BiKT: Enabling Bidirectional Knowledge Transfer Between Pretrained Models and Sequential Downstream Tasks

Hang Zeng[1], **Chaoyue Niu**[1*], **Fan Wu**[1], **Shaojie Tang**[2]
Leihao Pei[3], **Chengfei Lv**[3], **Guihai Chen**[1]
[1]Shanghai Jiao Tong University, [2]University of Texas at Dallas, [3]Alibaba Group
{nidhogg, rvince}@sjtu.edu.cn

Abstract

Adapting pretrained models to downstream tasks is important in practical applications. Existing frameworks adapt from an initial pretrained model to each downstream task directly, but ignore the sequential nature of the downstream tasks and their feedback effect on the pretrained model. In this work, we propose a new framework, called BiKT, to enable bidirectional knowledge transfer between pretrained models and downstream tasks in rounds. We model each downstream task in the current round as a target task for adaptation and treat all the tasks in the previous rounds as source tasks for feedback. We design a feedback algorithm by multi-task learning over the labeled data of the source tasks, where task-specific prompts are plugged into the backbone network for decoupling task-exclusive knowledge from task-shared knowledge. We further utilize the good initiation of the new backbone network updated in the feedback phase and the trained prompts of the source tasks for adaptation. Evaluation over 9 GLUE datasets, 6 SuperGLUE datasets, and 8 other datasets using models with different pretraining levels and different parameter scales shows remarkable improvement in full-shot and few-shot adaptation settings.

1 Introduction

Language models are initially pretrained on a diverse corpus and then adapted to various downstream tasks in practical applications. Finetuning (Radford et al., 2018) and prompt tuning (Lester et al., 2021) are two typical and widely used adaptation methods. In particular, finetuning updates all the parameters of a pretrained language model, shifting the whole backbone network towards a specific downstream task. In contrast, prompt tuning freezes the backbone network and tunes a small number of parameters inserted to the pretrained model for each downstream task.

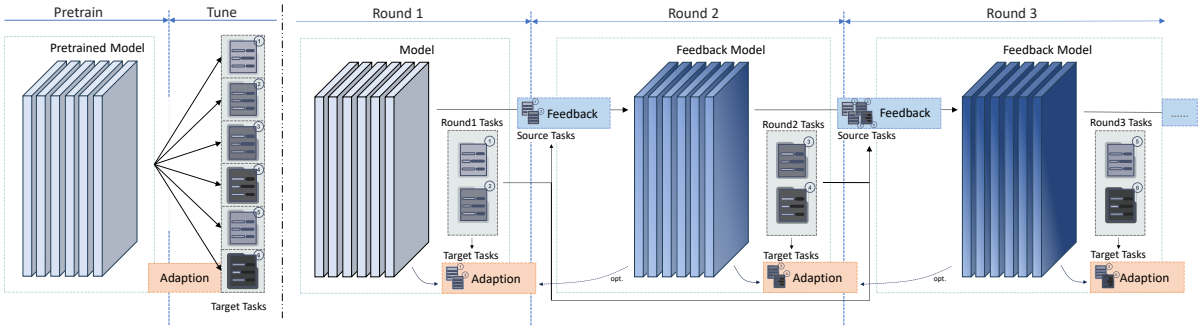
As shown in Figure 1a, the current pretrain-then-tune paradigm directly adapts a pretrained model to each downstream task, which is a one-time and unidirectional knowledge transfer. Multiple downstream tasks are treated independently as different ending points with the initial pretrained model as the same starting point of adaptation. Nevertheless, in practice, the deployment requirements of the pretrained model in different application scenarios are raised over time rather than all at once. Therefore, as depicted in Figure 1b, the downstream tasks should come in a sequential way, and a certain task for adaptation has some available previous tasks that have been adapted to ahead of this task.

In this work, we consider how to enable the bidirectional knowledge transfer between a pretrained model and downstream tasks. Such a problem is well-motivated. On the one hand, different from pretraining over public corpus, the downstream tasks in practical applications directly serve a large scale of users and receive their feedback, thereby generating massive fresh labeled samples. These high-quality samples can be exploited to continuously improve the generalization ability of the pretrained model in the real open world. On the other hand, the pretrained model, which has been optimized over previous downstream tasks, can become a better starting point of adaptation to the current task, facilitating knowledge transfer from upstream to downstream in the sequence of tasks.

To deal with the problem above, we propose a new framework, called BiKT, including the feedback phase from sequential downstream tasks to pretraining and the adaptation phase from pretraining to downstream tasks, thereby boosting the generalization ability of the pretrained model and improving the adaptation performance. We first model the sequential nature of downstream tasks in different rounds. We call a certain task in the current round to be adapted to as the target task and treat the tasks ahead of the target task in the previous

*C. Niu is the corresponding author (rvince@sjtu.edu.cn).

Findings of the Association for Computational Linguistics: EACL 2024, pages 3156–3171



(a) Previous paradigm (b) New adaptation and feedback paradigm in BiKT

Figure 1: Unidirectional adaptation to each downstream task independently in conventional finetuning and prompt tuning frameworks (left) vs. Bidirectional feedback and adaptation between a pretrained model and sequential downstream tasks in the proposed BiKT (right).

rounds as the source tasks for feedback. We then design a multi-task feedback algorithm over the source tasks for the pretrained model with trainable prompts in an incremental way. The newly inserted prompts are task-specific, whereas the backbone network of the pretrained model is task-shared, eliminating the interference of task-specific knowledge on the generalization of the pretrained model. After feedback, we further leverage the latest backbone network and the trained prompts of the source tasks from the feedback phase as good model initialization for adapting to target tasks.

We summarize the key contributions as follows:

- We, for the first time, formulate the bidirectional relationship between a pretrained model and downstream tasks under the sequential modeling of downstream tasks. In contrast, existing pretrain-then-tune work did not have the feedback phase and studied only the unidirectional adaptation from the initial pretrained model to each individual downstream task.
- We propose BiKT, comprising (1) a multi-task feedback algorithm over downstream tasks with task-specific prompts to decouple task-exclusive and task-shared knowledge, while enhancing the pretrained model by absorbing task-shared knowledge; and (2) an adaptation algorithm based on the new pretrained model from the feedback phase for initialization.
- We extensively evaluate BiKT using 23 public datasets with sub-billion scale models (i.e., BERT-base and RoBERTa-base) and billion scale models (i.e., Qwen1.5 and Phi-1.5). Evaluation results reveal that BiKT outperforms vanilla prompt tuning and finetuning by more than 5.8% and 1.4%, respectively.

2 Related Work

In this section, we briefly review related work.

Downstream Task Adaptation. Existing full parameter finetuning methods and parameter-efficient finetuning methods (e.g., prompt tuning (Lester et al., 2021), LoRA (Hu et al., 2022), $(IA)^3$ (Liu et al., 2022), BitFit (Zaken et al., 2022)) mainly focused on one-step adaptation and started from the initial pretrained model and end at each individual downstream task, failing to leverage historical tasks to boost the performance of the pretrained model when adapting to new tasks.

For the setting of multiple downstream tasks, one line of work (Vu et al., 2022; Asai et al., 2022; Wang et al., 2023b) adopted multi-task learning to learn a task-shared prompt from source tasks as good initialization when adapting to target tasks with prompt tuning, but kept the backbone network unchanged. In contrast, BiKT executes in rounds, not only initializing the prompts for each target task with the learnt prompts of source tasks but also adopting the new backbone network optimized in the feedback phase. Another line of work (Wei et al., 2022; Sanh et al., 2022) proposed multi-task instruction finetuning with different manual prompts for different downstream tasks, enabling the pretrained model to follow human instructions more effectively. These work ideally requires downstream tasks to arrive all at once. In contrast, BiKT models the practical sequential nature of downstream tasks and supports feedback and adaptation in rounds. The feedback algorithm of BiKT also takes task-specific soft prompts rather than manual hard instructions.

Continual Learning. The original goal is to memorize both old tasks and new tasks (Wang

et al., 2023a). Existing work proposed regularization (Liu et al., 2021), rehearsal (Rebuffi et al., 2017), or parameter freezing (Wang et al., 2022) to avoid catastrophic forgetting of the old tasks. In contrast, the goal of the feedback phase in BiKT is to memorize task-shared knowledge into the backbone network for better adaptation performance, rather than to memorize specific old/source tasks.

Another variant is continual pretraining. Existing work proposed to continuously add newly acquired unlabeled data for pretraining (Gururangan et al., 2020; Ke et al., 2023; Cossu et al., 2022) or incrementally apply unsupervised pretraining objectives (Sun et al., 2020). In contrast, BiKT mines task-shared knowledge from the labeled data with different distributions from downstream tasks.

3 Problem Formulation

We formulate the problem of bidirectional knowledge transfer between a pretrained model and sequential downstream tasks, including the feedback link from the downstream tasks to the pretrained model and the adaptation link from the pretrained model to the downstream tasks.

3.1 Sequential Downstream Tasks Modeling

We let $1, 2, \ldots, T$ denote T rounds of downstream tasks in total. In round t, a batch of n_t downstream tasks come, and their corresponding datasets are denoted as $\mathcal{D}^t = \{D_1^t, D_2^t, ..., D_{n_t}^t\}$. We let $D_i^t = (\mathbf{X}_i^t, \mathbf{Y}_i^t) = \{x_{ij}^t, y_{ij}^t\}_{j=1}^{m_i^t}$ denote the dataset of the i-th downstream task in round t with m_i^t samples in total, where x_{ij}^t and y_{ij}^t denote the feature vectors and the label of the j-th sample. For a certain downstream task with its dataset $\mathcal{T}_i^t = D_i^t \in \mathcal{D}^t$, we call it the *target task* to be adapted to, while calling the tasks ahead of the target task in the previous $t-1$ rounds as the *source tasks* for feedback, the datasets of which are denoted as $\mathcal{S}^t = \{\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^{t-1}\}$.

3.2 From Downstream Tasks to Pretraining

We first consider how to exploit source tasks to improve the pretrained model before target task adaptation. We let Θ^0 denote the initial pretrained model and let Θ^t denote the new pretrained model in round t. We formulate the sequential feedback problem of downstream tasks on the pretrained model round by round as follows.

In round 1, there is no source task (i.e., $\mathcal{S}^1 = \varnothing$), and the pretrained model is not updated (i.e., $\Theta^1 = \Theta^0$). Then, in round 2, given the source tasks with the datasets $\mathcal{S}^2$, the goal is to design a feedback algorithm F that takes the pretrained model Θ^1 and $\mathcal{S}^2$ as inputs and outputs a new model

$$\Theta^2 = F(\mathcal{S}^2; \Theta^1), \tag{1}$$

which enhances the generalization ability over target tasks. In round 3, the new pretrained model Θ^2 is further updated to Θ^3 with the source task datasets $\mathcal{S}^3$. Following the same reasoning, in round t, Θ^{t-1} functions as the starting model and is updated to

$$\Theta^t = F(\mathcal{S}^t; \Theta^{t-1}). \tag{2}$$

In fact, the feedback algorithm needs to continuously update the pretrained model with the datasets of new downstream tasks, where the starting model for feedback in the current round comes from the ending model that has been optimized over the source tasks in the previous rounds. In contrast, there was no feedback phase in the conventional adaptation paradigm, where the source tasks were not exploited to update the pretrained model in each round (i.e., $\forall t, \Theta^t = \Theta^0$).

3.3 From Pretraining to Downstream Tasks

We then consider how to adapt to the dataset of each target task $\mathcal{T}_i^t$ in round t. The adaptation should be based on the new pretrained model Θ^t obtained from the feedback phase. In contrast, conventional adaptation paradigm starts from the initial pretrained model Θ^0.

Intuitively, as shown in Figure 1a, existing adaptation methods, such as finetuning and prompt tuning, directly go from the initial pretrained model to each target task by ignoring the source tasks in the previous rounds, implying that each downstream task is independent from all the other tasks. In contrast, as shown in Figure 1b, this work adopts sequential task modeling, and for each target task, all the previous tasks are exploited in the feedback phase, thereby improving adaptation performance.

4 Design of BiKT

4.1 Design Objective

The workflow of BiKT is depicted in Figure 2. The key design goal is to boost the pretrained model with historical downstream tasks (i.e., source tasks) in sequence to generalize better on new downstream tasks (i.e., target tasks).

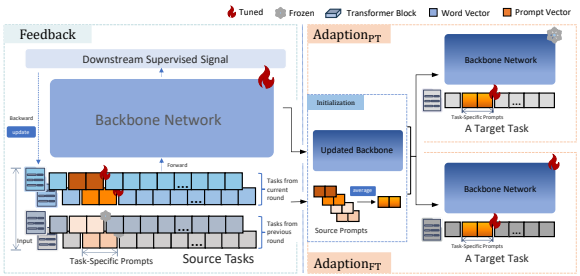


Figure 2: General workflow of BiKT. For the feedback phase of BiKT (left), multiple source tasks are used to update the task-shared backbone network and the task-specific prompts of source tasks from current round. For the adaptation phase of BiKT (right), the model initialization is from feedback phase, and different tuning methods can be applied.

To design the feedback algorithm F for the source tasks on the pretrained model, we first introduce the optimization objective. In round t, the starting model Θ^{t-1} from the previous round will be optimized over the datasets of the source tasks $\mathcal{S}^t = \{\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^{t-1}\} = \{(\mathbf{X}_i^k, \mathbf{Y}_i^k)_{i=1}^{i=n_k} | k = 1, 2, \ldots, t-1\}$, formally,

$$\max_{\Delta\Theta^t} \prod_{k=1}^{t-1} \prod_{i=1}^{n_k} P\left(\mathbf{Y}_i^k | \mathbf{X}_i^k; \Theta^{t-1} + \Delta\Theta^t\right). \quad (3)$$

where $\Delta\Theta^t$ denotes the model update of Θ^{t-1}. The new model $\Theta^t = \Theta^{t-1} + \Delta\Theta^t$ with the feedback of the source tasks not only serves as the initialization for adapting to each target task in round t as follows, but also will function as the starting model of the feedback algorithm in round $t+1$. During the whole feedback process, to decouple task-shared knowledge from task-specific knowledge, task-specific prompts can be plugged into the backbone network for different source tasks.

Then, for a certain target task $\mathcal{T}_i^t = D_i^t = (\mathbf{X}_i^t, \mathbf{Y}_i^t) \in \mathcal{D}^t$ in round t, the objective of the adaptation algorithm is

$$\max_{\Delta\Theta_i^t} P(\mathbf{Y}_i^t | \mathbf{X}_i^t; \Theta^t + \Delta\Theta_i^t), \quad (4)$$

where $\Delta\Theta_i^t$ denotes the model update of Θ^t for $\mathcal{T}_i^t$.

From the objective functions, we can see that the feedback algorithm is essentially a multi-task learning process over the source tasks, while the adaptation algorithm is a single-task tuning process for each individual target task. For each round with a batch of downstream tasks, the latest model after the feedback process can be immediately used for adaptation and also helps to restart the next feedback process, enabling bidirectional knowledge transfer between a pretrianed model and the downstream tasks.

Algorithm 1 Feedback of Source Tasks

Require: Source tasks in round t: $S^t = \{\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^{t-1}\}$; Backbone network updated in round $t-1$: Θ^{t-1}; Already learnt prompts for the downstream tasks in previous $t-2$ rounds: $\{\Phi_i^k|_{i=1}^{i=n_k}, k = 1, \ldots, t-2\}$

1: Randomly mix the training data from S^t;
2: Insert prompts for each source task;
3: Initialize the backbone network with Θ^{t-1};
4: **if** $t \geq 2$ **then**
5: Initialize the prompts of the downstream tasks in previous $t-1$ rounds with $\{\Phi_i^k|_{i=1}^{i=n_k}, k = 1, \ldots, t-2\}$, and freeze these prompts;
6: **end if**
7: Freeze the word embedding layer;
8: Train the model with the mixed data.

4.2 Feedback with Trainable Prompts

The input data of the feedback algorithm F are the random mixture of the datasets of the source tasks. The backbone networks take the Transformer encoder architecture for discriminative models (Devlin et al., 2019; Liu et al., 2019) and the decoder-only architecture for generative models (Bai et al., 2023; Li et al., 2023). Classification headers are added for discriminative models to identify different kinds of labels from different tasks, but are not required for generative models since both inputs and labels of samples are transformed into plain text format.

During the feedback phase, pretrained backbone network is finetuned over the mixed downstream tasks. However, not all the knowledge of a task is helpful for the others. Our goal is to accumulate the general knowledge shared by different source tasks. To separate task-specific knowledge from task-shared knowledge, we introduce soft prompts for each source task. In particular, the task-specific prompt is prepended to the input embeddings of the backbone network. Therefore, in round t, the full model comprises the task-shared backbone network Θ^{t-1} and the prompts for the source tasks $\mathcal{S}^t$, including the trainable prompts for n_{t-1} downstream tasks in round $t-1$, denoted as $\Phi_i^{t-1}|_{i=1}^{i=n_{t-1}}$, and the already trained and now frozen prompts for tasks in previous $t-2$ rounds, denoted as

Algorithm 2 Adaptation to Target Tasks

Require: A target task in round t: $\mathcal{T}_i^t = D_i^t$; Backbone network updated in round t: Θ^t; Learnt prompts for downstream tasks in previous $t-1$ rounds: $\{\Phi_i^k|_{i=1}^{i=n_k}, k = 1, \ldots, t-1\}$

1: Insert prompt for $\mathcal{T}_i^t$;
2: Initialize the backbone network with Θ^t;
3: Initialize the prompt with

$$\Phi_i^t = \sum_{k=1}^{t-1} \sum_{i=1}^{n_k} \frac{1}{\sum_{k=1}^{t-1} n_k} \Phi_i^k;$$

4: (Optional) Freeze the backbone network;
5: Train the model with D_i^t.

$\{\Phi_i^k|_{i=1}^{i=n_k}, k = 1, \ldots, t-2\}$. To achieve more stable learning of prompts, we freeze the word embedding layer of backbone network.

The training loss for discriminative models is

$$-\sum_{k=1}^{t-1} \sum_{i=1}^{n_k} \sum_{j=1}^{m_i^k} \log P(y_{ij}^k | x_{ij}^k; \Theta^{t-1}, \Phi_i^k), \quad (5)$$

where (x_{ij}^k, y_{ij}^k) denotes a labeled sample from the i-th task in round k, and the task-specific prompt Φ_i^{t-1} is updated only over the dataset of the i-th downstream task in round $t-1$. The loss function for generative models is

$$-\sum_{k=1}^{t-1} \sum_{i=1}^{n_k} \sum_{j=1}^{m_i^k} \log \prod_{w=1}^{|y_{ij}^k|} P(y_{ijw}^k | x_{ij}^k, y_{ij,<w}^k; \Theta^{t-1}, \Phi_i^k), \quad (6)$$

where y_{ijw}^k denotes the w-th word in the label y_{ij}^k, $y_{ij,<w}^k$ denotes the words ahead of the w-th word, and $|y_{ij}^k|$ denotes the size of words in y_{ij}^k.

We summarize the feedback design in Algorithm 1. The time complexity of T-round feedback is $O(Tn)$, where $n = \sum_{t=1}^{T} n_t$ denotes the total number of downstream tasks. Please refer to Appendix A for detailed analysis.

4.3 Feedback-Based Model Adaptation

In round t, after getting the new backbone network Θ^t based on the feedback algorithm, we use it as an initialization to adapt to each target task. For the target task with the dataset $\mathcal{T}_i^t = D_i^t \in \mathcal{D}^t$, we still insert the task-specific prompt Φ_i^t ahead of the input embeddings of the backbone network. In addition to the random initialization way in vanilla prompt tuning, the prompt Φ_i^t of the target task can leverage the learnt prompts of the source tasks, namely, $\{\Phi_i^k|_{i=1}^{i=n_k}, k = 1, \ldots, t-1\}$. Empirically, averaging all the learnt prompts is a simple but effective way for initialization, formally,

$$\Phi_i^t = \sum_{k=1}^{t-1} \sum_{i=1}^{n_k} \frac{1}{\sum_{k=1}^{t-1} n_k} \Phi_i^k. \quad (7)$$

Then, the adaptation to the target task can be achieved by finetuning both the backbone network and the task-specific prompt or freezing the backbone network and only tuning the prompt, denoted as **BiKT$_{\text{FT}}$** and **BiKT$_{\text{PT}}$**, respectively. The loss functions for the adaptation algorithm are the same as those in the feedback algorithm above.

We summarize the adaptation design in Algorithm 2. Compared with vanilla prompt tuning or finetuning to the target task, the key difference is the good initialization for both the backbone network and the task-specific prompt from the feedback algorithm over the source tasks, which is important in both full-shot and few-shot settings.

Round 1		Round 2			Round 3		
Task	#Train	Task	#Train	#Valid	Task	#Train	#Valid
COLA	8551	BoolQ	9427	3270	ANLI-R1	16946	1000
MNLI	392702	CB	250	56	ANLI-R2	45460	1000
MRPC	3668	COPA	800	200	ANLI-R3	100459	1200
QNLI	104743	MultiRC	27243	4848	SciTail	23587	1304
QQP	363846	RTE	2490	277	WinoGrande	10234	1267
RTE	2490	WiC	5428	638			
SST-2	67349	SNLI	549367	9842			
STS-B	5749	PAWS	49401	8000			
WNLI	635	IMDB	25000	25000			

Table 1: The statistics of the datasets for the source tasks and the target tasks in the evaluation.

5 Evaluation

We evaluate BiKT over a wide range of downstream tasks using various pretrained models.

5.1 Experimental Setups

Datasets. We take 23 nature language datasets for downstream tasks, including CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI, RTE, and WNLI from the GLUE benchmark (Wang et al., 2018); BoolQ, CB, COPA, MultiRC, RTE, and WiC from the SuperGLUE benchmark (Wang et al., 2019); SNLI (Bowman et al., 2015); PAWS-Wiki (Zhang et al., 2019); IMDB (Maas et al., 2011); ANLI (Nie et al., 2020); SciTail (Khot et al., 2018); and WinoGrande (Sakaguchi et al., 2020). The default sizes of the training set and the validation set

for each dataset are listed in Table 1. The detailed task types and data distributions of all tasks are listed in Table 8.

Pretrained Models and Prompts. We take different pretrained models with varying sizes, including BERT-base with 109M parameters, RoBERTa-base with 125M parameters, Qwen1.5-1.8B with 1.8B parameters, and Phi-1.5 with 1.3B parameters. All the pretrained checkpoints are loaded from huggingface[1]. For each task-specific prompt, the default size is set to 20.

Task Configuration. For the discriminative models of BERT and RoBERTa, all the downstream tasks are transformed into classification tasks, except for STS-B, which is treated as a regression task. The corresponding multi-task learning architecture shares the bottom network layers and adds task-specific headers at the top. For the generative models of Qwen and Phi, all the downstream tasks are cast as text generation, following the text-to-text approach (Raffel et al., 2020). The corresponding multi-task learning architecture shares the same model without headers. We directly evaluate the accuracy of the generated text, rather than using the output logits of language modeling head for preset labels (Gao et al., 2023).

We divide the downstream tasks and their datasets into three rounds. Since the order of raising the deployment requirements of pretrained model in application scenarios is natural, we allow each round to involve different types of tasks. In particular, round 1 involves 9 datasets from GLUE as target tasks; round 2 involves SNLI, PAWS, IMDB, and 6 datasets from SuperGLUE as target tasks, and the source tasks are from round 1; and round 3 involves 3 ANLI datasets, SciTail, and WinoGrande as target tasks, and the source tasks are from round 1 and round 2. For few-shot (i.e., K-shot) learning in the adaptation to each target task, we randomly choose K training samples in total uniformly from different classes. We consistently use the full validation set of each target task to evaluate the model performance.

Baselines. We introduce the following baselines for comparison: (1) **Finetuning (FT)**, which tunes the initial pretrained model over each target task's training set; (2) **Finetuning with Prompt ($\mathbf{FT_{PT}}$)**, which plugs the randomly initialized prompt to the backbone network of the initial pretrained model and tunes both the backbone network and the prompt over each target task's training set; (3) **Prompt Tuning (PT)** (Lester et al., 2021), which differs from FT_{PT} in that only the prompt is tuned while the backbone network is frozen; and (4) **Prompt Tuning with Multi-Task Learning Initialization ($\mathbf{PT_{MT}}$)** (Vu et al., 2022), which differs from PT in that the initialization of each target task's prompt is a shared prompt learnt over multiple source tasks.

To validate the extensibility of BiKT, we also replace PT or FT in the adaptation algorithm of BiKT with parameter-efficient finetuning methods: (5) **LoRA** (Hu et al., 2022), which injects low rank decomposition matrices into each layer of the backbone; (6) **(IA)**^{3} (Liu et al., 2022), which scales activations by learned vectors; and (7) **BitFit** (Zaken et al., 2022), which only tunes the bias terms.

Comparison Fairness We note that all existing methods followed the conventional adaptation paradigm that considered the unidirectional adaptation from a pretrained model to each downstream task independently. Thus, depending on whether there is a feedback phase before adaptation, we can classify the baselines we used into two categories for fair comparison.

The first category of baselines, including FT, FT_{PT}, PT (Lester et al., 2021), LoRA (Hu et al., 2022), $(\text{IA})^3$ (Liu et al., 2022), and BitFit (Zaken et al., 2022), followed the conventional adaptation paradigm, which directly adapted a pretrained model to each downstream task without feedback. The fair comparison with these baselines validates the necessity of the proposed feedback algorithm of BiKT in the adaptation phase by updating the model with the datasets of the previous tasks, and also reveals the good compatibility of BiKT with different tuning methods. For the second category of baselines, including PT_{MT} (Vu et al., 2022) and BiKT without task-specific prompt, we have modeled existing methods into a feedback phase, after which the adaptation performance over the new model are used for comparison. The fair comparison with these baselines validates the necessity of separating the task-specific prompts and the task-shared backbone.

Implementation Details. We implement BiKT and all the baselines in PyTorch. The workstation has 8 NVIDIA V100 32G GPUs. For all methods, we use the AdamW optimization scheme. For the finetuning type of algorithms, including the base-

[1]https://huggingface.co

Model & Method	Round 2										Round 3					
	BoolQ	CB	COPA	MRC	RTE	WiC	SNLI	PAWS	IMDB	Avg.	ANLI-R1	ANLI-R2	ANLI-R3	SciTail	WinoGrande	Avg.
BERT-base (FT)	73.1	**84.5**	59.2	68.5	61.7	66.0	90.4	92.6	**93.9**	76.7	47.0	**48.5**	48.7	94.0	56.9	59.0
BERT-base (FT_{PT})	71.5	70.8	54.7	68.8	58.7	62.9	90.2	92.6	93.7	73.8	46.3	47.4	48.5	93.6	54.8	58.3
BERT-base ($BiKT_{FT}$)	**78.9**	80.4	**66.5**	**70.2**	**77.1**	**67.2**	**90.9**	**92.7**	93.6	**79.7**	**47.3**	47.4	**49.1**	**94.8**	**58.1**	**59.4**
$BiKT_{FT}$ vs. FT	**+5.8**	-4.1	**+7.3**	**+1.7**	**+15.4**	**+1.2**	**+0.5**	**+0.1**	-0.3	**+3.0**	**+0.3**	-1.1	**+0.4**	**+0.8**	**+1.2**	**+0.4**
$BiKT_{FT}$ vs. FT_{PT}	**+7.4**	**+9.6**	**+11.8**	**+1.4**	**+18.4**	**+4.3**	**+0.7**	**+0.1**	-0.1	**+5.9**	**+1.0**	**+0.0**	**+0.6**	**+1.2**	**+3.3**	**+1.1**
BERT-base (PT)	65.0	72.0	55.3	62.9	59.2	61.4	84.3	83.0	91.9	70.5	41.9	40.5	42.0	87.2	54.8	53.3
BERT-base (PT_{MT})	71.5	74.4	62.8	**65.9**	68.3	65.1	83.4	82.9	**92.3**	74.1	42.1	40.5	41.6	91.0	**56.5**	54.3
BERT-base ($BiKT_{PT}$)	**72.9**	**85.1**	**66.5**	65.2	76.3	**65.2**	**86.3**	**85.1**	92.2	**77.2**	**42.7**	**40.8**	**42.7**	**92.6**	56.2	**55.0**
$BiKT_{PT}$ vs. PT	**+7.9**	**+13.1**	**+11.2**	**+2.3**	**+17.1**	**+3.8**	**+2.0**	**+2.1**	**+0.3**	**+6.7**	**+0.8**	**+0.3**	**+0.7**	**+5.4**	**+1.4**	**+1.7**
$BiKT_{PT}$ vs. PT_{MT}	**+1.4**	**+10.7**	**+3.7**	-0.7	**+8.0**	**+0.1**	**+2.9**	**+2.2**	-0.1	**+3.1**	**+0.6**	**+0.3**	**+1.1**	**+1.6**	-0.3	**+0.7**
RoBERTa-base (FT)	**81.5**	**94.0**	56.7	**79.0**	78.2	**71.5**	**91.4**	**94.6**	**95.4**	82.5	**52.8**	47.4	48.3	**95.7**	55.8	60.0
RoBERTa-base (FT_{PT})	81.4	92.9	60.3	71.5	77.1	71.1	91.3	94.7	95.4	81.7	52.7	**48.3**	49.0	95.7	57.2	60.6
RoBERTa-base ($BiKT_{FT}$)	81.1	91.7	**71.2**	78.1	**84.4**	68.0	91.3	94.3	94.8	**83.9**	52.1	47.9	**49.1**	95.5	**59.4**	**60.8**
$BiKT_{FT}$ vs. FT	-0.4	-2.3	**+14.5**	-0.9	**+6.2**	-3.5	-0.1	-0.3	-0.6	**+1.4**	-0.7	**+0.5**	**+0.8**	-0.2	**+3.6**	**+0.8**
$BiKT_{FT}$ vs. FT_{PT}	-0.3	-1.2	**+10.9**	**+6.6**	**+7.3**	-3.1	**+0.0**	-0.4	-0.6	**+2.2**	-0.6	-0.4	**+0.1**	-0.2	**+2.2**	**+0.2**
RoBERTa-base (PT)	67.5	70.8	54.8	58.4	61.0	62.3	86.6	88.0	94.5	71.6	41.7	39.4	40.4	90.8	56.5	53.8
RoBERTa-base (PT_{MT})	74.9	**91.1**	**73.0**	64.9	75.8	**70.0**	86.7	88.5	**94.8**	80.0	44.2	40.1	39.7	92.1	**58.3**	54.9
RoBERTa-base ($BiKT_{PT}$)	**75.6**	87.5	71.7	**71.2**	**83.6**	66.9	**87.9**	**89.9**	93.4	**80.8**	**46.2**	**41.3**	**42.4**	**93.2**	57.4	**56.1**
$BiKT_{PT}$ vs. PT	**+8.1**	**+16.7**	**+16.9**	**+12.8**	**+22.6**	**+4.6**	**+1.3**	**+1.9**	-1.1	**+9.2**	**+4.5**	**+1.9**	**+2.0**	**+2.4**	**+0.9**	**+2.3**
$BiKT_{PT}$ vs. PT_{MT}	**+0.7**	-3.6	-1.3	**+6.3**	**+7.8**	-3.1	**+1.2**	**+1.4**	-1.4	**+0.8**	**+2.0**	**+1.2**	**+2.7**	**+1.1**	-0.9	**+1.2**
Qwen1.5-1.8B (FT)	79.6	90.6	81.3	80.4	73.4	**72.3**	90.9	94.4	95.5	84.3	52.0	50.2	52.2	94.6	61.3	62.1
Qwen1.5-1.8B (FT_{PT})	74.1	93.8	76.6	78.0	74.7	71.3	90.1	94.2	95.2	83.1	51.7	51.0	**53.3**	94.5	62.6	62.6
Qwen1.5-1.8B ($BiKT_{FT}$)	**80.7**	**96.1**	**82.0**	**80.6**	**86.5**	71.6	**91.3**	**94.6**	**95.5**	**86.5**	**57.8**	**51.6**	52.1	**94.9**	**63.8**	**64.0**
$BiKT_{FT}$ vs. FT	**+1.1**	**+5.5**	**+0.7**	**+0.2**	**+13.1**	-0.7	**+0.4**	**+0.2**	**+0.0**	**+2.2**	**+5.8**	**+1.4**	-0.1	**+0.3**	**+2.5**	**+1.9**
$BiKT_{FT}$ vs. FT_{PT}	**+6.6**	**+2.3**	**+5.4**	**+2.6**	**+11.8**	**+0.3**	**+1.2**	**+0.4**	**+0.3**	**+3.4**	**+6.1**	**+0.6**	-1.2	**+0.4**	**+1.2**	**+1.4**
Qwen1.5-1.8B (PT)	74.8	67.2	57.8	80.4	75.7	62.5	89.3	**92.4**	96.0	77.4	48.5	46.0	48.8	94.7	52.1	58.0
Qwen1.5-1.8B (PT_{MT})	73.3	89.1	59.4	79.9	78.8	61.7	90.0	90.9	**96.2**	79.9	50.7	**46.4**	44.7	94.5	57.0	58.7
Qwen1.5-1.8B ($BiKT_{PT}$)	**75.2**	**92.2**	**65.6**	**82.5**	**87.5**	**67.8**	**90.6**	92.1	95.7	**83.2**	**58.7**	45.9	**50.7**	**94.9**	**59.6**	**62.0**
$BiKT_{PT}$ vs. PT	**+0.4**	**+25.0**	**+7.8**	**+2.1**	**+11.8**	**+5.3**	**+1.3**	-0.3	-0.3	**+5.8**	**+10.2**	-0.1	**+1.9**	**+0.2**	**+7.5**	**+4.0**
$BiKT_{PT}$ vs. PT_{MT}	**+1.9**	**+3.1**	**+6.2**	**+2.6**	**+8.7**	**+6.1**	**+0.6**	**+1.2**	-0.5	**+3.3**	**+8.0**	-0.5	**+6.0**	**+0.4**	**+2.6**	**+3.3**
Phi-1.5 (FT)	80.0	92.2	88.3	77.8	78.6	70.5	92.0	94.3	**95.0**	85.4	49.8	48.2	49.3	95.4	69.2	62.4
Phi-1.5 (FT_{PT})	78.7	94.6	89.4	77.6	77.9	71.3	89.3	94.1	94.5	85.3	52.6	**49.3**	49.9	**95.6**	69.4	63.4
Phi-1.5 ($BiKT_{FT}$)	**81.4**	**96.1**	**91.3**	**80.1**	**84.9**	**71.7**	**92.3**	**94.6**	94.4	**87.4**	**53.9**	48.9	**51.0**	95.3	**69.6**	**63.7**
$BiKT_{FT}$ vs. FT	**+1.4**	**+3.9**	**+3.0**	**+2.3**	**+6.3**	**+1.2**	**+0.3**	**+0.3**	-0.6	**+2.0**	**+4.1**	**+0.7**	**+1.7**	-0.1	**+0.4**	**+1.3**
$BiKT_{FT}$ vs. FT_{PT}	**+2.7**	**+1.5**	**+1.9**	**+2.5**	**+7.0**	**+0.4**	**+3.0**	**+0.5**	-0.1	**+2.1**	**+1.3**	-0.4	**+1.1**	-0.3	**+0.2**	**+0.3**
Phi-1.5 (PT)	69.1	67.2	54.7	77.0	66.9	60.8	88.0	86.0	93.8	73.7	45.5	41.6	**47.3**	93.4	**64.8**	58.5
Phi-1.5 (PT_{MT})	69.6	62.5	55.5	74.9	62.5	61.4	87.7	88.4	94.4	73.0	43.1	**42.3**	43.8	94.6	64.3	57.6
Phi-1.5 ($BiKT_{PT}$)	**75.3**	**85.9**	**80.5**	**78.9**	**80.9**	**66.6**	**90.1**	**89.4**	**94.6**	**82.5**	**50.9**	41.6	47.1	**95.0**	62.9	**59.5**
$BiKT_{PT}$ vs. PT	**+6.2**	**+18.7**	**+25.8**	**+1.9**	**+14.0**	**+5.8**	**+2.1**	**+3.4**	**+0.8**	**+8.8**	**+5.4**	**+0.0**	-0.2	**+1.6**	-1.9	**+1.0**
$BiKT_{PT}$ vs. PT_{MT}	**+5.7**	**+23.4**	**+25.0**	**+4.0**	**+18.4**	**+5.2**	**+2.4**	**+1.0**	**+0.2**	**+9.5**	**+7.8**	-0.7	**+3.3**	**+0.4**	-1.4	**+1.9**

Table 2: The full-shot performance of our BiKT and the baselines with BERT-base, RoBERTa-base, Phi-1.5, and Qwen1.5-1.8B in terms of accuracy (%) when adapting to different target tasks.

lines FT, FT_{PT}, the feedback algorithm of BiKT, and the adaptation algorithm of $BiKT_{FT}$, the learning rate is set to 2e-5, and the epoch number is set to 10. For the prompt tuning type of algorithms, including the baselines PT, PT_{MT}, and the adaptation algorithm of $BiKT_{PT}$, the learning rate is set to 2e-3, and the epoch number is set to 20. For LoRA, $(IA)^3$, BitFit and the BiKT version of them, the learning rate is set to 2e-4, and the epoch number is set to 20. The target modules for LoRA, $(IA)^3$, $BiKT_{LoRA}$ and $BiKT_{(IA)^3}$ include the query, key, value matrix, and the dense MLP in a transformer block. Specially, we use the CausalModel for training and inferring on Phi and Qwen. The parameters of Phi and Qwen are quantized to FP16.

5.2 Evaluation Results and Analysis

5.2.1 Full-Shot Performance

We first evaluate BiKT and the baselines using BERT-base, RoBERTa-base, Qwen1.5-1.8B, and Phi-1.5 with full training set for adaptation. Table 2 shows the adaptation performance of each target task in round 2 and round 3.

From Table 2, we can observe that for both the finetuning type and the prompt tuning type of algorithms, BiKT achieves the best performance for most of the target tasks and achieves the best on average. In particular, for different pretrained models, $BiKT_{FT}$ averagely outperforms FT by 1.4% – 3.0% in round 2 and by 0.4% – 1.9% in round 3; and $BiKT_{PT}$ averagely outperforms PT by 5.8% – 9.2% in round 2 and by 1.0% – 4.0% in round 3. We can also observe from Table 2 that compared with PT_{MT}, $BiKT_{PT}$ improves the adaptation performance in round 2 by 3.1%, 0.8%, 3.3%, and 9.5% using BERT-base, RoBERTa-base, Qwen1.5, and Phi-1.5, respectively; and improves the adaptation performance in round 3 by 0.7%, 1.2%, 3.3%, and 1.9%, respectively. These results demonstrate that the feedback design of BiKT in round 2 and

round 3 provides better initialization for adapting to target tasks.

By closely examining the finetuning type of algorithms, we can find that BiKT_{FT} outperforms FT by 1.4% – 3.0% in round 2, while FT_{PT} underperforms FT by 0.1% – 2.9% after adding task-specific prompt. This validates that the improvement of the adaptation algorithm in BiKT mainly comes from the optimization of the backbone network in the feedback phase over the source tasks rather than purely adding the randomly initialized prompt for each target task in the adaptation phase.

5.2.2 Few-Shot Performance

We then evaluate BiKT_{FT}, FT, BiKT_{PT}, and PT using Qwen1.5-1.8B and Phi-1.5 in K-shot learning settings, where K ranges from 16, 32, to 100. The results are shown in Table 3.

The first key observation from Table 3 is that BiKT_{FT} outperforms FT by 2.0% – 9.9% and 12.8% – 14.6% over Qwen1.5-1.8B and Phi-1.5 in round 2 under different few-shot settings, respectively; and improves the performance by 1.1% – 3.7% and 2.8% – 3.9% in round 3, respectively. The second key observation is that BiKT_{PT} outperforms PT by 7.4% – 18.5% and 3.3% – 18.0% in different rounds and different few-shot settings over Qwen1.5-1.8B and Phi-1.5, respectively. We can also find a performance drop of PT over Phi-1.5 due to overfitting, when K increases.

The results above reveal that the feedback algorithm of BiKT enables robust few-shot ability in the adaptation phase, whereas the original pretrained models even with higher pretraining levels cannot.

Model & Method	Round 2 Avg.			Round 3 Avg.		
	16-Shot	32-Shot	100-Shot	16-Shot	32-Shot	100-Shot
Qwen1.5-1.8B (FT)	62.7	72.3	77.2	45.5	47.5	51.2
Qwen1.5-1.8B (BiKT_{FT})	**72.6**	**74.7**	**79.2**	**49.2**	**50.0**	**52.3**
BiKT_{FT} vs. FT	**+9.9**	**+2.4**	**+2.0**	**+3.7**	**+2.5**	**+1.1**
Qwen1.5-1.8B (PT)	54.0	56.2	58.4	41.6	41.8	43.5
Qwen1.5-1.8B (BiKT_{PT})	**71.9**	**74.7**	**74.5**	**49.0**	**50.7**	**53.2**
BiKT_{PT} vs. PT	**+17.9**	**+18.5**	**+16.1**	**+7.4**	**+8.9**	**+9.7**
Phi-1.5 (FT)	58.7	60.7	66.9	44.2	44.4	47.3
Phi-1.5 (BiKT_{FT})	**72.5**	**75.3**	**79.7**	**47.9**	**47.2**	**51.2**
BiKT_{FT} vs. FT	**+13.8**	**+14.6**	**+12.8**	**+3.7**	**+2.8**	**+3.9**
Phi-1.5 (PT)	54.5	52.8	57.7	44.3	42.5	41.8
Phi-1.5 (BiKT_{PT})	**66.8**	**67.1**	**75.7**	**47.6**	**47.7**	**47.2**
BiKT_{PT} vs. PT	**+12.3**	**+14.3**	**+18.0**	**+3.3**	**+5.2**	**+5.4**

Table 3: The few-shot performance of our BiKT and the baselines with Qwen1.5-1.8B and Phi-1.5.

5.2.3 Ablation Study

Impact of Task-Specific Prompt. To verify the necessity of task-specific prompt in the feedback algorithm over the source tasks, we show the adaptation performance of BiKT_{PT} to each target task with and without task-specific prompt in the feedback phase. The prompts in the adaptation phase of BiKT_{PT} for the target tasks are randomly initialized. From the results in Table 4, we can see that introducing the task-specific prompt in the feedback phase improves the adaptation performance by 1.0% and 1.1% for BERT-base and Qwen1.5-1.8B, respectively. These results validate the functionality of task-specific prompt in separating the general knowledge from task-specific knowledge to enable more focused optimization of the backbone network in the feedback algorithm.

Model	Without Prompt	With Prompt	Δ
BERT-base	76.2	**77.2**	**+1.0**
Qwen1.5-1.8B	82.1	**83.2**	**+1.1**

Table 4: The adaptation performance of BiKT_{PT} with and without task-specific prompt in the feedback.

Impact of Task Order. To explore the impact of task order, we switch the downstream tasks in round 1 and those in round 2. In particular, we regard tasks in round 2 as source tasks for feedback and tasks in round 1 as target tasks for adaptation. We report the model adaptation performance using BERT-base and Qwen1.5-1.8B in the Table 5. We can observe that the average accuracy of BiKT_{PT} is still 2.0% and 7.2% higher than PT over BERT-base and Qwen1.5-1.8B respectively, which indicates that BiKT gains consistent improvements with different task order.

Model & Method	Avg.
BERT-base (PT)	74.0
BERT-base (BiKT_{PT})	**76.0**
BiKT_{PT} vs. PT	**+2.0**
Qwen1.5-1.8B (PT)	78.4
Qwen1.5-1.8B (BiKT_{PT})	**85.6**
BiKT_{PT} vs. PT	**+7.2**

Table 5: The performance of BiKT over BERT-base and Qwen1.5-1.8B by changing task order.

5.2.4 Extensibility with Tuning Methods

To validate the extensibility of the adaptation algorithm in BiKT, we replace the default prompt tuning module with LoRA, $(\text{IA})^3$, or BitFit. We take five tasks used in $(\text{IA})^3$ (Liu et al., 2022) as the target tasks to evaluate the adaptation performance in round 2. The results are shown in Table 6.

We can observe that BiKT with different tuning methods are better than applying these methods on

the original pretrained model. In particular, BiKT with LoRA outperforms LoRA by 8.0% and 4.5%, BiKT with $(IA)^3$ outperforms $(IA)^3$ by 8.0% and 3.7%, and BiKT with BitFit outperforms BitFit by 7.3% and 4.6% on average over BERT-base and Qwen1.5-1.8B, respectively. These results demonstrate that BiKT is compatible with different tuning methods and improves the adaptation performance by taking the optimized backbone network and the trained prompts from the previous feedback phase.

Model & Method	BoolQ	CB	COPA	RTE	WiC	Avg.
BERT-base (LoRA)	72.6	69.6	58.0	62.5	64.8	65.5
BERT-base ($\text{BiKT}_{\text{LoRA}}$)	**77.8**	**76.8**	**68.0**	**76.8**	**68.3**	**73.5**
$\text{BiKT}_{\text{LoRA}}$ vs. LoRA	**+5.2**	**+7.2**	**+10.0**	**+14.3**	**+3.5**	**+8.0**
BERT-base ($(IA)^3$)	71.2	69.6	56.5	63.9	64.7	65.4
BERT-base ($\text{BiKT}_{(IA)^3}$)	**77.4**	**76.8**	**69.0**	**75.7**	**66.9**	**73.2**
$\text{BiKT}_{(IA)^3}$ vs. $(IA)^3$	**+5.3**	**+7.2**	**+12.5**	**+11.8**	**+2.2**	**+8.0**
BERT-base (BitFit)	70.0	71.4	56.0	60.4	64.4	64.4
BERT-base ($\text{BiKT}_{\text{BitFit}}$)	**75.9**	**76.8**	**65.5**	**73.9**	**66.6**	**71.7**
$\text{BiKT}_{\text{BitFit}}$ vs. BitFit	**+5.9**	**+5.4**	**+9.5**	**+13.5**	**+2.2**	**+7.3**
Qwen1.5-1.8B (LoRA)	76.8	73.4	**78.1**	79.5	**68.4**	75.3
Qwen1.5-1.8B ($\text{BiKT}_{\text{LoRA}}$)	**79.0**	**90.6**	77.3	**86.1**	66.1	**79.8**
$\text{BiKT}_{\text{LoRA}}$ vs. LoRA	**+2.2**	**+17.2**	-0.8	**+6.6**	-2.3	**+4.5**
Qwen1.5-1.8B ($(IA)^3$)	71.9	92.2	74.2	78.8	**65.3**	76.5
Qwen1.5-1.8B ($\text{BiKT}_{(IA)^3}$)	**76.1**	**93.8**	**80.5**	**85.8**	64.8	**80.2**
$\text{BiKT}_{(IA)^3}$ vs. $(IA)^3$	**+4.2**	**+1.6**	**+6.3**	**+7.0**	-0.5	**+3.7**
Qwen1.5-1.8B (BitFit)	73.4	71.9	**77.3**	78.8	63.1	72.9
Qwen1.5-1.8B ($\text{BiKT}_{\text{BitFit}}$)	**76.7**	**89.1**	71.9	**85.8**	**64.1**	**77.5**
$\text{BiKT}_{\text{BitFit}}$ vs. BitFit	**+3.3**	**+17.2**	-5.4	**+7.0**	**+1.0**	**+4.6**

Table 6: The improvement of the full-shot adaptation performance over five tasks used in $(IA)^3$ with BERT-base, Qwen1.5-1.8B and different tuning methods.

5.2.5 Extended Study

Efficient Feedback As mentioned in Section 4.2, the time complexity of T-round feedback is $O(Tn)$. To improve efficiency, we let the feedback algorithm of BiKT run in an incremental way, which we call BiKT^{Inc}. Similarly to BiKT_{PT}, we also define $\text{BiKT}^{\text{Inc}}_{\text{PT}}$. More specifically, the feedback in round t runs only on the newly introduced datasets in round $t-1$, rather than the datasets of all the previous rounds. As a result, the dataset of each downstream task will be traversed only once, and the overall complexity is reduced to $O\left(\sum_{t=1}^{T} 1 \times n_t\right) = O(n)$, which is independent of T and is more efficient. We also evaluate the model performance with leveraging the datasets of all the previous rounds for feedback and with leveraging the datasets of the previous one round. In particular, we compare the feedback effect in round 3 with leveraging the datasets of round 2 (i.e., BiKT^{Inc}) and that with leveraging the datasets of both round 1 and round 2 (i.e., BiKT). Table 7 shows the model adaptation performance for each downstream task in round 3 using BERT-base. We can observe that $\text{BiKT}^{\text{Inc}}_{\text{PT}}$ is averagely 1.3% less accurate than BiKT_{PT} for feedback, but still outperforms the baseline PT with no feedback by 0.54% on average.

Model & Method	ANLI-R1	ANLI-R2	ANLI-R3	SciTail	WinoGrande	Avg.
BERT-base (PT)	**38.6**	**39.9**	40.4	86.7	52.1	51.54
BERT-base (BiKT_{PT})	38.4	38.9	**42.7**	**92.7**	**54.2**	**53.38**
BERT-base ($\text{BiKT}^{\text{Inc}}_{\text{PT}}$)	38.3	38.7	41.7	89.0	52.7	52.08

Table 7: The comparison between the feedback effect with leveraging the datasets of round 2 (i.e., BiKT^{Inc}) and that with leveraging the datasets of both round 1 and round 2 (i.e., BiKT).

6 Conclusion

In this work, we have studied the relationship between pretrained models and downstream tasks. We first have modeled the sequential nature of downstream tasks and have proposed the new framework of bidirectional knowledge transfer, called BiKT, including the two phases of adaptation and feedback. We have designed a multi-task feedback learning algorithm with trainable task-specific prompts as well as a model adaptation algorithm with feedback-based initialization. Evaluation results over several datasets using different models with varying sizes have demonstrated the effectiveness of BiKT in both full-shot and few-shot settings as well as the remarkable advantage over conventional finetuning and prompt tuning methods.

Limitations

We regard soft prompts as the descriptions of tasks to separate task-specific knowledge and task-shared knowledge from downstream tasks, which is, however, just a hypothesis without theoretical proof. We have provided detailed discussion about this hypothesis in Appendix B.

Acknowledgments

This work was supported in part by National Key R&D Program of China (No. 2022ZD0119100), China NSF grant No. 62025204, No. 62202296, and No. 62272293, Alibaba Innovation Research (AIR) Program, and SJTU-Huawei Explore X Gift Fund. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government.

References

Eneko Agirre, Llu'is M'arquez, and Richard Wicentowski, editors. 2007. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. Association for Computational Linguistics, Prague, Czech Republic.

Akari Asai, Mohammadreza Salehi, Matthew E. Peters, and Hannaneh Hajishirzi. 2022. ATTEMPT: parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 6655–6672. Association for Computational Linguistics.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *CoRR*, abs/2309.16609.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Andrea Cossu, Tinne Tuytelaars, Antonio Carta, Lucia C. Passaro, Vincenzo Lomonaco, and Davide Bacciu. 2022. Continual pre-training mitigates forgetting in language and vision. *CoRR*, abs/2205.09357.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Suchin Gururangan, Ana Marasovic, 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, ACL 2020, Online, July 5-10, 2020*, pages 8342–8360. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. 2023. Continual pre-training of language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5189–5197. AAAI Press.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing,*

EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 3045–3059. Association for Computational Linguistics.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need II: phi-1.5 technical report. *CoRR*, abs/2309.05463.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Huihui Liu, Yiding Yang, and Xinchao Wang. 2021. Overcoming catastrophic forgetting in graph neural networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 8653–8661. AAAI Press.

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.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. 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 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

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*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5533–5542. IEEE Computer Society.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A continual pre-training framework for language understanding. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8968–8975. AAAI Press.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. Spot: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5039–5059. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. 2023a. A comprehensive survey of continual learning: Theory, method and application. *CoRR*, abs/2302.00487.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogério Feris, Huan Sun, and Yoon Kim. 2023b. Multitask prompt tuning enables parameter-efficient transfer learning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. 2022. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 139–149. IEEE.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Adina Williams, Nikita Nangia, and Samuel 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, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1–9. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. 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 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

A Time Complexity Analysis

Time Complexity of Adaptation. For adaptation to the dataset of each downstream task, BiKT leverages the latest backbone network and the trained prompts of the source tasks from the feedback phase for good initialization. Thus, we do not introduce extra overhead compared with conventional adaptation methods, which are based on the initial pretrained model.

Time Complexity of Feedback. For the feedback phase in round t, the trainable parameters include only the backbone network and the prompts for the downstream tasks in round $t-1$. Thus, the size of trainable parameters does not increase with the number of rounds and the feedback phase is scalable from model. Regarding the datasets, BiKT leverages the datasets of all the previous rounds for feedback to achieve better model performance. For T rounds in total, n_t downstream tasks in round t, and $\sum_{t=1}^{T} n_t = n$ downstream tasks in total, the dataset of each downstream task in round t will be traversed by the model for the later $T-t$ rounds. Hence, the overall complexity is $\sum_{t=1}^{T}(T-t)n_t \leq T\sum_{t=1}^{T} n_t = O(Tn)$. To improve efficiency, we also introduce BiKT$^{\text{Inc}}$ in Section 5.2.5 and reduce the time complexity to $O(n)$, which is independent of T.

B Explanation for Separation of Knowledge

We interpret the separation of task-specific knowledge and task-shared knowledge from the back-propagation perspective. The backbone network is shared among tasks, while prompts are independent among different tasks. Thus, during back-propagation, prompts are only affected by the gradient from the corresponding task's data, while the task-shared backbone is affected by the gradient generated by all the tasks' data. Formally, we have $\Phi_i = \Phi_i + \Delta\Phi_i$, $\Theta = \Theta + \sum_i \Delta\Theta_i$ where Φ_i denotes the prompts of the i-th source task, $\Delta\Phi_i$ is the gradient for the prompt of the i-th source task, Θ denotes the backbone network, and $\Delta\Theta_i$ is the gradient for backbone network using the data of the i-th source task. We can observe that the backbone network is optimized over the mixed distribution of all the tasks' data in the direction of the aggregate gradient, while the prompt is optimized over a specific task's data distribution.

In addition, the ablation study of the impact of task-specific prompts validates the functionality of task-specific prompts in separating the general knowledge from task-specific knowledge to enable more focused optimization of the backbone network in the feedback algorithm.

C Dataset Details

We conduct our experiments on 23 datasets and the detailed task types and data distributions of all tasks are listed in Table 8. 9 datasets as target tasks for round 1 and as source tasks for round 2 and round 3 are from GLUE benchmark, which is a wide-ranging collection of natural language understanding tasks:

- **CoLA** (Warstadt et al., 2019) is a linguistic acceptability analysis task. The input is one short sentence. The labels are {*ungrammatical*, *grammatical*} in language and {0, 1} for discriminative model.
- **MNLI** (Williams et al., 2018) is a natural language inference task. The inputs are two sentences. The labels are {*neutral*, *contradiction*, *entailment*} in language and {0, 1, 2} for discriminative model.
- **MRPC** (Dolan and Brockett, 2005) is a paraphrase detection task. The inputs are two sentences from news. The labels are {*not equivalent*, *equivalent*} in language and {0, 1} for discriminative model.
- **QNLI** is a question-answer inference task. The input are one question and one answer sentence from SQuAD v1.0 (Rajpurkar et al., 2016). The labels are {*not equivalent*, *equivalent*} in language and {0, 1} for discriminative model.
- **QQP** is a question equivalent analysis task. The inputs are two questions. The labels are {*not duplicates*, *duplicates*} in language and {0, 1} for discriminative model.
- **RTE** is a textual entailment analysis task. The input are two sentences. The labels are {*not equivalent*, *equivalent*} in language and {0, 1} for discriminative model.
- **SST-2** (Socher et al., 2013) is a sentiment analysis task. The input is one sentence from movie reviews. The labels are {*negative*, *positive*} in language and {0, 1} for discriminative model.

Round 1 Task	Task Type	Text Corpora
CoLA	Linguistic Acceptability Analysis Task	NLP books and journals
MNLI	Natural Language Inference Task	Speech, fiction, and government reports
MRPC	Paraphrase Detection Task	Online news
QNLI	Question-Answer Inference Task	Questions and paragraphs of Wikipedia
QQP	Question Equivalent Analysis Task	QA pair from Quora, a community question and answer site
RTE	Textual Entailment Analysis Task	News and Wikipedia
SST-2	Sentiment Analysis Task	Movie comment
STS-B	Textual Similarity Analysis Task	Headlines of news, video and image
WNLI	Natural Language Inference Task	Mannually created competition data
Round 2 Task	**Task Type**	**Text Corpora**
BoolQ	Question-Answering Task	Google search engine and Wikipedia page
CB	Textual Entailment Task	News articles, fiction and dialogue
COPA	Causal Inference Task	Personal stories written in Internet weblogs
MultiRC	Question-Answering Task	News
RTE	Textual Entailment Analysis Task	News and Wikipedia page
WiC	Word Sense Disambiguation Task	English lexicographic resource, verb-based resource and Wiktionary
SNLI	Natural Language Inference Task	Human-written English sentence pairs
PAWS	Paraphrase Identification Task	Wikipedia pages
IMDB	Sentiment Classification Task	Movie comment
Round 3 Task	**Task Type**	**Text Corpora**
ANLI-R1	Natural Language Inference Task	Wikipedia
ANLI-R2	Natural Language Inference Task	Wikipedia
ANLI-R3	Natural Language Inference Task	Wikipedia, news, fiction, manually annotated sub-corpus, and WikiHow
SciTail	Textual Entailment Task	Text corpus of web sentences
WinoGrande	Commonsense Reasoning Task	WikiHow

Table 8: Detailed task types and data distributions of all tasks in our experiments.

- **STS-B** (Agirre et al., 2007) is a textual similarity analysis task. The inputs are two sentences extracted from news headlines, video captions, image captions, and natural language inference data. The label is a number ranging from 0 to 5 and number in string form for generative models.

- **WNLI** is a natural language inference task. The input are two sentences. The labels are {*not entailment*, *entailment*} in language and {0, 1} for discriminative model. Although some work excluded WNLI due to its adversarial training and validation splits, we still use it as source task to prove the stability of our methods.

6 datasets as target tasks for round 2 and as source tasks for round 3 are from SuperGLUE benchmark, which is a more difficult benchmark for natural language processing:

- **BoolQ** (Clark et al., 2019) is a question-answering task. The inputs are one question and one passage. The labels are {*no*, *yes*} in language and {0, 1} for discriminative model.

- **CB** (De Marneffe et al., 2019) is a textual entailment task. The inputs are one premise and one hypothesis. The labels are {*neutral*, *contradiction*, *entailment*} in language and {0, 1, 2} for discriminative model.

- **COPA** (Roemmele et al., 2011) is a causal inference task. The inputs are one question subject, one premise and two choices. The labels are {*choice1*, *choice2*} in language. As for discriminative model, we convert one sample into two data and each data contains only one choice. If the choice is correct, the label is 1; otherwise, it is 0.

- **MultiRC** (Khashabi et al., 2018) is a question-answering task. The inputs are one paragraph, one question and one answer. The labels are {*false*, *true*} in language and {0, 1} for discriminative model.

- **RTE** is a textual entailment analysis task. The input are two sentences. The labels are {*not entailment*, *entailment*} in language and {0, 1} for discriminative model.

- **WiC** (Pilehvar and Camacho-Collados, 2019) is word sense disambiguation task. The input are two sentences. The labels are {*false*,

K-Shot	Model & Method	Round 2										Round 3					
		BoolQ	CB	COPA	MRC	RTE	WiC	SNLI	PAWS	IMDB	Avg.	ANLI-R1	ANLI-R2	ANLI-R3	SciTail	WinoGrande	Avg.
16	Qwen1.5-1.8B (FT)	**62.9**	65.6	**68.0**	51.9	56.2	**53.0**	53.3	59.1	**94.3**	62.7	33.2	36.3	35.5	72.1	**50.2**	45.5
	Qwen1.5-1.8B (BiKT$_{\text{FT}}$)	60.8	**82.8**	66.4	**54.7**	**83.1**	52.7	**86.9**	**72.4**	93.7	**72.6**	**45.5**	**38.6**	**38.0**	**74.6**	49.5	**49.2**
32	Qwen1.5-1.8B (FT)	**66.8**	**81.2**	**70.3**	66.8	71.6	**55.6**	75.7	68.2	**94.2**	72.3	37.7	**39.6**	35.2	73.4	51.5	47.5
	Qwen1.5-1.8B (BiKT$_{\text{FT}}$)	64.9	**81.2**	64.1	**66.9**	**85.6**	54.2	**87.9**	**73.6**	93.8	**74.7**	**46.2**	36.0	**37.3**	**78.3**	**52.3**	**50.0**
100	Qwen1.5-1.8B (FT)	**64.6**	**95.3**	**79.7**	70.0	75.3	**59.2**	80.1	76.2	**94.7**	77.2	42.0	33.6	**41.0**	88.1	**51.1**	51.2
	Qwen1.5-1.8B (BiKT$_{\text{FT}}$)	63.1	89.1	77.3	**74.1**	**85.6**	58.8	**88.0**	**82.8**	93.9	**79.2**	**48.9**	**34.8**	39.6	**88.7**	49.7	**52.3**
16	Qwen1.5-1.8B (PT)	62.4	51.6	**56.2**	57.2	52.2	50.2	35.8	53.3	66.8	54.0	33.1	34.8	33.6	56.1	**50.4**	41.6
	Qwen1.5-1.8B (BiKT$_{\text{PT}}$)	**62.7**	**79.7**	50.0	**73.2**	**80.0**	**53.3**	**85.5**	**70.5**	**92.4**	**71.9**	**41.2**	**35.3**	**39.2**	**79.8**	49.6	**49.0**
32	Qwen1.5-1.8B (PT)	62.4	50.0	54.7	56.4	57.2	53.9	36.0	54.4	80.5	56.2	35.7	**34.6**	33.2	53.3	**52.4**	41.8
	Qwen1.5-1.8B (BiKT$_{\text{PT}}$)	**62.7**	**82.8**	**63.3**	**73.3**	**80.3**	**56.2**	**86.4**	**74.8**	**92.3**	**74.7**	**46.6**	34.0	**35.9**	**86.2**	50.7	**50.7**
100	Qwen1.5-1.8B (PT)	62.4	59.4	57.8	55.9	55.9	54.1	37.9	55.7	86.8	58.4	37.6	33.5	33.8	60.3	**52.3**	43.5
	Qwen1.5-1.8B (BiKT$_{\text{PT}}$)	**62.6**	**81.2**	**67.2**	**76.1**	**84.4**	**59.2**	**86.3**	**59.8**	**93.6**	**74.5**	**49.7**	**34.9**	**39.4**	**90.0**	52.0	**53.2**
16	Phi-1.5 (FT)	61.9	58.9	58.7	58.5	54.6	50.0	44.7	52.1	88.7	58.7	35.1	34.1	34.6	62.0	**55.2**	44.2
	Phi-1.5 (BiKT$_{\text{FT}}$)	**63.3**	**80.4**	**68.3**	**72.2**	**73.6**	**55.6**	**81.2**	**65.1**	**92.6**	**72.5**	**36.0**	**34.6**	**36.9**	**79.5**	52.5	**47.9**
32	Phi-1.5 (FT)	54.3	66.1	61.5	59.4	54.6	51.2	56.5	55.6	87.1	60.7	33.8	**35.6**	31.1	68.6	**52.8**	44.4
	Phi-1.5 (BiKT$_{\text{FT}}$)	**66.8**	**85.7**	**76.9**	**68.2**	**75.7**	**54.7**	**84.7**	**72.7**	**92.5**	**75.3**	**43.9**	**35.6**	**31.4**	**75.3**	49.7	**47.2**
100	Phi-1.5 (FT)	62.1	76.8	79.8	60.5	57.5	55.9	64.5	55.6	89.6	66.9	37.0	35.4	36.0	73.0	**54.9**	47.3
	Phi-1.5 (BiKT$_{\text{FT}}$)	**66.0**	**92.9**	**84.6**	**73.5**	**80.7**	**59.8**	**87.0**	**79.4**	**93.1**	**79.7**	**41.3**	**36.6**	**37.9**	**86.4**	54.0	**51.2**
16	Phi-1.5 (PT)	59.5	51.6	**54.7**	53.6	55.0	50.6	37.8	54.6	72.7	54.5	34.8	**36.0**	35.7	63.8	**51.4**	44.3
	Phi-1.5 (BiKT$_{\text{PT}}$)	**61.5**	**73.4**	**54.7**	**54.4**	**71.2**	**55.9**	**85.0**	**56.9**	**88.5**	**66.8**	**35.9**	32.7	**37.0**	**81.5**	50.5	**47.6**
32	Phi-1.5 (PT)	49.0	39.1	48.4	55.5	**58.1**	**52.0**	37.0	55.4	80.3	52.8	35.0	**35.6**	**33.5**	56.5	**52.2**	42.5
	Phi-1.5 (BiKT$_{\text{PT}}$)	**61.7**	**68.8**	**68.8**	**71.7**	56.9	51.2	**77.5**	**55.9**	**91.7**	**67.1**	**35.7**	35.1	33.1	**83.9**	50.4	**47.7**
100	Phi-1.5 (PT)	46.9	67.2	57.0	50.1	59.4	53.8	44.3	55.7	84.7	57.7	35.4	33.9	35.0	54.9	49.7	41.8
	Phi-1.5 (BiKT$_{\text{PT}}$)	**64.1**	**93.8**	**75.8**	**68.8**	**79.1**	**54.8**	**83.5**	**68.8**	**92.3**	**75.7**	**39.4**	**34.8**	**36.7**	**73.1**	**51.9**	**47.2**

Table 9: Detailed results of Table 3. All results are based on prompt tuning with 20 prompt vectors. Under {16, 32, 100}-shot settings, BiKT$_{\text{PT}}$ outperforms PT over Qwen1.5-1.8B and Phi-1.5.

true} in language and {0, 1} for discriminative model.

3 datasets as target tasks for round 2 and as source tasks for round 3 are publicly available:

- **SNLI** (Bowman et al., 2015) is a natural language inference task. The inputs are two sentences. The labels are {*neutral*, *contradiction*, *entailment*} in language and {0, 1, 2} for discriminative model.
- **PAWS-Wiki** (Zhang et al., 2019) is a paraphrase identification task. The inputs are two sentences from Wikipedia pages. The labels are {*not entailment*, *entailment*} in language and {0, 1} for discriminative model.
- **IMDB** (Maas et al., 2011) is a sentiment classification task. The input is one sentence from movie reviews. The labels are {*negative*, *positive*} in language and {0, 1} for discriminative model.

5 datasets as target tasks for round 3 are also publicly available:

- **ANLI-R1** (Nie et al., 2020) is the first round of ANLI, which is a natural language inference task. We use the test part as validation dataset. The inputs are one premise and one hypothesis. The labels are {*n*, *c*, *e*} in language and {0, 1, 2} for discriminative model.
- **ANLI-R2** is the second round of ANLI. We use the test part as validation dataset. The format of inputs and labels is same with ANLI-R1.
- **ANLI-R3** is the third round of ANLI. We use the test part as validation dataset. The format of inputs and labels is same with ANLI-R1.
- **SciTail** (Khot et al., 2018) is a textual entailment task. The inputs are two sentences. The labels are {*neutral*, *entails*} in language and {0, 1} for discriminative model.
- **WinoGrande** (Sakaguchi et al., 2020) is a commonsense reasoning task. The inputs are one sentence to be filled in and two options. We format options into one sentence with its index. The labels are {*option1*, *option2*} in language and {0, 1} for discriminative model.

D Detailed Evaluation Results

Table 9 shows details of the results under few-shot learning settings. We can observe that under all few-shot settings, BiKT$_{\text{PT}}$ outperforms PT and BiKT$_{\text{FT}}$ outperforms FT. Table 10 shows the full results of ablation experiments. The results reveal that adding task-specific prompts for feedback has a positive effect and can incorporate task features into the model. Table 11 shows the full results of PT and BiKT$_{\text{PT}}$ on BERT-base with reverse order

of tasks. Although the tasks in round 1 and tasks in round 2 are switched, BiKT$_{\text{PT}}$ still outperform PT.

Model	Method	BoolQ	CB	COPA	MRC	RTE	WiC	SNLI	PAWS	IMDB	Avg.
BERT-base	without prompt	71.6	82.7	62.8	63.9	**77.0**	64.2	**86.5**	**85.2**	**92.2**	76.2
	with prompt	**72.9**	**85.1**	**66.5**	**65.2**	76.3	**65.2**	86.3	85.1	**92.2**	**77.2**
Qwen1.5-1.8B	without prompt	76.8	85.9	60.9	82.9	86.9	66.9	**90.8**	91.5	**95.9**	82.1
	with prompt	**75.2**	**92.2**	**65.6**	**82.5**	**87.5**	**67.8**	90.6	**92.1**	95.7	**83.2**

Table 10: Detailed results of Table 4. The adaptation results are based on prompt tuning with 20 prompt vectors. Adding prompts has positive influence on the feedback algorithm.

Model & Method	COLA	MNLI	MRPC	QNLI	RTE	STS-B	QQP	SST-2	Avg.
BERT-base (PT)	**44.1**	73.0	72.8	**85.4**	61.4	81.3	**83.6**	90.0	74.0
BERT-base (BiKT$_{\text{PT}}$)	43.2	**74.6**	**81.9**	84.7	**65.3**	**85.0**	82.9	**90.0**	**76.0**
BiKT$_{\text{PT}}$ vs. PT	-0.9	**+1.6**	**+9.1**	-0.7	**+3.9**	**+3.7**	-0.7	**+0.0**	**+2.0**
Qwen1.5-1.8B (PT)	72.5	84.9	78.1	80.3	60.4	72.9	84.6	93.2	78.4
Qwen1.5-1.8B (BiKT$_{\text{PT}}$)	**79.1**	**85.9**	**80.5**	**90.5**	**84.4**	**82.6**	**86.2**	**95.4**	**85.6**
BiKT$_{\text{PT}}$ vs. PT	**+6.6**	**+1.0**	**+2.4**	**+10.2**	**+24.0**	**+9.7**	**+1.6**	**+2.2**	**+7.2**

Table 11: Detailed results of Table 5. The performance of BiKT over BERT-base and Qwen1.5-1.8B by changing task order.