

A APPENDIX

A.1 LANGUAGE MODELING EXPERIMENT

A.1.1 DATA

Open WebText is an open source effort to reproduce OpenAI’s WebText dataset. The dataset is created by extracting Reddit post urls from the Reddit submissions dataset³. These links are then deduplicated, filtered to exclude non-html content, and shuffled randomly. Near-duplicate documents are identified using local-sensitivity hashing. They are hashed into sets of 5-grams and all documents that had a similarity threshold of greater than 0.5 were removed. All language modeling datasets were tokenized based on byte-level BPE (Sennrich et al., 2016) with a vocabulary size of 50257 (Radford et al., 2019). The max sequence length of the input training sample is 1024.

A.1.2 TRAINING

Our implementation is based on Huggingface Transformers⁴. The GPT-2 base model consists 12 layers and has 12 attention heads in each attention module. The input and intermediate hidden dimension in the feed-forward network is 768 and 1024, respectively. We use mixed precision training and train on 8 80G Nvidia A100 GPUs. Detailed hyper-parameters are summarized in Table 10.

Table 10: Hyper-parameters for training GPT-2₆ on Open WebText.

Hyper-parameters	Stage I	Stage II
Dropout	0.1	0.1
Warmup Ratio	0.05	0.05
Learning Rates	0.00025	0.00025
Batch Size	4000	4000
Weight Decay	0	0
Training Epochs	1	4
Learning Rate Decay	Linear	Linear
Adam ϵ	1×10^{-6}	1×10^{-6}
Adam β_1	0.9	0.9
Adam β_2	0.98	0.98

A.2 NATURAL LANGUAGE UNDERSTANDING EXPERIMENT

A.2.1 DATA

GLUE is a commonly used natural language understanding benchmark containing nine tasks. The benchmark includes question answering (Rajpurkar et al., 2016b), linguistic acceptability (CoLA, Warstadt et al. 2019), sentiment analysis (SST, Socher et al. 2013), text similarity (STS-B, Cer et al. 2017), paraphrase detection (MRPC, Dolan & Brockett 2005), and natural language inference (RTE & MNLI, Dagan et al. 2006; Bar-Haim et al. 2006; Giampiccolo et al. 2007; Bentivogli et al. 2009; Williams et al. 2018) tasks. Details of the GLUE benchmark, including tasks, statistics, and evaluation metrics, are summarized in Table 14.

SQuAD 1.1/2.0 is the Stanford Question Answering Dataset (SQuAD) v1.1 and v2.0 (Rajpurkar et al., 2018; 2016a), two popular machine reading comprehension benchmarks from approximately 500 Wikipedia articles with questions and answers obtained by crowdsourcing. The SQuAD v2.0 dataset includes unanswerable questions about the same paragraphs.

³<https://files.pushshift.io/reddit/submissions/>

⁴<https://github.com/huggingface/transformers/tree/v4.17.0>

A.2.2 MODEL

We initialize the teacher for each target task as a DeBERTaV3-base model fine-tuned on the target task. We fine-tune the model by adding a target task classification head on top of the last layer. The detailed hyper-parameters are listed in Table 13. We initialize the student for each target task as a pre-trained DeBERTaV3-xsmall model.

A.2.3 TRAINING

We follow the hyper-parameter configurations listed in Table 13 for both the Stage I and Stage II training. Our implementation is based on Huggingface Transformers. We use mixed precision training and train on 8 32G Nvidia V100 GPUs.

For Stage I, we empirically observe that if we first fine-tune the student on the target task, then train the filters on top of the fine-tuned student, the distillation performance would improve. We hypothesize that the student filters can learn to capture more task-relevant knowledge if the student is properly initialized on the target task. As a result, we also fine-tune the student model following the hyper-parameter configuration listed in Table 13 before Stage I. Since most task-specific distillation baselines often adopt a pre-trained model as the student initialization, we show that adopting a fine-tuned model as the student initialization will not largely influence the final distillation performance and the comparison is still fair. As shown in Table 11, initializing the student model with fine-tuned weights leads to a comparable performance to initializing the student model with pre-trained weights.

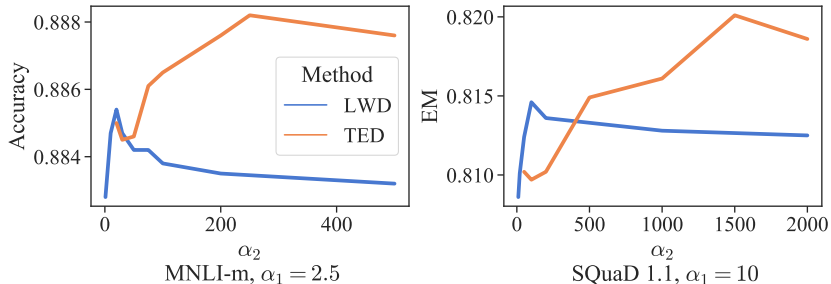
Table 11: Performance comparison of initializing the student with fine-tuned and pre-trained weights.

Method	Θ_s Fine-tuned?	MNLI-m/mm Acc	QQP Acc	QNLI Acc	SST-2 Acc	RTE Acc	Avg Score
LWD	✗	88.8/88.3	91.8	92.9	93.9	80.2	89.5
Abl.	✓	88.7/88.5	92.0	92.8	93.5	79.5	89.3

A.2.4 HYPER-PARAMETER STUDY

We further investigate whether TED is sensitive to α_2 , the hyper-parameter that control the strength of \mathcal{D}_{TED} . Figure 3 shows the performance of the DeBERTaV3-xsmall student on MNLI-m and SQuAD v1.0 under different values of α_2 . TED shows consistently gains over a wide range of values of α_2 .

Figure 3: Evaluation performance of DeBERTaV3-xsmall under different values of α_2 .



A.2.5 BERT EXPERIMENTS

Model. We initialize the teacher model with a pre-trained 12-layer BERT-base model that has been fine-tuned on the target task (BERT-base₁₂). The teacher model contains 110M parameters and has a hidden dimension of $d_t = 768$. We initialize the student model with 6 selected layers from the fine-tuned teacher model (BERT-base₆). Specifically, we define the layer mapping function

$M(k) = 2k - 1$ for $k \leq K/2$ and $M(k) = 2k$ for $k > K/2$, which is the same as Sanh et al. 2019. The fine-tuning hyper-parameters are listed in Table 12.

Stage I. We initialize each task-aware filter of the teacher with size $d_t \times d_t$. We fix the fine-tuned teacher and train the filters following the hyper-parameter configurations listed in Table 12. We directly take the trained k -th filter of the teacher as the k -th filter of the student without further training.

Stage II. We distill the student model and its filters following the hyper-parameter configurations listed in Table 12. Our implementation is based on Huggingface Transformers. We conduct all experiments using mixed precision training on 8 32G Nvidia V100 GPUs.

Table 12: Hyper-parameters for fine-tuning BERT-base₁₂ on MNLI.

Hyper-parameters	BERT-base
Dropout of Task Layer	0.1
Warmup Steps	1000
Learning Rates	3×10^{-5}
Batch Size	32
Weight Decay	0
Training Epochs	3
Learning Rate Decay	Linear
Adam ϵ	1×10^{-6}
Adam β_1	0.9
Adam β_2	0.98

A.3 ANALYSIS

A.3.1 TED ALLEVIATES THE CAPACITY GAP ISSUE

Model. We initialize the teacher model with a pre-trained 24-layer DeBERTaV3-large model that has been fine-tuned on the target task. The teacher model contains 435M parameters and has a hidden dimension $d_t = 1024$. We initialize the student model with a 12-layer DeBERTaV3-xsmall model. The student model contains 70M parameters and has a hidden dimension $d_s = 384$. We define the layer mapping function $M(k) = 2k - 1$ for $k \leq K/2$ and $M(k) = 2k$ for $k > K/2$, which is the same as Sanh et al. 2019. The fine-tuning hyper-parameters are listed in Table 13.

Stage I. We initialize each filter of the teacher model with the size $d_t \times d_t$ and each filter of the student model with the size $d_s \times d_t$. We fix the model parameters of the teacher and the student and train their filters following the hyper-parameters summarized in Table 13.

Stage II. We distill the student model and its filters following the hyper-parameters listed in Table 13. Our implementation is based on Huggingface Transformers. We conduct all experiments using mixed precision training on 8 32G Nvidia V100 GPUs.

Table 13: Hyper-parameters for fine-tuning DeBERTaV3 models on the downstream tasks.

Hyper-parameters	DeBERTaV3-large	DeBERTaV3-base	DeBERTaV3-xsmall
Dropout of Task Layer	{0.05, 0.1}	{0.05, 0.1, 0.15}	{0.05, 0.1, 0.15}
Learning Rates	$\{6, 7, 10\} \times 10^{-6}$	$\{1, 1.5, 2, 2.5, 3, 4, 5\} \times 10^{-5}$	$\{3, 3.5, 5, 6, 8, 9\} \times 10^{-5}$
Batch Size	{32, 64}	{12, 16, 32, 64}	{12, 16, 32, 64}
Weight Decay	0	0	0
Training Epochs	{2, 6, 8}	{2, 3, 6, 8}	{2, 3, 6, 8}
Learning Rate Decay	Linear	Linear	Linear
Adam ϵ	1×10^{-6}	1×10^{-6}	1×10^{-6}
Adam β_1	0.9	0.9	0.9
Adam β_2	0.98	0.98	0.98

Table 14: Summary of the GLUE benchmark.

Corpus	Task	#Train	#Dev	#Test	#Label	Metrics
Single-Sentence Classification (GLUE)						
CoLA	Acceptability	8.5k	1k	1k	2	Matthews corr
SST	Sentiment	67k	872	1.8k	2	Accuracy
Pairwise Text Classification (GLUE)						
MNLI	NLI	393k	20k	20k	3	Accuracy
RTE	NLI	2.5k	276	3k	2	Accuracy
QQP	Paraphrase	364k	40k	391k	2	Accuracy/F1
MRPC	Paraphrase	3.7k	408	1.7k	2	Accuracy/F1
QNLI	QA/NLI	108k	5.7k	5.7k	2	Accuracy
Text Similarity (GLUE)						
STS-B	Similarity	7k	1.5k	1.4k	1	Pearson/Spearman corr