Are Intermediate Layers and Labels Really Necessary? A General Language Model Distillation Method

Shicheng Tan[*1], **Weng Lam Tam**[2], **Yuanchun Wang**[3], **Wenwen Gong**[4], **Shu Zhao**[†1], **Peng Zhang**[2], **Jie Tang**[†4]
[1]Anhui University, [2]Zhipu.AI, [3]Renmin University of China, [4]Tsinghua University
tsctan@foxmail.com,{rainatam9784,frederickwang99}@gmail.com
wenweng@mail.tsinghua.edu.cn,zhaoshuzs2002@hotmail.com
peng.zhang@zhipuai.cn,jietang@tsinghua.edu.cn

Abstract

The large scale of pre-trained language models poses a challenge for their deployment on various devices, with a growing emphasis on methods to compress these models, particularly knowledge distillation. However, current knowledge distillation methods rely on the model's intermediate layer features and the golden labels (also called hard labels), which usually require aligned model architecture and enough labeled data respectively. Moreover, the parameters of vocabulary are usually neglected in existing methods. To address these problems, we propose a general language model distillation (GLMD) method that performs two-stage word prediction distillation and vocabulary compression, which is simple and surprisingly shows extremely strong performance. Specifically, GLMD supports more general application scenarios by eliminating the constraints of dimension and structure between models and the need for labeled datasets through the absence of intermediate layers and golden labels. Meanwhile, based on the long-tailed distribution of word frequencies in the data, GLMD designs a strategy of vocabulary compression through decreasing vocabulary size instead of dimensionality. Experimental results show that our method outperforms 25 state-of-the-art methods on the SuperGLUE benchmark, achieving an average score that surpasses the best method by 3%. [1]

1 Introduction

The exponential increase in the scale of pre-trained language models has impeded their deployment on a wider range of devices. To mitigate the inference cost of large-scale pre-trained language models, researchers have increasingly focused on model compression methods, aiming to compress a large model into a small one with as little performance loss as possible (Li et al., 2022). While model compression can yield very small models, maintaining performance without degradation is still a challenging task, particularly when the large and small models have a significant discrepancy in parameter size (Li et al., 2021). There are various methods of model compression (Wang and Yoon, 2022), including network pruning (Huang et al., 2022), quantization (Boo et al., 2021), neural architecture search (Elsken et al., 2019), parameter sharing (Lan et al., 2020), matrix decomposition (Tahaei et al., 2022), and knowledge distillation (Liu et al., 2022b; Zhang et al., 2022) etc. Currently, knowledge distillation is an important research direction, which allows for the transfer of knowledge from a large model (the teacher) to a small one (the student).

There are two main optimization objectives of the earliest knowledge distillation methods (Hinton et al., 2015): increasing the similarity between the student's prediction probabilities for the task and those of the teacher (soft targets); increasing the similarity between the student's predictions and the golden labels (hard targets). When the knowledge distillation method is applied to language models, there are typically two directions for improvement: leveraging the intermediate layer features of the teacher model, such as hidden states and attention, to obtain additional hidden state knowledge (Sun et al., 2019; Liu et al., 2022a); and refining the two objectives (soft targets and hard targets) and weights of objectives (Lu et al., 2021; Zhou et al., 2022). As shown in Table 1, these methods all rely on intermediate layer features or hard labels of the model. However, using intermediate layer features and hard labels is often accompanied by certain limitations, such as the requirement for the teacher and student models to have the same structure and dimensions, or the need for additional data and labels. These limitations make the implementation

[*]This work was done when the author visited Zhipu.AI.
[†]Corresponding authors.
[1]The code is available at https://github.com/aitsc/GLMKD.

Findings of the Association for Computational Linguistics: ACL 2023, pages 9678–9696
July 9-14, 2023

Methods	Inter	Soft	Hard
PKD (Sun et al., 2019)	✓	✓	✓
DistilBERT (Sanh et al., 2019)	✓	✓	✓
Theseus (Xu et al., 2020)	✓		✓
TinyBERT (Jiao et al., 2020)	✓	✓	
SID (Aguilar et al., 2020)	✓	✓	
MobileBERT (Sun et al., 2020b)	✓		✓
CoDIR (Sun et al., 2020a)	✓	✓	✓
MiniLM (Wang et al., 2020)	✓		
MiniLMv2 (Wang et al., 2021)	✓		
ALP-KD (Passban et al., 2021)	✓	✓	✓
LRC-BERT (Fu et al., 2021)	✓	✓	✓
Annealing-KD (Jafari et al., 2021)		✓	✓
CKD (Park et al., 2021)	✓	✓	✓
Universal-KD (Wu et al., 2021b)	✓	✓	✓
Meta-KD (Pan et al., 2021)	✓	✓	
HRKD (Dong et al., 2021)	✓	✓	
RW-KD (Lu et al., 2021)		✓	✓
MetaDistil (Zhou et al., 2022)		✓	✓
DIITO (Wu et al., 2022)	✓	✓	✓
Continuation-KD (Jafari et al., 2022)		✓	✓
RAIL-KD (Haidar et al., 2022)	✓	✓	✓
MGSKD (Liu et al., 2022a)	✓	✓	
TMKD (Yang et al., 2020)		✓	✓
MT-BERT (Wu et al., 2021a)	✓	✓	✓
RL-KD (Yuan et al., 2021)		✓	✓
Uncertainty (Li et al., 2021)		✓	✓

Table 1: Almost all knowledge distillation methods for language models are based on either intermediate layer features (Inter) or hard labels (Hard). Soft denotes soft labels, specifically, the logits of the teacher model in downstream task loss.

of distillation complex and hinder the applicability of these methods to a wider range of models and data. Moreover, existing methods often reduce the parameter scale of the model by decreasing the number of layers and hidden dimensions, neglecting the impact of vocabulary size.

To address these problems, we propose a general language model distillation (GLMD) method that performs two-stage (pre-training and task-specific stages) word prediction distillation and vocabulary compression. Specifically, GLMD distills the model using only the language modeling word prediction logits during the pre-training stage, which is similar to the soft labels used in general methods. The key to this stage is that we distill both masked and unmasked tokens. In the task-specific stage (fine-tuning), GLMD distills both the language modeling word prediction logits and the soft labels. The language modeling word prediction logits is crucial in this stage, making the distillation more consistent between the pre-training and task-specific stages. In these two stages, GLMD eliminates the need for complicated intermediate layers and golden labels and does not require the selection of intermediate layers or labeled dataset. Meanwhile, GLMD uses the teacher vocabulary to map low-frequency words to the most similar high-frequency words, further compressing the model with almost no performance loss.

In summary, our major contributions are:

- We propose a general language model distillation (GLMD) method that saves the tedious work on intermediate layer features and golden labels, and does not require the selection of intermediate layers or labeled dataset. We demonstrate through analysis that GLMD allows models to autonomously learn intermediate layer features that are similar to those of the teacher.

- We propose a vocabulary compression strategy based on the long-tailed distribution of words in data, which reduces the vocabulary size without reducing dimensions of the model. Additionally, our vocabulary compression strategy can be used in conjunction with other dimensionality reduction strategies with very little performance loss.

- We verify that GLMD outperforms 25 state-of-the-art model distillation methods on the SuperGLUE benchmark, achieving an average score that surpasses the best method by 3%. Furthermore, our vocabulary compression strategy also outperforms other 2 dimensionality reduction strategies. We also investigate distillation of ultra-large-scale language models (10B-scale) for the first time.

2 Related work

Language Model Distillation Since the introduction of knowledge distillation to pre-trained language models by PKD (Sun et al., 2019), an increasing number of researchers have recognized the importance of knowledge distillation. During the early stage of the research, PD (Turc et al., 2019) employed simple baseline (soft targets) distillation for language models, resulting in a relatively limited transfer of knowledge for the model. Subsequent research had primarily focused on the use of intermediate layer features in language models (Xu et al., 2020; Sanh et al., 2019), including distillation of models during pre-training stage (Sun et al., 2020b), task-specific stage (Aguilar et al., 2020), and two-stage (Jiao et al., 2020) approaches. Given

the typically large amount of intermediate layer features, some work had utilized features from only a single intermediate layer (Wang et al., 2020, 2021), while other work had examined methods for reducing the scale of features (Liu et al., 2021). Recent work has explored ways to utilize better intermediate layer features, for example, CoDIR (Sun et al., 2020a) and LRC-BERT (Fu et al., 2021) utilized cross-sample feature relationships through contrastive learning; ALP-KD (Passban et al., 2021) and Universal-KD (Wu et al., 2021b) combined all intermediate layer features through attention mechanisms; Meta-KD (Pan et al., 2021) and HRKD (Dong et al., 2021) used meta-learning to assign appropriate weights to intermediate layer features; RAIL-KD (Haidar et al., 2022) randomly selected different intermediate layers for distillation; CKD (Park et al., 2021) and MGSKD (Liu et al., 2022a) used some variety of similarity calculation methods for intermediate layer features; DIITO (Wu et al., 2022) allowed student models to learn counterfactual outputs by swapping intermediate layer features between different samples.

However, the use of intermediate layer features has additional limitations, such as requiring the same model structure (Sun et al., 2020b) for both teacher and student, or requiring linear transformations (Jiao et al., 2020) to ensure consistency in dimensions between teacher and student. There were also methods that only used soft and hard targets, for example, Annealing-KD (Jafari et al., 2021) and Continuation-KD (Jafari et al., 2022) gradually increased the weight of soft targets through simulated annealing; RW-KD (Lu et al., 2021) adjusted the weight of soft and hard targets through meta-learning and a dev set; MetaDistil (Zhou et al., 2022) allowed the teacher to learn how to output better soft labels through meta-learning and a quiz set. These approaches relied on hard labels and may have even required additional datasets for partitioning. Additionally, there had been approaches that distilled multiple teachers (Yang et al., 2020; Wu et al., 2021a; Yuan et al., 2021; Li et al., 2021) or teacher assistants (Mirzadeh et al., 2020; Son et al., 2021) at the same time, but they still relied on intermediate layer features or hard labels. In comparison, GLMD can achieve the strongest performance in a more broadly applicable context without intermediate layer features or hard labels.

Vocabulary Compression Vocabulary compression refers to reducing the parameter size of the vocabulary in a language model. In the knowledge distillation of language models, reducing the parameter size of the model is mainly used to reduce the number of model layers or dimensions (Jiao et al., 2020; Wang et al., 2021). MobileBERT (Sun et al., 2020b) and ALBERT (Lan et al., 2020) independently reduced the dimensions of the vocabulary to achieve vocabulary compression. MobileBERT needed to restore the dimension of the vocabulary in the calculation of the pre-training loss due to the requirement to ensure consistency between the vocabulary dimension and the model output dimension. On the other hand, ALBERT used a linear layer to alter the output dimension of the model. However, these vocabulary compression methods only reduced the dimensionality and ultimately required dimensionality restoration. In contrast, our vocabulary compression method reduces the number of words through mapping, further compressing the model with almost no impact on performance.

3 Preliminaries

In this section, we introduce the objective function for knowledge distillation of language models, and formalize the language modeling word prediction logits.

3.1 Knowledge Distillation

Knowledge distillation aims to transfer the knowledge of the teacher T to the student S. The knowledge and transfer method can be formalized as model features and distance metrics respectively. Formally, knowledge distillation for language models typically consists of the following three objective functions:

$$\begin{aligned}\mathcal{L}_{\text{soft}} &= \tau^2 KL(\sigma(f_l^S(\mathbf{x})/\tau), \sigma(f_l^T(\mathbf{x})/\tau)) \\ \mathcal{L}_{\text{hard}} &= CE(\sigma(f_l^S(\mathbf{x})), \mathbf{y}) \\ \mathcal{L}_{\text{inter}} &= d(f^S(\mathbf{H}^S), f^T(\mathbf{H}^T))\end{aligned} \tag{1}$$

where τ denotes the softening parameter (temperature), $KL(\cdot,\cdot)$ denotes the KL divergence, σ denotes the softmax function, $\mathbf{x} \in \mathbb{R}^l$ denotes the input sequence (token ids) of length l for the language model, $f_l^S(\mathbf{x})$ and $f_l^T(\mathbf{x})$ denote the logits output by the student and the teacher before computing the task loss respectively, $CE(\cdot,\cdot)$ denotes the cross entropy, $\mathbf{y}$ denotes the hard labels, $d(\cdot)$ denotes the distance metric (e.g., KL divergence and mean square error), $\mathbf{H}^S$ and $\mathbf{H}^T$ denote the intermediate layer features (e.g., hidden states and

attention) of the student and the teacher respectively, $f^S(\cdot)$ and $f^T(\cdot)$ denote custom transformations (e.g. linear transformations) of the student and teacher features, respectively.

Currently, mainstream methods employ different combinations and weighting schemes of the three objective functions in the pre-training and task-specific stages. For example, TinyBERT (Jiao et al., 2020) optimizes $\mathcal{L}_{\text{inter}}$ in the pre-training stage and optimizes $\mathcal{L}_{\text{inter}}$ and $\mathcal{L}_{\text{soft}}$ in the task-specific stage, while MetaDistil (Zhou et al., 2022) only optimizes $\mathcal{L}_{\text{soft}}$ and $\mathcal{L}_{\text{hard}}$ in the task-specific stage. Notably, to ensure feature dimension matching between the teacher and student, $\mathcal{L}_{\text{inter}}$ relies on complex custom transformations, such as linear transformations ($f(\mathbf{H}) = \mathbf{WH}$) and pair-wise scaled dot-product ($f(\mathbf{H}) = \mathbf{HH}^T/\sqrt{dimensionality}$). In contrast, our method does not rely on $\mathcal{L}_{\text{inter}}$ and $\mathcal{L}_{\text{hard}}$.

3.2 Language Modeling Word Prediction Logits

Language modeling typically refers to unsupervised tasks in the pre-training stage, such as causal language modeling for GPT (Radford et al., 2018), masked language modeling for BERT (Devlin et al., 2019), and autoregressive blank filling for GLM (Du et al., 2022). This process typically requires a decoder to decode the model's output into a prediction logits for each word. The decoder is typically a linear transformation using the vocabulary parameters as weights. The language modeling word prediction logits can be formulated as follows:

$$LM(\mathbf{x}) = f_t(\mathbf{x})\mathbf{W}_v^T \tag{2}$$

where $\mathbf{W}_v \in \mathbb{R}^{v\times h}$ denotes the vocabulary parameters (weight of embeddings-layer), and $f_t(\mathbf{x}) \in \mathbb{R}^{l\times h}$ denotes the output of the final layer of transformer. The scalar value l denotes the length of the text sequence, v denotes the number of tokens in the vocabulary, and h denotes the dimensionality of the hidden layer. It is worth noting that the $LM(\cdot)$ can also be computed at the task-specific stage.

4 Method

In this section, we propose a general language model distillation (GLMD) method with two-stage word prediction distillation and vocabulary compression. Figure 1 shows the overview framework of GLMD, which implements a vocabulary compression strategy while performing a two-stage

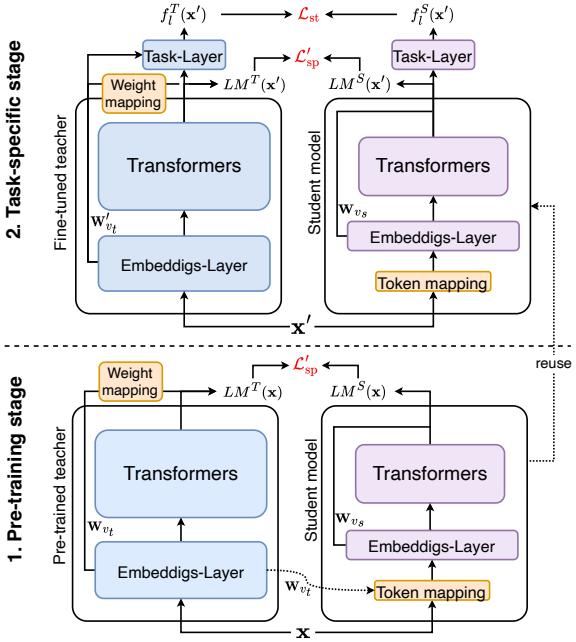

Figure 1: The framework of the GLMD. The task-layer aims to convert the $f_t(\cdot)$ and $\mathbf{W}_v$ into the logits $f_l(\cdot)$ of downstream task loss.

word prediction distillation process. We next provide a detailed description of these two components.

4.1 Two-stage Word Prediction Distillation

To eliminate the reliance on intermediate layer features and hard labels in distillation, we propose a two-stage word prediction distillation process based on the language modeling word prediction logits. It allows teacher models and students models to have different model structures and does not need the selection of intermediate layers. This process makes the distillation goal more closely aligned with the model's task and makes the model's distillation more consistent across both the pre-training and task-specific stages. During the pre-training stage, we optimize the student model with the objective function $\mathcal{L}'_{\text{sp}}$. We then optimize the student again using $\mathcal{L}'_{\text{sp}}$ during the task-specific stage. After these two training phases, we finally optimize the student with the objective function $\mathcal{L}_{\text{st}}$ from the task-specific stage. Our objective functions $\mathcal{L}_{\text{st}}$ and $\mathcal{L}'_{\text{sp}}$ are defined as:

$$\begin{aligned}
\mathcal{L}_{\text{st}} &= \mathcal{L}_{\text{soft}} \\
\mathcal{L}'_{\text{sp}} &= \mathcal{L}_{\text{sp}} \odot \mathbf{m}_p \\
\mathcal{L}_{\text{sp}} &= \tau^2 KL(\sigma(\tfrac{LM^S(\mathbf{x})}{\tau}), \sigma(\tfrac{LM^T(\mathbf{x})}{\tau}))
\end{aligned} \tag{3}$$

where $\odot$ denotes the Hadamard product, and $\mathbf{m}_p \in \mathbb{R}^l$ denotes the mask vector, which only masks the pad while preserving the masked and unmasked tokens. We find that unmasked tokens are typically not predicted by language modeling, but they can provide more knowledge for distillation.

$\mathcal{L}'_{\text{sp}}$ and $\mathcal{L}_{\text{st}}$ represent the soft targets for the pre-training and task-specific stages, respectively. It is worth noting that $\mathcal{L}'_{\text{sp}}$ can be used in both pre-training and task-specific stages, making the optimization objectives more consistent across the two stages.

4.2 Vocabulary Compression

To further compress the parameter scale of the model, we propose a vocabulary compression strategy that reduces the number of tokens in the vocabulary. Because word frequencies have a long-tailed distribution, some low-frequency words can still be understood by the language model after being replaced with similar words. Let the compression rate of the vocabulary be r_v, and the number of all tokens before compression be v. We sort the tokens in the pre-trained corpus according to their frequency of occurrence. The tokens ranking in the top vr_v are treated as the compressed tokens $\mathbf{w}_c \in \mathbb{R}^{vr_v}$, while the remaining tokens $\mathbf{w}_m \in \mathbb{R}^{v(1-r_v)}$ are to be mapped. Figure 1 illustrates the key aspect of our vocabulary compression strategy, which includes replacing low-frequency words with similar words through **token mapping** and aligning the weight matrix of the embedding-layer through **weight mapping**. The token mapping aims to map $w_m \in \mathbf{w}_m$ to $w_c \in \mathbf{w}_c$, and the mapping function is defined as:

$$f_{tm}(w_m) = \underset{w_c \in \mathbf{w}_c}{\text{argmax}}(Sim(v(w_m), v(w_c))) \quad (4)$$

where $v(w_m)$ and $v(w_c)$ denote the token vectors of w_m and w_c in the pre-trained teacher's vocabulary weight $\mathbf{W}_{v_t}$, respectively. $Sim(\cdot,\cdot)$ is a function of calculating similarity using the inner product, which is similar to the decoding in Equation 2. The weight mapping aims to remove $\mathbf{w}_m$ from $\mathbf{W}_{v_t}$, and the mapping function is defined as:

$$f_{wm}(\mathbf{W}_{v_t}) = \mathbf{W}_{v_t}[\mathbf{w}_c] \quad (5)$$

where $[\cdot]$ denotes a slicing operation, specifically obtaining all vector of $w_c \in \mathbf{w}_c$ from $\mathbf{W}_{v_t}$.

Models	#Params	#Dimensions	#Layers
T_1	110M	768	12
T_1 for MobileBERT	293M	1024	24
T_2	340M	1024	24
T_3	10B	4096	48
A_1	200M	896	14
A_2	110M	768	12
S_1	22M	384	6
S_1 for MobileBERT	25M	128	24
S_2	66M	768	6
S_3	2B	2048	36

Table 2: The parameter sizes, hidden layer dimensions, and number of transformer layers of the teacher models (T_1, T_2, T_3), teacher assistant models (A_1, A_2), and student models (S_1, S_2, S_3) used by all methods.

5 Experiments

In this section, we demonstrate the effectiveness of the GLMD on models with different parameter scales (110M, 340M, and 10B) and analyze the role of different components and why they are effective. All experiments were conducted on 40 NVIDIA A100 GPUs and completed within 4 months, utilizing the PyTorch framework.

5.1 Experimental Setup

Datasets To evaluate our GLMD method, we conduct experiments on the more challenging SuperGLUE (Wang et al., 2019a) benchmark instead of GLUE (Wang et al., 2019b). The use of more difficult tasks allows for a better display of the discrepancy between different distillation methods. We use the average score across 8 tasks in SuperGLUE as the evaluation metric. We use BooksCorpus (Zhu et al., 2015) and English Wikipedia as the data (19GB) for the distillation of pre-training stage for all methods.

Baselines We compare 25 commonly used distillation methods as listed in Table 3. We provide a more detailed description of these methods in Appendix A.1.

Language Models Student and teacher models of all methods have the standard GLM (Du et al., 2022) architecture. GLM (General Language Model) is a more advanced language model that inherits the advantages of both autoencoding and autoregression. We choose GLM for two reasons: it performs stronger than the commonly used BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019); it has open-source pre-trained language models with 10B-scale or even 100B-scale

parameters (Zeng et al., 2022). All pre-trained models, with the exception of MobileBERT, which was trained by us based on GLM, were obtained from the official GLM website [2]. Both the teacher and student models were trained with half-precision floating-point (fp16). The model sizes used in this paper are shown in Table 2.

Hyperparameters For our method, the temperatures τ for the loss $\mathcal{L}'_{\text{sp}}$ and $\mathcal{L}_{\text{st}}$ are set to 15 and 1, respectively. All baselines use the best parameters from their respective papers. For all methods that rely on the pre-training stage, the batch size, peak learning rate, and number of iterations are set to 64, 4e-4, and 150000, respectively. For all single-teacher methods, we use grid search to find the best parameters during the task-specific stage, including the learning rate {5e-6,1e-5,2e-5} and batch size {16,32}. The multi-teacher and teacher assistant methods are similar to the single-teacher methods in the core method, with differences in the weighting and assistant of teachers. The other parameters for the task-specific stage are kept consistent with the fine-tuned teacher, using the best parameters provided by GLM. The results for all experiments (w/o T_3-S_3) are the average of 3 random seeds. For more details on the hyperparameters, refer to Appendix A.2.

5.2 Main Results

In Table 3, we report the average scores of all methods on the SuperGLUE dev set. GLMD$_{-\text{vc}}$ denotes GLMD without vocabulary compression strategy. GLMD$_{-\text{vc+mo}}$ and GLMD$_{-\text{vc+al}}$ denote the use of MobileBERT and ALBERT vocabulary compression strategies on GLMD, respectively. GLMD$_{+\text{al}}$ denotes the combination of ALBERT and our vocabulary compression strategies on GLMD.

GLMD achieves the highest performance among 25 baselines on T_1-S_1, T_1-S_2, and T_2-S_2 scales, with a 0.1%, 0.1%, and 3.1% improvement over the best method (TinyBERT), respectively. More importantly, in a fair environment without vocabulary compression, GLMD$_{-\text{vc}}$ outperforms the best method by 0.7%, 0.7%, and 3.0%, respectively. This demonstrates that high-performance distillation does not necessarily require intermediate layer features or hard labels, whether reducing the number of layers or the dimensionality of the student model. GLMD significantly outperforms TinyBERT in the distillation process on the scale of

[2] https://github.com/THUDM/GLM

Methods	SuperGLUE		
Fine-tuned teacher	T_1	T_2	T_3
GLM (Du et al., 2022)	71.7	77.1	89.0
Single-teacher	T_1-S_1	T_1-S_2	T_2-S_2
(compression ratio)	(0.2)	(0.5)	(0.2)
(parameter size of the student)	(22M)	(66M)	(66M)
KD (Hinton et al., 2015)	52.0	52.4	53.2
PD (Turc et al., 2019)	61.0	62.0	61.4
PKD (Sun et al., 2019)	-	66.2	-
DistilBERT (Sanh et al., 2019)	-	63.8	-
Theseus (Xu et al., 2020)	-	66.1	-
TinyBERT (Jiao et al., 2020)	67.3	70.8	69.1
SID (Aguilar et al., 2020)	-	55.1	-
MobileBERT$_{25\text{M}}$ (Sun et al., 2020b)	65.1	-	-
MiniLM (Wang et al., 2020)	61.6	63.8	62.4
MiniLMv2 (Wang et al., 2021)	62.1	63.7	66.1
ALP-KD (Passban et al., 2021)	-	64.3	-
LRC-BERT (Fu et al., 2021)	58.4	62.3	60.8
Annealing-KD (Jafari et al., 2021)	61.0	63.3	63.1
CKD (Park et al., 2021)	61.9	63.3	62.6
Universal-KD (Wu et al., 2021b)	52.9	66.2	54.5
DIITO (Wu et al., 2022)	-	66.8	-
Continuation-KD (Jafari et al., 2022)	61.0	62.7	61.7
RAIL-KD (Haidar et al., 2022)	61.2	63.6	60.4
MGSKD (Liu et al., 2022a)	58.8	63.5	59.9
GLMD$_{-\text{vc}}$ *(ours)*	**67.9**	**71.5**	72.1
GLMD *(ours)*	67.4	70.9	**72.2**
(parameter size of the student)	(16M)	(55M)	(55M)
Single-teacher (10B-scale)	T_3-S_3 (2B)		
TinyBERT (Jiao et al., 2020)	65.09 (w/o ReCoRD)		
GLMD$_{-\text{vc}}$ *(ours)*	**84.76** (w/o ReCoRD)		
Multi-teacher	(T_1, T_2)-S_2 (66M)		
TMKD (Yang et al., 2020)	65.6		
MT-BERT (Wu et al., 2021a)	59.1		
RL-KD (Yuan et al., 2021)	65.3		
Uncertainty (Li et al., 2021)	65.4		
Teacher assistants	T_2-A_1-A_2-S_2 (66M)		
TAKD (Mirzadeh et al., 2020)	54.5		
DGKD (Son et al., 2021)	54.0		
Vocabulary compression	T_1-S_1, $r_v = 0.5$ (16M)		
GLMD$_{-\text{vc+mo}}$ (Sun et al., 2020b)	67.0		
GLMD$_{-\text{vc+al}}$ (Lan et al., 2020)	67.3		
GLMD$_{+\text{al}}$ *(ours)* (14M)	**67.4**		

Table 3: The average scores of all methods on the SuperGLUE dev set. The inference speed is essentially proportional to the scale of the student's parameters. Detailed results for each dataset of SuperGLUE can be found in Appendix C.

Methods	SuperGLUE
Two-stage word prediction distillation	T_1-S_1
GLMD$_{-\text{vc}}$	**67.9**
w/o $\mathbf{m}_p$ in $\mathcal{L}'_{\text{sp}}$	67.3
w/o unmasked tokens in $\mathbf{m}_p$ of $\mathcal{L}'_{\text{sp}}$	66.2
w/o $\mathcal{L}'_{\text{sp}}$ in task-specific stage	64.4
add same inter loss as TinyBERT	67.2
add $\mathcal{L}_{hard}$ in task-specific stage	67.5
replace KL with MSE in $\mathcal{L}'_{\text{sp}}$	66.7
replace KL with MSE in $\mathcal{L}_{\text{st}}$	66.7
Vocabulary compression	T_1-S_1,$r_v = 0.5$
GLMD	**67.4**
replace $Sim(\cdot)$ with Cosine similarity	66.8
replace $Sim(\cdot)$ with - Euclidean distance	66.3
replace $f_{map}(\cdot)$ with [UNK] token id	65.1
add token mapping for teacher	65.5

Table 4: Ablation study on the SuperGLUE dev set.

10B to 2B, indicating that TinyBERT is not suitable for ultra-large-scale model distillation on the SuperGLUE benchmark. The use of vocabulary compression in GLMD still maintains strong competitiveness in further compressing the model. GLMD outperforms the best vocabulary compression strategy (GLMD$_{-\text{vc}+\text{al}}$) by 0.1% on T_1-S_1 scale, confirming that reducing the vocabulary size is an effective strategy. It is worth noting that our vocabulary compression strategy can be combined with other dimensionality reduction methods, such as GLMD$_{+\text{al}}$, which can maintain the original performance even with only one-fourth of the vocabulary parameters. Additionally, some recent baselines did not show the strongest performance, we discuss more factors affecting baseline performance in appendix B.

5.3 Ablation Study

After having validated the effectiveness of GLMD and GLMD$_{-\text{vc}}$, we further analyze in Table 4 the key design factors that impact the performance of the two components in greater detail. **(1) Two-stage word prediction distillation.** The results indicate that both removing $\mathbf{m}_p$ (row 4) or removing unmasked tokens (row 5) from $\mathbf{m}_p$ do not perform as well as GLMD$_{-\text{vc}}$ (row 3), which confirms the effectiveness of $\mathbf{m}_p$ in $\mathcal{L}'_{\text{sp}}$. The use of $\mathbf{m}_p$ in $\mathcal{L}'_{\text{sp}}$ in the task-specific stage makes the distillation of the student more consistent in the pre-training and task-specific stages, which is verified by row 6. The performance degradation observed upon incorporating intermediate layer features (row 7) or hard labels (row 8) into the loss function in GLMD$_{-\text{vc}}$ further confirms that such features and labels are not necessary. Additionally, we find that the KL divergence performed better than the MSE (mean square error) in both $\mathcal{L}'_{\text{sp}}$ and $\mathcal{L}_{\text{st}}$ (rows 9 and 10). **(2) Vocabulary compression.** In addition to mapping low-frequency tokens to similar tokens using the decoder approach, we also attempt to use Cosine similarity (row 13), Euclidean distance (row 14), and direct replacement with [UNK] (row 15) to map similar tokens. We found that these mapping methods did not perform as well as GLMD (row 12), which may be because the mapping method used in GLMD is closer to the decoding approach used in language modeling task. The result of line 12 outperforming line 16 verifies that token mapping is only applicable for students.

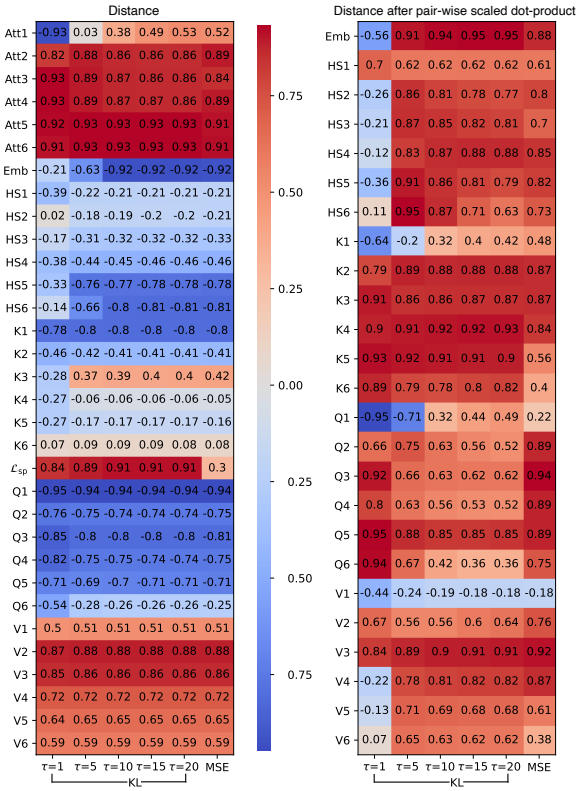

Figure 2: Spearman correlation coefficient of the $\mathcal{L}'_{\text{sp}}$ ($\tau = 15$) with the distance between teacher and student features during pre-training stage of GLMD$_{-\text{vc}}$ (T_1-S_2). Let the intermediate feature be $\mathbf{H} \in \mathbb{R}^{l \times h}$, and the distance after pair-wise scaled dot-product is calculated by first computing $f(\mathbf{H}) = \frac{\mathbf{H}\mathbf{H}^T}{\sqrt{h}}$. HS1, Att1, Q1, K1, and V1 denote the hidden state, attention scores, query, key, and value of the first layer transformer, respectively. Emb denotes the output of the embedding-layer.

6 Analysis

In this section, we analyze the reasons behind the work of GLMD and the impact of hyperparameters on performance in GLMD.

6.1 Why Does GLMD Work?

Compared to methods using only soft or hard labels, the $\mathcal{L}'_{\text{sp}}$ in GLMD$_{-\text{vc}}$ clearly provides more knowledge, but it is still unclear why the intermediate feature is not necessary. We hypothesize that $\mathcal{L}'_{\text{sp}}$ reduces inductive bias and allows the model to spontaneously learn intermediate features that should be similar to the teacher. To verify this hypothesis, we calculate the spearman correlation between the distance $d(f^S(\mathbf{H}^S), f^T(\mathbf{H}^T))$ and $\mathcal{L}'_{\text{sp}}$ during pre-training stage of GLMD$_{-\text{vc}}$. The red part in Figure 2 shows that as $\mathcal{L}'_{\text{sp}}$ decreases, not all the distance of features between teacher and student is getting close during distillation so that it

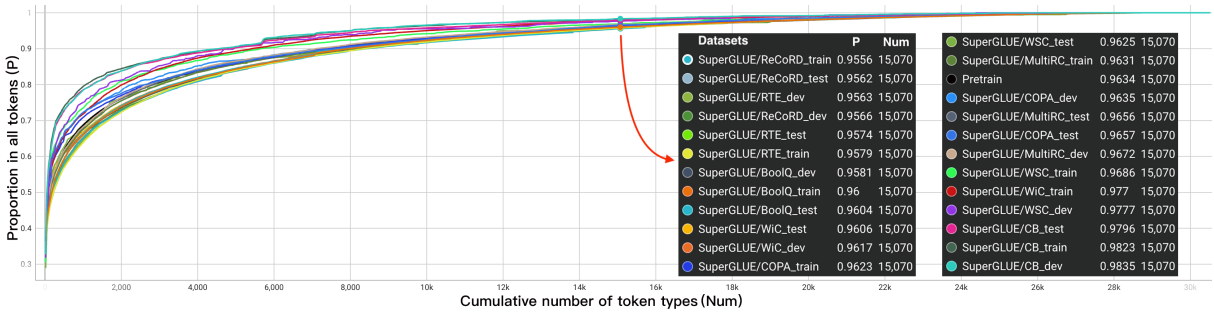

Figure 3: At least half of the tokens can cover 95.56% of the dataset corpus. We count the number of times a token appeared in the pre-trained corpus and sorted them in decreasing order on the x-axis. The y-axis shows the proportion of the 8 datasets (divided into train/dev/test) in SuperGLUE benchmark that can be covered by x tokens.

may not be necessary to draw all the intermediate features close as in existed methods, supporting our hypothesis.

We hypothesize that the success of the vocabulary compression strategy is based on the long-tail distribution of tokens, where some low-frequency tokens can still be understood by the language model after being replaced with similar tokens. Figure 3 verifies the long-tail distribution of tokens. The result in row 16 of Table 4 shows that using token mapping for teacher results in a decrease in performance. This verifies that even when some low-frequency tokens are replaced with similar tokens, students can still learn the meaning of these tokens from teachers without token mapping.

Methods	**SuperGLUE**
Two-stage word prediction distillation	T_1-S_1
GLMD$_{-\text{vc}}$	**67.9**
set $\tau = 1$ in $\mathcal{L}'_{\text{sp}}$	65.0
set $\tau = 5$ in $\mathcal{L}'_{\text{sp}}$	67.0
set $\tau = 10$ in $\mathcal{L}'_{\text{sp}}$	66.9
set $\tau = 20$ in $\mathcal{L}'_{\text{sp}}$	67.6
set $\tau = 5$ in $\mathcal{L}_{\text{st}}$	67.5
set $\tau = 10$ in $\mathcal{L}_{\text{st}}$	67.5
set $\tau = 15$ in $\mathcal{L}_{\text{st}}$	67.5
set $\tau = 20$ in $\mathcal{L}_{\text{st}}$	67.5
set $\tau = 100$ in $\mathcal{L}_{\text{st}}$	67.8
set $\tau = 200$ in $\mathcal{L}_{\text{st}}$	67.4
set $\tau = 1000$ in $\mathcal{L}_{\text{st}}$	67.6
set batch size = 8 in pre-training stage	64.7
set batch size = 16 in pre-training stage	66.2
set batch size = 32 in pre-training stage	67.1
set batch size = 128 in pre-training stage	67.6
Vocabulary compression	T_1-S_1
GLMD	**67.4**
set $r_v = 0.75$	67.2
set $r_v = 0.25$	65.8
set $r_v = 0.1$	64.6

Table 5: Hyper-parameter analysis. All combinations use the best hyperparameters of GLMD$_{-\text{vc}}$.

6.2 Hyper-parameter Analysis

In Table 5, we analyze the impact of these hyperparameters on performance: τ in $\mathcal{L}'_{\text{sp}}$ and $\mathcal{L}_{\text{st}}$, batch size in pre-training stage, and r_v in GLMD$_{-\text{vc}}$. We find that the temperature hyperparameter (τ) has a significant impact on the performance of $\mathcal{L}'_{\text{sp}}$ (rows 4-7) but little effect on $\mathcal{L}_{\text{st}}$ (rows 8-14). Similarly, in $\mathcal{L}'_{\text{sp}}$, we observe that the batch size during the pre-training stage is roughly proportional to performance (rows 15-18). The compression ratio (r_v) in our vocabulary compression strategy (rows 21-23) also follows this trend as a higher r_v results in more parameters being retained. It is worth noting that the teacher models (T_1 and T_2) required a batch size of 1024 during the pre-training process, which is significantly larger than the batch size we used in distillation.

6.3 Limitation

Due to limitations in time and computational resources, we limited our experiments to using GLM and SuperGLUE benchmark[3]. While transformer-based language models and the SuperGLUE benchmark are representative, further validation is necessary when applied to a wider range of models and tasks. Additionally, we found that the performance of GLMD$_{-\text{vc}}$ (10B→2B) at 85.28% was marginally lower than that of GLM-2B at 85.91%. However, it's noteworthy that GLM-2B leverages a substantially greater scale in the pre-training stage with a batch size, iterations, and GPU count of 7168, 17k, and 224 respectively, far exceeding the respective parameters of 64, 15k, and 8 employed by GLMD$_{-\text{vc}}$ (10B→2B) in its distillation during the pre-training stage. We plan to further investigate these potential limitations in future work.

[3]Given the requirement for grid search and seed averaging, we have run over a thousand SuperGLUE averages.

7 Conclusions

In this paper, we introduce a general language model distillation method called GLMD. GLMD has two main advantages: improving distillation performance without relying on intermediate layer features and hard labels and reducing vocabulary parameters without reducing dimensions. We also had two important findings: distillation of intermediate layer features is unnecessary, and a vocabulary compression strategy that reduces the number of tokens is feasible and can be combined with a method that reduces dimensions. In the future, we plan to explore model distillation on a 100B-scale and apply it to more real-world scenarios.

Ethical Statement

This paper aims to compress language models using knowledge distillation methods, and the proposed method do not raise ethical problems or potential biases. All language models, baselines, and datasets used in this work are publicly available and widely used.

Acknowledgements

This work is supported by Technology and Innovation Major Project of the Ministry of Science and Technology of China under Grant 2020AAA0108400 and 2020AAA0108402, the Natural Science Foundation of China under Grant No. 61836013, the Major Program of the National Social Science Foundation of China under Grant No. 18ZDA032, and funds from CCF-Zhipu.AI and Beijing Academy of Artificial Intelligence (BAAI). The GPUs used are sponsored by Zhipu.AI.

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge distillation from internal representations. In *AAAI*, pages 7350–7357.

Yoonho Boo, Sungho Shin, Jungwook Choi, and Wonyong Sung. 2021. Stochastic precision ensemble: Self-knowledge distillation for quantized deep neural networks. In *AAAI*, pages 6794–6802.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Chenhe Dong, Yaliang Li, Ying Shen, and Minghui Qiu. 2021. Hrkd: Hierarchical relational knowledge distillation for cross-domain language model compression. In *EMNLP*, pages 3126–3136.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *ACL*, pages 320–335.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *JMLR*, 20:55:1–55:21.

Hao Fu, Shaojun Zhou, Qihong Yang, Junjie Tang, Guiquan Liu, Kaikui Liu, and Xiaolong Li. 2021. Lrc-bert: Latent-representation contrastive knowledge distillation for natural language understanding. In *AAAI*, pages 12830–12838.

Md. Akmal Haidar, Nithin Anchuri, Mehdi Rezagholizadeh, Abbas Ghaddar, Philippe Langlais, and Pascal Poupart. 2022. Rail-kd: Random intermediate layer mapping for knowledge distillation. In *NAACL*, pages 1389–1400.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Shaoyi Huang, Dongkuan Xu, Ian En-Hsu Yen, Yijue Wang, Sung-En Chang, Bingbing Li, Shiyang Chen, Mimi Xie, Sanguthevar Rajasekaran, Hang Liu, and Caiwen Ding. 2022. Sparse progressive distillation: Resolving overfitting under pretrain-and-finetune paradigm. In *ACL*, pages 190–200.

Aref Jafari, Ivan Kobyzev, Mehdi Rezagholizadeh, Pascal Poupart, and Ali Ghodsi. 2022. Continuation kd: Improved knowledge distillation through the lens of continuation optimization. In *EMNLP*, page 5260–5269.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. In *EACL*, pages 2493–2504.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *EMNLP*, pages 4163–4174.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*.

Lei Li, Yankai Lin, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. 2021. Dynamic knowledge distillation for pre-trained language models. In *EMNLP*, pages 379–389.

Zheng Li, Zijian Wang, Ming Tan, Ramesh Nallapati, Parminder Bhatia, Andrew O. Arnold, Bing Xiang, and Dan Roth. 2022. Dq-bart: Efficient sequence-to-sequence model via joint distillation and quantization. In *ACL*, pages 203–211.

Chang Liu, Chongyang Tao, Jiazhan Feng, and Dongyan Zhao. 2022a. Multi-granularity structural knowledge distillation for language model compression. In *ACL*, pages 1001–1011.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yuanxin Liu, Fandong Meng, Zheng Lin, Weiping Wang, and Jie Zhou. 2021. Marginal utility diminishes: Exploring the minimum knowledge for bert knowledge distillation. In *ACL*, pages 2928–2941.

Yufan Liu, Jiajiong Cao, Bing Li, Weiming Hu, and Stephen Maybank. 2022b. Learning to explore distillability and sparsability: a joint framework for model compression. *TPAMI*, pages 1–18.

Peng Lu, Abbas Ghaddar, Ahmad Rashid, Mehdi Rezagholizadeh, Ali Ghodsi, and Philippe Langlais. 2021. Rw-kd: Sample-wise loss terms re-weighting for knowledge distillation. In *EMNLP*, pages 3145–3152.

Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. 2020. Improved knowledge distillation via teacher assistant. In *AAAI*, pages 5191–5198.

Haojie Pan, Chengyu Wang, Minghui Qiu, Yichang Zhang, Yaliang Li, and Jun Huang. 2021. Meta-kd: A meta knowledge distillation framework for language model compression across domains. In *ACL*, pages 3026–3036.

Geondo Park, Gyeongman Kim, and Eunho Yang. 2021. Distilling linguistic context for language model compression. In *EMNLP*, pages 364–378.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. Alp-kd: Attention-based layer projection for knowledge distillation. In *AAAI*, pages 13657–13665.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Wonchul Son, Jaemin Na, Junyong Choi, and Wonjun Hwang. 2021. Densely guided knowledge distillation using multiple teacher assistants. In *ICCV*, pages 9375–9384.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. In *EMNLP*, pages 4322–4331.

Siqi Sun, Zhe Gan, Yuwei Fang, Yu Cheng, Shuohang Wang, and Jingjing Liu. 2020a. Contrastive distillation on intermediate representations for language model compression. In *EMNLP*, pages 498–508.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020b. Mobilebert: a compact task-agnostic bert for resource-limited devices. In *ACL*, pages 2158–2170.

Marzieh S. Tahaei, Ella Charlaix, Vahid Partovi Nia, Ali Ghodsi, and Mehdi Rezagholizadeh. 2022. Kroneckerbert: Significant compression of pre-trained language models through kronecker decomposition and knowledge distillation. In *NAACL*, pages 2116–2127.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Lin Wang and Kuk-Jin Yoon. 2022. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *TPAMI*, 44(6):3048–3068.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *ACL*, pages 2140–2151.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *NeurIPS*.

Chuhan Wu, Fangzhao Wu, and Yongfeng Huang. 2021a. One teacher is enough? pre-trained language model distillation from multiple teachers. In *ACL*, pages 4408–4413.

Yimeng Wu, Mehdi Rezagholizadeh, Abbas Ghaddar, Md. Akmal Haidar, and Ali Ghodsi. 2021b. Universal-kd: Attention-based output-grounded intermediate layer knowledge distillation. In *EMNLP*, pages 7649–7661.

Zhengxuan Wu, Atticus Geiger, Joshua Rozner, Elisa Kreiss, Hanson Lu, Thomas Icard, Christopher Potts, and Noah D. Goodman. 2022. Causal distillation for language models. In *NAACL*, pages 4288–4295.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing bert by progressive module replacing. In *EMNLP*, pages 7859–7869.

Ze Yang, Linjun Shou, Ming Gong, Wutao Lin, and Daxin Jiang. 2020. Model compression with two-stage multi-teacher knowledge distillation for web question answering system. In *WSDM*, pages 690–698.

Fei Yuan, Linjun Shou, Jian Pei, Wutao Lin, Ming Gong, Yan Fu, and Daxin Jiang. 2021. Reinforced multi-teacher selection for knowledge distillation. In *AAAI*, pages 14284–14291.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Peng Zhang, Yuxiao Dong, and Jie Tang. 2022. GLM-130B: an open bilingual pre-trained model. *CoRR*, abs/2210.02414.

Minjia Zhang, Uma-Naresh Niranjan, and Yuxiong He. 2022. Adversarial data augmentation for task-specific knowledge distillation of pre-trained transformers. In *AAAI*, pages 11685–11693.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2022. Bert learns to teach: Knowledge distillation with meta learning. In *ACL*, pages 7037–7049.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, pages 19–27. IEEE Computer Society.

A Implementation Details

In this section, we provide a detailed overview of all baselines and hyperparameters for the benefit of researchers interested in a deeper analysis.

A.1 Baselines

In Table 6, we show the differences between 25 baseline methods in the features used. Using only hard labels for the training process is equivalent to pre-training or fine-tuning without distillation. Of these methods, 22 are specifically designed for language models, while the remaining 3 (KD, TAKD, and DGKD) are from computer vision. Figure 4 illustrates the differences between our vocabulary compression strategy and the other two strategies. Next, we provide a brief overview of these methods, as well as some strategies we adopted and adaptations for GLM.

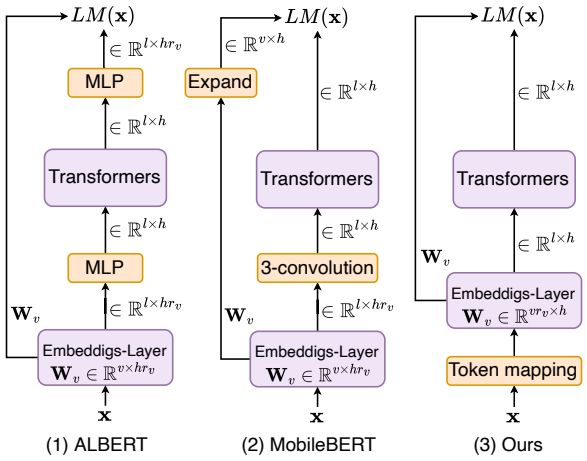

Figure 4: Details of three types of vocabulary compression strategies on the student. Where l denotes the length of the input sequence $\mathbf{x}$ (token ids) , and r_v denotes the compression rate of the vocabulary. It can be seen that our vocabulary compression method is characterized by reducing the number v of tokens rather than dimensions h.

KD (Hinton et al., 2015) was originally from computer vision and was not designed for the pre-training stage. We used randomly initialized parameters during the pre-training stage.

PD (Turc et al., 2019) removed the use of hard labels from KD, but used a pre-trained student model to initialize the student for the task-specific stage. The same hyperparameters are used for pre-training of the student model, regardless of whether distillation is performed.

PKD (Sun et al., 2019) was based on KD and added distillation loss for the [CLS] token in the intermediate layers. It was the first approach to initialize the student model for the task-specific stage by assigning some of the fine-tuned teacher's parameters to the student.

DistilBERT (Sanh et al., 2019) was the first approach to use distillation during the pre-training stage and only require fine-tuning during the task-specific stage.

Theseus (Xu et al., 2020) implemented distillation by continuously replacing intermediate layers in the teacher with smaller intermediate layers.

TinyBERT (Jiao et al., 2020) was the first approach to use distillation during both the pre-training and task-specific stages. We did not use data augmentation here.

SID (Aguilar et al., 2020) gradually increased the the number of layers for distillation as the number of epochs increased. We used the Exp3.4 strat-

Methods	Pre-training					Task-specific					Task-specific 2		
	Emb	Att	HS	Soft	Hard	Emb	Att	HS	Soft	Hard	HS	Soft	Hard
KD (Hinton et al., 2015)	random parameters								✓	✓			
PD (Turc et al., 2019)					✓				✓				
PKD (Sun et al., 2019)	truncated teacher parameters							✓	✓	✓			
DistilBERT (Sanh et al., 2019)			✓	✓	✓					✓			
Theseus (Xu et al., 2020)	truncated teacher parameters							✓		✓			
TinyBERT (Jiao et al., 2020)	✓	✓	✓			✓	✓	✓				✓	
SID (Aguilar et al., 2020)					✓		✓	✓	✓				
MobileBERT (Sun et al., 2020b)	✓	✓	✓		✓					✓			
MiniLM (Wang et al., 2020)		✓								✓			
MiniLMv2 (Wang et al., 2021)		✓								✓			
ALP-KD (Passban et al., 2021)	truncated teacher parameters							✓	✓	✓			
LRC-BERT (Fu et al., 2021)					✓			✓			✓	✓	✓
Annealing-KD (Jafari et al., 2021)					✓				✓				✓
CKD (Park et al., 2021)					✓	✓		✓	✓	✓			
Universal-KD (Wu et al., 2021b)	truncated teacher parameters							✓	✓				✓
DIITO (Wu et al., 2022)			✓	✓	✓					✓			
Continuation-KD (Jafari et al., 2022)					✓				✓	✓			
RAIL-KD (Haidar et al., 2022)	same as DistilBERT							✓	✓	✓			
MGSKD (Liu et al., 2022a)	same as TinyBERT					✓		✓				✓	
TMKD (Yang et al., 2020)				✓					✓	✓			
MT-BERT (Wu et al., 2021a)	truncated teacher parameters							✓	✓	✓			
RL-KD (Yuan et al., 2021)	truncated teacher parameters								✓	✓		✓	✓
Uncertainty (Li et al., 2021)	truncated teacher parameters								✓	✓			
TAKD (Mirzadeh et al., 2020)	truncated teacher parameters								✓	✓			
DGKD (Son et al., 2021)	truncated teacher parameters								✓	✓			

Table 6: The features used in the distillation process for the baselines implemented in GLM. A model undergoes up to three training processes (pre-training, task-specific, task-specific 2). Emb, Att, HS, Soft, and Hard denote the output of the embedding layer, attention layer (including query, key, and value), hidden state, soft labels, and hard labels, respectively.

egy from the original paper.

MobileBERT (Sun et al., 2020b) implemented a large-scale reduction of model parameters without reducing the number of model layers using an inverted-bottleneck structure. Since it required modifying the teacher's structure, we spent a week using the same hyperparameters as GLM-Large to pre-train an inverted-bottleneck structure of GLM-Large on 16 NVIDIA A100 GPUs. We used the PKT (progressive knowledge transfer) strategy from the original paper.

MiniLM (Wang et al., 2020) distilled the attention probability matrix and the value matrix of the final layer transformer during the pre-training stage.

MiniLMv2 (Wang et al., 2021) replaced the attention probability matrix from MiniLM with query and key matrices, and modified the distillation of the final layer to other layers.

ALP-KD (Passban et al., 2021) fused the features of all layers in the teacher model through an attention mechanism, allowing each layer in the student model to capture information from all layers in the teacher.

LRC-BERT (Fu et al., 2021) constructed a loss on intermediate layer features based on contrastive learning, causing the intermediate layer features of the teacher model for other samples in the same batch to be dissimilar to the intermediate layer features of the student model for the current sample. We did not use gradient perturbation as in the original work.

Annealing-KD (Jafari et al., 2021) gradually increased the weight of teacher features during the process of distilling soft targets.

Continuation-KD (Jafari et al., 2022) built upon Annealing-KD by merging two training processes at the task-specific stage, resulting in the weight of hard targets increasing with the number of iterations. In addition, soft targets were not used when the value of the soft target loss was relatively small.

CKD (Park et al., 2021) used the distance between any two or three tokens in the hidden state features as context features for the teacher and student, and then the distance between the teacher and student on these context features was used as the loss. CKD proposed task-agnostic and task-specific distillation losses, and we used task-specific distillation loss.

Universal-KD (Wu et al., 2021b) used a simi-

lar attention mechanism to ALP-KD, but applied an additional linear transformation to the intermediate layer features to ensure consistency in the hidden state dimensions of the teacher and student. The original paper provided three strategies for constructing the loss, and we adopted Universal-KD$^{(IL)}$.

DIITO (Wu et al., 2022) allowed the student model to learn counterfactual outputs by exchanging intermediate layer features between different samples. This process required two forward propagations per batch, the first to extract intermediate layer features and the second to exchange them. The original paper provided multiple strategies for aligning and exchanging the intermediate layer features, and we adopted DIITO$_{\text{FULL}} + \mathcal{L}_{\text{Cos}}^{\text{DIITO}}$.

RAIL-KD (Haidar et al., 2022) randomly used different intermediate layers of the teacher for distillation in each epoch of training in order to improve the generalization ability of the student model. It used a pre-trained distilled model of DistilBERT to initialize the task-specific stage. In cases where initialization with DistilBERT was not possible due to dimensional constraints (e.g. T_1-S_1 and T_2-S_2), we used MiniLM for initialization.

MGSKD (Liu et al., 2022a), based on CKD, used avg pooling to transform the hidden state features into features with three levels of granularity (token, span, sample) and constructed the loss on different layers using these granularities separately. For span representation, we randomly selected the token spans whose start positions and lengths are sampled from some distributions.

TMKD (Yang et al., 2020) introduced the multi-teacher method for language model distillation for the first time, with the aim of making the output of the student model as close as possible to the output of all the teacher models. There are two differences in our implementation compared to the original method: (1) We were unable to implement the multi-header layer, which transforms the output of the student model, due to the differences between GLM and BERT. (2) Since the original pre-training data is not publicly available, we used the same pre-trained corpus as other methods.

MT-BERT (Wu et al., 2021a) first used co-finetuning to fine-tune all the teachers simultaneously, and then used the reciprocal of each teacher's loss on the task as the weight for the loss between each teacher and the student. Due to the differences between GLM and BERT, the use of co-finetuning significantly degraded the performance of the teacher models, so we did not use co-finetuning.

RL-KD (Yuan et al., 2021) used reinforcement learning to select appropriate teachers for distillation at each iteration, and the final loss was the average of the loss between each selected teacher and the student. We used the $reward_1$ from the original paper as the method for calculating the reward.

Uncertainty (Li et al., 2021) used the entropy of the student's predicted results as a criterion for selecting the teacher at each iteration. The lower the entropy, the more confident the student was and the more it learned from the larger scale teacher, a process referred to as dynamic teacher adoption. We employed the hard selection strategy from the original paper.

TAKD (Mirzadeh et al., 2020), for the first time, used a teacher assistant approach in which the teacher was distilled to a mid-sized teacher assistant before being distilled to the student, rather than distilling the teacher directly to the student.

DGKD (Son et al., 2021), building upon TAKD, used all previously distilled teachers and assistants to distill the current model. It randomly discarded teachers or assistants at each iteration to serve as a regularizer.

A.2 Hyperparameters

To ensure the reproducibility of all methods, we present in Table 7 the learning rates and batch sizes for each method on each dataset in the SuperGLUE benchmark, including the hyperparameters obtained via grid search. Table 8 further shows the additional hyperparameters for the task-specific stage, which follow the settings of GLM.

B Additional Analysis

In this section, we further explore the various factors that influence the performance of the baselines and examine the necessity of intermediate layer features.

B.1 What Factors Affect Performance?

In the implementation of baselines, we discover that certain methods for initializing the student parameters led to a decrease in performance, surpassing even the advantages brought about by the method innovation. Specifically, these include the following three ways: **(1) using truncated teacher**

Methods	Size	ReCoRD	COPA	WSC	RTE	BoolQ	WiC	CB	MultiRC
		bs/lr	bs/lr	bs/lr	bs/lr	bs/lr	bs/lr	bs/lr	bs/lr
GLM (teacher)	T_1	bs (batch size) = 16, lr (learning rate) = 1E-5							
	T_2								
	T_3	bs = 64, lr = 1E-5							
PKD	T_1-S_2	32/2E-05	32/2E-05	16/2E-05	32/5E-06	16/1E-05	16/5E-06	16/2E-05	32/2E-05
DistilBERT		16/1E-05	16/2E-05	16/1E-05	16/5E-06	32/2E-05	32/2E-05	32/2E-05	16/1E-05
Theseus		32/2E-05	16/1E-05	16/1E-05	32/1E-05	16/1E-05	32/1E-05	16/2E-05	32/5E-06
SID		16/2E-05	32/5E-06	16/5E-06	16/2E-05	16/2E-05	16/2E-05	16/1E-05	16/2E-05
ALP-KD		16/2E-05	16/1E-05	16/2E-05	16/2E-05	16/2E-05	32/2E-05	16/2E-05	32/2E-05
DIITO		16/5E-06	32/1E-05	16/2E-05	16/1E-05	16/2E-05	16/1E-05	16/1E-05	16/5E-06
MobileBERT	T_1-S_1	16/1E-05	16/1E-05	32/2E-05	32/2E-05	32/2E-05	32/1E-05	32/2E-05	16/5E-06
KD	T_1-S_1	16/2E-05	16/2E-05	16/2E-05	32/5E-06	32/1E-05	32/5E-06	16/2E-05	16/5E-06
	T_1-S_2	16/5E-06	16/2E-05	16/1E-05	16/2E-05	16/2E-05	16/5E-06	16/2E-05	16/5E-06
	T_2-S_2	16/2E-05	16/1E-05	16/1E-05	32/1E-05	16/1E-05	16/5E-06	16/1E-05	32/5E-06
PD	T_1-S_1	32/2E-05	16/1E-05	16/2E-05	16/1E-05	16/2E-05	16/5E-06	16/2E-05	32/1E-05
	T_1-S_2	16/1E-05	32/5E-06	16/2E-05	16/1E-05	32/1E-05	16/5E-06	16/2E-05	16/5E-06
	T_2-S_2	16/1E-05	32/5E-06	16/2E-05	16/1E-05	16/2E-05	16/5E-06	32/2E-05	16/1E-05
TinyBERT	T_1-S_1	32/1E-05	16/5E-06	16/2E-05	32/1E-05	32/2E-05	16/2E-05	16/1E-05	32/2E-05
	T_1-S_2	32/1E-05	16/5E-06	32/5E-06	16/2E-05	16/1E-05	16/5E-06	16/1E-05	16/1E-05
	T_2-S_2	32/1E-05	32/5E-06	32/2E-05	32/1E-05	16/1E-05	16/1E-05	16/2E-05	16/2E-05
	T_3-S_3	same as GLM (teacher, T_1)							
MiniLM	T_1-S_1	16/2E-05	32/2E-05	16/1E-05	16/5E-06	32/2E-05	32/5E-06	16/2E-05	16/1E-05
	T_1-S_2	16/2E-05	32/1E-05	32/2E-05	32/1E-05	16/1E-05	16/1E-05	32/1E-05	32/2E-05
	T_2-S_2	16/1E-05	32/5E-06	16/2E-05	32/1E-05	16/5E-06	16/5E-06	16/1E-05	16/5E-06
MiniLMv2	T_1-S_1	16/1E-05	16/1E-05	32/1E-05	32/1E-05	32/2E-05	32/1E-05	16/2E-05	16/5E-06
	T_1-S_2	16/1E-05	16/1E-05	16/5E-06	32/2E-05	16/2E-05	32/2E-05	16/1E-05	16/1E-05
	T_2-S_2	16/1E-05	16/2E-05	16/2E-05	32/5E-06	16/2E-05	16/2E-05	16/2E-05	32/2E-05
LRC-BERT	T_1-S_1	16/2E-05	16/2E-05	32/2E-05	16/5E-06	16/2E-05	16/5E-06	16/5E-06	16/2E-05
	T_1-S_2	16/2E-05	32/1E-05	16/2E-05	32/1E-05	16/2E-05	16/5E-06	16/2E-05	16/5E-06
	T_2-S_2	16/2E-05	32/5E-06	32/5E-06	32/5E-06	16/2E-05	32/5E-06	32/2E-05	16/2E-05
Annealing-KD	T_1-S_1	16/2E-05	32/5E-06	32/2E-05	32/2E-05	16/1E-05	32/5E-06	32/2E-05	32/5E-06
	T_1-S_2	16/2E-05	16/5E-06	16/2E-05	16/2E-05	16/2E-05	32/5E-06	16/1E-05	32/5E-06
	T_2-S_2	16/2E-05	16/1E-05	32/2E-05	32/5E-06	32/2E-05	32/2E-05	32/1E-05	16/5E-06
CKD	T_1-S_1	32/1E-05	16/5E-06	16/2E-05	16/5E-06	16/2E-05	32/5E-06	32/2E-05	16/5E-06
	T_1-S_2	32/2E-05	16/2E-05	16/5E-06	16/1E-05	16/2E-05	16/1E-05	16/1E-05	32/2E-05
	T_2-S_2	16/1E-05	32/1E-05	16/1E-05	32/1E-05	16/2E-05	32/1E-05	16/2E-05	32/2E-05
Universal-KD	T_1-S_1	16/2E-05	32/2E-05	32/2E-05	32/5E-06	16/2E-05	16/1E-05	32/2E-05	16/2E-05
	T_1-S_2	32/2E-05	32/5E-06	32/5E-06	32/1E-05	32/5E-06	16/5E-06	16/1E-05	16/1E-05
	T_2-S_2	16/5E-06	32/1E-05	32/2E-05	16/2E-05	16/1E-05	16/5E-06	32/1E-05	32/2E-05
Continuation-KD	T_1-S_1	16/2E-05	32/5E-06	16/2E-05	32/2E-05	32/2E-05	32/2E-05	16/2E-05	32/1E-05
	T_1-S_2	16/2E-05	32/1E-05	16/1E-05	16/1E-05	16/2E-05	32/1E-05	16/1E-05	16/5E-06
	T_2-S_2	16/1E-05	16/1E-05	16/2E-05	16/2E-05	16/1E-05	16/5E-06	16/1E-05	32/5E-06
RAIL-KD	T_1-S_1	16/1E-05	32/1E-05	32/2E-05	32/1E-05	16/1E-05	32/5E-06	16/2E-05	32/2E-05
	T_1-S_2	16/1E-05	16/1E-05	16/2E-05	16/5E-06	32/2E-05	16/1E-05	32/1E-05	32/2E-05
	T_2-S_2	32/2E-05	16/2E-05	32/2E-05	16/5E-06	16/1E-05	16/5E-06	16/2E-05	16/1E-05
MGSKD	T_1-S_1	32/1E-05	16/5E-06	32/2E-05	32/1E-05	16/1E-05	32/1E-05	32/5E-06	32/5E-06
	T_1-S_2	16/5E-06	16/2E-05	32/2E-05	16/5E-06	16/5E-06	16/1E-05	32/2E-05	32/5E-06
	T_2-S_2	32/2E-05	32/5E-06	32/5E-06	32/5E-06	16/2E-05	16/5E-06	16/2E-05	16/2E-05
GLMD$_{-\text{vc}}$	T_1-S_1	16/2E-05	32/2E-05	32/2E-05	16/5E-06	16/2E-05	16/2E-05	32/2E-05	16/1E-05
	T_1-S_2	16/2E-05	16/5E-06	32/2E-05	16/2E-05	16/2E-05	16/2E-05	32/2E-05	16/5E-06
	T_2-S_2	16/1E-05	32/1E-05	32/2E-05	32/2E-05	16/2E-05	16/2E-05	32/2E-05	16/5E-06
	T_3-S_3	same as GLM (teacher, T_1)							
GLMD	T_1-S_1	same as GLMD$_{-\text{vc}}$ (T_1-S_1)							
	T_1-S_2	same as GLMD$_{-\text{vc}}$ (T_1-S_2)							
	T_2-S_2	same as GLMD$_{-\text{vc}}$ (T_2-S_2)							
TMKD MT-BERT RL-KD Uncertainty	(T_1, T_2)-S_2	same as GLM (teacher, T_1)							
TAKD DGKD	T_2-A_1-A_2-S_2	same as KD (T_1-S_2)							
GLMD$_{-\text{vc+mo}}$ GLMD$_{-\text{vc+al}}$ GLMD$_{+\text{al}}$	T_1-S_1	same as GLMD (T_1-S_1)							

Table 7: Hyperparameters for all methods in Table 3 on the 8 datasets of the SuperGLUE benchmark.

Hyperparameters	ReCoRD	COPA	WSC	RTE	BoolQ	WiC	CB	MultiRC
Sequence length	512	256	128	256	256	256	256	512
Epochs	5	50	50	50	20	30	50	15
Dropout	0.1							
Attention Dropout	0.1							
Warmup Ration	0.1							
Weight Decay	0.1							
Learning Rate Decay	Linear							
Adam ϵ	1E-8							
Adam β_1	0.9							
Adam β_2	0.999							
Gradient Clipping	0.1							

Table 8: Other hyperparameters for the task-specific stage.

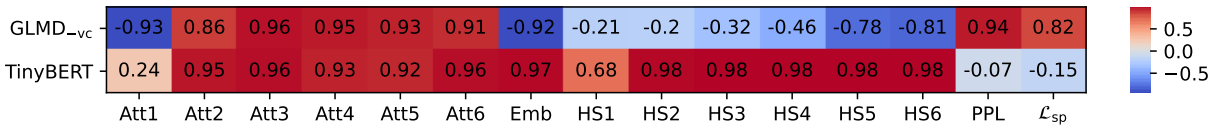

Figure 5: Spearman correlation coefficient of the loss function values with the distance between teacher and student features during pre-training stage of GLMD$_{-vc}$ and TinyBERT (T_1-S_2). Att1 and HS1 denote the attention scores and hidden states, respectively, of the first layer transformer. Emb denotes the output of the embedding layer. The KL divergence with a temperature τ of 1 is used to compute the distance between the teacher and student models in the Att, while the mean square error is used for the HS and Emb. PPL denotes the language modeling perplexity of the student on the validation set.

parameters in the case of different dimensions of teacher and student hidden layers. Many methods that do not distill in the pre-training stage will use the parameters of the first few layers of the fine-tuned teacher as student parameters in a task-specific stage. In the case of different dimensions of teacher and student hidden layers, we can only truncate some parameters per layer. As shown in Table 3, Universal-KD (Wu et al., 2021b) in T_1-S_2 performs much better than T_1-S_2 and T_2-S_2, TAKD (Mirzadeh et al., 2020) and DGKD (Son et al., 2021) also performs badly due to this reason. **(2) Using less data to pre-train a student model as initialization.** To ensure fairness, regardless of whether distillation is used, we set the batch size of all methods in the pre-training stage to 64, which is equivalent to only using one-sixteenth of the full data (batch size=1024). Some methods that use a pre-trained student as initialization for a task-specific stage may be affected by this, for example, PD (Turc et al., 2019), SID (Aguilar et al., 2020), LRC-BERT (Fu et al., 2021), Annealing-KD (Jafari et al., 2021), Continuation-KD (Jafari et al., 2022), and CKD (Park et al., 2021). **(3) Randomly initializing the parameters of the student model.** As can be seen from Table 3, KD (Hinton et al., 2015) using random initialization are obviously inferior to PD (Turc et al., 2019) using pre-trained students and soft labels.

The above analysis demonstrates that methods without pre-training distillation are sensitive to the initialization of the student's parameters. To achieve optimal performance, methods based on truncating teacher parameters require the hidden dimensions of the teacher and student to be identical or else, other methods would require a significant cost in pre-training a student model. Therefore, utilizing a subset of the corpus for knowledge distillation during the pre-training stage is a more favorable option.

B.2 Why are Intermediate Layers not Necessary?

In Section 6.1, we have verified that the $\mathcal{L}'_{sp}$ in GLMD$_{-vc}$ can enable the model to spontaneously learn intermediate layer features that should be similar to those of the teacher. We further validate in Figure 5 that training with a loss function focused on the intermediate layer features does not lead to a reduction in $\mathcal{L}'_{sp}$ and cannot even lower the perplexity (PPL) of language modeling. In the process of distillation using GLMD$_{-vc}$ and TinyBERT methods, we have quantified the spearman correlation between the distance of student and teacher features, including the perplexity of the student model on the validation set, and the loss function values. We observe that there is no correlation between

the loss function values of TinyBERT and $\mathcal{L}'_{\text{sp}}$, nor with the perplexity of the validation set. This suggests that we may not require a significant inductive bias towards the intermediate layer features.

C Detailed Results

Due to space constraints, we do not present results for all datasets in the SuperGLUE benchmark in the main text but only show the averages. Table 9 shows the results for all methods on each dataset in the SuperGLUE benchmark, rounded to two decimal places.

Methods	Size	ReCoRD	COPA	WSC	RTE	BoolQ	WiC	CB	MultiRC	avg
		F1/Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	F1/Acc.	$F1_a$/EM	
GLM (teacher)	T_1	72.80/72.17	66.00	77.88	72.92	79.39	66.14	88.19/91.07	72.32/26.34	71.72
	T_2	80.08/79.54	78.00	81.73	79.78	82.63	70.06	86.33/89.29	76.39/37.67	77.11
	T_3	94.56/94.13	97.00	95.19	90.61	88.81	74.61	96.05/94.64	88.10/63.38	88.96
PKD	T_1-S_2	61.77/60.99	60.00	65.38	68.83	77.73	65.78	82.76/85.12	69.99/22.67	66.17
DistilBERT		59.79/59.05	65.00	68.59	60.89	73.39	60.34	77.48/83.33	66.98/17.38	63.78
Theseus		57.07/56.33	61.67	66.35	68.11	77.81	64.37	89.14/87.50	69.08/21.79	66.09
SID		27.17/26.19	65.00	65.06	58.12	69.33	57.16	51.02/73.81	59.26/14.55	55.08
ALP-KD		57.72/56.90	60.67	64.74	68.11	77.20	64.79	74.82/79.76	68.21/19.90	64.27
DIITO		63.71/63.00	72.00	69.23	65.46	75.46	60.76	86.75/85.12	66.28/17.63	66.77
MobileBERT	T_1-S_1	59.29/58.61	65.33	68.59	58.97	74.61	63.85	86.65/88.69	66.87/19.41	65.14
KD	T_1-S_1	22.43/21.74	58.67	64.74	54.75	65.98	56.69	65.56/72.02	46.20/2.66	52.02
	T_1-S_2	22.66/21.99	61.67	63.46	54.63	66.07	57.05	61.75/72.02	51.98/2.41	52.41
	T_2-S_2	23.07/22.38	59.33	65.71	54.15	66.12	56.79	69.19/74.40	56.56/1.50	53.21
PD	T_1-S_1	46.54/45.90	66.33	67.95	58.48	69.93	59.67	81.78/80.95	63.91/12.42	61.01
	T_1-S_2	54.36/53.59	65.67	66.67	59.45	69.82	59.20	80.13/81.55	65.97/15.29	62.03
	T_2-S_2	54.00/53.22	65.33	64.74	60.29	69.94	58.41	76.66/79.76	66.05/15.15	61.39
TinyBERT	T_1-S_1	56.13/55.53	69.00	69.87	70.04	76.93	65.41	84.86/85.12	70.60/22.39	67.32
	T_1-S_2	65.60/64.88	70.33	75.00	71.96	77.97	67.87	89.58/89.88	71.37/25.74	70.83
	T_2-S_2	66.61/65.86	67.00	63.46	71.12	79.98	66.46	90.56/88.69	70.42/27.35	69.09
	T_3-S_3	-	61.00	65.38	67.87	74.46	63.32	70.49/78.57	70.76/27.39	65.09
MiniLM	T_1-S_1	51.01/50.27	63.67	60.90	67.63	73.72	61.29	68.87/77.98	66.93/16.19	61.60
	T_1-S_2	60.00/59.24	62.00	63.46	67.63	75.88	64.99	67.63/79.17	67.36/19.66	63.81
	T_2-S_2	57.07/56.39	62.33	64.42	66.43	74.00	60.92	65.38/79.76	66.81/17.45	62.44
MiniLMv2	T_1-S_1	51.85/51.25	65.00	60.58	67.63	75.17	61.70	65.81/79.76	66.26/17.98	62.07
	T_1-S_2	60.88/60.16	62.00	62.82	66.67	76.73	63.69	66.38/76.79	68.68/21.65	63.65
	T_2-S_2	64.08/63.29	59.33	65.71	66.19	77.26	65.05	86.84/85.71	68.40/21.44	66.05
LRC-BERT	T_1-S_1	40.44/39.69	64.33	66.03	54.87	68.84	56.74	78.68/80.36	59.66/13.61	58.38
	T_1-S_2	55.10/54.44	65.67	66.67	56.56	74.86	57.63	80.27/81.55	65.75/16.16	62.25
	T_2-S_2	51.83/51.17	66.33	63.78	58.12	72.02	59.04	68.08/75.00	66.69/21.13	60.78
Annealing-KD	T_1-S_1	49.18/48.55	66.33	65.38	58.97	69.68	58.36	82.73/81.55	63.96/12.91	61.02
	T_1-S_2	56.08/55.39	69.33	66.67	58.97	70.57	59.82	85.78/85.12	66.26/13.92	63.33
	T_2-S_2	55.57/54.89	69.00	68.59	58.24	71.48	58.88	83.11/83.33	66.85/13.43	63.10
CKD	T_1-S_1	48.82/48.27	66.33	64.74	59.57	70.65	60.76	87.02/85.71	63.72/12.28	61.87
	T_1-S_2	56.35/55.65	65.00	66.67	61.25	71.63	58.83	88.61/84.52	66.11/15.22	63.33
	T_2-S_2	56.29/55.57	65.00	65.71	58.00	71.39	58.46	86.17/84.52	66.49/14.62	62.55
Universal-KD	T_1-S_1	24.08/23.27	61.00	66.03	55.48	65.93	56.79	60.38/73.21	53.17/2.17	52.92
	T_1-S_2	58.67/57.83	58.67	66.67	70.16	77.56	65.52	87.52/85.71	69.96/22.63	66.22
	T_2-S_2	24.51/23.71	64.00	67.63	55.84	66.47	58.52	66.11/75.60	56.39/1.22	54.53
Continuation-KD	T_1-S_1	48.63/48.01	66.33	66.03	58.97	69.12	58.31	83.72/81.55	63.38/13.26	61.00
	T_1-S_2	55.61/54.91	68.67	64.74	58.72	71.42	58.25	85.61/83.93	66.64/13.33	62.73
	T_2-S_2	55.15/54.38	67.00	65.38	57.64	70.91	58.20	78.79/80.36	66.80/13.96	61.73
RAIL-KD	T_1-S_1	51.49/50.83	62.67	67.31	64.50	71.93	60.45	65.91/75.60	65.06/16.79	61.21
	T_1-S_2	59.85/59.19	66.67	70.19	60.53	69.00	60.34	78.98/83.33	66.55/15.60	63.56
	T_2-S_2	50.26/49.51	62.00	65.06	59.33	72.30	59.46	73.17/78.57	63.57/15.01	60.40
MGSKD	T_1-S_1	34.03/33.26	65.33	61.86	64.98	70.20	61.23	70.98/77.98	64.38/12.63	58.78
	T_1-S_2	50.29/49.49	65.00	65.06	65.94	73.31	63.17	83.89/84.52	67.32/15.56	63.50
	T_2-S_2	43.68/42.88	63.00	63.78	57.76	72.46	60.55	85.64/85.71	55.27/10.74	59.94
$GLMD_{-vc}$	T_1-S_1	59.06/58.33	74.00	72.12	66.19	76.17	64.26	86.96/86.90	67.97/22.11	**67.92**
	T_1-S_2	68.66/67.90	72.00	75.96	70.16	78.92	67.08	91.59/89.88	70.29/27.11	**71.48**
	T_2-S_2	70.51/69.82	72.67	73.08	71.60	79.96	66.93	93.08/91.67	71.76/28.86	72.13
	T_3-S_3	89.35/88.50	89.00	86.54	87.36	86.24	74.14	100.00/100.00	84.78/55.30	**85.28**
GLMD	T_1-S_1	57.98/57.23	71.33	68.27	65.10	76.41	63.95	91.99/90.48	68.61/21.79	67.39
	T_1-S_2	66.13/65.44	72.33	75.00	71.12	78.30	66.25	89.61/90.48	70.53/25.78	70.87
	T_2-S_2	68.97/68.32	75.33	74.36	71.84	80.37	65.78	92.90/90.48	71.64/28.09	**72.23**
TMKD	(T_1,T_2)-S_2	65.77/65.09	70.33	63.14	66.91	75.37	63.38	70.22/79.17	68.76/22.77	65.63
MT-BERT		46.81/46.08	59.00	63.46	65.46	66.90	62.33	78.76/80.36	57.53/2.06	59.12
RL-KD		59.78/58.99	58.33	66.03	69.07	77.93	65.78	76.87/82.74	69.24/22.21	65.26
Uncertainty		58.52/57.67	59.33	64.10	70.16	77.55	65.78	80.85/83.33	69.47/22.49	65.39
TAKD	T_2-A_1-A_2-S_2	25.50/24.69	60.33	66.03	55.11	66.39	57.94	76.28/76.79	55.90/1.50	54.52
DGKD		23.68/22.96	61.00	66.99	55.96	65.71	58.73	75.45/75.60	48.06/1.50	54.00
$GLMD_{-vc+mo}$	T_1-S_1	59.56/58.85	67.67	70.51	68.35	77.41	64.99	81.48/82.74	68.78/22.74	67.00
$GLMD_{-vc+al}$		59.79/59.13	69.67	65.38	71.12	76.95	64.37	84.81/88.10	68.76/21.76	67.33
$GLMD_{+al}$		58.74/58.06	70.67	70.19	69.55	76.82	63.11	85.18/85.71	68.24/22.14	**67.42**

Table 9: Detailed results for all methods in Table 3 on the 8 datasets of the SuperGLUE benchmark.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
6.3 Limitation

☑ A2. Did you discuss any potential risks of your work?
Ethical Statement

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and 1 Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

5 Experiments

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
5 Experiments

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
5.1 Experimental Setup and A.2 Hyperparameters

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
5.1 Experimental Setup

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
5.1 Experimental Setup

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.