Learning to Teach with Student Feedback

Anonymous ACL submission

Abstract

Knowledge distillation (KD) has gained much attention due to its effectiveness in compressing large-scale pre-trained models. In typical KD methods, the small student model is trained to match the soft targets generated by the big teacher model. However, the interaction between student and teacher is one-way. The teacher is usually fixed once trained, resulting in static soft targets to be distilled. This one-way interaction leads to the teacher's inability to perceive the characteristics of the student and its training progress. To address this issue, we propose Interactive Knowledge Distillation (IKD), which also allows the teacher to learn to teach from the feedback of the student. In particular, IKD trains the teacher model to generate specific soft target at each training step for a certain student. Joint optimization for both teacher and student is achieved by two iterative steps: a course step to optimize student with the soft target of teacher, and an exam step to optimize teacher with the feedback of student. IKD is a general framework that is orthogonal to most existing knowledge distillation methods. Experimental results show that IKD outperforms traditional KD methods on various NLP tasks.

1 Introduction

Large-scale pre-trained language models (PLMs) such as GPT (Radford et al., 2019), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) have achieved significant improvement on various NLP tasks. Despite their power, they are computationally expensive due to their enormous size, which limits their deployment in real-time scenarios.

As an effective technique to tackle this problem, Knowledge Distillation (KD) (Bucila et al., 2006; Hinton et al., 2015) has gained much attention in the community. In common KD methods for compressing large-scale PLMs, the original large-scale PLM often serves as the teacher model, which is first trained on the downstream task then uses its generated soft targets to teach the student model, which is usually a smaller PLM. Previous works such as DistilBERT (Sanh et al., 2019), BERT-PKD (Sun et al., 2019), MobileBERT (Sun et al., 2020), and TinyBERT (Jiao et al., 2020) have explored informative features that can be distilled, including the output logits, word embeddings, hidden states, attention maps, etc.

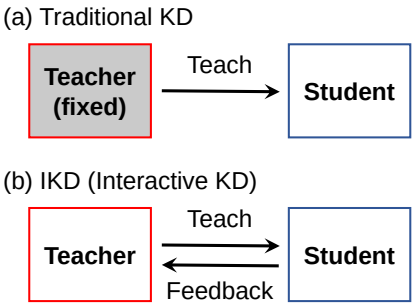


Figure 1: Comparison between traditional KD methods and our proposed IKD. (a) Traditional KD methods use an one-way interaction between teacher and student while the teacher model is static during teaching student. (b) IKD builds a co-interactive channel to achieve joint training of both teacher and student.

However, in these existing methods the interaction between teacher and student remains one-way. The teacher model is usually fixed once trained on the downstream data, resulting in static soft targets regardless of the characteristics of the student and its training progress. Though there are works that insert a scheduled temperature into the Softmax function to control the smoothness of the soft targets such as Annealing-KD (Jafari et al., 2021), the design of the temperature and its schedule does not utilize the feedback from the student and highly depends on expert experience.

In this paper, we propose the Interactive Knowledge Distillation (IKD) to implement a bidirectional interaction between teacher and student. IKD allows the teacher to learn to teach based on the feedback from the student. In particular, the teacher

model is trained to generate specific soft targets at each training step with the help of meta-learning, especially MAML (Finn et al., 2017). Figure 1 illustrates main difference between our proposed IKD and traditional KD methods.

The central idea of IKD is to make the student model generalize well (like achieving a lower loss) on a batch of unseen samples after learned from the teacher model, while the measurable performance on unseen samples is back-propagated to the teacher model to update its teaching strategy. More specifically, IKD consists of two update steps: *course* step and *exam* step. **In course step**, the student is trained to match the soft targets generated by the teacher on a batch of data called course data. **In exam step**, the student after one gradient step is evaluated on another batch of data called exam data, the cross entropy of the student on the exam data provides meta-gradients to be back-propagated to the teacher model such that the teacher model could get updated to generate better soft targets. By iteratively conducting the two steps, we can continually improve the student model via the soft targets generated by the teacher, and improve the teacher model via the feedback generated by the student.

We conduct experiments on various NLP tasks. Experimental results on GLUE benchmark demonstrate that IKD consistently outperforms vanilla knowledge distillation. Further analysis is then conducted to shed some light on the dynamic soft targets and the student feedback.

To sum up, our contributions are as follows:

- We propose a co-interactive method for teacher-student framework, namely Interactive Knowledge Distillation (IKD).
- We take an approximation to convert the iterative optimization into a joint optimization, which is more efficient for training.
- Our proposed IKD is orthogonal to most existing knowledge distillation works that distill different feature sets, in which we empirically evaluated the effectiveness of IKD based on vanilla knowledge distillation.

2 Method

The intuition behind our framework is straightforward: First, the teacher teaches the *course* to the student who then updates its knowledge according to the course. Second, the student takes *exams* and produces scores for the teacher to adjust its teaching strategy. Such two steps are common in real-world education.

Based on the intuition above, IKD can be formulated in the context of machine learning as follows. Denote the teacher model and the student model as f and g respectively. Assume teacher model f is parameterized by θ, student model g is parameterized by ϕ. In the course step, we optimize the student model $g(\phi)$ by vanilla knowledge distillation. The student model is trained to match the soft targets generated by the teacher model on a batch of data drawn from the course data set D_{course}. In the exam step, we evaluate the student model on a batch of data drawn from the exam data set D_{exam}. The test score of the exam, which can be instantiated as cross entropy, provides meta gradients to optimize the teacher model $f(\theta)$. The overall illustration of our method is depicted by Figure 2.

2.1 Course Step: Student Optimization by Vanilla Knowledge Distillation

In our setting, the teacher model $f(\theta)$ is implemented as a deep encoder such as BERT (Devlin et al., 2019), and the student model $g(\phi)$ is implemented as a lightweight encoder such as a Transformer (Vaswani et al., 2017) encoder with fewer layers.

Given a training sample (x_i, y_i) drawn from the course data set $D_{course} = \{x_i, y_i\}_{i=1}^{N}$, the teacher model computes a contextualized embedding $h_i^T = f(x_i; \theta)$[1], which is then fed into a Softmax layer to obtain the probability for each category, i.e. the soft targets, that is $y_i^T = \text{Softmax}(W^T h_i^T)$, where W^T is a learnable parameter matrix and $W^T h_i^T$ is the logits. For simplicity, $W^T h_i^T$ is abbreviated to z_i^T. In practice, the temperature is often introduced as a hyper-parameter to control the smoothness of the soft targets,

$$y_i^T = \text{Softmax}(z_i^T / T) = \frac{e^{z_{i,c}^T / T}}{\sum_{c'=1}^{C} e^{z_{i,c'} / T}}, \quad (1)$$

where C denotes the number of classes, c is the correct class.

Similarly, we can obtain the output of the student model $y_i^S = \text{Softmax}(W^S h_i^S)$, where $h_i^S = g(x_i; \phi)$. In vanilla knowledge distillation, the KL divergence of the teacher's prediction and the stu-

[1] We use the `[CLS]` token embedding of the last layer as the sentence representation.

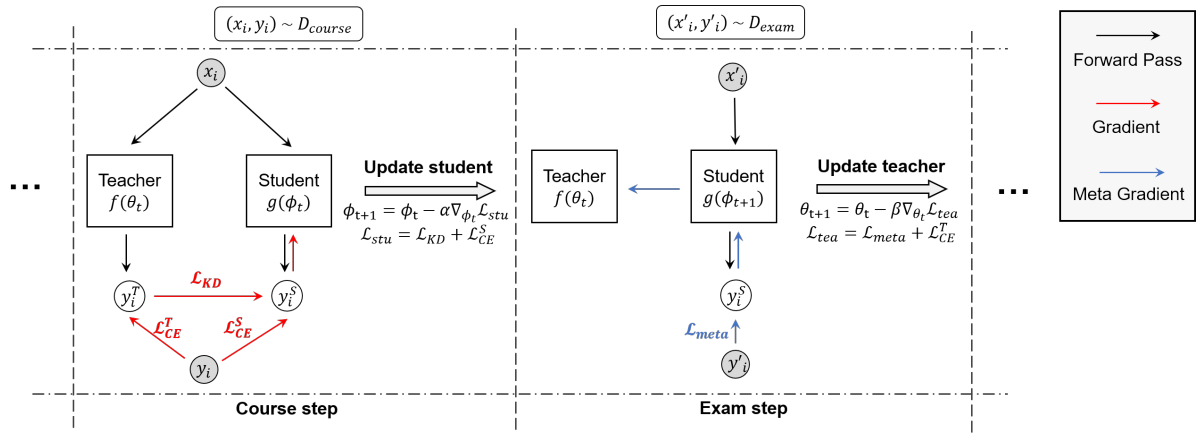


Figure 2: Illustration of our proposed Interactive Knowledge Distillation (IKD). IKD consists of two iterative steps: (a) Course step to update student with the soft targets generated by the teacher, (b) Exam step to update teacher with the feedback (i.e. cross entropy on the exam data) produced by the student. By iterating the course step and the exam step, both of student and teacher can be optimized.

dent's prediction should be minimized,

$$\mathcal{L}_{KD} = \mathrm{KL}(y^T, y^S) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y^T_{i,j} \log \frac{y^S_{i,j}}{y^T_{i,j}}, \tag{2}$$

where KL means Kullback-Leibler divergence.

In practice, the cross entropy between the student's prediction and the ground truth is also incorporated to be minimized, i.e.,

$$\mathcal{L}^S_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} \log y^S_{i,j}. \tag{3}$$

Thus the overall loss function for the student model can be written as:

$$\mathcal{L}_{stu} = \lambda\mathcal{L}_{KD} + (1-\lambda)\mathcal{L}^S_{CE}, \tag{4}$$

where λ is a hyper-parameter to balance the knowledge distillation loss and the cross entropy loss.

The student model is then updated by taking one gradient descent step with the loss function above, i.e.

$$\phi_{t+1} = \phi_t - \alpha\nabla_{\phi_t}\mathcal{L}_{stu}, \tag{5}$$

where α is the learning rate of the student model.

2.2 Exam Step: Teacher Optimization by Student's Feedback

It is expected that the student with the post-update parameters will generalize well on unseen samples. Therefore, the updated student is then evaluated on another batch of samples drawn from the exam data set $D_{exam} = \{x'_i, y'_i\}_{i=1}^{M}$. The performance of the student is measured by the cross entropy loss on the batch of exam data, which is called *meta loss* since it is calculated on the post-update parameter ϕ_{t+1} thus provides meta gradient w.r.t. the teacher's parameters.

$$\mathcal{L}_{meta} = \mathrm{CE}(y', y'^S_{t+1}) \tag{6}$$

$$= -\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{C} y'_{i,j} \log y'^S_{i,j}(\phi_{t+1}). \tag{7}$$

Our goal is to minimize $\mathcal{L}_{meta}$ by optimizing the teacher model, therefore the gradient of $\mathcal{L}_{meta}$ is w.r.t. the teacher's parameters θ, so is a second-order gradient (gradient of gradient). To explicitly show this, we can re-write $y'^S_{i,j}(\phi_{t+1})$ in the above equation as:

$$y'^S_{i,j}(\phi_{t+1}) = y'^S_{i,j}(\phi_t - \alpha\nabla_{\phi_t}\mathcal{L}_{stu}(\theta_t)), \tag{8}$$

in which we know that the gradient can flow into θ_t through the knowledge distillation in the course step. Figure 3 shows the computational graph of the forward pass and the backward pass.

Although it is computationally available by conducting an additional backward pass under current deep learning libraries that support automatic differentiation, we use the first-order approximation for its efficiency (Finn et al., 2017; Nichol et al., 2018).

In the context of meta-learning, the teacher can be regarded as the meta-learner, while D_{course} and D_{exam} can be analogous to support set and query set respectively.

Also, the standard cross entropy loss between the teacher's prediction and the ground truth can

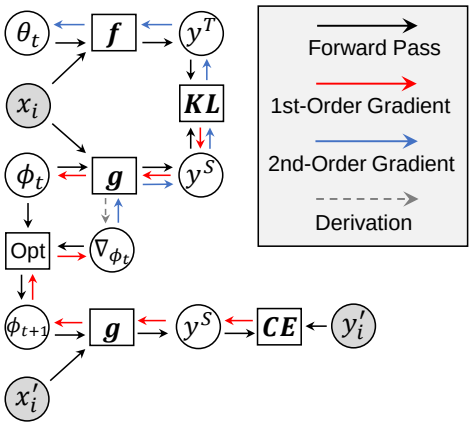


Figure 3: Computational graph to show how gradients are back-propagated into the teacher's parameters. Circles represent variable and squares represent operation. "Opt" stands for the optimizer, which is one gradient descent step. For simlicity, we omit the cross entropy loss for the student $\mathcal{L}_{CE}^{S}$ and for the teacher $\mathcal{L}_{CE}^{T}$ in the course step.

be incorporated,

$$\mathcal{L}_{CE}^{T} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j}\log y_{i,j}^{T}. \tag{9}$$

Thus, the overall loss function for the teacher model is

$$\mathcal{L}_{tea} = \gamma\mathcal{L}_{meta} + (1-\gamma)\mathcal{L}_{CE}^{T}, \tag{10}$$

where γ is another hyper-parameter to balance the meta loss and the cross entropy loss.

Thus, the teacher model can also be optimized by taking gradient descent:

$$\theta_{t+1} = \theta_t - \beta\nabla_{\theta_t}\mathcal{L}_{tea}, \tag{11}$$

where β is the learning rate of the teacher model.

Putting the student optimization in course step and the teacher optimization in exam step together, the overall loss function of IKD is defined as

$$\mathcal{L}_{IKD} = \mathcal{L}_{stu} + \mathcal{L}_{tea}. \tag{12}$$

By iterating the course step and the exam step, the student model learns from the soft targets generated by the teacher while the teacher model learns from the feedback generated by the student.

2.3 Discussion and Implementation

To take a closer look, in this section we further analyze the expanded form of the meta-gradient, which provides insights into the *student's feedback* and sheds light on how to take an approximation to achieve efficient training.

According to the chain rule, we can unfold the meta gradient $\nabla_{\theta_t}\mathcal{L}_{meta}$ as follows[2],

$$\begin{aligned}\nabla_{\theta_t}\mathcal{L}_{meta} &= \nabla_{\phi_{t+1}}\mathcal{L}_{meta}\cdot\nabla_{\theta}\phi_{t+1}\\ &= \nabla_{\phi_{t+1}}\mathcal{L}_{meta}\cdot(-\alpha\nabla_{\theta}\nabla_{\phi_t}\mathcal{L}_{stu})\\ &= \alpha\nabla_{\phi_{t+1}}\mathcal{L}_{meta}\cdot\nabla_{\theta}\nabla_{\phi_t}\log y_t^S\cdot y_t^T\\ &= \underbrace{\alpha\nabla_{\phi_{t+1}}\mathcal{L}_{meta}\cdot\nabla_{\phi_t}\log y_t^S}_{\text{feedback}}\cdot\nabla_{\theta_t}y_t^T.\end{aligned} \tag{13}$$

Note that $\nabla_{\phi_{t+1}}\mathcal{L}_{meta}$ is the Jacobi matrix of shape $|\phi|$ and $\nabla_{\phi_t}\log y_t^S$ is of shape $|\phi|\times C$.[3] Therefore, $\nabla_{\phi_{t+1}}\mathcal{L}_{meta}\cdot\nabla_{\phi_t}\log y_t^S$ is actually a weighting vector of shape C to adjust the gradient of teacher's prediction w.r.t. its parameters $\nabla_{\theta_t}y_t^T$. It is easy to find that the weighting vector only depends on the student's gradient, therefore it is exactly the student's feedback.

The behavior relationship between the weighting vector and the teacher's prediction is analyzed. It is worth noticing that the weighting vector should be detached from the computational graph to forbid the gradient flow into the student. By this, the meta loss will only contribute to the update of the teacher's parameters.

In practice, the difference between ϕ_t and ϕ_{t+1} is usually very small, which motivates us to take an approximation $\phi_{t+1}\approx\phi_t$ such that $\nabla_{\phi_{t+1}}\mathcal{L}_{meta}\approx\nabla_{\phi_t}\mathcal{L}_{meta}$. By this approximation, Eq. (13) can be simplified as

$$\nabla_{\theta_t}\mathcal{L}_{meta}\approx\alpha\nabla_{\phi_t}\mathcal{L}_{meta}\cdot\nabla_{\phi_t}\log y_t^S\cdot\nabla_{\theta_t}y_t^T. \tag{14}$$

Besides, note that $\nabla_{\phi_t}\mathcal{L}_{meta}$, $\nabla_{\phi_t}\log y_t^S$ and $\nabla_{\theta_t}y_t^T$ are mutually independent, therefore we can calculate them independently and achieve joint optimizing in an efficient way. In our experiments we set $D_{course}=D_{exam}=D_{train}$ for efficient optimization. Though, it should be mentioned that in our framework the course data set can be unlabeled, i.e. $D_{course}=\{x_i\}_{i=1}^{N}$. In this case, the cross entropy loss between the student's prediction and the ground truth is removed such that the loss for the student is exactly the knowledge distillation loss. Thus, only the supervision signal of D_{exam} is required. We leave IKD in this semi-supervised setting as future work.

[2]For the simplicity of derivation, we set N and M to 1, and omit the constant scalar λ.

[3]For matrix multiplication, $\nabla_{\phi_t}\log y_t^S$ should be in the transpose form, which is omitted here for the simplicity.

3 Experiments

Though IKD is orthogonal to most current knowledge distillation methods to compressing large-scale PLMs such as DistilBERT (Sanh et al., 2019) and TinyBERT (Jiao et al., 2020), which distill different feature sets (such as logits, embeddings, hidden states, attention maps, etc.), it is exhaustive to test IKD with these different distillation features and their combinations. Thus, in our experiments we mainly evaluate the effectiveness of IKD based on the vanilla knowledge distillation and BERT-PKD (Sun et al., 2019).

3.1 Datasets and Models

We conduct experiments on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019) with the backbone of BERT (Devlin et al., 2019), and on ChemProt dataset (Schneider et al., 2020) and SciCite dataset (Cohan et al., 2019) with backbone of SciBERT (Beltagy et al., 2019).

GLUE Benchmark We use seven text classification tasks in the GLUE benchemark: The Corpus of Linguistic Acceptability (CoLA) (Warstadt et al., 2019), The Stanford Sentiment Treebank (SST-2) (Socher et al., 2013), Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005), Quora Question Pairs (QQP) (Wang et al., 2019), Multi-Genre Natural Language Inference (MNLI) (Williams et al., 2018), Question Natural Language Inference (QNLI) (Rajpurkar et al., 2016) and Recognizing Textual Entailment (RTE) (Wang et al., 2019).

Domain Tasks We use ChemProt (Schneider et al., 2020) and SciCite (Cohan et al., 2019) to evaluate the performance of our method on domain specific tasks. ChemProt consists of 1,820 PubMed abstracts with chemical-protein interactions annotated by domain experts. SciCite is a dataset containing 11k annotated citation intents in biochemical and computer science domains. The standard metrics are micro F1 for ChemProt and macro F1 for SciCite. We evaluate our method on both tasks with the backbone of SciBERT (Beltagy et al., 2019).

3.2 Experimental Setup

Our implementation is based on PyTorch (Paszke et al., 2019). The training and evaluation are performed on a single RTX 2080Ti or RTX 3090 GPU. For downstream tasks in GLUE we first fine-tune the teacher model for 3 epochs with learning rate of 2e-5. For domain tasks the teacher is fine-tuned for 4 epochs. The batch size is 32 for all tasks. The student model is initialized with the parameters from the first k layers of BERT base(12 layer), where k is the number of the student's layer. In our experiments we mainly evaluate our method when $k = 4$ and $k = 6$. The total number of parameters of BERT_{24}, BERT_{12}, BERT_6 and Bert_4 are 340M, 110M, 67M and 52M respectively. The batch size is set to 4 for all downstream tasks. We conduct 3 experiments for each dataset and the median is reported.

3.3 Experimental Results

We report our results on GLUE benchmark. The results are presented in Table 1. We denote BERT_k as a BERT with only k layers. BERT_4-FT is to directly fine-tune the first 4 layers of the pre-trained BERT. BERT_4-KD represents student trained with vanilla knowledge distillation loss, i.e., Eq. (4). We also exhibit the reported result of TAKD (Mirzadeh et al., 2019) and Annealing KD (Jafari et al., 2021) for a direct comparision.

Results on GLUE Benchmark As shown in Table 1, IKD surpasses the vanilla knowledge distillation method on all tasks. For RTE, IKD outperforms directly finetuning by 3.2% and knowledge distillation by 2.9%. For QNLI and MRPC, although the performance of knowledge distillation compared with directly fine-tuning is even detrimental, IKD can consistently improve the students. We also evaluate our method on the GLUE test server and we show the results in Table 1. On the GLUE leaderboard our method consistently improves the performance.

Results on Domain Tasks Experimental results on the ChemProt and SciCite test set are shown in Table 2, which demonstrate the effectiveness of IKD on domain specific task. It's worth noting that on the SciCite dataset our approach is even able to make the student outperform the teacher.

Compatibility with Patient Knowledge Distillation To evaluate the compatibility of IKD with more competitive distillation methods, we further apply IKD on BERT-PKD (Sun et al., 2019) (denoted as BERT-PKD-IKD) with teacher model as BERT-base (12 layers) and student as BERT_6. BERT-PKD exploits information from hidden fea-

Method	MNLI (393k)	QQP (364k)	QNLI (105k)	SST-2 (67k)	CoLA (8.5k)	MRPC (3.7k)	RTE (2.5k)	Macro Score
Dev Set								
$BERT_{24}$ (Teacher)	87.0	88.7	92.6	93.2	63.4	88.8	70.0	83.4
$BERT_4$-TAKD†	72.4	**87**	82.6	89.1	34.2	85.2	59.6	72.9
$BERT_4$-Annealing KD†	74.4	86.5	83.1	**89.4**	**36.0**	**86.2**	61.0	73.8
$BERT_4$-FT	78.2	85.9	85.4	88.0	28.6	85.1	63.2	73.5
$BERT_4$-KD	78.5	86.4	85.2	88.8	30.4	84.6	63.5	73.9
$BERT_4$-IKD	**79.1**	86.5	**85.5**	89.1	30.9	86.0	**66.4**	**74.7**
Test Set								
$BERT_4$-KD	78.2	68.8	84.6	**90.1**	27.7	82.5	**63.5**	70.8
$BERT_4$-IKD	**78.4**	**68.9**	**85.3**	**90.1**	**30.6**	**83.2**	62.9	**71.3**

Table 1: Comparisons between IKD and Vanilla KD on the dev and test set of GLUE tasks. The teacher network is $BERT_{24}$ and the student network is $BERT_4$. FT indicates that the student is directly fine-tuned without any soft targets. KD means that the student learns using vanilla knowledge distillation. IKD reprensents our method. The best results for each task are in-bold. Methods with † denote results reported from Jafari et al. (2021).

Method	ChemProt (4.2k)	SciCite (7.3k)
$SciBERT_{12}$(Teacher)	84.5	86.1
$SciBERT_6$-FT	78.1	85.0
$SciBERT_6$-KD	79.3	85.7
$SciBERT_6$-IKD	**79.9**	**86.6**

Table 2: Experimental results of SciBERT on the test set of ChemProt and SciCite.

Method	QNLI	SST-2	MRPC	RTE
$BERT_{12}$(Teacher)	90.4	92.4	88.9	68.2
$BERT_6$-PKD	88.3	89.1	87.6	66.8
$BERT_6$-PKD-IKD	**88.7**	**90.3**	**87.7**	**67.1**

Table 3: Performance based on patient knowledge distillation on GLUE dev set. PKD represents patient knowledege distillation and PKD-IKD means combination of PKD and IKD.

tures for distillation. We follow PKD-Skip strategy which is shown to be better than PKD-Last as suggested by Sun et al. (2019). Experimental results are listed in Table 3, which demonstrate the versatility of IKD with stronger baselines.

4 Analysis

In this section, we take a closer look at IKD by conducting empirical analysis to provide some insights to understand how it works. In particular, we focus on the dynamic of generating soft targets and the feedback from the student.

Dataset	Epoch 1	Epoch 2	Epoch 3
MRPC	0.2637	0.1076	0.0343
RTE	0.4461	0.2313	0.0845
ChemProt	0.5049	0.2686	0.1451

Table 4: The average entropy of output distribution for each epoch on MRPC, RTE and ChemProt dataset.

4.1 About the Soft Targets

Since IKD allows the teacher to dynamically adjust its soft targets by using the feedback from the student, it would be interesting to see how the output of the teacher changes as the training goes on. In Figure 4 we visualize the output soft targets of the teacher output in different training epochs. We can find that the soft targets are relatively smooth at the beginning and become sharper as the training progresses. This is consistent with the hand-designed smoothing schedules (Dogan et al., 2020; Jafari et al., 2021), which introduce a hyper-parameter to control how quickly the soft targets converge to the one-hot distribution. In contrast, our method automatically learns to adjust the smoothness of the soft targets based on the feedback from the student. In Table 4 we further use entropy to measure the smoothness of output targets in different epochs on full training sets of MRPC, RTE and ChemProt. The average entropy goes down over the training process as shown, which confirms our observation as well.

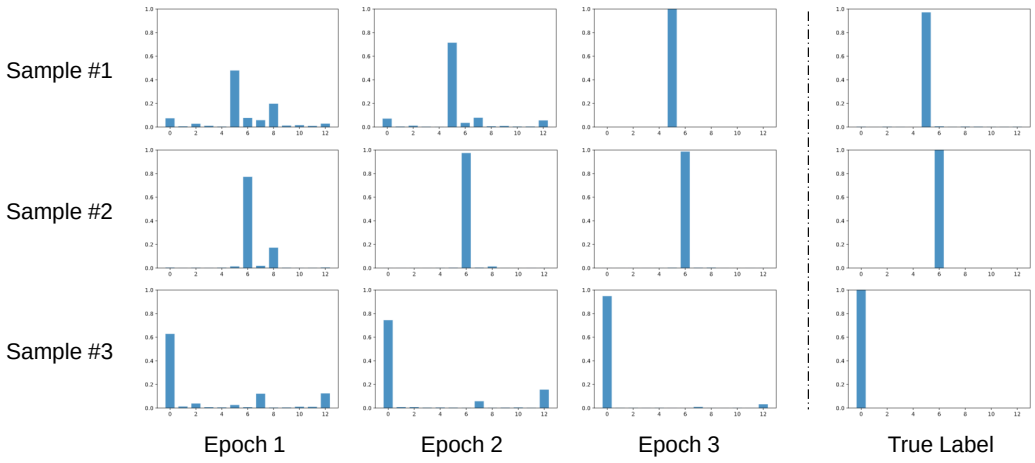


Figure 4: Visualization of the generated soft targets of three training epochs on ChemProt dataset produced by the corresponding teacher. We can find that the soft targets are softer at the early stage and become sharp as the training goes on. Finally the soft targets are quite similar to the one-hot distribution.

4.2 About the Feedback

As mentioned in 2.3, the gradient of the meta loss can be viewed as a weighted sum of the gradient of teacher's prediction w.r.t. its parameters, i.e. $\nabla_\theta y^T$ (See Eq. (14)). The weighting vector $\alpha \nabla_\phi \mathcal{L}_{meta} \cdot \nabla_\phi \log y^S$ can be regarded as the feedback of the student. Assume $fb = \alpha \nabla_\phi \mathcal{L}_{meta} \cdot \nabla_\phi \log y^S$, Eq. (14) can be written as

$$\nabla_\theta \mathcal{L}_{meta} \approx fb \cdot \nabla_\theta y^T = \nabla_\theta \sum_{i=1}^{C} fb_i \cdot y_i^T, \quad (15)$$

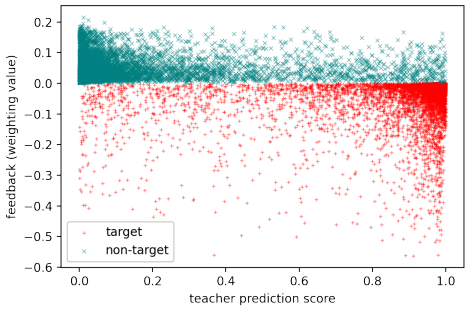
where $fb \in \mathbb{R}^C$, $y^T \in \mathbb{R}^C$, C is the number of classes.

In this section, we attempt to analyze the relationship between the student's feedback fb and the teacher's prediction y^T. In particular, we plot (fb_i, y_i^T) for each $i \in \{1, \cdots, C\}$ in Figure 5. The data points are collected from the 7,500 training steps on CoLA dataset. CoLA is a binary classification task so we can plot $C = 2$ data points for each training step such that there are 15,000 data points in total.

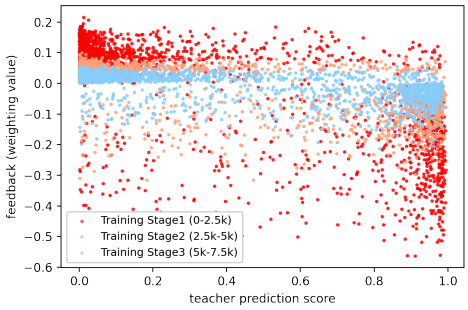
Note that The greater the student's feedback fb_i for class i, the greater the penalty of teacher's prediction on class i. As shown in Figure 5a, the student's feedback is always non-positive for teacher's prediction on target label, which can also be revealed by the expanded form of the feedback:

$$\alpha \nabla_\phi \mathcal{L}_{meta} \nabla_\phi \log y_c^S = -\alpha (\nabla_\phi \log y_c^S)^2, \quad (16)$$

where c is the target label. Besides, we can see that the student's feedback is usually non-negative



(a) Feedback of target vs. non-target.



(b) Feedback in different training stage.

Figure 5: Visualization results to show the relationship between student feedback and teacher prediction scores on CoLA dataset.

for teacher's prediction on non-target label. Thus, the teacher's prediction is always encouraged to approach the one-hot distribution where only the target label is 1, which has been verified in analysis. Also, we find that the range of feedback on target prediction is approximately three times larger than the range of feedback on non-target prediction.

In addition, the student's feedback can be different even when the teacher's predictions are the same. To take a deeper look, we demonstrate the value of feedback of the student at different training stage in Figure 5b. We can see that the student feedback converges to zero as the training goes, which implies that IKD works mainly at the immediate training stage.

5 Related Work

Our method draws on the idea of meta-learning and applies it to knowledge distillation for NLU. Thus, there are two lines of work that are related: (a) Knowledge distillation methods for language understanding, (b) Optimization-based meta-learning. We will introduce the two lines of work in Transformer Based KD and Optimization-Based Meta-Learning, and the intersect of the both lines in Teacher-Student Framework with Meta-Learning.

5.1 Knowledge Distillation for NLP

Knowledge distillation (KD) was originally proposed in domains other than NLP (Bucila et al., 2006; Hinton et al., 2015). With the emergence of large-scale pre-trained language models (PLMs) such as GPT (Radford et al., 2019), BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019), KD has gain much attention in the NLP community due to its simplicity and efficiency in compressing the large PLMs. Much effort has been devoted to compress BERT via KD, including BERT-PKD (Sun et al., 2019), DistilBERT (Sanh et al., 2019), MobileBERT (Sun et al., 2020), and TinyBERT (Jiao et al., 2020). These methods have achieved great success by exploring informative features to be distilled, such as word embeddings, attention maps, hidden states, and output logits. Different from these work, our proposed IKD focuses on the interaction between teacher and student therefore is orthogonal to these work. Gou et al. (2021) provides a survey of online knowledge distillation where both the teacher and the student are updated simulatenously. Our IKD can be regarded as a special case of online KD.

5.2 Optimization-Based Meta-Learning

Optimization-based meta-learning intends to learn a group of initial parameters that can fast converge on a new task with only a few gradient steps, in which the learner and meta-learner share the same architecture. As a representative method, Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) uses a nested loop to find an optimal set of parameters on a variety of learning tasks, such that they can adapt well after a small number of training steps. To tackle the high-order derivative, Finn et al. (2017) also propose First-Order MAML (FOMAML) to perform the first-order approximation to reduce the computation. Similar to MAML, Reptile (Nichol et al., 2018) utilizes the weights in the inner loop instead of gradient to update meta parameters and achieves competitive performance on several few-shot classification benchmarks.

5.3 Teacher-Student Framework with Meta-Learning

Meta-learning has been proved efficient when combined with the teacher-student framework. Liu et al. (2020) employ meta-learning to optimize a label generator that generates dynamic soft targets for intermediate layers. Their method is proposed for self-boosting. Pan et al. (2020) focus on training a meta-teacher possessing multi-domain knowledge by metric-based meta-learning and then teach single-domain students. Meta Pseudo Labels (Pham et al., 2020) leverage a teacher network to generate pseudo labels on unsupervised images to enlarge the training set of the student. They aim to train a pseudo labels generator to while IKD dedicates to compress a large model into a smaller one. Thus the pseudo label generator is trained from scratch and shares same architecture with the student while in our IKD the teacher must be a larger pretrained model which leads to the impossibility of implementation of gradient substitution in MAML.

6 Conclusion

This paper proposes the Interactive Knowledge Distillation (IKD) to empower the teacher to learn to teach with student feedback. Inspired by MAML, IKD involves two optimization steps: (1) a course step for student optimization with the guidance of teacher and (2) an exam step for teacher optimization with the feedback from student. IKD shows superiority over vanilla KD on a suit of NLP tasks. By iterating the course step and exam step we can jointly optimize the teacher and student. In addition, we have discussed a semi-supervised generalization of our method, in which the course data is allowed to be unlabeled.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. 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 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 535–541. ACM.

Arman Cohan, Waleed Ammar, Madeleine van Zuylen, and Field Cady. 2019. Structural scaffolds for citation intent classification in scientific publications. 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 3586–3596, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Ürün Dogan, Aniket Anand Deshmukh, Marcin Machura, and Christian Igel. 2020. Label-similarity curriculum learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXIX*, volume 12374 of *Lecture Notes in Computer Science*, pages 174–190. Springer.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

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

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 2493–2504. Association for Computational Linguistics.

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.

Benlin Liu, Yongming Rao, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. 2020. Metadistiller: Network self-boosting via meta-learned top-down distillation. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XIV*, volume 12359 of *Lecture Notes in Computer Science*, pages 694–709. Springer.

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. *CoRR*, abs/1907.11692.

Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. 2019. Improved knowledge distillation via teacher assistant.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999.

Haojie Pan, Chengyu Wang, Minghui Qiu, Yichang Zhang, Yaliang Li, and Jun Huang. 2020. Meta-kd: A meta knowledge distillation framework for language model compression across domains. *CoRR*, abs/2012.01266.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. 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 8024–8035.

Hieu Pham, Qizhe Xie, Zihang Dai, and Quoc V. Le. 2020. Meta pseudo labels. *CoRR*, abs/2003.10580.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Elisa Terumi Rubel Schneider, João Vitor Andrioli de Souza, Julien Knafou, Lucas Emanuel Silva e Oliveira, Jenny Copara, Yohan Bonescki Gumiel, Lucas Ferro Antunes de Oliveira, Emerson Cabrera Paraiso, Douglas Teodoro, and Cláudia Maria Cabral Moro Barra. 2020. BioBERTpt - a Portuguese neural language model for clinical named entity recognition. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 65–72, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

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.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2158–2170. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

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.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Trans. Assoc. Comput. Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.