Knowledge Inheritance for Pre-trained Language Models

Anonymous ACL submission

Abstract

Recent explorations of large-scale pre-trained language models (PLMs) such as GPT-3 have revealed the power of PLMs with huge amounts of parameters, setting off a wave of training ever-larger PLMs. However, training a large-scale PLM requires tremendous amounts of computational resources, which is time-consuming and expensive. In addition, existing large-scale PLMs are mainly trained from scratch individually, ignoring the availability of many existing well-trained PLMs. To this end, we explore the question that how can previously trained PLMs benefit training larger PLMs in future. Specifically, we introduce a novel pre-training framework named "knowledge inheritance" (KI), which combines both self-learning and teacher-guided learning to efficiently train larger PLMs. Experimental results demonstrate the superiority of our KI framework. We also conduct empirical analyses to explore the effects of teacher PLMs' pre-training settings, including model architecture, pre-training data, etc. Finally, we show that KI can well support lifelong learning and knowledge transfer. All source code and model parameters will be available to advance further research explorations.

1 Introduction

Recently, huge efforts have been devoted to pre-trained language models (PLMs), targeting at acquiring versatile syntactic and semantic knowledge from large-scale corpora (Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2019). By taking full advantage of the rich knowledge distributed in these PLMs, the state-of-the-art across a wide range of NLP tasks is continuously being pushed. Up to now, it has become a consensus in the NLP community to use PLMs as the backbone for downstream tasks. Despite the great follow-up efforts of exploring various pre-training techniques and model architectures, researchers find that simply enlarging the model capacity, data size, and training steps can further improve the performance of PLMs (Kaplan et al., 2020). This discovery sets off a wave of training large-scale PLMs, from GPT-3 (Brown et al., 2020) with hundreds of billions of parameters, to Switch-Transformer (Fedus et al., 2021) with trillions of parameters.

Although these huge PLMs have shown awesome performance, especially the amazing ability of zero-shot and few-shot learning, training large-scale PLMs requires tremendous amounts of computational resources. For example, about $10,000$ GPUs were used to train GPT-3, costing millions of dollars at a rough estimate. Therefore, severe environmental concerns on the prohibitive computational costs have been raised. Moreover, existing PLMs are generally trained from scratch individually, ignoring the availability of many well-trained PLMs. In contrast, humans can leverage the knowledge summarized by their predecessors to learn new tasks, so that the learning process could become efficient. This leaves us an important question: how can previously trained PLMs benefit learning larger PLMs in future?

We argue that the implicit knowledge distributed in different PLMs is inheritable. In order to train a larger PLM, it is worth reusing the knowledge summarized and organized by an existing well-trained PLM, which is similar to the learning process of human beings. More specifically, different from learning from scratch, we introduce a novel pre-training framework, named "**knowledge inheritance**" (KI), which combines both self-learning and teacher-guided learning to efficiently train larger PLMs. Intuitively, such a process of inheriting knowledge from teachers is much more efficient and effective than the common practice of self-learning.

To some extent, the process of KI is similar to Knowledge Distillation (KD) (Hinton et al., 2015), which transfers the knowledge from a high-capacity teacher model to a more compact student model.

However, conventional KD methods presume that teacher models play pivotal roles in mastering knowledge, and student models with smaller capacities generally cannot match their teachers in performance. When it comes to the scenario of KI, since student models have larger capacities, the performance of teacher models is no longer an "upper bound" of student models, leading to many challenges that have not been encountered in KD.

In addition, as more and more PLMs with different pre-training settings (model architectures, training data, training strategies, etc) emerge, it is unclear how these different settings will affect the performance of KI. Besides, human beings excel at learning knowledge in a lifelong manner, that is, incrementally acquiring, refining, and transferring knowledge. In real scenarios where data is streaming, whether larger PLMs can continuously inherit the special skills from multiple smaller teachers and evolve is unanswered. Lastly, the ability to hand down knowledge from generation to generation is also vital for PLMs, which is not considered in conventional KD methods either.

In this paper, we propose a general KI framework that leverages previously trained PLMs for training larger ones. We carry out thorough experiments to rigorously evaluate the feasibility of KI. We also systematically conduct empirical analyses to show the effects of various teacher pre-training settings, which may indicate how to select the most appropriate PLM as the teacher for KI. We further extend the above framework and show that an already trained large PLM can continuously inherit new knowledge from multiple models pre-trained on different specific domains; the newly learned knowledge can further be passed down to descendants. This demonstrates that our KI framework can well support lifelong learning and knowledge transfer, providing a promising direction to share and exchange the knowledge learned by different models and continuously promote their performance.

2 Knowledge Inheritance Framework

Background. A PLM $\mathcal{M}$ generally consists of an embedding layer and N Transformer layers. Given a textual input $\mathbf{x} = \{x^1, \ldots, x^n\}$ and the corresponding label $\mathbf{y} \in \mathbb{R}^K$, where K is the number of classes for the specific pre-training task, e.g., the vocabulary size for masked language modeling (MLM) (Devlin et al., 2019), $\mathcal{M}$ first converts $\mathbf{x}$ to an embedding matrix $\mathbf{H}_0 = [\mathbf{h}_0^1, ..., \mathbf{h}_0^n]$, which is then encoded by the Transformer layers into representations $\mathbf{H}_l = [\mathbf{h}_l^1, ..., \mathbf{h}_l^n]$ at different levels as follows: $[\mathbf{h}_l^1, \ldots, \mathbf{h}_l^n] = \text{Transformer}_l([\mathbf{h}_{l-1}^1, \ldots, \mathbf{h}_{l-1}^n])$, where $l \in \{1, 2, ..., N\}$. Upon these representations, a classifier $\mathcal{F}$ is applied to produce task-specific logits $\mathbf{z}^j = [z_1^j, ..., z_K^j] = \mathcal{F}(\mathbf{h}_N^j)$ for token x^j. Each logit is converted to a probability distribution $\mathcal{P}(x^j; \tau) = [p_1(x^j; \tau), ..., p_K(x^j; \tau)]$ by comparing with other logits using a softmax function with temperature τ. $\mathcal{M}$ is pre-trained with the objective $\mathcal{L}_{\text{SELF}}(\mathbf{x}, \mathbf{y}) = \mathcal{H}(\mathbf{y}, \mathcal{P}(\mathbf{x}; \tau))$, where $\mathcal{H}$ is the loss function, e.g., cross-entropy for MLM.

Knowledge Inheritance. The goal of knowledge inheritance is to train a large PLM $\mathcal{M}_L$ on the corpora $\mathcal{D}_L = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{|\mathcal{D}_L|}$. The common practice of training $\mathcal{M}_L$ is to directly optimize $\mathcal{L}_{\text{SELF}}$ on $\mathcal{D}_L$. We first consider a simple scenario that we have a well-trained small PLM $\mathcal{M}_S$ optimized on $\mathcal{D}_L$ with the self learning objective (such as MLM) $\mathcal{L}_{\text{SELF}}$. Since we have already trained a smaller PLM $\mathcal{M}_S$, it is worth inheriting the knowledge summarized and organized by $\mathcal{M}_S$, so that $\mathcal{M}_L$ can at least achieve the same performance as the teacher requiring minor effort. Such a process is far more efficient than learning on $\mathcal{M}_L$'s own, which is aligned with humans' learning experience that, having a knowledgeable teacher to guide students and clarify their faults is more effective than self-learning. More specifically, imparting $\mathcal{M}_S$'s knowledge to $\mathcal{M}_L$ on $\mathcal{D}_L$ is implemented by minimizing the Kullback-Leibler (KL) divergence between two probability distributions output by $\mathcal{M}_S$ and $\mathcal{M}_L$ on the same input $\mathbf{x}_i \in \mathcal{D}_L$, i.e., $\mathcal{L}_{\text{KI}}(\mathbf{x}_i; \mathcal{M}_S) = \tau^2 \text{KL}(\mathcal{P}_{\mathcal{M}_S}(\mathbf{x}_i; \tau) || \mathcal{P}_{\mathcal{M}_L}(\mathbf{x}_i; \tau))$. In addition to teacher-guided learning, $\mathcal{M}_L$ is also encouraged to conduct self-learning by optimizing $\mathcal{L}_{\text{SELF}}(\mathbf{x}_i, \mathbf{y}_i)$. To control how much we want to trust the knowledge from the teacher, we set an inheritance rate α to balance $\mathcal{L}_{\text{SELF}}$ and $\mathcal{L}_{\text{KI}}$:

$$
\begin{aligned}
\mathcal{L}(\mathcal{D}_L; \mathcal{M}_S) &= \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_L} (1-\alpha) \mathcal{L}_{\text{SELF}}(\mathbf{x}_i, \mathbf{y}_i) + \alpha \mathcal{L}_{\text{KI}}(\mathbf{x}_i; \mathcal{M}_S) \\
&= \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_L} (1-\alpha) \mathcal{H}(\mathbf{y}_i, \mathcal{P}_{\mathcal{M}_L}(\mathbf{x}_i; 1)) \\
&+ \alpha \tau^2 \text{KL}(\mathcal{P}_{\mathcal{M}_S}(\mathbf{x}_i; \tau) || \mathcal{P}_{\mathcal{M}_L}(\mathbf{x}_i; \tau))).
\end{aligned} \tag{1}
$$

Dynamic Inheritance Rate. However, since larger models generally converge faster and can achieve better final performance (Li et al., 2020b), the PLM $\mathcal{M}_L$ can be seen as a student, who is a fast learner, but with poorer knowledge at first. This is different from conventional KD, where the teacher's ability is the upper bound for the student. In KI, since the student's learning ability is better than the teacher, it becomes more and more knowledgeable during the learning process, and will surpass the teacher eventually. Thus, it is necessary to encourage $\mathcal{M}_L$ increasingly learning knowledge on its own, not only memorizing the teacher's instructions. This can be done by dynamically changing the inheritance rate α to balance $\mathcal{L}_{\text{SELF}}$ and $\mathcal{L}_{\text{KI}}$. Additionally, after $\mathcal{M}_L$ has surpassed its teacher, it no longer needs the guidance from $\mathcal{M}_S$ and should conduct pure self-learning from then on. To implement this, for a total training steps of T, we choose the α_t that is linearly decayed with a slope of $\frac{\alpha_T}{T}$ and the student only inherits knowledge from the teacher for $\frac{T}{\alpha_T}$ steps, and then conducts pure self-learning, i.e., $\alpha_t = \max(1 - \alpha_T \times \frac{t}{T}, 0)$. Specifically, at step t, the loss function for inheriting knowledge of $\mathcal{M}_S$ on $\mathcal{D}_L$ is formulated as follows:

$$\mathcal{L}(\mathcal{D}_L; \mathcal{M}_S) = \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_L} (1-\alpha_t)\mathcal{L}_{\text{SELF}}(\mathbf{x}_i, \mathbf{y}_i) + \alpha_t \mathcal{L}_{\text{KI}}(\mathbf{x}_i; \mathcal{M}_S). \tag{2}$$

Note the logits of $\mathcal{M}_S$ on $\mathcal{D}_L$ can be pre-computed and saved offline so that we do not need to re-compute the inference of $\mathcal{M}_S$ when training $\mathcal{M}_L$. This process is done once and for all. KI does not require the access to $\mathcal{M}_S$'s parameters, which may be not available due to privacy issues.

Diverse Teachers & Domains. In real world scenarios, we generally have a series of well-trained smaller PLMs $\overline{\mathcal{M}_S} = \{\mathcal{M}_S^1, ..., \mathcal{M}_S^{N_S}\}$, each having been optimized on $\overline{\mathcal{D}_S} = \{\mathcal{D}_S^1, ..., \mathcal{D}_S^{N_S}\}$, respectively, and thus gained sufficient knowledge on the corresponding corpus. Considering that the PLMs in $\overline{\mathcal{M}_S}$, consisting of varied model architectures, are pre-trained on different corpora of various sizes and domains with arbitrary strategies, thus the knowledge they master is also manifold, making it beneficial to let $\mathcal{M}_L$ continuously absorb knowledge from each teacher. In addition, $\mathcal{M}_L$'s pre-training data $\overline{\mathcal{D}_L}$ may also consist of massive, heterogeneous corpora from multiple sources, i.e., $\overline{\mathcal{D}_L} = \{\mathcal{D}_L^1, ..., \mathcal{D}_L^{N_L}\}$. Due to the difference between $\overline{\mathcal{D}_L}$ and $\overline{\mathcal{D}_S}$, $\mathcal{M}_S$ may be required to transfer its knowledge on instances unseen during its pre-training. Ideally, we want $\mathcal{M}_S$ to teach the courses it is skilled in so that $\mathcal{M}_L$ can make the best of teacher models. To better summarize the hybrid knowledge of $\overline{\mathcal{D}_L}$, it is essential to choose the most appropriate teacher $\mathcal{M}_S^* = \texttt{optimal}(\overline{\mathcal{M}_S}|\mathcal{D}_L^*)$ for each composition $\mathcal{D}_L^* \in \overline{\mathcal{D}_L}$, where $\texttt{optimal}$ denotes the teacher selection strategy. We will analyze the effects that contribute to the optimal strategy in the next section. The overall formulation for inheriting knowledge from $\overline{\mathcal{M}_S}$ on $\overline{\mathcal{D}_L}$ is:

$$\mathcal{L}(\overline{\mathcal{D}_L}; \overline{\mathcal{M}_S}) = \sum_{i=1}^{N_L} \mathcal{L}(\mathcal{D}_L^i; \texttt{optimal}(\overline{\mathcal{M}_S}|\mathcal{D}_L^i)) \tag{3}$$

3 Experiments

In this section, we first present a preliminary experiment to demonstrate the effectiveness of KI framework in § 3.1. Then we conduct empirical analyses to show the effects of different pre-training settings of the teacher models in § 3.2. Finally, we show KI can well support lifelong learning and knowledge transfer so that PLMs can continuously absorb knowledge from multiple teachers in § 3.3, and PLMs can accumulate knowledge over generations in § 3.4. All these results show that our KI framework can make training larger PLMs effective and efficient by taking advantage of existing smaller PLMs. For a fair comparison, we train all models in the same computation environment with 8 NVIDIA 32GB V100 GPUs. Detailed hyper-parameters for pre-training are listed in our appendix.

3.1 Preliminary Experiments

Setting. Our KI framework is agnostic to the specific self-supervised pre-training task. Without loss of generality, we focus on the representative MLM task in the main paper and discuss auto-regressive language modeling in our appendix. We use the model structure of RoBERTa (Liu et al., 2019). In § 3.1, we first choose RoBERTa$_{\texttt{BASE}}$ (denoted as `BASE`) as the teacher ($\mathcal{M}_S$) architecture and RoBERTa$_{\texttt{LARGE}}$ (denoted as `LARGE`) as the student ($\mathcal{M}_L$) architecture.

For pre-training data, we use the concatenation of Wikipedia and BookCorpus (Zhu et al., 2015) same as BERT (Devlin et al., 2019), with roughly $3,400$M tokens in total. The training-validation ratio is set to $199:1$. All models are trained for 125k steps, with a batch size of $2,048$ and a sequence length of 512, and we ensure that they have well converged in the end. Note the whole training computational cost is approximately equivalent to that of BERT. We first pre-train $\mathcal{M}_S$ and then pre-train $\mathcal{M}_L$ by inheriting $\mathcal{M}_S$'s knowledge under KI (denoted as "`BASE` $\rightarrow$ `LARGE`"). We compare it with "`LARGE`" that only conducts self-learning from beginning to end.

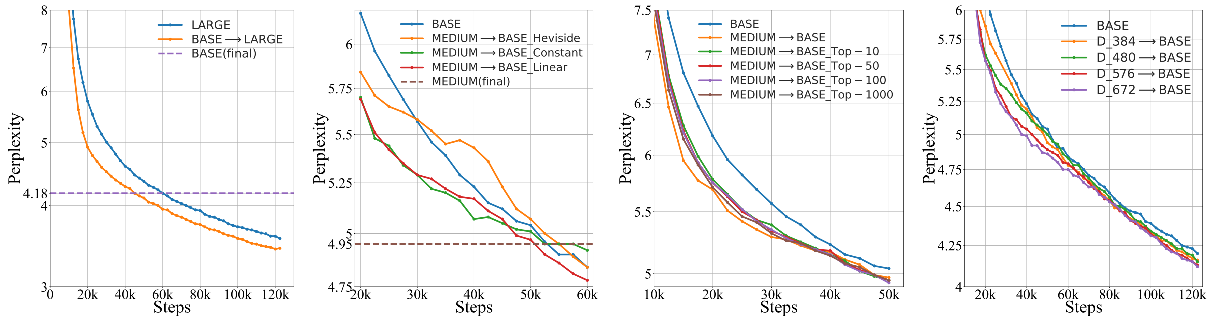

Figure 1: From left to right: (1) the validation PPL curve for pre-training $\mathcal{M}_L$ under KI framework (BASE $\rightarrow$ LARGE) and the self-learning baseline (LARGE). The teacher's (BASE) performance is 4.18. (2) Pre-training BASE under KI with three strategies for the inheritance rate α_t: Linear, Heviside and Constant. The teacher's (MEDIUM) performance is 4.95. (3) Pre-training BASE under KI with top-K logits, we vary K in $\{10, 50, 100, 1000\}$, respectively. (4) Effects of $\mathcal{M}_S$'s model architecture (width).

For evaluation, we report the MLM validation perplexity (PPL) during pre-training and the downstream performance on development sets of eight GLUE (Wang et al., 2019) tasks. Note compared with the self-learning baseline, in KI, the logits output by $\mathcal{M}_L$ are additionally used to calculate $\mathcal{L}_{\text{KI}}$, we empirically find that the additional computations caused by it are almost negligible compared with the cumbersome computations in Transformer blocks. Therefore, it requires almost the same computational cost between KI and the baseline for each step. Hence, we report the performance w.r.t training step (Li et al., 2020a), while the performance w.r.t. FLOPs (Schwartz et al., 2019) and wall-clock time (Li et al., 2020b) can be roughly obtained by stretching the figure horizontally.

Overall Results. As shown in Figure 1 and Table 1, we can find that: (1) **training $\mathcal{M}_L$ under KI framework converges faster than the self-learning baseline**, indicating that inheriting the knowledge from an existing teacher is far more efficient than solely learning such knowledge. That is, to achieve the same level of validation PPL, KI requires fewer computational costs. Specifically, with the guidance of $\mathcal{M}_S$, whose validation PPL is 4.18, BASE $\rightarrow$ LARGE achieves a validation PPL of 3.41 at the end of pre-training, compared with baseline (LARGE) 3.58. After BASE $\rightarrow$ LARGE breaks away from teacher-guided learning at step 40k, it improves the validation PPL from 4.60 (LARGE) to 4.28, which is almost the performance when the baseline LARGE conducts self-learning for 55k steps, thus saving roughly 27.3% pre-training computational costs[1]. (2) **$\mathcal{M}_L$ trained under KI framework achieves better performance than the baseline on downstream tasks at each step**. We also found empirically that, under the same setting (e.g., data, hyper-parameters and model architectures), lower validation PPL generally indicates better downstream task performance. Since the performance gain in downstream tasks is consistent with that reflected in the validation PPL, we only show the latter for the remaining experiments due to the length limit. (3) **More evident improvements for larger PLMs**. We experiment on different sizes of $\mathcal{M}_S$ and $\mathcal{M}_L$ in our appendix to further demonstrate the universal superiority of KI over self-learning. We also find that with the size of both $\mathcal{M}_S$ and $\mathcal{M}_L$ growing, the improvements from KI become more evident. Concerning the energy cost, for the remaining experiments, unless otherwise specified, we choose MEDIUM (9 layers, 576 hidden size) as $\mathcal{M}_S$ and BASE as $\mathcal{M}_L$.

Effects of Inheritance Rate. In KI, we set α_t in Eq. (2) to be linearly decayed (denoted as Linear) to gradually encourage $\mathcal{M}_L$ exploring knowledge on its own. We analyze whether this design is essential for our framework by comparing it with two other strategies: the first is to only learn from the teacher at first and change to pure self-learning (denoted as Heviside) at the 35k-th step; the second is to use a constant ratio $(1:1)$ between $\mathcal{L}_{\text{SELF}}$ and $\mathcal{L}_{\text{KI}}$ throughout the whole training process (denoted as Constant). We can conclude from Figure 1 that: (1) **annealing at first is necessary**. The validation PPL curve of Linear converges the fastest, while Heviside tends to increase after $\mathcal{M}_L$ stops learning from the teacher, indicating

[1] If we load BASE and compute its inference during pre-training, 18.7% FLOPs can be saved roughly, since the forward passes of the small teacher also take up a small part.

Step	Model	CoLA	MNLI	QNLI	RTE	SST-2	STS-B	MRPC	QQP	Avg
5k	LARGE	0.0	73.5	81.7	53.0	81.7	45.8	71.4	87.5	61.8
	BASE → LARGE	**17.4**	**75.8**	**83.4**	**54.7**	**85.7**	**72.0**	**72.6**	**88.6**	**68.8**
45k	LARGE	61.8	84.9	91.7	63.4	92.9	88.6	87.7	**91.5**	82.8
	BASE → LARGE	**64.3**	**85.9**	**92.2**	**75.3**	**93.2**	**89.3**	**89.4**	91.5	**85.2**
85k	LARGE	64.5	86.8	92.7	69.7	93.5	89.9	89.7	91.7	84.8
	BASE → LARGE	**65.7**	**87.2**	**93.0**	**77.0**	**94.3**	**90.0**	**90.4**	**91.8**	**86.2**
125k	LARGE	64.3	87.1	**93.2**	73.4	94.1	90.3	**90.1**	91.8	85.5
	BASE → LARGE	**67.7**	**87.7**	93.1	**74.9**	**94.8**	**90.6**	88.2	**91.9**	**86.1**

Table 1: Downstream performances on GLUE tasks (dev). Our KI framework takes fewer pre-training steps to get a high score after fine-tuning. More detailed results at different pre-training steps are illustrated in our appendix.

that, due to the difference between teacher-guided learning and self-learning, annealing at first is necessary so that the performance won't decay at the transition point (35k-th step). (2) **Teacher-guided learning is redundant after $\mathcal{M}_L$ surpasses $\mathcal{M}_S$.** Although `Constant` performs well in the beginning, its PPL gradually becomes even worse than the other two strategies. The reason is that, after $\mathcal{M}_L$ has already surpassed $\mathcal{M}_S$, it will be encumbered by keeping following guidance from $\mathcal{M}_S$.

Saving Storage Space with Top-K Logits. Loading the teacher $\mathcal{M}_S$ repeatedly for knowledge inheritance is cumbersome, and an alternative way is to pre-compute and save the predictions of $\mathcal{M}_S$ offline once and for all. We show that using the information of top-K logits (Tan et al., 2019) can reduce the memory footprint without much performance decrease. Specifically, we save only top-K probabilities of $\mathcal{P}_S(x^j;\tau)$ followed by re-normalization, instead of the full distribution over all tokens. For RoBERTa, the dimension of $\mathcal{P}_S(x^j;\tau)$ is decided by its vocabulary size, which is around $50,000$. We thus vary K in $\{10, 50, 100, 1000\}$ to see its effects in Figure 1, from which we observe that: **top-K logits contain the vast majority of information**. Choosing a relatively small K (e.g., 10) is already good enough for inheriting knowledge from the teacher without much performance decrease compared with the full distribution. It demonstrates that, for $\mathcal{P}(x^j;\tau)$, the vast majority of information is contained in the top-K probabilities, while the tail probabilities tend to be some noise, which is aligned with previous observations (Tan et al., 2019) to some extent.

3.2 The Effects of $\mathcal{M}_S$'s Pre-training Setting

Existing PLMs are typically trained under quite different settings, and it is unclear how these different settings will affect the performance of KI. To this end, we conduct thorough experiments to analyze the effects of several representative factors: model architecture, pre-training data, $\mathcal{M}_S$'s pre-training step (appendix) and batch size (appendix).

Effects of Model Architecture. Large PLMs generally converge faster and achieve lower validation PPL, which means they are fast learners with more knowledge acquired through pre-training, and thus serving as more competent teachers. We experiment with two widely chosen architecture variations, i.e., width (hidden size) and depth (number of layers), to explore the effects of different model architectures. We choose `BASE` (12 layer, 768 hidden size) as $\mathcal{M}_L$'s architecture, and choose the architecture of $\mathcal{M}_S$ to differ from $\mathcal{M}_L$ in either width or depth. Specifically, for $\mathcal{M}_S$, we vary the width in $\{384, 480, 576, 672\}$, and the depth in $\{4, 6, 8, 10\}$, respectively, and pre-train $\mathcal{M}_S$ under the same setting as $\mathcal{M}_L$. The validation PPL curve for each teacher model is shown in our appendix, from which we observe that deeper / wider teachers with more parameters converge faster and achieve lower final validation PPL during pre-training. After that, we pre-train $\mathcal{M}_L$ under KI leveraging these teacher models. As shown in Figure 1 and 2, **choosing a wider / deeper teacher further accelerates $\mathcal{M}_L$'s convergence**, demonstrating the benefits of learning from a more knowledgeable teacher. Since the performance of PLMs is weakly related to model shape but highly related to model sizes (Li et al., 2020b), it is always a better strategy to choose the teacher with more parameters if other settings are kept the same. In experiments, we also find empirically that, the optimal duration for teacher-guided learning should be longer for larger teachers, which means it takes more time to learn from a more knowledgeable teacher.

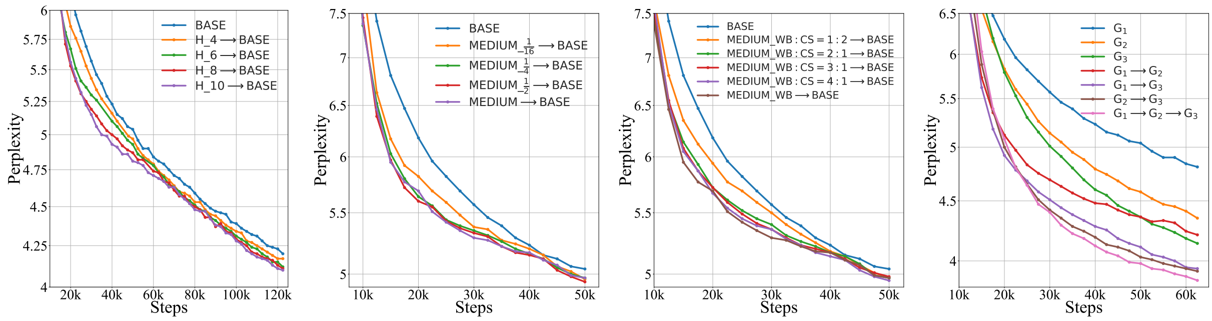

Figure 2: From left to right: (1) effects of $\mathcal{M}_S$'s model architecture (depth). (2) Effects of $\mathcal{M}_S$'s pre-training data size. (3) Effects of $\mathcal{M}_S$'s data domain. (4) Knowledge inheritance over generations.

Effects of Pre-training Data. In previous experiments, we assume $\mathcal{M}_L$ is pre-trained on the same corpus as $\mathcal{M}_S$, i.e., $\mathcal{D}_L = \mathcal{D}_S$. However, in real world scenarios, it may occur that the pre-training corpus used by both $\mathcal{M}_L$ and $\mathcal{M}_S$ is mis-matched, due to several factors: (1) **data size.** When training larger models, the pre-training corpus is often enlarged to improve downstream performance, i.e., $|\mathcal{D}_S| \ll |\mathcal{D}_L|$; (2) **data domain.** PLMs are trained on heterogeneous corpora from various sources (e.g., news articles, literary works, etc.) with different genres, i.e., $\mathcal{P}_{\mathcal{D}_S} \neq \mathcal{P}_{\mathcal{D}_L}$. The different knowledge contained in each domain may affect PLMs' generalization in downstream tasks. The existence of the above factors may hinder the successful knowledge transferring by requiring $\mathcal{M}_S$ to teach courses it is not skilled in. We thus design experiments to analyze the effects of these factors, with two observations concluded:

• **Obs. 1: PLMs can image the big from the small for in-domain data**. To evaluate the effects of data size, we first pre-train teacher models on different partitions of the original training corpus under the same setting by randomly sampling $\{\frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, \frac{1}{1}\}$ of it, resulting in teacher models with final validation PPL of $\{5.43, 5.15, 5.04, 4.98, 4.92\}$, respectively. The final validation PPL increases as we shrink the size of $\mathcal{M}_S$'s pre-training corpus, which implies that training with less data weakens the teacher's ability. Next, we compare the differences when their knowledge is inherited by $\mathcal{M}_L$. As reflected in Figure 2, however, the performance of KI is not substantially undermined until only $\frac{1}{16}$ of the original data is leveraged by the teacher. This indicates that PLMs can well image the overall data distribution even if it only sees a small part. Hence, when training larger PLMs, unless the data size is extensively enlarged, its impact can be ignored.

• **Obs. 2: Inheriting on similar domain improves performance**. To evaluate the effects of data domain, we experiment on the cases where the pre-training corpus used by $\mathcal{M}_S$ and $\mathcal{M}_L$ has domain mis-match. Specifically, keeping data size the same, we mix Wikipedia and BookCorpus (WB) used previously with computer science (CS) papers from S2ORC (Lo et al., 2020), whose domain is very different from WB, using different proportions, i.e., WB : CS $= \{1:2, 2:1, 3:1, 4:1\}$, respectively. We pre-train $\mathcal{M}_S$ on the constructed corpora, then test the performance when $\mathcal{M}_L$ inherits these teachers' knowledge on the WB domain data. As shown in Figure 2, with the domain of the constructed corpus $\mathcal{M}_S$ is trained on becoming gradually similar to WB, the benefits from KI become more obvious, which means it is essential that both $\mathcal{M}_S$ and $\mathcal{M}_L$ are trained on similar domain of data, so that $\mathcal{M}_S$ can successfully impart knowledge to $\mathcal{M}_L$ by teaching the "right" course. We further study the **data privacy** issue in our appendix and find that, as long as $\mathcal{D}_L$ and $\mathcal{D}_S$ share the same domain, whether they have data overlap or not is not a serious issue for $\mathcal{M}_S$ to teach $\mathcal{M}_L$. This is extremely meaningful when organizations aim to share the knowledge of their trained PLMs without exposing either the pre-training data or the model parameters due to privacy concerns.

3.3 Continual Knowledge Inheritance across Domain

With streaming data of various domains continuously increasing rapidly, training domain-specific PLMs and storing the model parameters for each domain can be prohibitively expensive. To this end, researchers recently demonstrate the feasibility of adapting PLMs to the target domain through continual pre-training (Gururangan et al., 2020). In this section, we further extend our KI framework

N_{tokens}		3,400M		200M		100M		40M		20M	
Metrics		F1	PPL	F1	PPL	F1	PPL	F1	PPL	F1	PPL
CS	SL	69.8	3.12	71.7	3.17	71.4	3.24	68.3	3.51	67.5	4.07
	KI	**72.9**	**3.06**	**72.6**	**3.09**	**71.9**	**3.11**	**71.1**	**3.21**	**70.8**	**3.37**
BIO	SL	84.0	2.67	82.8	2.72	83.2	2.83	83.3	3.16	82.7	3.81
	KI	**84.5**	**2.65**	**83.4**	**2.66**	**83.9**	**2.69**	**83.6**	**2.82**	**83.5**	**3.01**

Table 2: The validation PPL (PPL) and downstream performance (F1) on the target domain (CS / BIO) after BASE_WB is post-trained for 4k steps with self-learning (SL) or knowledge inheritance (KI). We experiment with different sizes of domain corpus. All downstream experiments are repeated 10 times with different seeds.

to a continual setting and demonstrate that domain adaptation for PLM can benefit from inheriting knowledge of existing domain experts.

Specifically, instead of training large PLMs from scratch, we focus on adapting BASE_WB, which has been well-trained on the concatenation of Wikipedia and BookCorpus (WB domain) for 125k steps, to two target domains, i.e., computer science (CS) and biomedical (BIO) papers from S2ORC (Lo et al., 2020). The proximity (vocabulary overlap) of three domains is listed in our appendix. We assume the existence of two domain experts MEDIUM_CS and MEDIUM_BIO, which have been trained on CS and BIO domain for 125k steps. Note their training computation is far less than BASE_WB due to fewer model parameters. Hence, either MEDIUM_CS or MEDIUM_BIO is no match for BASE_WB in WB domain but has richer knowledge in CS / BIO domain. For evaluation, we compare both (1) the MLM validation PPL on the target domain and (2) the performance (test F1) on downstream tasks, i.e. ACL-ARC (Jurgens et al., 2018) for CS domain and CHEMPROT (Kringelum et al., 2016) for BIO domain. Before adaptation, BASE_WB achieves a PPL of 5.41 / 4.86 and F1 of 68.5 / 81.6 on CS / BIO domain, while MEDIUM_CS achieves 2.95 (PPL) and 69.4 (F1) on CS domain, MEDIUM_BIO achieves 2.55 (PPL) and 83.6 (F1) on BIO domain. This demonstrates the superiority of two teachers over the student in their own domain despite their smaller model capacity.

We compare two strategies for continual pre-training: (1) only conducting self-learning on the target domain and (2) inheriting knowledge from well-trained domain teachers. Specifically, BASE_WB is post-trained for additional 4k steps on either CS or BIO domain to learn new knowledge. In addition, considering that in real world scenarios, it can be hard to retrieve enough pre-training data for a special domain, due to some privacy issues, hence, we conduct experiments with different sizes of domain corpus. The results are listed in Table 2, from which we observe that:

(1) **KI is more training-efficient**. Compared with self-learning, inheriting knowledge from domain teachers achieves lower final validation PPL and improved performance in domain-specific downstream tasks, indicating that, for domain adaptation, KI is more training-efficient than self-learning so that large PLMs can absorb more domain knowledge under the same training budget. By inheriting knowledge from domain teachers, large PLMs can further surpass their teachers in dealing with the specific domain. (2) **KI is more data-efficient**. The validation PPL gap between KI and SL is further enlarged when there is less domain-specific data available for adaptation, which means KI is more stable and data-efficient especially in low-resource settings, where domain data is relatively hard to collect. In other words, since the domain teacher has acquired rich knowledge, only providing a portion of domain-specific data is enough for satisfactory adaptation performance under KI, while self-learning exhibits overfitting to some extent. We further show in appendix that (1) there may exist catastrophic forgetting problem (McCloskey and Cohen, 1989) on the source domain during adaptation, and (2) large PLMs can simultaneously absorb knowledge from multiple domain teachers and thus become omnipotent.

3.4 Knowledge Inheritance over Generations

Human beings can inherit the knowledge from their antecedents, refine it and pass it down to their offsprings, so that knowledge can gradually accumulate over generations. Inspired by this, we investigate whether PLMs also have this kind of pattern. Specifically, we experiment with the knowledge inheritance among three generations of

PLMs with roughly 1.7x growth in model size: G_1 (`BASE`, 125M), G_2 (`BASE_PLUS`, 211M) and G_3 (`LARGE`, 355M), whose architectures are listed in our appendix. All models are trained for 125k steps with a batch size of $2,048$ on the same corpus. We compare the differences among (1) self-learning for each generation (denoted as G_1, G_2 and G_3), (2) knowledge inheritance over two generations (denoted as $G_1 \to G_2$, $G_1 \to G_3$ and $G_2 \to G_3$), and (3) knowledge inheritance over three generations (denoted as $G_1 \to G_2 \to G_3$), where G_2 first inherit the knowledge from G_1, refine it by additional self-exploring and pass its knowledge down to G_3. The results are drawn in Figure 2. Comparing the performance of G_2 and $G_1 \to G_2$, G_3 and $G_1 \to G_3$, or G_3 and $G_2 \to G_3$, we can again demonstrate the superiority of KI over self-training as concluded before. Comparing the performance of $G_1 \to G_3$ and $G_1 \to G_2 \to G_3$, or $G_2 \to G_3$ and $G_1 \to G_2 \to G_3$, it is observed that the performance of G_3 benefits from the involvements of both G_1 and G_2, which means knowledge is accumulating through more generations' involvements.

4 Related Work

Pre-training models on the unlabeled text and then performing task-specific fine-tuning have become the dominant method for NLP field, such as GPT (Radford et al., 2018), BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019b). Thenceforth, numerous efforts have been devoted to investigate better PLMs, including designing effective model architectures (Tay et al., 2021), formalizing novel pre-training objectives (Raffel et al., 2019; Clark et al., 2020; Lewis et al., 2020), applying additional supervision from knowledge base (Zhang et al., 2019; Qin et al., 2021; Wang et al., 2021; Peters et al., 2019), etc. In spite of these efforts, researchers find that the performance of PLMs can be improved by directly increasing the model size, data size and training steps (Liu et al., 2019; Raffel et al., 2019; Kaplan et al., 2020; Radford et al., 2019; Lan et al., 2020), sparking a recent wave of training ever-larger PLMs. For instance, the revolutionary OpenAI GPT-3 (Brown et al., 2020), which contains 175 billion parameters and is pre-trained on 570GB textual data, shows strong capabilities for language understanding and generation.

Nevertheless, larger models require greater computational demands (Patterson et al., 2021). To this end, researchers propose to accelerate pre-training by mixed-precision training (Shoeybi et al., 2019; Micikevicius et al., 2018), distributed training (Shoeybi et al., 2019; Huang et al., 2019; Shazeer et al., 2018), large batch optimization (You et al., 2020) and architecture innovation (layer sharing (Lan et al., 2020) and progressive layer dropping (Zhang and He, 2020)). Another line of methods (Gong et al., 2019; Gu et al., 2021) proposes to pre-train larger PLMs progressively. They first train a small PLM, and then gradually increase the depth or width of the network based on parameter initialization. However, they have strict requirements of the architectures of both models, which makes progressive training hard to be implemented practically for the goal of KI. In addition, progressive training is not applicable for absorbing knowledge from multiple teacher models and continual KI. More detailed comparisons between KI and progressive training are explained in our appendix.

Our work is also related to Knowledge Distillation (KD) (Hinton et al., 2015), which aims to compress a large model into a fast-to-execute one. KD has renewed a surge of interest in PLMs recently. Some explore KD at different training phases, e.g., pre-training (Sanh et al., 2019), downstream fine-tuning (Sun et al., 2019; Krishna et al., 2020), or both of them (Jiao et al., 2020); others explore distilling not only the final logits output by the large PLM, but also the intermediate hidden representations (Sun et al., 2019; Sanh et al., 2019; Jiao et al., 2020; Sun et al., 2020; Zhang et al., 2020). Previous work also indicates the relation between KD and label smoothing (Shen et al., 2021), however, we show in our appendix that the improvements of KI are not because of benefiting from optimizing smoothed targets, which impose regularization.

5 Conclusion

In this work, we propose a general KI framework that leverages previously trained PLMs for training larger ones, and conduct thorough experiments to demonstrate its feasibility. In addition, we comprehensively analyze various pre-training settings of the teacher model that may affect KI's performance. Finally, we extend KI and show that it can well support continual learning and knowledge transfer so that large PLMs can continuously absorb knowledge from multiple small teachers. In general, we provide a promising direction to share and exchange the knowledge learned by different models and continuously promote their performance.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *ArXiv preprint*, abs/2101.03961.

Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born-again neural networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1602–1611. PMLR.

Linyuan Gong, Di He, Zhuohan Li, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Efficient training of BERT by progressively stacking. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2337–2346. PMLR.

Xiaotao Gu, Liyuan Liu, Hongkun Yu, Jing Li, Chen Chen, and Jiawei Han. 2021. On the transformer growth for progressive BERT training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5174–5180, Online. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *ArXiv preprint*, abs/1503.02531.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Xu Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. 2019. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 103–112.

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 *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv preprint*, abs/2001.08361.

Jens Kringelum, Sonny Kim Kjaerulff, Søren Brunak, Ole Lund, Tudor I Oprea, and Olivier Taboureau. 2016. Chemprot-3.0: a global chemical biology diseases mapping. *Database*, 2016.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. 2020. Thieves on sesame street! model extraction of bert-based apis. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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 *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer

Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Mengtian Li, Ersin Yumer, and Deva Ramanan. 2020a. Budgeted training: Rethinking deep neural network training under resource constraints. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez. 2020b. Train big, then compress: Rethinking model size for efficient training and inference of transformers. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5958–5968. PMLR.

Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2021. A survey of transformers. *ArXiv preprint*, abs/2106.04554.

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*, abs/1907.11692.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training. *ArXiv preprint*, abs/2104.10350.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Yujia Qin, Yankai Lin, Ryuichi Takanobu, Zhiyuan Liu, Peng Li, Heng Ji, Minlie Huang, Maosong Sun, and Jie Zhou. 2021. ERICA: Improving entity and relation understanding for pre-trained language models via contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3350–3363, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv preprint*, abs/1910.10683.

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

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2019. Green ai. *ArXiv preprint*, abs/1907.10597.

Noam Shazeer, Youlong Cheng, Niki Parmar, Dustin Tran, Ashish Vaswani, Penporn Koanantakool, Peter Hawkins, HyoukJoong Lee, Mingsheng Hong, Cliff Young, Ryan Sepassi, and Blake A. Hechtman. 2018. Mesh-tensorflow: Deep learning for supercomputers. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10435–10444.

Zhiqiang Shen, Zechun Liu, Dejia Xu, Zitian Chen, Kwang-Ting Cheng, and Marios Savvides. 2021. Is label smoothing truly incompatible with knowledge distillation: An empirical study. *arXiv preprint arXiv:2104.00676*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *ArXiv preprint*, abs/1909.08053.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

Siqi Sun, Zhe Gan, Yuwei Fang, Yu Cheng, Shuohang Wang, and Jingjing Liu. 2020. Contrastive distillation on intermediate representations for language model compression. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 498–508, Online. Association for Computational Linguistics.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yi Tay, Mostafa Dehghani, Jai Prakash Gupta, Vamsi Aribandi, Dara Bahri, Zhen Qin, and Donald Metzler. 2021. Are pretrained convolutions better than pretrained transformers? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4349–4359, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Chenglin Yang, Lingxi Xie, Siyuan Qiao, and Alan L Yuille. 2019a. Training deep neural networks in generations: A more tolerant teacher educates better students. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5628–5635.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. 2019. Reducing bert pre-training time from 3 days to 76 minutes. *ArXiv preprint*, abs/1904.00962.

Yang You, Jing Li, Sashank J. Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training BERT in 76 minutes. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Minjia Zhang and Yuxiong He. 2020. Accelerating training of transformer-based language models with progressive layer dropping. *ArXiv preprint*, abs/2010.13369.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. Ternary-BERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, Online. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

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 *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

Appendices

A Additional Experiments and Analysis

A.1 Effects of Model Size

We experiment on four PLMs with roughly 1.7x growth in model size: $\mathcal{M}_1$ (RoBERTa$_{\texttt{MEDIUM}}$, 73.5M), $\mathcal{M}_2$ (RoBERTa$_{\texttt{BASE}}$, 125M), $\mathcal{M}_3$ (RoBERTa$_{\texttt{BASE_PLUS}}$, 211M) and $\mathcal{M}_4$ (RoBERTa$_{\texttt{LARGE}}$, 355M), whose architectures are listed in Table 6. We first pre-train a teacher PLM $\mathcal{M}_i$ ($\mathcal{M}_S$) for 125k steps with a batch size of $2,048$ under the same setting then train a larger one $\mathcal{M}_{i+1}$ ($\mathcal{M}_L$) by inheriting $\mathcal{M}_i$'s knowledge under KI framework (denoted as $\mathcal{M}_i \rightarrow \mathcal{M}_{i+1}, i \in \{1,2,3\}$). We compare $\mathcal{M}_i \rightarrow \mathcal{M}_{i+1}$ with $\mathcal{M}_{i+1}$ that conducts self-learning from beginning to end. As shown in Figure 3, the superiority of KI is observed across all models. In addition, with the overall model size of $\mathcal{M}_S$ and $\mathcal{M}_L$ gradually increasing, the benefits of KI become more evident, reflected in the broader absolute gap between the PPL curve of $\mathcal{M}_i \rightarrow \mathcal{M}_{i+1}$ and $\mathcal{M}_{i+1}$ when i gradually grows. This implies that with the advance of computing power in future, training larger PLMs will benefit more and more from our KI framework.

A.2 Effects of $\mathcal{M}_S$'s Pre-training Steps

Longer pre-training has been demonstrated as an effective way for PLMs to achieve better performance (Liu et al., 2019) and thus become more knowledgeable. To evaluate the benefits of more pre-training steps for $\mathcal{M}_S$, we first vary RoBERTa$_{\texttt{MEDIUM}}$'s pre-training steps in $\{$62.5k, 125k, 250k, 500k$\}$, and keep all other settings the same. After pre-training, these teacher models achieve the final validation PPL of $\{5.25, 4.92, 4.72, 4.51\}$, respectively. Then we compare the performances when RoBERTa$_{\texttt{BASE}}$ learn from these teacher models and visualize the results in Figure 3, from which we can conclude that, inheriting knowledge from teachers with longer pre-training time (steps) helps $\mathcal{M}_L$ converge faster. However, such a benefit is less and less obvious as $\mathcal{M}_S$'s pre-training steps increase, which means after enough training computations invested, the teacher model enters a plateau of convergence in validation PPL, and digging deeper in knowledge becomes even harder. The bottleneck more lies in other factors, e.g., the size and diversity of pre-training data, which hinder $\mathcal{M}_S$ from becoming more knowledgeable. We also found empirically that, after being pre-trained for 125k steps on the corpus with a batch size of $2,048$, all the models used in this paper have well converged, and longer pre-training only results in limited performance gain in either PPL or downstream performance.

A.3 Effects of $\mathcal{M}_L$'s Batch Size

Batch size is highly related to PLM's training efficiency, and previous work (Liu et al., 2019; Li et al., 2020b; You et al., 2019) found that slow-but-accurate large batch sizes can bring improvements to model training, although the improvements become marginal after increasing the batch size beyond a certain point (around $2,048$). BERT (Devlin et al., 2019) is pre-trained for $1,000$k steps with a batch size of 256, and the computational cost is equivalent to training for 125k steps with a batch size of $2,048$ (Liu et al., 2019), which is the pre-training setting chosen in our main paper. Choosing RoBERTa$_{\texttt{MEDIUM}}$ as the teacher model and RoBERTa$_{\texttt{BASE}}$ as the student model, in Figure 3 we compare the validation PPL as we vary the batch size in $\{256, 512, 1024, 2,048\}$, controlling for the number of passes through the pre-training corpus. We also vary the peak learning rate in $\{1.0 \times 10^{-4}, 2.5 \times 10^{-4}, 3.8 \times 10^{-4}, 5.0 \times 10^{-4}\}$ and pre-train for $\{1,000\text{k}, 500\text{k}, 250\text{k}, 125\text{k}\}$ steps, respectively, when increasing the batch size. We observe that increasing the batch size results in improved final validation PPL, which is aligned with previous findings (Liu et al., 2019). When adjusting batch size, KI accelerates the convergence unanimously, and its benefits become more evident when training with a smaller batch size, reflected in the absolute improvement in final validation PPL. We hypothesize that this is because learning from the smoothed target probability of KI, containing rich *secondary information* (Yang et al., 2019a) or *dark knowledge* (Furlanello et al., 2018), makes the pre-training process more stable. The student PLM is prevented from fitting to unnecessarily strict distributions and can thus learn faster.

A.4 Experiments on GPT

To demonstrate that our KI framework is agnostic to the specific self-supervised pre-training task, in addition to the experiments on MLM in the main paper, we conduct experiments on auto-regressive language modeling and choose GPT (Radford et al., 2018) as the PLM structure. Specifically, experimenting on the same pre-training corpus, we

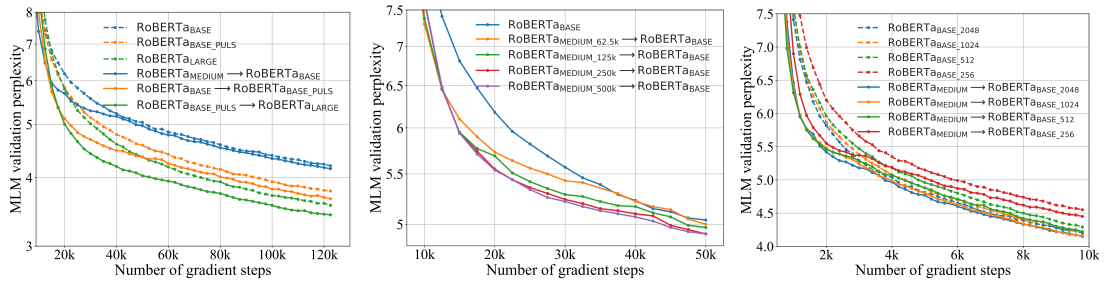

Figure 3: Left: effects of $\mathcal{M}_L$'s model size. Middle: effects of $\mathcal{M}_S$'s number of pre-training steps. Right: effects of $\mathcal{M}_L$'s batch size.

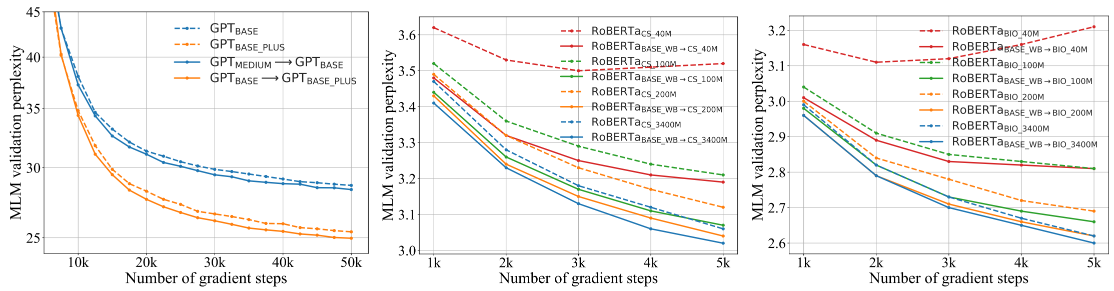

Figure 4: Left: experiments of auto-regressive language modeling for GPT. Middle & Right: adapting RoBERTa$_{\texttt{BASE_WB}}$ to CS (middle) / BIO (right) domain with different number of training steps on different sizes of domain data. We compare two strategies: self-learning and KI. For example, RoBERTa$_{\texttt{CS_3400M}}$ denotes post-training RoBERTa$_{\texttt{BASE_WB}}$ with the self-learning strategy on the $3,400$M token CS domain corpus. RoBERTa$_{\texttt{BASE_WB}\rightarrow\texttt{CS_3400M}}$ denotes post-training RoBERTa$_{\texttt{BASE_WB}}$ with the KI strategy on the $3,400$M token CS domain corpus.

choose three architectures: GPT$_{\texttt{MEDIUM}}$, GPT$_{\texttt{BASE}}$ and GPT$_{\texttt{BASE_PLUS}}$ with their architecture hyper-parameters specified in Table 6. We experiment with GPT$_{\texttt{MEDIUM}}$ $\rightarrow$ GPT$_{\texttt{BASE}}$ and GPT$_{\texttt{BASE}}$ $\rightarrow$ GPT$_{\texttt{BASE_PLUS}}$, and compare them with the self-training baseline GPT$_{\texttt{BASE}}$ and GPT$_{\texttt{BASE_PLUS}}$, respectively. All the teacher models are pre-trained for 62.5k steps with a batch size of $2,048$. As reflected in Figure 4, training larger GPTs under our KI framework converges faster than the self-learning baseline, which demonstrates KI is agnostic to the specific pre-training task and PLM structure chosen. We expect future work to explore KI with other pre-training tasks and PLM structures.

A.5 Additional Experiments for Continual Knowledge Inheritance across Domain

Different Number of Post-training Steps. In the main paper, we adapt RoBERTa$_{\texttt{BASE_WB}}$ to either CS or BIO domain by post-training it for 4k steps. We further vary the number of training steps in $\{1\text{k}, 2\text{k}, 3\text{k}, 4\text{k}, 5\text{k}\}$ and visualize the validation PPL in Figure 4. We also experiment on different sizes of domain corpus, i.e., $3,400$M, 200M, 100M, 40M tokens, respectively, as done in the main paper. We observe that generally the validation PPL on each domain decreases with the training step growing, and the performance of KI is always better than self-learning. The improvement of KI over self-learning is further enlarged when there is less target domain data available, demonstrating that KI is more data-efficient and can work well in low-resource settings. In addition, self-learning exhibits overfitting problems when the data size of the target domain is relatively small, which is not observed under our KI framework, which means KI can mitigate overfitting under low-resource settings.

Domain	Strategy	$3,400$M	200M	100M	40M
CS	SL	6.71	7.01	7.39	8.77
	KI	8.63	9.39	9.48	9.87
BIO	SL	7.29	6.61	8.16	10.34
	KI	10.74	10.78	10.93	11.66

Table 3: The validation PPL on the source domain (WB) after RoBERTa$_{\texttt{BASE_WB}}$ is post-trained on the target domain (CS / BIO) with self-learning (SL) and knowledge inheritance (KI).

Catastrophic Forgetting on the Source Domain. Table 3 lists the validation PPL on the source domain (WB) after RoBERTa$_{\texttt{BASE_WB}}$ is post-trained on the target domain (CS / BIO) with self-learning (SL) and knowledge inheritance (KI) for 4k steps. We show the results w.r.t. different sizes of domain corpus (3,400M, 200M, 100M and 40M tokens). We observe that after domain adaptation, the validation PPL on the source domain increases, which means PLMs may forget some key knowledge on the source domain when learning new knowledge in the target domain, i.e., the catastrophic forgetting problem. In addition, we find that the problem is more evident for KI than self-learning. Although we found empirically this problem can be largely mitigated by "reviewing" the lessons learned previously, we argue that our main goal in this paper is to let large PLMs efficiently and effectively absorb new knowledge, and we expect future work to further explore how to mitigate the catastrophic forgetting thoroughly.

Experiments on PLM adaptation towards multiple domains. In the main paper, we investigate the PLM adaptation towards one domain. Taking a step further, we explore whether KI could benefit PLM adaptation towards multiple domains when there exist domain teachers. Specifically, keeping the experimental settings the same, we adapt RoBERTa$_{\texttt{BASE_WB}}$ to synthetic domain data (BIO : CS = 1 : 1) to absorb knowledge from two domains simultaneously (for KI, we assume $\mathcal{M}_L$ is trained with the optimal teacher selection strategy, i.e., each teacher imparts the knowledge on its own domain data). From Table 4, we observe RoBERTa$_{\texttt{BASE_WB}}$ achieves improved performance on both domains after being taught by two teachers simultaneously. This demonstrates large PLMs can simultaneously absorb knowledge from multiple domains and thus become omnipotent. Compared with self-learning, KI is still a better choice. However, simultaneous learning overfits training data more easily and its performance on either domain is no match for learning only one domain at a time.

A.6 Detailed Downstream Performances on GLUE Tasks

Figure 5 visualizes in detail the downstream performance of RoBERTa$_{\texttt{LARGE}}$ and RoBERTa$_{\texttt{BASE}}$ $\rightarrow$ RoBERTa$_{\texttt{LARGE}}$ on the dev sets of six GLUE tasks at different pre-training steps with an interval of 5k. It can be observed that the downstream performance of RoBERTa$_{\texttt{BASE}}$ $\rightarrow$ RoBERTa$_{\texttt{LARGE}}$ rises faster than the baseline, which means it takes fewer pre-training steps for our KI framework to get a high score in downstream tasks. Aligned with previous findings (Li et al., 2020b), we found MNLI and SST-2 to be the most stable tasks in GLUE, whose variances are lower.

We also list the average GLUE performance for RoBERTa$_{\texttt{BASE}}$ $\rightarrow$ RoBERTa$_{\texttt{LARGE}}$ and the baseline RoBERTa$_{\texttt{LARGE}}$ in Table 5, from which we observe that the baseline at 70k-th step achieves almost the same GLUE performance as our method at 40k-th step, which means our framework saves around 42.9% FLOPs, much higher than the reported 27.3% FLOPs saved based on the pre-training PPL metric in the main paper. In addition, our method achieves almost the same GLUE performance as the baseline at the final step (125k) with only 70k steps, which means our framework saves 44% FLOPs in total. Both the perplexity in the pre-training stage and performance in downstream tasks can be chosen as the evaluation metric for measuring the computational cost savings. However, in this paper, we choose the former because it is more stable and accurate than the latter. We find empirically that some GLUE tasks like CoLA have higher variances than others, which might make the measurement inaccurate.

Besides, when discussing the effects of model architectures in the main paper, we only show the validation PPL of each model during pre-training, we visualize the corresponding downstream performance (MNLI) in Figure 6, from which it can be observed that learning from teacher models with more parameters helps achieve better downstream performance at the same pre-training step. In general, we observe that, under our setting, the performance gain in downstream tasks is aligned with that reflected in validation PPL during pre-training.

A.7 Teacher Models' Validation PPL Curves during Pre-training for "Effects of Model Architecture"

Figure 6 visualizes the validation PPL curves for all the teacher models used in the experiments on the effects of model architecture. The teacher models differ from RoBERTa$_{\texttt{BASE}}$ in either the depth or width. Specifically, we vary the depth in $\{4, 6, 8, 10\}$ (denoted as $\{$RoBERTa$_{\texttt{H_4}}$, RoBERTa$_{\texttt{H_6}}$, RoBERTa$_{\texttt{H_8}}$, RoBERTa$_{\texttt{H_10}}\}$), and the width in $\{384, 480, 576, 672\}$ (de-

$\mathbf{N_{tokens}}$	3,400M				200M				100M				40M			
Metrics	F1_C	PPL_C	F1_B	PPL_B	F1_C	PPL_C	F1_B	PPL_B	F1_C	PPL_C	F1_B	PPL_B	F1_C	PPL_C	F1_B	PPL_B
SL	71.7	3.15	83.7	2.71	70.5	3.97	82.7	3.36	67.7	5.95	81.7	4.84	68.3	11.7	81.1	10.5
KI	**72.2**	**3.15**	**83.9**	**2.70**	**71.8**	**3.42**	**83.1**	**2.92**	**69.8**	**3.90**	**82.6**	**3.32**	**69.1**	**5.70**	**81.3**	**4.64**

Table 4: The results when RoBERTa$_{\text{BASE_WB}}$ is post-trained on the synthetic domain data with self-learning (SL) or knowledge inheritance (KI). We report both validation PPL (PPL_B / PPL_C) and downstream performance (F1_B / F1_C) for BIO / CS domain. We observe that SL exhibits serious overfitting when data is relatively scarce.

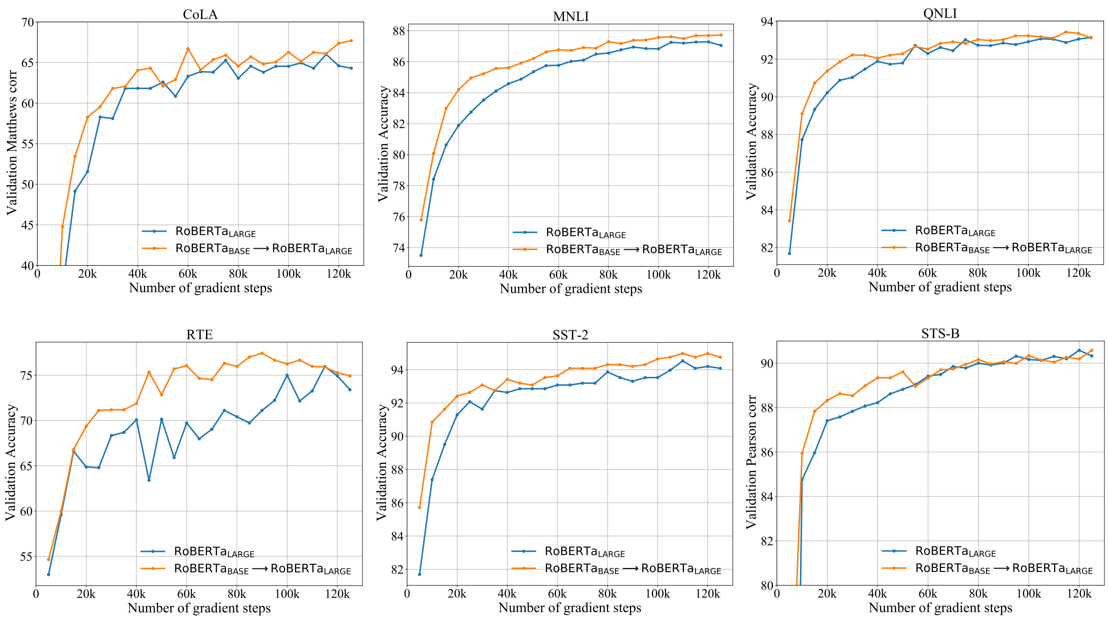

Figure 5: Downstream performance visualization on six GLUE tasks comparing RoBERTa$_{\text{LARGE}}$ and RoBERTa$_{\text{BASE}}$ $\rightarrow$ RoBERTa$_{\text{LARGE}}$. For CoLA, RTE, SST-2 and STS-B, we repeat fine-tuning for 5 times; for MNLI and QNLI, we repeat fine-tuning for 3 times.

Step	RoBERTa$_{\text{BASE}}$	RoBERTa$_{\text{BASE}}$ $\rightarrow$ RoBERTa$_{\text{LARGE}}$
5k	61.8	68.8
10k	75.6	78.1
15k	79.3	81.5
20k	80.4	82.8
25k	81.7	83.6
30k	82.4	83.9
35k	83.1	84.1
40k	83.6	84.5
45k	82.8	85.2
50k	83.9	84.6
55k	83.4	85.2
60k	84.0	85.7
65k	84.1	85.3
70k	84.3	85.5
75k	85.0	85.8
80k	84.7	85.8
85k	84.8	86.2
...	...	...
125k	85.5	86.1

Table 5: Average GLUE performance comparing both RoBERTa$_{\text{BASE}}$ and RoBERTa$_{\text{BASE}}$ $\rightarrow$ RoBERTa$_{\text{LARGE}}$ at different pre-training steps.

noted as {RoBERTa$_{\text{D_384}}$, RoBERTa$_{\text{D_480}}$, RoBERTa$_{\text{D_576}}$, RoBERTa$_{\text{D_672}}$}). Generally, PLMs with larger model parameters converge faster and achieve better final performance.

A.8 Effects of Data Privacy

In the main paper, we investigate the effects of both the **data size** and **data domain** for the pre-training data. However, even if both size and domain of $\mathcal{M}_S$ and $\mathcal{M}_L$'s data are ensured to be the same, it may be hard to retrieve the pre-training corpus used by $\mathcal{M}_S$ due to privacy reasons, with an extreme case: $\mathcal{D}_L \cap \mathcal{D}_S = \emptyset$, which is dubbed as **data privacy issue**. To evaluate its effects, we experiment in an extreme case where the pre-training corpus of $\mathcal{M}_S$ and $\mathcal{M}_L$ has no overlap at all. To avoid the influences of size and domain, we randomly split the WB domain training corpus $\mathcal{D}$ into two halves ($\mathcal{D}_A$ and $\mathcal{D}_B$) and pre-train two teacher models (denoted as RoBERTa$_{\text{MEDIUM_A}}$ and RoBERTa$_{\text{MEDIUM_B}}$) on

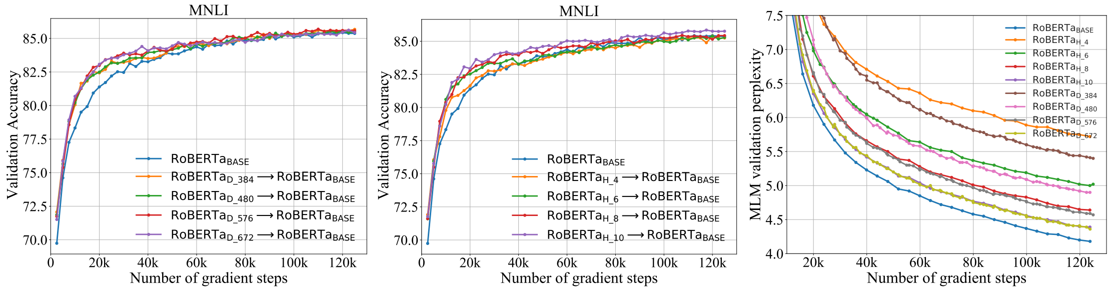

Figure 6: Left & Middle: downstream performances corresponding to the experiments on effects of $\mathcal{M}_S$'s model architecture (width (left) & depth (middle)). Right: validation PPL during pre-training for the teacher models used in experiments of effects of teacher model architecture.

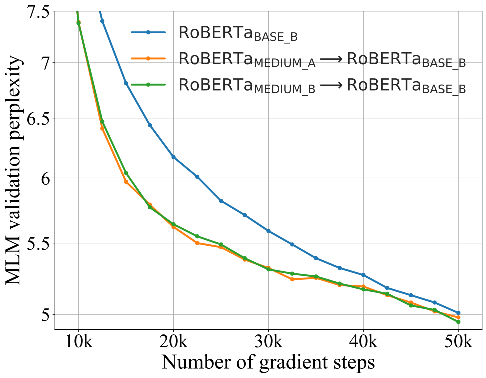

Figure 7: Effects of data privacy.

them. After pre-training, both of them achieve almost the same final PPL (4.99) on the same validation set. They are then inherited by the student model RoBERTa$_{\texttt{BASE}}$ on $\mathcal{D}_B$ (denoted as RoBERTa$_{\texttt{MEDIUM_A}}$ $\rightarrow$ RoBERTa$_{\texttt{BASE_B}}$ and RoBERTa$_{\texttt{MEDIUM_B}}$ $\rightarrow$ RoBERTa$_{\texttt{BASE_B}}$), which is exactly the pre-training corpus of RoBERTa$_{\texttt{MEDIUM_B}}$ and has no overlap with that of RoBERTa$_{\texttt{MEDIUM_A}}$. We also choose $\mathcal{M}_L$ that conducts pure self-learning on $\mathcal{D}_B$ as the baseline (denoted as RoBERTa$_{\texttt{BASE_B}}$). It is observed from Figure 7 that, there is little difference between the validation PPL curves of RoBERTa$_{\texttt{MEDIUM_A}}$ $\rightarrow$ RoBERTa$_{\texttt{BASE_B}}$ and RoBERTa$_{\texttt{MEDIUM_B}}$ $\rightarrow$ RoBERTa$_{\texttt{BASE_B}}$, indicating that whether the pre-training corpus of $\mathcal{M}_S$ and $\mathcal{M}_L$ has overlap or not is not important as long as they share the same domain. This is extremely meaningful when organizations aim to share the knowledge of their trained PLMs without exposing either the pre-training data or the model parameters due to privacy concerns. In other words, as long as the recipients prepare pre-training data in similar domain, the knowledge can be successfully inherited by receiving $\mathcal{M}_S$'s predictions.

B Pre-training Hyper-parameters

Table 6 describes the architectures we used for all models in this paper, covering the details for the total number of trainable parameters (n_{params}), the total number of layers (n_{layers}), the number of units in each bottleneck layer (d_{model}), the total number of attention heads (n_{heads}), the inner hidden size of FFN layer (d_{FFN}) and the learning rate when batch size is set to $2,048$ (lr). We set the dropout rate for each model to 0.1, weight decay to 0.01 and use linear learning rate decay. We adopt Adam as the optimizer, warm up learning rate for the first 10% steps then linearly decay it. The hyper-parameters for Adam optimizer is set to $1 \times 10^{-6}, 0.9, 0.98$ for $\epsilon, \beta_1, \beta_2$, respectively. As mentioned in the main paper, all experiments are done in the same computation environment with 8 NVIDIA 32GB V100 GPUs and it takes around 1 week to pre-train RoBERTa$_{\texttt{BASE}}$ and 2 weeks to pre-train RoBERTa$_{\texttt{LARGE}}$. It has been shown by previous work (Kaplan et al., 2020) that, within a reasonably broad range, the validation PPL is not sensitive to these parameters. All the pre-training implementations are based on `fairseq`[2] (Ott et al., 2019) (MIT-license).

Table 7 describes the total number of pre-training steps for each ($\mathcal{M}_L$, $\mathcal{M}_S$) pair chosen in our experiments. Within a reasonably broad range, the performance of KI is not sensitive to its choice.

C Fine-tuning Hyper-parameters

Table 8 describes the hyper-parameters for ACL-ARC, CHEMPROT and GLUE tasks. The selection of these hyper-parameters closely follows (Liu et al., 2019) and (Gururangan et al., 2020). The implementations of ACL-ARC and CHEMPROT

[2] https://github.com/pytorch/fairseq

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{FFN}	lr (bs $= 2,048$)
RoBERTa$_{\texttt{MEDIUM}}$	73.5M	9	576	12	3072	5.0×10^{-4}
RoBERTa$_{\texttt{D_d}}$	-	12	d	12	3072	5.0×10^{-4}
RoBERTa$_{\texttt{H_h}}$	-	h	768	12	3072	5.0×10^{-4}
RoBERTa$_{\texttt{BASE}}$	125M	12	768	12	3072	5.0×10^{-4}
RoBERTa$_{\texttt{BASE_PLUS}}$	211M	18	864	12	3600	3.5×10^{-4}
RoBERTa$_{\texttt{LARGE}}$	355M	24	1024	16	4096	2.5×10^{-4}
GPT$_{\texttt{MEDIUM}}$	72.8M	9	576	12	3072	5.0×10^{-4}
GPT$_{\texttt{BASE}}$	124M	12	768	12	3072	5.0×10^{-4}
GPT$_{\texttt{BASE_PLUS}}$	209M	18	864	12	3600	3.5×10^{-4}

Table 6: Model architectures for all the models we used in this paper.

$\mathcal{M}_L$	$\mathcal{M}_S$	Steps of teacher-guided learning
RoBERTa$_{\texttt{BASE}}$	RoBERTa$_{\texttt{MEDIUM}}$	35k
	RoBERTa$_{\texttt{D_384}}$	28k
	RoBERTa$_{\texttt{D_480}}$	40k
	RoBERTa$_{\texttt{D_576}}$	70k
	RoBERTa$_{\texttt{D_672}}$	85k
	RoBERTa$_{\texttt{H_4}}$	22k
	RoBERTa$_{\texttt{H_6}}$	35k
	RoBERTa$_{\texttt{H_8}}$	55k
	RoBERTa$_{\texttt{H_10}}$	65k
RoBERTa$_{\texttt{BASE_PLUS}}$	RoBERTa$_{\texttt{BASE}}$	55k
RoBERTa$_{\texttt{LARGE}}$	RoBERTa$_{\texttt{BASE}}$	40k
	RoBERTa$_{\texttt{BASE_PLUS}}$	65k
	RoBERTa$_{\texttt{BASE}}$ $\rightarrow$ RoBERTa$_{\texttt{BASE_PLUS}}$	75k
GPT$_{\texttt{BASE}}$	GPT$_{\texttt{MEDIUM}}$	10k
GPT$_{\texttt{BASE_PLUS}}$	GPT$_{\texttt{BASE}}$	15k

Table 7: The total number of steps for teacher-guided learning for different ($\mathcal{M}_L$, $\mathcal{M}_S$) pairs.

are based on (Gururangan et al., 2020)[3]; the implementations of GLUE tasks are based on `fairseq`[4] (Ott et al., 2019) (MIT-license).

D Domain Proximity of WB, CS and BIO

Table 9 lists the domain proximity (vocabulary overlap) of WB, CS and BIO used in this paper.

E Evaluation Metrics for the Computational Costs Saved

As stated in the main paper, training RoBERTa$_{\texttt{LARGE}}$ under the knowledge inheritance framework saves roughly 27.3% pre-training computations (FLOPs) at the step of 55k, where the teacher-guided learning ends. Since we trained all models under the same hardware environment, choosing the evaluation metric of FLOPs is equivalent to wall-clock time, i.e., our framework saves RoBERTa$_{\texttt{LARGE}}$ roughly 27.3% training time, which is around 28.4 hours in our setting (8 V100 GPU for training RoBERTa$_{\texttt{LARGE}}$). Since for both our method and the baseline method, it takes nearly the same training time/FLOPs for each step, thus, the "training-time/FLOPs vs. PPL figure" can be easily obtained by stretching the horizontal axis linearly in "step vs. PPL figure".

In addition, since the training time of PLMs can vary greatly in different hardware environments, there are many factors that should be considered, e.g., the choice of the GPU, the number of GPU used, whether PLMs are trained distributedly across multiple servers (synchronizing gradients for large PLMs may involve much longer time between different servers for communication), etc. Therefore, we believe the metric of FLOPs is more suitable for future research comparison.

F Comparison between Knowledge Inheritance and Progressive Training

"Progressive Training" first trains a small PLM, and then gradually increases the depth or width of the

[3] https://github.com/allenai/dont-stop-pretraining

[4] https://github.com/pytorch/fairseq

HyperParam	ACL-ARC & CHEMPROT	GLUE
Learning Rate	2×10^{-5}	$\{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$
Batch Size	256	$\{16, 32\}$
Weight Decay	0.1	0.1
Max Epochs	10	10
Learning Rate Decay	Linear	Linear
Warmup Ratio	0.06	0.06

Table 8: Hyper-parameters for fine-tuning RoBERTa on ACL-ARC, CHEMPROT and GLUE.

	WB	CS	BIO
WB	100%	19.1%	25.6%
CS	19.1%	100%	22.5%
BIO	25.6%	22.5%	100%

Table 9: Domain proximity (vocabulary overlap) among three domains (WB, CS, BIO) discussed in this paper. Following (Gururangan et al., 2020), we create the vocabulary for each domain by considering the top 10k most frequent words (excluding stopwords).

network based on parameter initialization. It is an orthogonal research direction against our "knowledge inheritance" framework, and has many limitations while our knowledge inheritance does not have as follows:

Architecture Mismatch. Existing parameter reusing methods (Gong et al., 2019; Gu et al., 2021) require that the architectures of both small PLMs and large PLMs are matched to some extent, however, our knowledge inheritance does not have such a requirement. For example, Gong et al. (2019); Gu et al. (2021) either requires the number of layers, or the hidden size / embedding size of a large PLM to be the integer multiples of that of a small PLM. Hence, it is not flexible to train larger PLMs with arbitrary architectures, making parameter reusing hard to be implemented practically. Besides, there are more and more advanced non-trivial Transformer modifications appearing (we refer to Lin et al. (2021) for details), which change the inner structures of a standard Transformer, e.g., pre-normalization, relative embedding, sparse attention, etc. It is non-trivial to directly transfer the parameters between two PLMs if they have different non-trivial inner structures. Nevertheless, our knowledge inheritance framework will not be influenced by such architectural mismatches.

Inability for Multi-to-one Knowledge Inheritance. It is non-trivial to support absorbing knowledge from multiple teacher models by jointly reusing their model parameters. Instead, it is easy to implement for knowledge inheritance. As shown in our experiments, we demonstrate that under our framework, large PLMs can simultaneously absorb knowledge from multiple teachers.

Inability for Continual Knowledge Inheritance. Parameter reusing is hard to support continual learning, which makes large PLMs absorb knowledge from small ones in a lifelong manner. In real-world scenarios, numerous PLMs of different architectures are trained locally with different data. These small PLMs can be seen as domain experts, and it is essential that larger PLMs can continuously benefit from these existing PLMs efficiently by incorporating their knowledge so that larger PLMs can become omnipotent. As described before, it is easy to implement for our framework and we have demonstrated the effectiveness.

Model Privacy. Parameter reusing requires the availability of the parameters of an existing PLM, which may be impractical due to some privacy issues, e.g., GPT-3 only provides API access for prediction instead of the model parameters. Instead, our knowledge inheritance framework does not presume access to an existing model parameter since the predictions of the small model can be pre-computed and saved offline. This superiority will further make it possible for API-based online knowledge transfer.

G Comparing Label Smoothing and Knowledge Inheritance

Previous work shows the relation between label smoothing and knowledge distillation to some extent (Shen et al., 2021). To demonstrate that the success of our KI is not because of learning from a more smoothed target, we conduct experiments comparing both label smoothing and our KI in Table 10. Specifically, for label smoothing, PLMs optimize a smoothed target $\mathbf{y}_i^S = (1-\alpha) * \mathbf{y}_i + \alpha * \vec{\mathbf{1}}/(K-1)$, where $\alpha = 0$ denotes learning from scratch with no label smoothing, larger α

Step	20k	40k	60k	80k	100k
$\alpha = 0.3$	8.68	7.29	6.90	6.57	6.26
$\alpha = 0.2$	7.27	6.47	5.95	5.68	5.46
$\alpha = 0.1$	6.71	5.74	5.35	5.06	4.86
$\alpha = 0$	6.13	5.21	4.83	4.57	4.36
KI	**5.69**	**5.17**	**4.78**	**4.52**	**4.32**

Table 10: Validation loss for training RoBERTa$_{\texttt{BASE}}$ with different strategies. **KI** denotes our knowledge inheritance framework, where RoBERTa$_{\texttt{MEDIUM}}$ is chosen as the teacher.

means a more smoothed target for PLMs to learn from, K denotes the vocabulary size. Specifically, we choose α from $\{0.1, 0.2, 0.3\}$. It can be concluded from the results in Table 10 that adding label smoothing into the pre-training objectives of PLMs leads to far worse performance than the vanilla baseline, which shows that the improvements of our knowledge inheritance framework are non-trivial: larger PLMs are indeed inheriting the "knowledge" from smaller ones, instead of benefiting from optimizing a smoothed target, which imposes regularization. To the best of our knowledge, there is little previous work that investigates the feasibility of label smoothing in the field of pre-trained language models, we expect future work to discuss it in detail.

H Limitations and Future Work

Being the first to systematically propose the idea of "knowledge inheritance for PLMs", we hope this work could launch an entirely new research area and enlighten further research attempts. Therefore, this paper focus on providing a general framework and a systematic empirical analysis.

There are some limitations which are not addressed in this paper and left as future work: (1) hyper-parameter choice: the total number of pre-training steps of teacher-guided learning is not a known prior and we need to change the hyper-parameter α_T under different circumstances. However, we found empirically that estimating the optimal choice of α_T is relatively easy, and within a reasonably broad range, the performance of KI is not sensitive to the choice of α_T. (2) Catastrophic forgetting problem: when adapted to a new domain, PLMs exhibit catastrophic forgetting problems on the source domain, which is not well-addressed in our paper. (3) Data privacy problem: in the main paper, we demonstrate that the knowledge of an existing PLM can be successfully extracted by saving its predictions on corpus unseen during its pre-training as long as the same domain is ensured. However, it does not mean the privacy of pre-training corpus used by the existing PLM is 100% preserved. In fact, it is still under-explored whether some malicious adversarial attacks can be applied to access the private data, causing potential privacy concerns. We expect future work to explore this direction and design corresponding defense strategies.

In general, we believe it a promising direction to share and exchange the knowledge learned by different models and continuously promote their performances. In future, we aim to explore the following directions. (1) The **efficiency** of KI, i.e., given limited computational budget and pre-training corpus, how to more efficiently absorb knowledge from teacher models. Potential solutions include denoising teacher models' predictions and utilizing more information from the teacher, i.e., the inner hidden units computed by the teacher. How to select the most representative data points for KI is also an interesting topic. (2) The **effectiveness** of KI under different circumstances, i.e., how can KI be applied if the teachers and the students are pre-trained on different vocabularies (e.g., from BERT to RoBERTa), languages, pre-training objectives (e.g., from GPT to BERT) and even modalities. In addition, in the main paper, we systematically analyze the effects of pre-training setting of the teacher model for KI. However, in real world scenarios, we need to consider these effects jointly to design the optimal teacher selection strategy. (3) How is PLMs trained with KI qualitatively different from the non-KI PLM apart from being faster to train, e.g. is KI PLM more robust to adversarial attacks?

Finally, we believe it is vital to use fair benchmarking that can accurately and reliably judge each KI algorithm. Towards this goal, we propose the following suggestions for future work: (1) Conduct all experiments under the same computation environment and report the pre-training hyper-parameters and hardware deployments in detail for future comparisons. (2) Evaluate the downstream tasks with multiple different random seeds and choose tasks (e.g. MNLI) that give relatively stable and consistent results, which could serve as better indicators for PLMs' effectiveness. In addition, it is also essential that PLMs are tested on diverse downstream tasks which evaluate PLMs' different abilities. (3) Save the checkpoint more frequently during pre-training and evaluate the downstream performance, which can better indicate the trend of PLMs' effectiveness. (4) Open-source all the codes and model parameters for future comparisons and deployments. In conclusion, we hope our efforts could facilitate future research attempts to improve the community's understanding and development of this important research direction.