	**90.4**	**96.5**	**86.1**	**90.7**



Purpose

An aim is to improve the training efficiency of the BERT model from in an algorithmic sense.

Fact: attentions are repeated not only across different heads, but also across different layers.



Algorithm description

If we have L-layer BERT, we can construct $2L$-layer BERT. Warm start from already trained BERT (**shallow model**). Sharing: i-th layer is copied into a $(L+i)$-th layer.

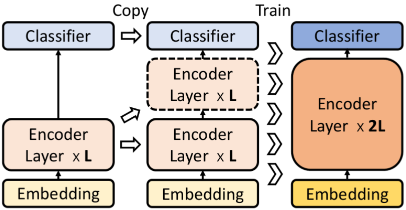


Figure 3. The diagram of the *stacking* algorithm.

Algorithm detailed

Algorithm 1 Progressive stacking

$M_0' \leftarrow \text{InitBERT}(L/2^k)$
$M_0 \leftarrow \text{Train}(M_0')$ {Train from scratch.}
for $i \leftarrow 1$ to k **do**
  $M_i' \leftarrow \text{Stack}(M_i)$ {Doubles the number of layers.}
  $M_i \leftarrow \text{Train}(M_i')$ {M_i has $L/2^{k-i}$ layers.}
end for
return M_k

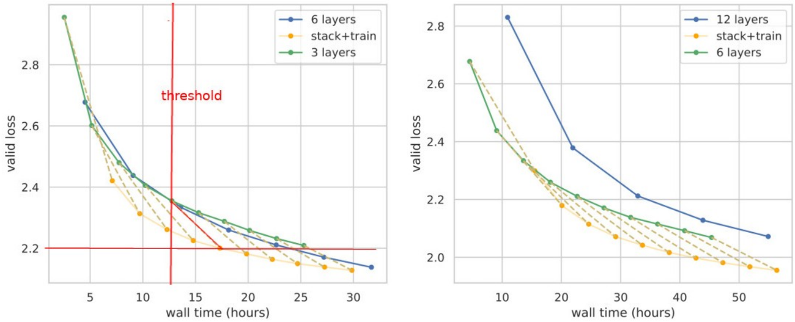
Results

Clearly, the information is being lost during training. That's why authors wanted to show closest results to the original BERT:

	CoLA 8.5k	SST-2 67k	MRPC 3.7k	STS-B 5.7k	QQP 364k	MNLI-m/mm 393k	QNLI 108k	RTE 2.5k	GLUE
ELMo-BiLSTM-Attn	33.6	90.4	84.4/78.0	74.2/72.3	63.1/84.3	74.1/74.5	79.8	58.9	70.0
OpenAI GPT	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.7/80.6	87.2	**69.1**	76.9
BERT-Base (original)	52.1	93.5	**88.9/84.8**	**87.1/85.8**	**71.2/89.2**	**84.6**/83.4	**90.5**	66.4	78.3
BERT-Base (baseline)	52.8	92.8	87.3/83.0	81.2/80.0	70.2/88.4	84.4/83.7	90.4	64.9	77.4
BERT-Base (stacking)	**56.2**	**93.9**	88.2/83.9	84.2/82.5	70.4/88.7	84.4/**84.2**	90.1	67.0	**78.4**

Switching time

Note: there is exists a threshold θ such that if we pick the *switching time* $t < \theta$, a progressive stacking algorithm will be trained fast than training from scratch.



Purpose

Usually PLMs (pretrained language models) are heavy. We aim to reduce a memory complexity of a model.
Two-stage learning framework:

- General distillation;
- Task-specific distillation;

Knowledge distillation: quick reminder

T – teacher network, S – student network. f^T, f^S are *behaviour* functions for teacher and student respectively (MHA or FFN outputs).
Formally, KD is an optimization task:

$$\mathcal{L}_{KD} = \sum_{x \in \mathcal{X}} L\left(f^S(x), f^T(x)\right) \to \min_{\theta_{f^S}, \theta_{f^T}}$$

How to choose properly:

- Loss function L?
- Behaviour functions f^T, f^S?

Problem formulation (layer mapping function)

Student network consists of M transformer layers, teacher – N layers. Let $g : \{0, ..., M+1\} \to \{0, ..., N+1\}$ be a mapping from a distillated student layer to an original teacher layer.

- $0 = g(0)$ (**embedding layer**)
- $N + 1 = g(M + 1)$ (**prediction layer**)

Problem formulation (final objective)

Introduce a final objective for a BERT distillation task:

$$\mathcal{L}_{\text{model}} = \sum_{m=0}^{M+1} \lambda_m \mathcal{L}_{\text{layer}}\left(S_m, T_{g(m)}\right)$$

S_m, $T_{g(m)}$ are the outputs of given layers. $\mathcal{L}_{\text{layer}}\left(S_m, T_{g(m)}\right)$ represents a loss on them, λ_m is an importance of certain layer.

Attention and hidden states distillation losses

An **attention distillation loss** is simply a $L2$-distance on attention matrices:

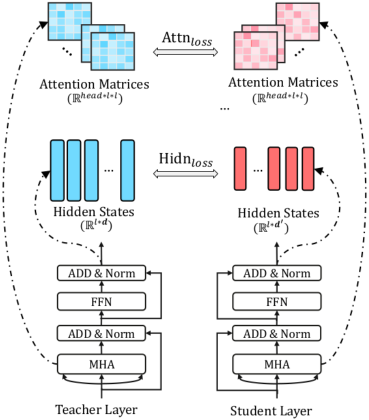
$$\mathcal{L}_{\text{attn}} = \frac{1}{h}\sum_{i=1}^{h} ||A_i^S - A_i^T||_2^2$$

Let $W_h \in \mathbb{R}^{d' \times d}$ be a learnable transformation from the student hidden states to the teacher hidden states. Then a **hidden distillation loss** could be defined:

$$\mathcal{L}_{\text{hidn}} = ||H^S \cdot W_h - H^T||_2^2$$

H^S, H^T represent hidden states of the networks.

Attention and hidden states distillation losses



Embedding layer and prediction layer distillation losses

Embedding distillation loss is similar to the hidden states based distillation:

$$\mathcal{L}_{\text{embd}} = ||E^S \cdot W_e - E^T||_2^2$$

Prediction distillation loss is taken from an original KD paper:

$$\mathcal{L}_{\text{pred}} = -\text{softmax}(z^T) \cdot \text{log\_softmax}\left(\frac{z^S}{t}\right)$$

TinyBERT Learning (general distillation)

- Teacher: original pre-trained BERT;
- Training data: arbitrary unlabeled text corpus;

Pre-trained TinyBERT is significantly worse than the original one by itself.

TinyBERT Learning (task-specific distillation)

- Mask a word in a sentence;
- Use pre-trained BERT as a prediction of M most-likely words;
- Replace current word to a randomly selected candidate (or save it with certain probability);
- Apply this transformation to each word of a sentence;
- Take your augmented sentence;

Problem: multiple sub-word pieces are not recognizable.
Solution: picking a similar word from GloVe.
And now just continue distilling with an augmented dataset.

Ablation studies

Table 5: Ablation studies of different procedures (i.e., TD, GD, and DA) of the two-stage learning framework. The variants are validated on the dev set.

System	MNLI-m	MNLI-mm	MRPC	CoLA	Average
TinyBERT	82.8	82.9	85.8	49.7	75.3
No GD	82.5	82.6	84.1	40.8	72.5
No TD	80.6	81.2	83.8	28.5	68.5
No DA	80.5	81.0	82.4	29.8	68.4

Table 6: Ablation studies of different distillation objectives in the TinyBERT learning. The variants are validated on the dev set.

System	MNLI-m	MNLI-mm	MRPC	CoLA	Average
TinyBERT	82.8	82.9	85.8	49.7	75.3
No Embd	82.3	82.3	85.0	46.7	74.1
No Pred	80.5	81.0	84.3	48.2	73.5
No Trm	71.7	72.3	70.1	11.2	56.3
No Attn	79.9	80.7	82.3	41.1	71.0
No Hidn	81.7	82.1	84.1	43.7	72.9

Table 7: Results (dev) of different mapping strategies.

System	MNLI-m	MNLI-mm	MRPC	CoLA	Average
TinyBERT (Uniform-strategy)	82.8	82.9	85.8	49.7	75.3
TinyBERT (Top-strategy)	81.7	82.3	83.6	35.9	70.9
TinyBERT (Bottom-strategy)	80.6	81.3	84.6	38.5	71.3

Conclusion: each part of the pipeline was important!

Final results

Table 2: Results are evaluated on the test set of GLUE official benchmark. All models are learned in a single-task manner. "-" means the result is not reported.

System	MNLI-m	MNLI-mm	QQP	SST-2	QNLI	MRPC	RTE	CoLA	STS-B	Average
$\text{BERT}_{\text{BASE}}$ (Google)	84.6	83.4	71.2	93.5	90.5	88.9	66.4	52.1	85.8	79.6
$\text{BERT}_{\text{BASE}}$ (Teacher)	83.9	83.4	71.1	93.4	90.9	87.5	67.0	52.8	85.2	79.5
$\text{BERT}_{\text{SMALL}}$	75.4	74.9	66.5	87.6	84.8	83.2	62.6	19.5	77.1	70.2
Distilled $\text{BiLSTM}_{\text{SOFT}}$	73.0	72.6	68.2	90.7	-	-	-	-	-	-
BERT-PKD	79.9	79.3	70.2	89.4	85.1	82.6	62.3	24.8	79.8	72.6
DistilBERT	78.9	78.0	68.5	91.4	85.2	82.4	54.1	32.8	76.1	71.9
TinyBERT	82.5	81.8	71.3	92.6	87.7	86.4	62.9	43.3	79.9	76.5

Note: it is possible to vary a size of TinyBERT. However, DistillBERT and BERT-PKD have fixed size.

References

Lan et al. (2020)
ALBERT: A Lite BERT for self-supervised learning of language representations

Hendrycks et al. (2018)
Gaussian Error Linear Units

Lai et al. (2017)
RACE: Large-scale ReAding Comprehension Dataset From Examinations

Jiao et al. (2019)
TinyBERT: Distilling BERT for Natural Language Understanding

Isola et al. (2019)
Revealing the Dark Secrets of BERT

Questions

1. What is the main problem of NSP in comparison with SOP? How to form a negative example in SOP?
2. Describe a work principle of a factorized embedding parameterization. What is an aim of applying it?
3. Which approaches of sharing parameters across layers do you know? Describe their ideas.
4. Write out all of the distillation losses formulas and explain their main components.