Efficient NLP Modeling and Training

— Advances in Huawei Noah's Ark Lab

Qun Liu (刘群)

Huawei Noah's Ark Lab

机器之心ACL2021论文分享会，2021-07-28

NOAH'S ARK LAB

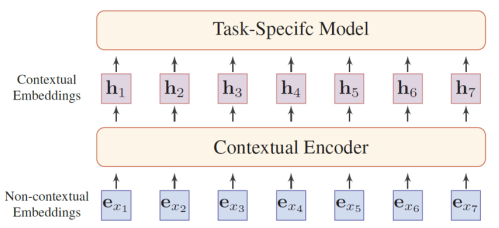
HUAWEI

Content

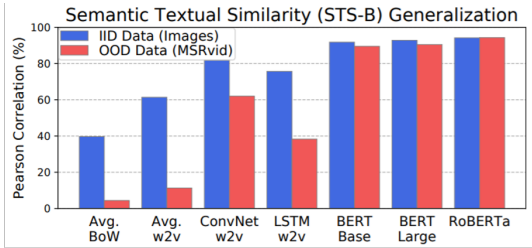
Content

预训练模型已经成为各种NLP任务的基石

- 各种预训练模型被各大公司竞相提出
- 先做大阶段：“大算力+大模型+大数据+创意任务”探索能力边界
- 再做小阶段：在各种下游任务上形成生产力（对话/阅读理解/搜索等）



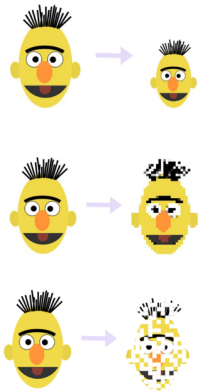
(Qiu et al., 2020)



(Hendrycks et al., 2020)

预训练模型压缩

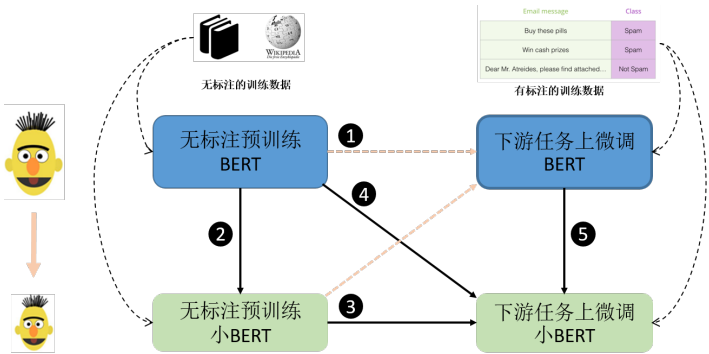
- 蒸馏(Knowledge Distillation)
  - 用一个小模型来模拟大模型
  - DistilBERT/BERT-PKD/TinyBERT
  - MobileBERT/MiniLM (Task agnostic)
- 量化(Quantization)
  - 用低bit来表示权重和激活函数
  - Q-BERT/Q8BERT
  - TernaryBERT
- 剪枝/可伸缩 (Pruning/Slimmable)
  - 将一些不重要的head/层/神经元去掉
  - LayerDrop/DynaBERT
- 其他：参数共享/矩阵分解/特征自动生成等



Content

Content

预训练模型蒸馏–知识迁移视角



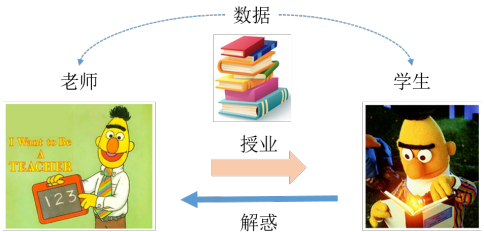
迁移1：基于无标注预训练的BERT到基于下游任务微调的BERT

迁移2+3：通过两步，将在无标注语料学到的知识迁移到小模型

迁移4：通过一步，将无标注语料学到的知识迁移到小模型

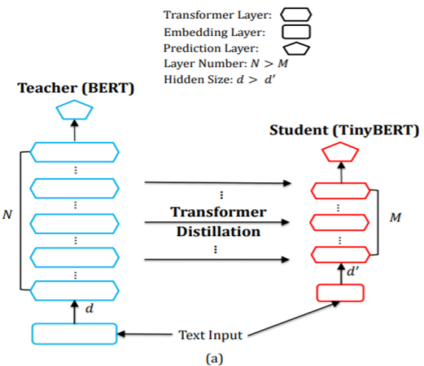
迁移5：将下游任务上的老师迁移到小模型

预训练模型蒸馏–面临的问题



- 好的老师模型：包含的知识应该更容易迁移到小模型，（比如：MobileBERT专门学习了瘦高的老师）
- 好的学生模型：应该稀疏、低比特、参数少，同时可以包含更多的知识；（比如：AdaBERT通过AutoML搜索学生结构）
- 好的学习方法：
  1. 定义更好的知识表示函数和损失函数；
  2. 通过各种策略学习怎么教给小模型，比如：逐步地将知识迁移到学生不同的层，找一个助教解决超大老师的知识不易于迁移的难题；
  3. 通过更加自动化的方法扩充语料，将更多的知识教给小模型。

预训练模型蒸馏–损失函数



$$\sum_{x \in X} \sum_{m=1}^{M+1} L(f(s_m(x)), f(t_{g(m)}(x)))$$

- X: Dataset
- m: index of a student layer
- s_m: the m^{th} student layer
- $t_{g(m)}$: the teacher layer corresponding to the m^{th} student layers
- $f(*)$: the knowledge function
- $L(*)$: the loss function

Content

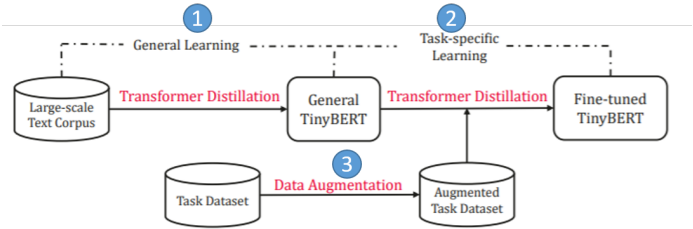
CKD: Combination of Layers

TinyBERT: Distilling BERT for Natural Language Understanding

Xiaoqi Jiao[1]*†, Yichun Yin[2]*‡, Lifeng Shang[2]‡, Xin Jiang[2]
Xiao Chen[2], Linlin Li[3], Fang Wang[1]‡ and Qun Liu[2]
[1]Key Laboratory of Information Storage System, Huazhong University of Science and Technology, Wuhan National Laboratory for Optoelectronics
[2]Huawei Noah's Ark Lab
[3]Huawei Technologies Co., Ltd.

Published in EMNLP 2020 Findings (Long paper)

TinyBERT知识蒸馏的基本流程



(1) 第一步蒸馏 GD(General Distillation)
- 将pre-trained teacher BERT知识迁移到general TinyBERT

(2) 第二步蒸馏 TD(Task-specific Distillation)
- 将fine-tuned teacher BERT知识迁移到fine-tuned TinyBERT

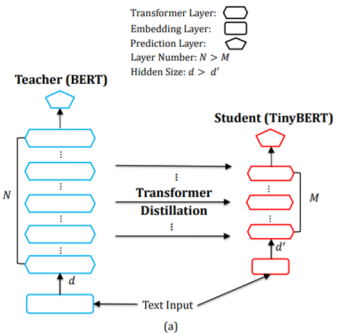
(3) 数据增强 DA(Data Augmentation)

TinyBERT知识蒸馏：损失函数

训练目标函数：

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

- 同时计算词嵌入层、Transformer层和预测层的损失
- 在Transformer层，同时计算隐状态的损失和注意力矩阵的损失



$$\mathcal{L}_{layer}(S_m, T_{g(m)}) = \begin{cases} \mathcal{L}_{embd}(S_0, T_0), & m = 0 \\ \mathcal{L}_{hidn}(S_m, T_{g(m)}) + \mathcal{L}_{attn}(S_m, T_{g(m)}), & M \geq m > 0 \\ \mathcal{L}_{pred}(S_{M+1}, T_{N+1}), & m = M+1 \end{cases}$$

TinyBERT知识蒸馏：数据增强

下游任务通常训练数据不足，我们采用数据增强方法生成更多的伪数据用于TD（面向任务的蒸馏）

- 对于下游任务的每一个训练样本，随机选择一些词语进行替换
- 替换时我们采用BERT和Glove，替换相近的词语
- 通过一个阈值控制被替换词语的比例
- 我们发现伪数据数量为原始下游任务训练数据20倍时蒸馏效果最好

[Mask][Mask][Mask][Mask]歌曲

[帮][我][搜][索]歌曲
[播][放][一][首]歌曲
[给][我][搜][索]歌曲
[给][我][播][放]歌曲
[给][我][放][首]歌曲
[给][我][唱][首]歌曲
[帮][我][播][放]歌曲

TinyBERT实验结果：GLUE

System	#Params	#FLOPS	Speedup	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg
$\text{BERT}_{\text{BASE}}$ (Teacher)	109M	22.5B	1.0x	83.9/83.4	71.1	90.9	93.4	52.8	85.2	87.5	67.0	79.5
$\text{BERT}_{\text{TINY}}$	14.5M	1.2B	9.4x	75.4/74.9	66.5	84.8	87.6	19.5	77.1	83.2	62.6	70.2
$\text{BERT}_{\text{SMALL}}$	29.2M	3.4B	5.7x	77.6/77.0	68.1	86.4	89.7	27.8	77.0	83.4	61.8	72.1
BERT_4-PKD	52.2M	7.6B	3.0x	79.9/79.3	70.2	85.1	89.4	24.8	79.8	82.6	62.3	72.6
DistilBERT_4	52.2M	7.6B	3.0x	78.9/78.0	68.5	85.2	91.4	32.8	76.1	82.4	54.1	71.9
$\text{MobileBERT}_{\text{TINY}}$†	15.1M	3.1B	-	81.5/81.6	68.9	**89.5**	91.7	**46.7**	80.1	**87.9**	65.1	**77.0**
TinyBERT_4 (ours)	14.5M	1.2B	9.4x	**82.5/81.8**	**71.3**	87.7	**92.6**	44.1	**80.4**	86.4	**66.6**	**77.0**
BERT_6-PKD	67.0M	11.3B	2.0x	81.5/81.0	70.7	89.0	92.0	-	-	85.0	65.5	-
DistilBERT_6	67.0M	11.3B	2.0x	82.6/81.3	70.1	88.9	92.5	49.0	81.3	86.9	58.4	76.8
TinyBERT_6 (ours)	67.0M	11.3B	2.0x	**84.6/83.2**	**71.6**	**90.4**	**93.1**	**51.1**	**83.7**	**87.3**	**70.0**	**79.4**

TinyBERT实际性能和应用情况

- 相比于BERT-base模型，4层TinyBERT模型加速9.4倍，参数量降为1/7，精度平均下降2.5%，优于现有的所有同量级的模型
- 6层TinyBERT模型精度接近BERT-base模型
- TinyBERT成为终端语音助手NLU算法核心，相对传统模型提高6-10%
- TinyBERT目前已成为预训练模型压缩的主流方案，被微软、谷歌、IBM等公司引用，多家媒体报道，被引用292次（up to 2021-07-30）
- 4层TinyBERT基于诺亚AI系统工程实验室开发的Bolt框架，对于常见的短文本（长度小于9个词）在2.5GHz的ARM A76单核float16推理时间仅有2ms, int8推理时间1.3ms ，模型存储只有10MB左右。

Tinybert: No.1 of the Most Influential EMNLP 2021 Papers

TABLE 1: Most Influential EMNLP Papers (2021-02)

YEAR	RANK	PAPER	AUTHOR(S)
2020	1	**TinyBERT: Distilling BERT For Natural Language Understanding** IF:4 Related Papers Related Patents Related Grants Related Orgs Related Experts Details *Highlight: To accelerate inference and reduce model size while maintaining accuracy, we first propose a novel Transformer distillation method that is specially designed for knowledge distillation (KD) of the Transformer-based models.*	XIAOQI JIAO et. al.

"Paper Digest Team analyze all papers published on EMNLP in the past years, and presents the 10 most influential papers for each year."

https://www.paperdigest.org/2021/02/most-influential-emnlp-papers/

Content

Layer Mapping Search for KD

Improving Task-Agnostic BERT Distillation with Layer Mapping Search

Xiaoqi Jiao[1]*, Huating Chang[2], Yichun Yin[3], Lifeng Shang[3]
Xin Jiang[3], Xiao Chen[3], Linlin Li[4], Fang Wang[1] and Qun Liu[3]
[1]Huazhong University of Science and Technology
[2]Zhejiang University
[3]Huawei Noah's Ark Lab
[4]Huawei Technologies Co., Ltd.

Accepted by Neurocomputing

Layer Mapping Search for KD

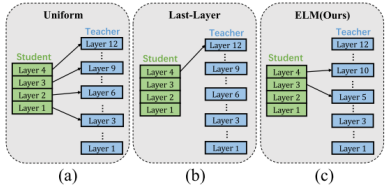


Figure 1: The diagram of different layer mapping strategies.

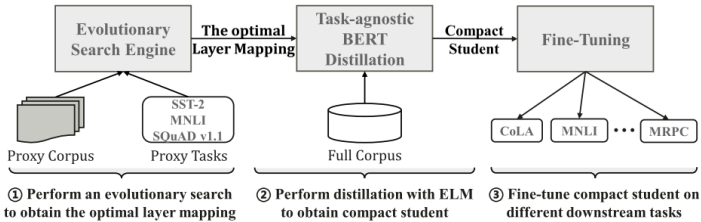


Figure 2: Overview of the layer mapping search for task-agnostic BERT distillation. We focus on the first stage to explore better layer mappings by the proposed approach under a proxy setting.

Layer Mapping Search for KD

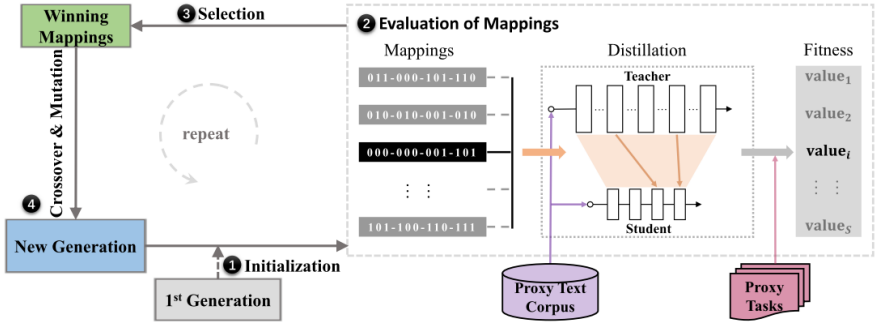


Figure 3: The overview of ELM (or the evolutionary search engine). The process includes four stages: 1) Start with a randomly initialized *generation*, the first generation, which consists of a set of *genes*. 2) Run the task-agnostic BERT distillation in parallel with different *genes* (layer mappings) on the proxy corpus to obtain corresponding students. 3) Pair each *gene* with a *fitness* value by fine-tuning the corresponding student on some representative proxy tasks. 4) Perform the genetic algorithm (GA) to select the *genes* and produce a new generation by the genetic operations *crossover* and *mutation*. By repeating stage 2 to 4, ELM would evolve better layer mappings automatically and find the optimal *gene*.

Layer Mapping Search for KD

Table 2: Comparison between student models with different layer mappings on the dev set. The fine-tuning results are averaged over 3 runs. The 4-layer student has an architecture of (M=4, d=312, d_i=1200, h=12), and the 6-layer student has an architecture of (M=6, d=384, d_i=1536, h=12).

Student	Strategy	Layer Mapping	SST-2 (acc)	MNLI (acc)	SQuAD v1.1 (EM/F1)	MRPC (acc/F1)	CoLA (mcc)	QNLI (acc)	QQP (acc/F1)	SQuAD v2.0 (EM/F1)	Avg
4-Layer	Uniform	(3,6,9,12)	87.4	77.0	66.7/77.4	76.5/84.6	21.3	84.9	86.0/81.7	58.9/62.5	72.1
	Last-Layer	(0,0,0,12)	88.1	77.6	69.2/79.4	83.8/88.6	21.4	85.7	87.2/82.9	59.4/63.3	73.4
	Contribution	(1,10,11,12)	86.8	76.1	64.4/76.4	79.4/86.3	15.5	85.8	86.1/81.4	61.6/65.1	71.7
	ELM (ours)	(0,0,5,10)	**89.9**	**78.6**	**71.5/81.2**	**85.0/89.5**	**23.9**	**86.0**	**87.9/83.6**	**62.9/66.2**	**74.9**
6-Layer	Uniform	(2,4,6,8,10,12)	90.7	81.2	76.0/84.6	85.0/89.6	27.2	89.2	88.2/84.1	68.0/71.3	77.2
	Last-Layer	(0,0,0,0,0,12)	89.8	81.3	76.0/84.7	85.5/89.7	34.3	89.0	88.7/84.6	68.7/71.9	78.2
	Contribution	(1,6,7,10,11,12)	90.0	80.9	75.0/84.1	84.6/89.3	28.5	88.8	88.0/84.2	66.5/70.0	77.0
	ELM (ours)	(0,5,0,0,0,10)	**91.5**	**82.4**	**77.2/85.7**	**86.0/90.1**	**36.1**	**89.3**	**89.2/85.4**	**70.3/73.2**	**79.2**

Content

CKD: Combination of Layers

Why Skip If You Can Combine: A Simple Knowledge Distillation Technique for Intermediate Layers

Yimeng Wu* **Peyman Passban*** **Mehdi Rezagholizadeh** **Qun Liu**
Huawei Noah's Ark Lab
firstname.lastname@huawei.com

Published in EMNLP 2020 Proceedings (Short paper)

CKD: Combination of Layers

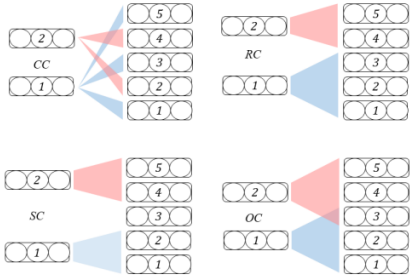


Figure 2: Different variations of *CKD*. $\mathcal{T}$ has 5 and $\mathcal{S}$ has 2 hidden layers. For the *CC* case $M(1) = \{1, 3, 5\}$.

Content

ALP-KD

ALP-KD: Attention-Based Layer Projection for Knowledge Distillation

Peyman Passban* **Yimeng Wu** **Mehdi Rezagholizadeh** **Qun Liu**
Huawei Noah's Ark Lab
passban.peyman@gmail.com
{yimeng.wu,mehdi.rezagholizadeh,qun.liu}@huawei.com

Published in AAAI 2021

ALP-KD

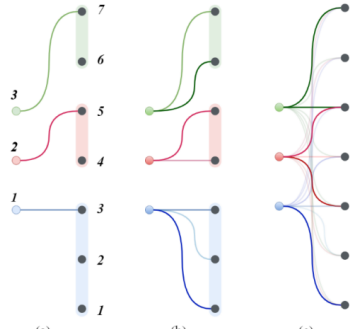


Figure 2: Three pairs of $\mathcal{S}$ and $\mathcal{T}$ networks with different forms of layer connections. In Figure 2a, teacher layers are divided into 3 buckets and only one layer from each bucket is connected to the student side, e.g. $h_{\mathcal{T}}^5$ is the source of distillation for $h_{\mathcal{S}}^2$ ($h_{\mathcal{T}}^5 \leftrightarrow h_{\mathcal{S}}^2$). In Figure 2b, a weighted average of teacher layers from each bucket is considered for distillation, e.g. $\mathcal{A}(2) = \{h_{\mathcal{T}}^4, h_{\mathcal{T}}^5\}$ and $\mathcal{C}^2 = \alpha_{24} h_{\mathcal{T}}^4 + \alpha_{25} h_{\mathcal{T}}^5$ ($\mathcal{C}^2 \leftrightarrow h_{\mathcal{S}}^2$). In Figure 2c, there is no bucketing and all teacher layers are considered for projection. Links with higher color intensities have higher attention weights.

ALP-KD

Problem	Model	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	Average
N/A	$\mathcal{T}_{\text{BERT}}$	57.31	83.39	86.76	91.25	90.96	68.23	92.67	88.82	82.42
N/A	$\mathcal{S}_{\text{NKD}}$	31.05	76.83	77.70	85.13	88.97	61.73	88.19	87.29	74.61
skip, search	$\mathcal{S}_{\text{RKD}}$	29.22	79.31	79.41	86.77	90.25	65.34	90.37	87.45	76.02
skip, search	$\mathcal{S}_{\text{PKD}}$	32.13	79.26	80.15	86.64	90.23	65.70	90.14	87.26	76.44
search	$\mathcal{S}_{\text{CKD-NO}}$	31.23	79.42	80.64	86.93	88.70	66.06	90.37	87.62	76.37
search	$\mathcal{S}_{\text{CKD-PO}}$	31.95	79.53	80.39	86.75	89.89	**67.51**	90.25	87.55	76.73
search	$\mathcal{S}_{\text{ALP-NO}}$	**34.21**	79.26	79.66	**87.11**	**90.72**	65.70	90.37	87.52	76.82
search	$\mathcal{S}_{\text{ALP-PO}}$	33.86	**79.74**	79.90	86.95	90.25	66.43	**90.48**	87.52	76.89
none	$\mathcal{S}_{\text{ALP}}$	33.07	79.62	**80.72**	87.02	90.54	67.15	90.37	**87.62**	**77.01**

Table 1: Except the teacher ($\mathcal{T}_{\text{BERT}}$) which is a 12-layer model, all other models have 4 layers. Apart from the number of layers, all students have the same architecture as the teacher. The first column shows what sort of problems each model suffers from. NKD stands for *No KD* which means there is no KD technique involved during training this student model. *NO* and *PO* are different configurations for mapping internal layers. Boldfaced numbers show the best student score for each column over the validation set. Scores in the first column are Matthew's Correlations. SST-B scores are Pearson correlations and the rest are accuracy scores.

Content

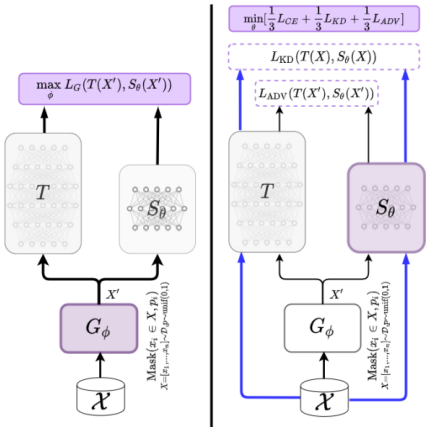
Mate-KD: Adversarial Data Augmentation for KD

MATE-KD: Masked Adversarial TExt, a Companion to Knowledge Distillation

Ahmad Rashid[1]*, Vasileios Lioutas[2]*†, Mehdi Rezagholizadeh[1]
[1]Huawei Noah's Ark Lab, [2]University of British Columbia
ahmad.rashid@huawei.com, contact@vlioutas.com,
mehdi.rezagholizadeh@huawei.com

Accepted by ACL 2021 Proceedings （Long paper)

Mate-KD: Adversarial Data Augmentation for KD



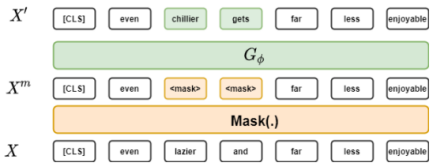
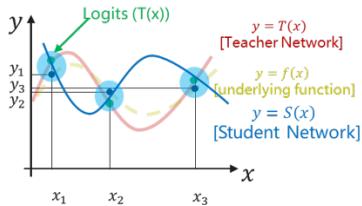


Figure 2: This figure illustrates how a training sample will be randomly masked and then fed to the text generator G_ϕ to get the pseudo training sample.



Mate-KD: Adversarial Data Augmentation for KD

Method	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	Score
$\text{RoBERTa}_{\text{Large}}$ (teacher)	68.1	96.4	91.9	92.3	91.5	90.2	94.6	86.3	85.28
DistilRoBERTa (student)	56.6	92.7	89.5	87.2	90.8	84.1	91.3	65.7	78.78
Student + FreeLB	58.1	93.1	90.1	88.8	90.9	84.0	91.0	67.8	80.01
Student + FreeLB + KD	58.1	93.2	90.5	88.6	91.2	83.7	90.8	68.2	80.06
Student + KD	60.9	92.5	90.2	89.0	91.6	84.1	91.3	71.1	80.77
Student + TinyBERT Aug + KD	61.3	93.3	90.4	88.6	91.7	84.4	91.6	72.5	81.12
Student + MATE-KD (Ours)	65.9	94.1	91.9	90.4	91.9	85.8	92.5	75.0	**82.64**

Table 1: Dev Set results for the GLUE benchmark. The score for the WNLI task is 56.3 for all models.

Original	**Generated**
the new insomnia is a surprisingly faithful remake of its chilly predecessor, and	sinister new insomnia shows a surprisingly terrible remake of its hilarious predecessor, and
beautifully shot, delicately scored and powered by a set of heartfelt performances	beautifully sublime, delicately scored, powered by great dozens of heartfelt performances
a perfectly pleasant if slightly pokey comedy	a 10 pleasant if slightly pokey comedy
that appeals to me	Federal appeals punished me
good news to anyone who's fallen under the sweet, melancholy spell of this unique director's previous films	good news for anyone who's fallen under the sweet, melancholy spell of this unique director's previous mistakes

Table 6: Examples of original and adversarially generated samples during training for the SST-2 dataset

Content

Minimax-*k*NN-DA: Sample Efficient Retrieval for Data Aug.

Not Far Away, Not So Close: Sample Efficient Nearest Neighbour Data Augmentation via MiniMax

Ehsan Kamalloo[*†◊] **Mehdi Rezagholizadeh**[*§] **Peyman Passban**[†§] **Ali Ghodsi**[‡¶]
[◊]Department of Computing Science, University of Alberta
[§]Huawei Noah's Ark Lab
[‡]David R. Cheriton School of Computer Science, Univeristy of Waterloo
[¶]Department of Statistics and Actuarial Science, Univeristy of Waterloo

Accepted by ACL 2021 Findings （Long paper）

Minimax-kNN-DA: Sample Efficient Retrieval for Data Aug.

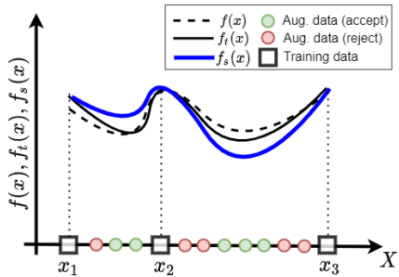


Figure 1: Data sparsity problem in KD; f, f_t, and f_s are representing the underlying function, teacher, and student outputs respectively. We show 10 augmented samples around x_2 with small circles on the X-axis. The green circles show the augmented samples which are selected by our MiniMax-kNN because these points correspond to maximum divergence regions of the teacher and student networks. The red circles are rejected augmented samples.

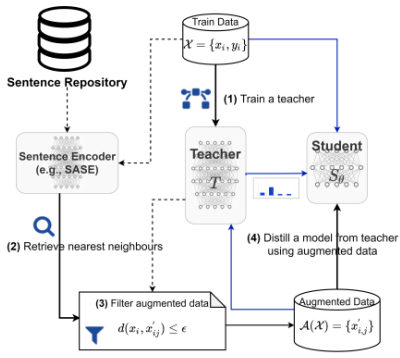


Figure 2: A schematic view of MiniMax-kNN

Minimax-*k*NN-DA: Sample Efficient Retrieval for Data Aug.

Model	SST-5	SST-2	TREC	CR	IMP
$\text{RoBERTa}_{\text{Large}}$ (Teacher)	57.6	96.2	98.0	94.1	90.0
DistilRoBERTa	52.9	93.5	96.0	**92.1**	86.8
DistilRoBERTa + KD	53.2	93.6	96.6	**92.1**	87.7
DistilRoBERTa + vanilla-8NN	55.2	94.7	97.0	91.3	88.4
AUG. SIZE (#forward / #backward pass)	8x / 8x	8x / 8x	8x / 8x	8x / 8x	8x / 8x
DistilRoBERTa + MiniMax-8NN*	**55.4**	**95.2**	**97.6**	91.6	**88.6**
AUG. SIZE (#forward / #backward pass)	5x / 4x	7x / 4x	8x / 4x	8x / 2x	8x / 1x

Table 2: Test accuracy (↑) on the downstream tasks (*denotes our approach and **bold** numbers indicate the best result—excluding the teacher—for each task).

Content

Annealing KD

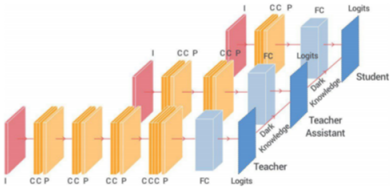
Annealing Knowledge Distillation

[1,2]Aref Jafari, [2]Mehdi Rezagholizadeh, [1]Pranav Sharma, [1,3]Ali Ghodsi

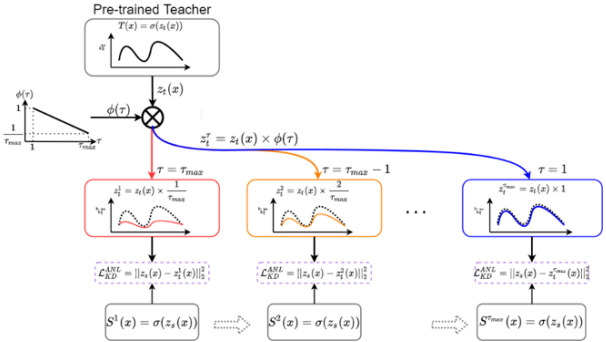
[1] David R. Cheriton School of Computer Science, University of Waterloo
[2]Huawei Noah's Ark Lab
[3]Department of Statistics and Actuarial Science, University of Waterloo

Published in EACL 2021

Annealing KD



Teacher Assistant KD



Annealing KD

Image is taken from: https://arxiv.org/pdf/1902.03393.pdf

Annealing KD

$$\mathcal{L} = (1-\lambda)\mathcal{L}_{CE} + \lambda\mathcal{L}_{KD}$$
$$\mathcal{L}_{CE} = H_{CE}\Big(y, (\sigma(z_s(x))\Big)$$
$$\mathcal{L}_{KD} = \mathcal{T}^2 KL\Big(\sigma(\frac{z_t(x)}{\mathcal{T}}), \sigma(\frac{z_s(x)}{\mathcal{T}})\Big)$$

Loss for Regular KD

$$\mathcal{L} = \begin{cases} \mathcal{L}_{\text{KD}}^{\text{Annealing}}(i), & \textbf{Stage I: } 1 \le \mathcal{T}_i \le \tau_{\max} \\ \mathcal{L}_{CE}, & \textbf{Stage II: } \mathcal{T}_n = 1 \end{cases}$$
$$\mathcal{L}_{\text{KD}}^{\text{Annealing}}(i) = ||z_s(x) - z_t(x) \times \Phi(\mathcal{T}_i)||_2^2$$
$$\Phi(\mathcal{T}) = 1 - \frac{\mathcal{T}-1}{\tau_{\max}}, 1 \le \mathcal{T} \le \tau_{\max}, \mathcal{T} \in \mathbb{N}$$

Loss for Annealing KD

Annealing KD

Table 3: DistilRoBERTa results for Annealing KD on dev set. F1 scores are reported for MRPC, pearson correlations for STB-B, and accuracy scores for all other tasks.

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI	Score
Teacher	68.1	86.3	91.9	92.3	96.4	94.6	91.5	90.22/89.87	56.33	85.29
From scratch	59.3	67.9	88.6	88.5	92.5	90.8	90.9	84/84	52.1	79.3
Vanilla KD	60.97	71.11	90.2	88.86	92.54	91.37	91.64	84.18/84.11	56.33	80.8
TAKD	61.15	71.84	89.91	88.94	92.54	91.32	**91.7**	83.89/84.18	56.33	80.85
Annealing KD	**61.67**	**73.64**	**90.6**	**89.01**	**93.11**	**91.64**	91.5	**85.34/84.6**	56.33	**81.42**

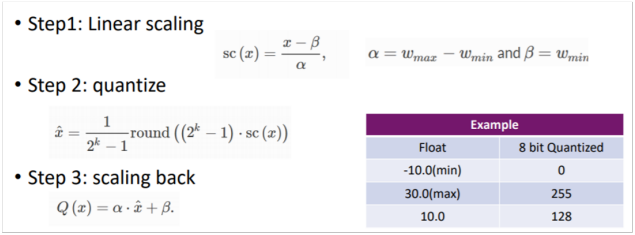
Table 4: Performance of DistilRoBERTa trained by annealing KD on the GLUE leaderboard compared with Vanilla KD and TAKD. We applied the standard tricks to all 3 methods and fine-tune RTE, MRPC and STS-B from trained MNLI student model.

KD Method	CoLA	MRPC	STS-B	SST-2	MNLI-m	MNLI-mm	QNLI	QQP	RTE	WNLI	Score
Vanilla KD	**54.3**	86/80.8	85.7/84.9	93.1	83.6	82.9	90.8	71.9/89.5	74.1	65.1	78.9
TAKD	53.2	86.7/82.7	85.6/84.4	93.2	83.8	83.2	**91**	72/89.4	**74.2**	65.1	79
Annealing KD	54	**88.0/83.9**	**87.0/86.6**	**93.6**	**83.8**	**83.9**	90.8	**72.6/89.7**	73.7	65.1	**79.5**

Content

预训练语言模型压缩：基于量化的技术

- 定义：减少数值表示所需要的比特数，Floating point representation -> Fixed point representation



- 代表性工作
  - Q8BERT [Zafrir elt al., 2019] :参数矩阵\词向量\activation均采用8bit；
  - QBERT [Shen et al., 2019] : 参数矩阵选择2或3bit; 词向量\activation采用8bit；
  - TernaryBERT [Zhang et al., 2020]: 参数矩阵和词向量采用2bit，activation采用8bit；

Content

TernaryBERT

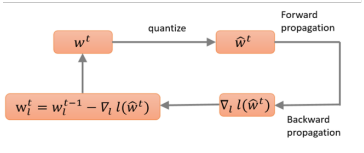
TernaryBERT: Distillation-aware Ultra-low Bit BERT

Wei Zhang[*], Lu Hou[*], Yichun Yin[*], Lifeng Shang, Xiao Chen, Xin Jiang, Qun Liu
Huawei Noah's Ark Lab

Published in EMNLP 2020 Proceedings (Long paper)

TernaryBERT：三值量化的预训练语言模型

将蒸馏与极低比特（1/2-bit）量化技术结合



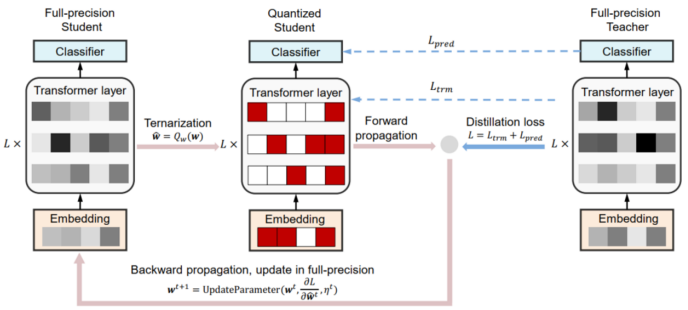


Figure 2: Depiction of the proposed distillation-aware ternatization of BERT model.

TernaryBERT：实验结果

- 模型权重W：2-bit量化，即-1,0,1
- 词向量矩阵E：2-bit量化，即-1,0,1
- 激活值A：8-bit量化

Table 1: Development set results of quantized BERT and TinyBERT on the GLUE benchmark. We abbreviate the quantization bits for weights of Transformer layers as "W-bit", word embedding as "E-bit", activations as "A-bit".

	W-E-A (#bits)	Size (MB)	MNLI-m/mm	QQP	QNLI	SST-2	CoLA	MRPC	STS-B	RTE
BERT	32-32-32	418 (×1)	84.5/84.9	87.5/90.9	92.0	93.1	58.1	90.6/86.5	89.8/89.4	71.1
Q-BERT	2-8-8	43 (×9.7)	76.6/77.0	-	-	84.6	-	-	-	-
Q2BERT	2-8-8	43 (×9.7)	47.2/47.3	67.0/75.9	61.3	80.6	0	81.2/68.4	4.4/4.7	52.7
TernaryBERT (TWN)	2-2-8	28 (×14.9)	83.3/83.3	86.7/90.1	91.1	92.8	**55.7**	91.2/87.5	**87.9/87.7**	**72.9**
TernaryBERT (LAT)	2-2-8	28 (×14.9)	**83.5/83.4**	86.6/90.1	**91.5**	92.5	54.3	91.1/87.0	87.9/87.6	72.2
TernaryTinyBERT (TWN)	2-2-8	18 (×23.2)	83.4/83.8	**87.2/90.5**	89.9	**93.0**	53.0	**91.5/88.0**	86.9/86.5	71.8
Q-BERT	8-8-8	106 (×3.9)	83.9/83.8	-	-	92.9	-	-	-	-
Q8BERT	8-8-8	106 (×3.9)	-/-	**88.0/-**	90.6	92.2	58.5	89.6/-	89.0/-	68.8
Ours (BERT)	8-8-8	106 (×3.9)	84.2/84.7	87.1/90.5	**91.8**	**93.7**	**60.6**	**90.8/87.3**	**89.7/89.3**	71.8
Ours (TinyBERT)	8-8-8	65 (×6.4)	**84.4/84.6**	87.9/91.0	91.0	93.3	54.7	90.0/89.4	91.2/87.5	**72.2**

TernaryBERT：精度与模型大小的平衡

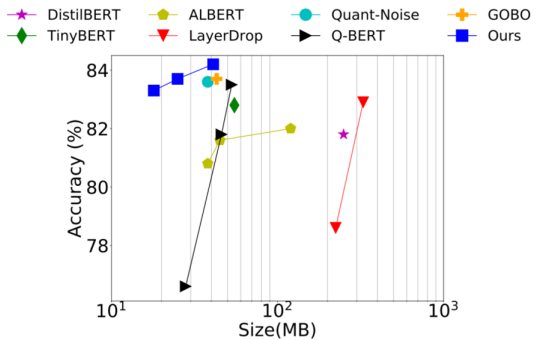


Figure 1: Model Size vs. MNLI-m Accuracy. Our proposed method outperforms other BERT compression methods. Details are in Section 4.4.

Method	W-E-A (#bits)	Size (MB)	Accuracy (%)
DistilBERT	32-32-32	250	81.6
TinyBERT	32-32-32	55	82.8
ALBERT-E64	32-32-32	38	80.8
ALBERT-E128	32-32-32	45	81.6
ALBERT-E256	32-32-32	62	81.5
ALBERT-E768	32-32-32	120	82.0
LayerDrop-6L	32-32-32	328	82.9
LayerDrop-3L	32-32-32	224	78.6
Quant-Noise	PQ	38	83.6
Q-BERT	2/4-8-8	53	83.5
Q-BERT	2/3-8-8	46	81.8
Q-BERT	2-8-8	28	76.6
GOBO	3-4-32	43	83.7
GOBO	2-2-32	28	71.0
3-bit BERT	3-3-8	41	84.2
3-bit TinyBERT	3-3-8	25	83.7
TernaryBERT	2-2-8	28	83.5
TernaryTinyBERT	2-2-8	18	83.4

Content

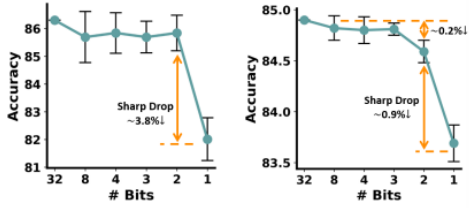
BinaryBERT

BinaryBERT: Pushing the Limit of BERT Quantization

Haoli Bai[1], **Wei Zhang**[2], **Lu Hou**[2], **Lifeng Shang**[2],
Jing Jin[3], **Xin Jiang**[2], **Qun Liu**[2], **Michael Lyu**[1], **Irwin King**[1]
[1] The Chinese University of Hong Kong
[2]Huawei Noah's Ark Lab, [3]Huawei Technologies Co., Ltd.
{hlbai, lyu, king}@cse.cuhk.edu.hk

Accepted by ACL 2021 Proceedings （Long paper)

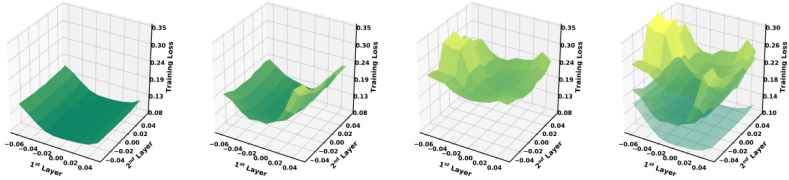
BinaryBERT



(a) MRPC. (b) MNLI-m.

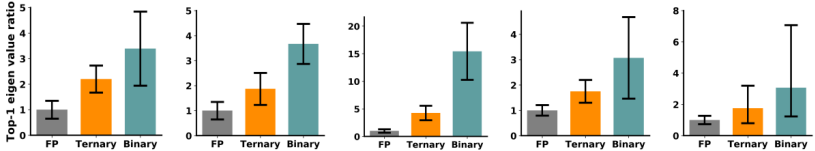
Figure 1: Performance of quantized BERT with varying weight bit-widths and 8-bit activation. We report the mean results with standard deviations from 10 seeds on MRPC and 3 seeds on MNLI-m, respectively.

BinaryBERT



(a) Full-precision Model. (b) Ternary Model. (c) Binary Model. (d) All Together.

Figure 2: Loss landscapes visualization of the full-precision, ternary and binary models on MRPC. For (a), (b) and (c), we perturb the (latent) full-precision weights of the value layer in the 1st and 2nd Transformer layers, and compute their corresponding training loss. (d) shows the gap among the three surfaces by stacking them together.



(a) MHA-QK. (b) MHA-V. (c) MHA-O. (d) FFN-Mid. (e) FFN-Out.

Figure 3: The top-1 eigenvalues of parameters at different Transformer parts of the full-precision (FP), ternary and binary BERT. For easy comparison, we report the ratio of eigenvalue between the ternary/binary models and the full-precision model. The error bar is estimated of all Transformer layers over different data mini-batches.

BinaryBERT

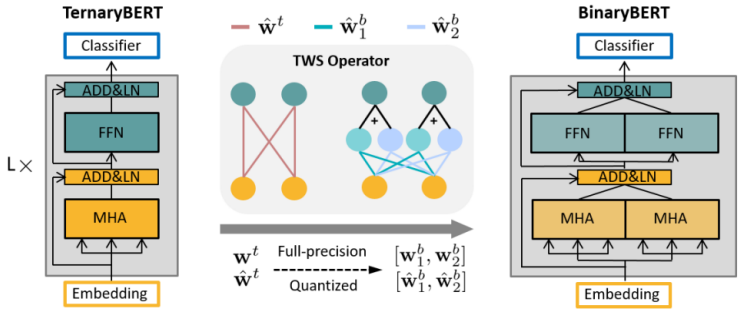


Figure 4: The overall workflow of training BinaryBERT. We first train a half-sized ternary BERT model, and then apply ternary weight splitting operator (Equations (6) and (7)) to obtain the latent full-precision and quantized weights as the initialization of the full-sized BinaryBERT. We then fine-tune BinaryBERT for further refinement.

BinaryBERT

#	Quant	#Bits (W-E-A)	Size (MB)	FLOPs (G)	DA	MNLI -m/mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
1	-	*full-prec.*	417.6	22.5	-	84.9/85.5	91.4	92.1	93.2	59.7	90.1	86.3	72.2	83.9
2	BWN	1-1-8	13.4	3.1	✗	84.2/84.0	91.1	90.7	92.3	46.7	86.8	82.6	68.6	80.8
3	TWS	1-1-8	16.5	3.1	✗	**84.2/84.7**	**91.2**	**91.5**	**92.6**	**53.4**	**88.6**	**85.5**	**72.2**	**82.7**
4	BWN	1-1-4	13.4	1.5	✗	83.5/83.4	90.9	90.7	**92.3**	34.8	84.9	79.9	**65.3**	78.4
5	TWS	1-1-4	16.5	1.5	✗	**83.9/84.2**	**91.2**	**90.9**	**92.3**	**44.4**	**87.2**	**83.3**	**65.3**	**79.9**
6	BWN	1-1-8	13.4	3.1	✓	84.2/84.0	91.1	91.2	92.7	54.2	88.2	**86.8**	70.0	82.5
7	TWS	1-1-8	16.5	3.1	✓	**84.2/84.7**	**91.2**	**91.6**	**93.2**	**55.5**	**89.2**	86.0	**74.0**	**83.3**
8	BWN	1-1-4	13.4	1.5	✓	83.5/83.4	90.9	91.2	92.5	51.9	87.7	85.5	70.4	81.9
9	TWS	1-1-4	16.5	1.5	✓	**83.9/84.2**	**91.2**	**91.4**	**93.7**	**53.3**	**88.6**	**86.0**	**71.5**	**82.6**

Table 1: Results on the GLUE development set. "#Bits (W-E-A)" represents the bit number for weights of Transformer layers, word embedding, and activations. "DA" is short for data augmentation. "Avg." denotes the average results of all tasks including MNLI-m and MNLI-mm. The higher results in each block are bolded.

BWN: vanilla binary training

#	Quant	#Bits (W-E-A)	Size (MB)	FLOPs (G)	DA	MNLI -m/mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
1	-	*full-prec.*	417.6	22.5	-	84.5/84.1	89.5	91.3	93.0	54.9	84.4	87.9	69.9	82.2
2	BWN	1-1-8	13.4	3.1	✗	83.3/83.4	88.9	**90.1**	92.3	38.1	81.2	**86.1**	63.1	78.5
3	TWS	1-1-8	16.5	3.1	✗	**84.1/83.6**	**89.0**	90.0	**93.1**	**50.5**	**83.4**	86.0	**65.8**	**80.6**
4	BWN	1-1-4	13.4	1.5	✗	83.5/82.5	**89.0**	**89.4**	92.3	26.7	78.9	84.2	59.9	76.3
5	TWS	1-1-4	16.5	1.5	✗	**83.6/82.9**	**89.0**	89.3	**93.1**	**37.4**	**82.5**	**85.9**	**62.7**	**78.5**
6	BWN	1-1-8	13.4	3.1	✓	83.3/83.4	88.9	**90.3**	91.3	48.4	**83.2**	**86.3**	66.1	80.1
7	TWS	1-1-8	16.5	3.1	✓	**84.1/83.5**	**89.0**	89.8	**91.9**	**51.6**	82.3	85.9	**67.3**	**80.6**
8	BWN	1-1-4	13.4	1.5	✓	83.5/82.5	**89.0**	**89.9**	92.0	45.0	81.9	85.2	64.1	79.2
9	TWS	1-1-4	16.5	1.5	✓	**83.6/82.9**	**89.0**	89.7	**93.1**	**47.9**	**82.9**	**86.6**	**65.8**	**80.2**

TWS: ternary weight splitting

Table 2: Results on the GLUE test set scored using the GLUE evaluation server.

BinaryBERT

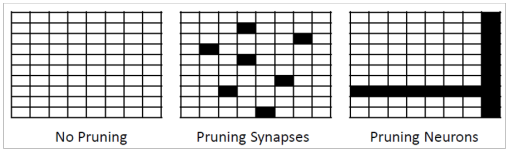
Method	#Bits (W-E-A)	Size (MB)	Ratio (↓)	SQuAD v1.1	MNLI -m
BERT-base	*full-prec.*	418	1.0	80.8/88.5	84.6
DistilBERT	*full-prec.*	250	1.7	79.1/86.9	81.6
LayerDrop-6L	*full-prec.*	328	1.3	-	82.9
LayerDrop-3L	*full-prec.*	224	1.9	-	78.6
TinyBERT-6L	*full-prec.*	55	7.6	79.7/87.5	82.8
ALBERT-E128	*full-prec.*	45	9.3	82.3/89.3	81.6
ALBERT-E768	*full-prec.*	120	3.5	81.5/88.6	82.0
Quant-Noise	PQ	38	11.0	-	83.6
Q-BERT	2/4-8-8	53	7.9	79.9/87.5	83.5
Q-BERT	2/3-8-8	46	9.1	79.3/87.0	81.8
Q-BERT	2-8-8	28	15.0	69.7/79.6	76.6
GOBO	3-4-32	43	9.7	-	83.7
GOBO	2-2-32	28	15.0	-	71.0
TernaryBERT	2-2-8	28	15.0	79.9/87.4	83.5
BinaryBERT	**1-1-8**	**17**	**24.6**	**80.8/88.3**	**84.2**
BinaryBERT	**1-1-4**	**17**	**24.6**	**79.3/87.2**	**83.9**

Table 4: Comparison with other state-of-the-art methods on development set of SQuAD v1.1 and MNLI-m.

Content

预训练语言模型压缩：基于剪枝的技术

- 定义：基于一定的准则（比如绝对值/重要性排序），去掉参数矩阵中冗余的部分；分为结构化和非结构化剪枝

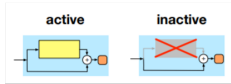


[Gupta and Agrawal, 2020]

- 代表性工作
  - 面向Attention Head的剪枝
    - [Michel et al., 2019]
  - 面向层的剪枝
    - LayerDrop [Fan et al., 2019]
    - Poor Man's BERT [Sajjad et al., 2020]
  - 其他：
    - 非结构化剪枝Compressing BERT [Gordon et al., 2020] .

$$I_h = \mathbb{E}_{x \sim X} \left| \mathrm{Att}_h(x)^T \frac{\partial \mathcal{L}(x)}{\partial \mathrm{Att}_h(x)} \right|$$

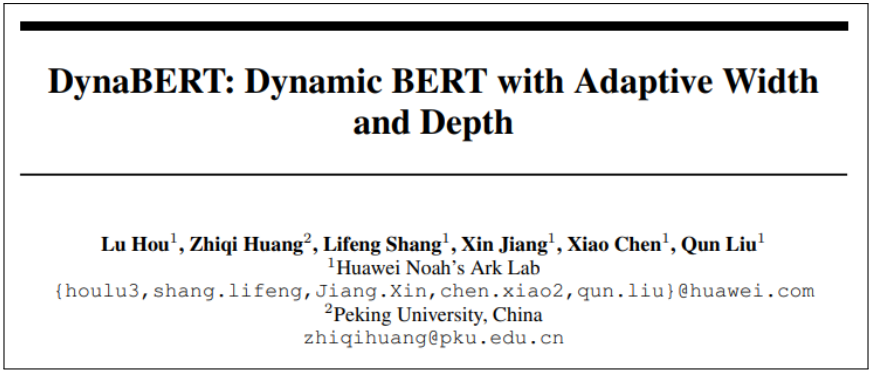
Head Importance Score
[Michel et al., 2019]



Random Structured Dropout
in LayerDrop [Fan et al., 2019]

Content

DynaBERT



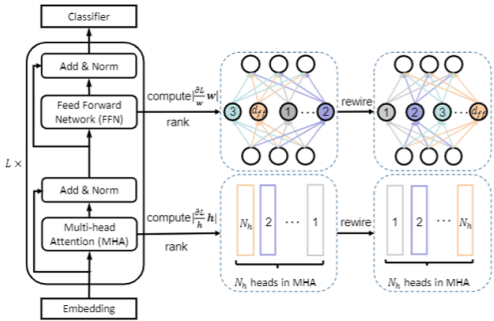
DynaBERT: Dynamic BERT with Adaptive Width and Depth

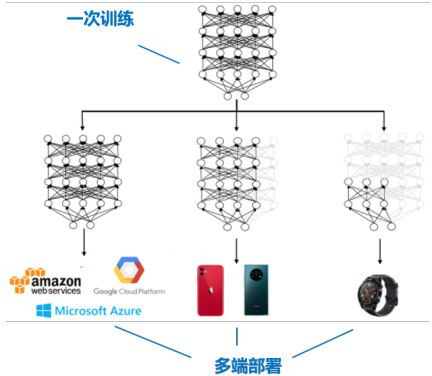
Lu Hou[1], **Zhiqi Huang**[2], **Lifeng Shang**[1], **Xin Jiang**[1], **Xiao Chen**[1], **Qun Liu**[1]
[1]Huawei Noah's Ark Lab
{houlu3,shang.lifeng,Jiang.Xin,chen.xiao2,qun.liu}@huawei.com
[2]Peking University, China
zhiqihuang@pku.edu.cn

Published in NeurIPS 2020 （Spotlight paper)

DynaBERT：动态可伸缩的预训练语言模型

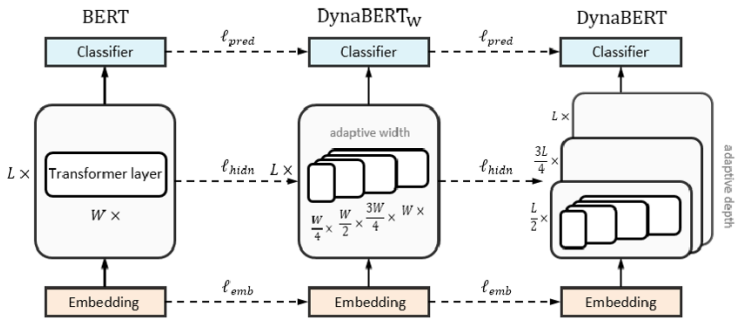
- 一次训练、多设备/多场景部署
- 部署时可以灵活选择不同的子网络进行推理
- 针对Transformer模型，定义网络的深度和宽度，根据神经元/头的重要性进行排序，使得重要的神经元/头被更多子网络共享





DynaBERT训练

- 先进行宽度可伸缩网络的训练，再通过宽度和深度同时可伸缩网络的训练。
- 借鉴TinyBERT的Transformer蒸馏技术。

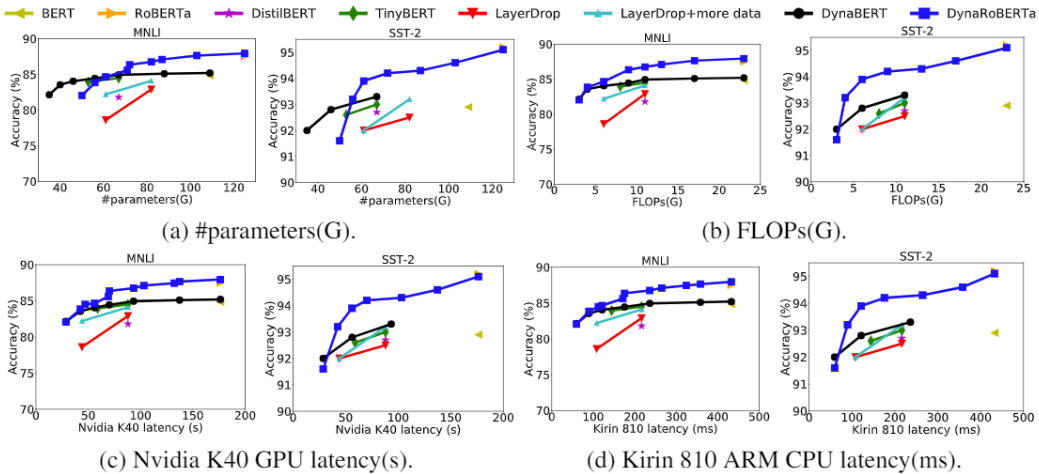


DynaBERT实验结果

Table 1: Development set results of the GLUE benchmark using DynaBERT and DynaRoBERTa with different width and depth multipliers (m_w, m_d).

Method		CoLA			STS-B			MRPC			RTE		
$\text{BERT}_{\text{BASE}}$		58.1			89.8			87.7			71.1		
	(m_w, m_d)	1.0x	0.75x	0.5x	1.0x	0.75x	0.5x	1.0x	0.75x	0.5x	1.0x	0.75x	0.5x
DynaBERT	1.0x	59.7	59.1	54.6	**90.1**	89.5	88.6	86.3	85.8	85.0	72.2	71.8	66.1
	0.75x	**60.8**	59.6	53.2	90.0	89.4	88.5	**86.5**	85.5	84.1	71.8	**73.3**	65.7
	0.5x	58.4	56.8	48.5	89.8	89.2	88.2	84.8	84.1	83.1	72.2	72.2	67.9
	0.25x	50.9	51.6	43.7	89.2	88.3	87.0	83.8	83.8	81.4	68.6	68.6	63.2
		MNLI-(m/mm)			QQP			QNLI			SST-2		
$\text{BERT}_{\text{BASE}}$		84.8/84.9			90.9			92.0			92.9		
	(m_w, m_d)	1.0x	0.75x	0.5x	1.0x	0.75x	0.5x	1.0x	0.75x	0.5x	1.0x	0.75x	0.5x
DynaBERT	1.0x	**84.9/85.5**	84.4/85.1	83.7/84.6	**91.4**	**91.4**	91.1	92.1	91.7	90.6	93.2	93.3	92.7
	0.75x	84.7/85.5	84.3/85.2	83.6/84.4	**91.4**	91.3	91.2	92.2	91.8	90.7	93.0	93.1	92.8
	0.5x	84.7/85.2	84.2/84.7	83.0/83.6	91.3	91.2	91.0	**92.2**	91.5	90.0	**93.3**	92.7	91.6
	0.25x	83.9/84.2	83.4/83.7	82.0/82.3	90.7	91.1	90.4	91.5	90.8	88.5	92.8	92.0	92.0

DynaBERT：精度与响应速度的平衡



(a) #parameters(G). (b) FLOPs(G).

(c) Nvidia K40 GPU latency(s). (d) Kirin 810 ARM CPU latency(ms).

DynaBERT：注意力矩阵可视化

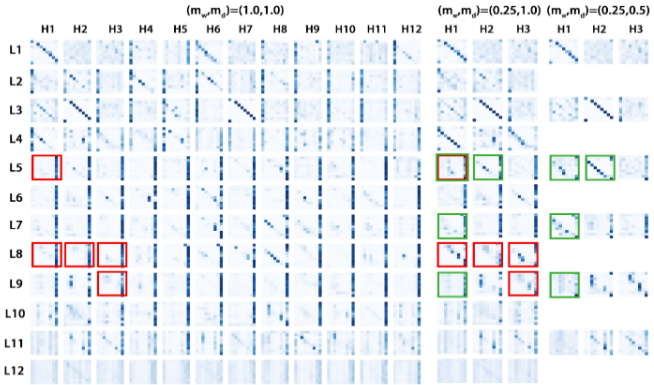


Figure 5: Attention maps in sub-networks with different widths and depths in DynaBERT trained on CoLA.

通过观察我们发现，小模型的注意力模式有可能融合了大模型的多个注意力模式。

Content

预训练语言模型压缩：其他技术

- 矩阵参数分解
  - ALBERT [Lan et al., 2020]
  - Tensorized Transformer [Ma et al., 2019]
- 参数共享
  - ALBERT [Lan et al., 2020]
  - Universal Transformer [Dehghani et al., 2018]
- 模型结构设计与搜索
  - MobileBERT [sun et al., 2021]
  - Transformer Lite [Wu et al., 2020]
  - AdaBERT [Chen et al., 2020]

Content

AutoTinyBERT

AutoTinyBERT: Automatic Hyper-parameter Optimization for Efficient Pre-trained Language Models

Yichun Yin[1], Cheng Chen[2*], Lifeng Shang[1], Xin Jiang[1], Xiao Chen[1], Qun Liu[1]
[1]Huawei Noah's Ark Lab
[2]Department of Computer Science and Technology, Tsinghua University
{yinyichun,shang.lifeng,jiang.xin,chen.xiao2,qun.liu}@huawei.com
c-chen19@mails.tsinghua.edu.cn

Accepted by ACL 2021 Proceedings （Long paper)

Content

GhostBERT

GhostBERT: Generate More Features with Cheap Operations for BERT

Zhiqi Huang[1], **Lu Hou**[2], **Lifeng Shang**[2], **Xin Jiang**[2], **Xiao Chen**[2], **Qun Liu**[2]
[1]Peking University, [2]Huawei Noah's Ark Lab
zhiqihuang@pku.edu.cn, {houlu3, shang.lifeng, jiang.xin, chen.xiao, qun.liu}@huawei.com

Accepted by ACL 2021 Proceedings （Long paper)

(slides made by Zhiqi Huang)

GhostBERT

- Redundant features (**feature maps, attention pattern**) are similar

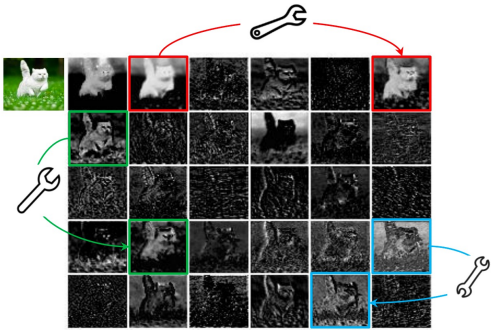


Figure1: Feature maps are similar

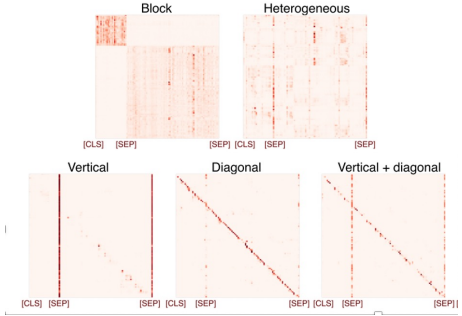


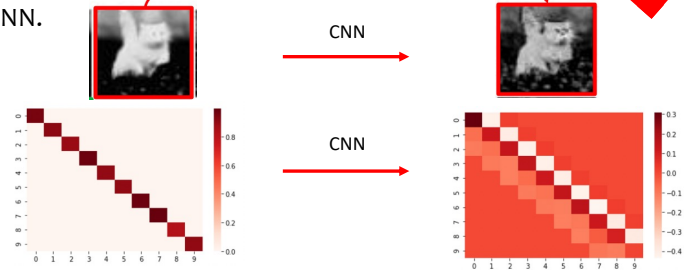
Figure2: Attention patterns in BERT

[1] Han et al., "GhostNet: More Features From Cheap Operations" CVPR 2021.
[2] Kovaleva et al., "Revealing the Dark Secrets of BERT." EMNLP 2019.

(slides made by Zhiqi Huang)

GhostBERT

- Can we directly discarded the redundant features?
  - The theory behind why more features help can be related to how over-parameterized neural networks benefit both training [1] and generalization [2].
- Can we use other operations to generate redundant features?
  - For example, CNN.

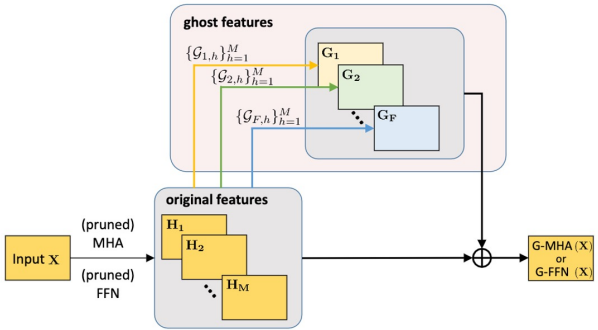


[3] Samet et al., "Overparameterized nonlinear learning: Gradient descent takes the shortest path?" *ICML* 2019.

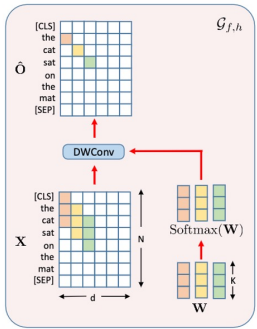
[4] Yuan et al., "Generalization bounds of stochastic gradient descent for wide and deep neural networks." *arXiv:1905.13210*, 2019.

(slides made by Zhiqi Huang)

GhostBERT



(a) Adding ghost modules $\{\mathcal{G}_{f,h}\}_{f=1,h=1}^{F,M}$ to MHA and FFN.



(b) Ghost Module $\mathcal{G}_{f,h}$.

Figure3: Using ghost modules to generate more features in BERT. G-MHA/FFN stands for Ghost-MHA/FFN.

(slides made by Zhiqi Huang)

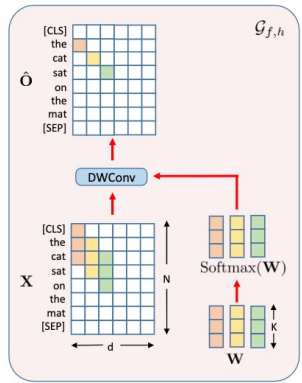
GhostBERT

- Convolution Type

$$O_{i,c} = \mathrm{DWConv}(\mathbf{X}_{:,c}, \mathbf{W}_{c,:}, i, c)$$
$$= \sum_{m=1}^{k} W_{c,m} \cdot X_{i-\lceil \frac{k+1}{2} \rceil + m, c}.$$

- Normalization

$$\hat{O}_{i,c} = \mathrm{DWConv}(\mathbf{X}_{:,c}, \mathrm{Softmax}(\mathbf{W}_{c,:}), i, c).$$



(b) Ghost Module $\mathcal{G}_{f,h}$.

(slides made by Zhiqi Huang)

GhostBERT

- 1. With only 55.3K more parameters (0.05% of BERT) and 14.2M more FLOPs (0.06% of BERT), adding ghost modules to pre-trained model increases the accuracy.

Model-Size	FLOPs(G)	#params(M)	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	Avg.	
BERT-base (Devlin et al., 2019)	22.5	110	84.5	92.0	90.9	71.1	92.9	87.8	58.1	89.8	83.4	
GhostBERT (m = 12/12)	22.5	110	84.7	92.3	91.1	71.8	93.0	88.0	63.6	89.7	**84.3**	+0.9
GhostBERT (m = 9/12)	16.9	88	84.8	92.1	91.2	72.6	92.6	87.5	61.1	89.8	84.0	
GhostBERT (m = 6/12)	11.3	67	84.7	92.2	91.2	72.2	92.9	87.3	58.1	89.2	83.5	
GhostBERT (m = 3/12)	5.8	46	84.3	91.6	91.4	72.9	94.6	86.5	53.9	89.2	83.1	
GhostBERT (m = 1/12)	2.0	32	82.8	90.0	90.5	66.1	92.8	86.0	46.1	87.8	80.3	
RoBERTa-base (Liu et al., 2019)	22.5	125	87.6	92.8	91.9	78.7	94.8	90.2	63.6	91.2	86.4	
GhostRoBERTa (m = 12/12)	22.5	125	88.0	93.1	91.9	80.5	95.3	90.7	65.0	91.3	**87.0**	+0.6
GhostRoBERTa (m = 9/12)	16.9	103	87.6	92.9	91.9	79.4	95.4	89.0	60.8	90.7	86.0	
GhostRoBERTa (m = 6/12)	11.3	82	86.8	92.6	91.6	77.6	94.4	89.7	57.6	90.3	85.1	
GhostRoBERTa (m = 3/12)	5.8	61	86.1	91.7	91.2	73.6	94.5	88.0	52.4	89.2	83.3	
GhostRoBERTa (m = 1/12)	2.0	47	82.1	89.2	90.5	66.1	93.7	83.3	39.8	87.4	79.0	
ELECTRA-small (Clark et al., 2020)	1.7	14	78.9	87.9	88.3	68.5	88.3	87.4	56.8	86.8	80.4	
GhostELECTRA-small (m = 4/4)	1.7	14	82.5	89.3	90.7	71.5	92.0	88.7	59.6	88.4	**82.8**	+2.4

(slides made by Zhiqi Huang)

GhostBERT

- 2. GhostBERT (m= 6/12) and GhostRoBERTa (m=9/12) get **similar** results to backbones
- 3. When the compression rate increases (i.e., **m = 3/12**, 1/12), we still achieve **99.6% performance** (resp. 96.3%) with only **25% FLOPs** (resp. 8%) of BERT-base.

Model-Size	FLOPs(G)	#params(M)	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	Avg.
BERT-base (Devlin et al., 2019)	22.5	110	84.5	92.0	90.9	71.1	92.9	87.8	58.1	89.8	83.4
GhostBERT (m = 12/12)	22.5	110	84.7	92.3	91.1	71.8	93.0	88.0	63.6	89.7	**84.3**
GhostBERT (m = 9/12)	16.9	88	84.8	92.1	91.2	72.6	92.6	87.5	61.1	89.8	84.0
GhostBERT (m = 6/12)	11.3	67	84.7	92.2	91.2	72.2	92.9	87.3	58.1	89.2	83.5
GhostBERT (m = 3/12)	5.8	46	84.3	91.6	91.4	72.9	94.6	86.5	53.9	89.2	83.1
GhostBERT (m = 1/12)	2.0	32	82.8	90.0	90.5	66.1	92.8	86.0	46.1	87.8	80.3
RoBERTa-base (Liu et al., 2019)	22.5	125	87.6	92.8	91.9	78.7	94.8	90.2	63.6	91.2	86.4
GhostRoBERTa (m = 12/12)	22.5	125	88.0	93.1	91.9	80.5	95.3	90.7	65.0	91.3	**87.0**
GhostRoBERTa (m = 9/12)	16.9	103	87.6	92.9	91.9	79.4	95.4	89.0	60.8	90.7	86.0
GhostRoBERTa (m = 6/12)	11.3	82	86.8	92.6	91.6	77.6	94.4	89.7	57.6	90.3	85.1
GhostRoBERTa (m = 3/12)	5.8	61	86.1	91.7	91.2	73.6	94.5	88.0	52.4	89.2	83.3
GhostRoBERTa (m = 1/12)	2.0	47	82.1	89.2	90.5	66.1	93.7	83.3	39.8	87.4	79.0
ELECTRA-small (Clark et al., 2020)	1.7	14	78.9	87.9	88.3	68.5	88.3	87.4	56.8	86.8	80.4
GhostELECTRA-small (m = 4/4)	1.7	14	82.5	89.3	90.7	71.5	92.0	88.7	59.6	88.4	**82.8**

(slides made by Zhiqi Huang)

GhostBERT

- Comparison with Other Compression Methods.

Model	FLOPs(G)	#params(M)	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	Avg.
BERT-base (Devlin et al., 2019)	22.5	110	84.6	90.5	89.2	66.4	93.5	84.8	52.1	85.8	80.9
RoBERTa-base (Liu et al., 2019)	22.5	125	86.0	92.5	88.7	73.0	94.6	86.5	50.5	88.1	82.5
ELECTRA-small (Clark et al., 2020)	1.7	14	79.7	87.7	88.0	60.8	89.1	83.7	54.6	80.3	78.0
TinyBERT$_6$ (Jiao et al., 2020)	11.3	67	84.6	90.4	89.1	70.0	93.1	87.3	51.1	83.7	81.2
TinyBERT$_4$ (Jiao et al., 2020)	1.2	15	82.5	87.7	89.2	66.6	92.6	86.4	44.1	80.4	78.7
ConvBERT-medium (Jiang et al., 2020)	4.7	17	82.1	88.7	88.4	65.3	89.2	84.6	56.4	82.9	79.7
ConvBERT-small (Jiang et al., 2020)	2.0	14	81.5	88.5	88.0	62.2	89.2	83.3	54.8	83.4	78.9
MobileBERT w/o OPT (Sun et al., 2020)	5.7	25	84.3	91.6	88.3	70.4	92.6	84.5	51.1	84.8	81.0
MobileBERT (Sun et al., 2020)	5.7	25	83.3	90.6	-	66.2	92.8	-	50.5	84.4	-
MobileBERT-tiny (Sun et al., 2020)	3.1	15	81.5	89.5	-	65.1	91.7	-	46.7	80.1	-
GhostBERT (m = 12/12)	22.5	110	84.6	91.1	89.3	70.2	93.1	86.9	54.6	83.8	81.7
GhostBERT (m = 9/12)	16.9	88	84.9	91.0	88.6	69.2	92.9	86.1	53.7	84.0	81.3
GhostBERT (m = 6/12)	11.3	67	84.2	90.8	89.1	69.6	93.1	84.0	53.4	83.1	80.9
GhostBERT (m = 3/12)	5.8	46	83.8	90.7	89	68.6	93.2	82.5	51.3	82.5	80.2
GhostBERT (m = 1/12)	2.0	32	82.5	89.3	88.7	65.0	92.9	81.0	41.3	80.0	77.6
GhostRoBERTa (m = 12/12)	22.5	125	87.9	93.0	89.6	74.6	95.1	88.0	52.4	88.3	83.6
GhostRoBERTa (m = 9/12)	16.9	103	87.7	92.6	89.5	73.0	94.5	85.7	51.9	87.1	82.8
GhostRoBERTa (m = 6/12)	11.3	82	86.3	92.1	89.5	71.5	94.5	86.8	51.2	87.0	82.4
GhostRoBERTa (m = 3/12)	5.8	61	85.5	91.2	89.1	68.5	93.4	85.3	48.9	84.7	80.8
GhostRoBERTa (m = 1/12)	2.0	47	81.3	88.6	88.5	62.8	92.1	82.8	39.7	81.8	77.2
GhostELECTRA-small (m = 4/4)	1.7	14	82.3	88.3	88.5	64.7	91.9	88.4	55.8	83.5	80.4

(slides made by Zhiqi Huang)

Content

Future Work

- Huge LMs like GPT-3 brings new challenges:
  - Model is too big to conduct full-model fine-tune
  - Inference is slow
- Ideas:
  - Fix backbone fine-tuning
    - Prompt (Prefix) tuning
    - Adaptor
  - KD with large gaps in model size
  - Train small models with authentic data generated by huge models

CFP of ENLSP 2021 Workshop (a NeurIPS Workshop)

Date: between Mon Dec 6th - Tue the 14th (to be determined)

```
--------------------------------------------------------------------------------
Schedule for contributed workshop papers:
Suggested Submission Date for Workshop Contributions: Sep 17, 2021
Suggested Acceptance date: October 15, 2021
Mandatory Accept/Reject Notification Date: Oct 22, 2021
--------------------------------------------------------------------------------
```

Website: https://neurips2021-nlp.github.io/

CFP of ENLSP 2021 Workshop (a NeurIPS Workshop)

Invited Speaker	Affiliation
Prof. Mirella Lapata	University of Edinburgh
Prof. Luke Zettlemoyer	University of Washington & Facebook
Prof. Kevin Duh	Johns Hopkins University
Dr. Mohammad Norouzi	Google Brain
Prof. Yejin Choi	University of Washington & Allen Institute for AI
Dr. Boxing Chen	Alibaba
Prof. Sameer Singh	University of California, Irvine (UCI)
Prof. Danqi Chen	Princeton University
Dr. Xin Jiang	Huawei Noah's Ark Lab
Prof. Xu Sun	Peking University
Prof. Barbara Plank	IT University of Copenhagen

Thank you!

把数字世界带入每个人、每个家庭、
每个组织，构建万物互联的智能世界。

Bring digital to every person, home and organization for a fully connected, intelligent world.

HUAWEI