P-Distill: Efficient and Effective Prompt Tuning using Knowledge Distillation

Anonymous ACL submission

Abstract

In the field of natural language processing (NLP), prompt-based learning is widely used for efficient parameter learning. However, this method has the drawback of shortening the input length by the extent of the attached prompt, leading to an inefficiency in utilizing the input space. In this study, we propose P-Distill, a novel approach that mitigates this limitation by utilizing knowledge distillation from a teacher model with extensive prompts to a student model with shorter prompts. We introduce two novel methods for prompt compression, including prompt initialization, and prompt distillation. Experiments across various NLP tasks demonstrate that P-Distill exhibits comparable or superior performance compared to other state-of-the-art prompt-based learning methods, even with significantly shorter prompts. We achieve a peak improvement of 1.90% even with the prompt lengths compressed to one-eighth. An additional study further provides insights into the distinct impact of each method on the overall performance of P-Distill. These results highlight the potential of P-Distill in facilitating efficient and effective training for a wide range of NLP models.

1 Introduction

Pre-trained language models (PLMs) have been effective in improving performances of various natural language processing (NLP) tasks (Devlin et al., 2019; Brown et al., 2020; Touvron et al., 2023). These models are fine-tuned by optimizing all parameters to enhance the performances of specific downstream tasks; however, fine-tuning requires significant computational resources while training. The need for significant computational resources for storage and training becomes a challenge, especially when fine-tuning large language models such as Llama2 (Touvron et al., 2023), which may not be readily available to most users.

To reduce computational costs, researchers have explored various methods for efficiently fine-tuning the parameters (Houlsby et al., 2019; Hu et al., 2021; Liu et al., 2022). In contrast to the traditional model fine-tuning that updates all parameters for a downstream task, P-tuning (Liu et al., 2022) fixes the pre-trained parameters and only trains the continuous prompts. These prompts are trainable embeddings attached at the beginning or throughout each layer of the model. P-tuning is computationally efficient, particularly for PLMs with a large number of parameters; however, it overlooks the inefficiency in input space utilization arising from attaching continuous prompts (Hu et al., 2021). Similar to the findings in the work (Liu et al., 2022), more challenging tasks require longer prompt lengths to achieve the optimal performance, as shown in Figure 1. For instance, in datasets such as CoNLL04, CoNLL05 WSJ, SQuAD 2.0, and OntoNotes 5.0, a prompt length of 128 is required to achieve the optimal performance.

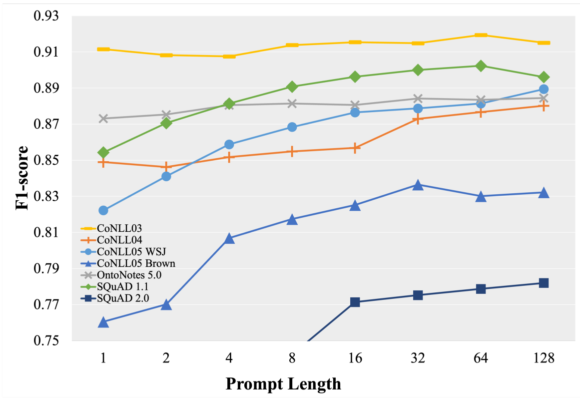

Figure 1: Performance variation in P-tuning across tasks based on the length of continuous prompts.

In this paper, we propose P-Distill, which is a novel prompt compression method to address the limitations of long prompts. Our method involves a two-step process where we first train a teacher model using P-tuning to achieve optimal performance with long prompts. We then transfer this knowledge to a student model with significantly shorter prompts through a distillation process. To ensure stability in training, we first perform prompt initialization based on the teacher model prompts. Then, we focus on distilling knowledge between the teacher and student models, specifically targeting the outputs of their intermediate and prediction layers. This is due to the impact of soft prompts on the hidden states within these layers, which subsequently influences the model's predictions. This method enables the compression of prompts to shorter lengths without a significant degradation in performance, thereby addressing the inefficiencies inherent in longer prompts.

To validate its effectiveness and efficiency, we evaluate P-Distill using various NLP benchmarks, including SuperGLUE (Wang et al., 2020), SQuAD (Rajpurkar et al., 2016, 2018), ReCoRD (Zhang et al., 2018), OntoNotes (Weischedel et al., 2013), and CoNLL (Tjong Kim Sang and De Meulder, 2003; Carreras and Màrquez, 2004, 2005; Pradhan et al., 2012a). Our results demonstrate that P-Distill exhibits comparable or superior performance than those of the existing state-of-the-art prompt-based models. To the best of our knowledge, this study is the first to train teacher prompts and transfer their knowledge to student prompts for the purpose of compressing prompts. The main contributions of this study are summarized as follows:

- We propose a method called P-Distill to compress the soft prompts, effectively mitigating the limitation of reducing the model's usable sequence length in prompt-based learning.
- We introduce a prompt distillation method utilizing teacher model's hidden state and prediction outputs, influenced by soft prompts, and propose a prompt initialization for stable prompt distillation.
- We validate P-Distill across multiple NLP benchmarks, demonstrating its ability to maintain or enhance accuracy while reducing prompt lengths by up to eight times.

The remainder of this paper is structured as follows: Section 2 provides the preliminaries; Section 3 describes a detailed description of the proposed method; Section 4 presents the experimental results and analysis, and Section 5 concludes the study.

2 Preliminaries

2.1 Pre-trained Language Models Based on the Transformer

The transformer model (Vaswani et al., 2023), comprising an encoder and decoder, is the fundamental architecture of the majority of recent PLMs, including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and GPT-3 (Brown et al., 2020). Each encoder and decoder consists of multiple transformer layers and incorporates key components, such as multi-head attention modules (MHA), feed-forward networks, layer normalization, and residual connections. A key component of this architecture is the multi-head attention mechanism, which computes attention weights using query (Q), key (K), and value (V) matrices. Mathematically, the attention function in multi-head attention can be represented as follows:

$$Att(x) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \tag{1}$$

where $\sqrt{d_k}$ is the scaling factor for gradient stabilization during training. This attention mechanism is crucial in understanding language and generating tasks by modulating the focus of the model on different parts of the input data.

2.2 Prompt-based Learning Methods

Prompt-based learning methods have emerged as an efficient alternative to full model fine-tuning, especially for PLMs (Liu et al., 2022, 2023). These methods use prompts to guide the model predictions for specific tasks. Several approaches (Jiang et al., 2020; Shin et al., 2020) employ discrete prompts, which are fixed templates added to the input. For example, in sentiment analysis, a template might be "This text *[Input Text]* expresses a *[MASK]* sentiment.". However, discrete prompts are limited in that their performances significantly depend on template selection. Advanced approaches, such as Prefix-Tuning (Li and Liang, 2021) and P-tuning (Liu et al., 2022), use continuous prompts that are trainable embeddings independent of the model vocabulary. Particularly, P-tuning attaches continuous prompts to each layer of the model, thereby influencing its behavior and enhancing its performance in downstream tasks. This approach is mathematically represented as:

$$T = \{h_{0:i}, e(x), h_{i+1:m}, e(y)\}, \tag{2}$$

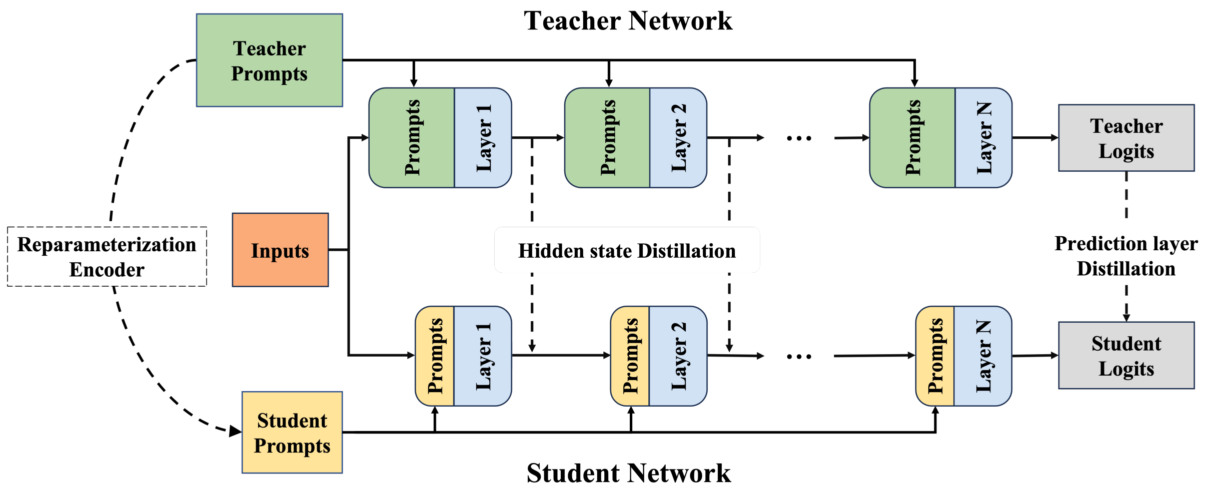

Figure 2: Overview of the proposed method, denoted as P-Distill. This method trains a teacher model to generate concise and effective prompts, followed by distilling the knowledge into a student model.

where h_i denotes the trainable embedding vector of the continuous prompts. These continuous prompts are integrated into the attention mechanism of the transformer model as follows:

$$head_l(x) = Att(xW^{(l)}, [P_k^{(l)} : K^{(l)}], [P_v^{(l)} : V^{(l)}]), \tag{3}$$

where $head_l(x)$ is the attention computation for each attention head l. The query vector $Q^{(l)}$ is generated using the input x and the weight matrix $W^{(l)}$. $K^{(l)}$ and $V^{(l)}$ are the key and value vectors for the l-th attention head, and $P_k^{(l)}$ and $P_v^{(l)}$ are the continuous prompts added to the key and value vectors of the l-th attention head, respectively. This integration enables the model to influence layers closer to the output, significantly affecting the final predictions.

2.3 Knowledge Distillation

In artificial intelligence, knowledge distillation (KD) is a technique for reducing the size of large models while preserving their performances (Jiao et al., 2020; Sun et al., 2020; Sanh et al., 2020; Hinton et al., 2015). During KD, a smaller student model is trained to internalize and emulate the complex decision-making patterns and behaviors of a larger teacher model. This process involves the behavior functions of the models, f^T and f^S, transforming inputs into informative representations, typically defined as the output of any layer within the model. These representations contain abundant information for model predictions. KD is quantified using loss functions, such as the Kullback-Leibler divergence (Kullback and Leibler, 1951) or Mean Squared Error (MSE) (Hinton et al., 2015), as follows:

$$L_{KD} = \sum_{x \in X} L(f^S(x), f^T(x)), \tag{4}$$

where x is the input, and X and L denote the dataset and the loss function, respectively. This approach enables the student model to gain a comprehensive understanding of various classes, enhancing its application in fields such as NLP.

3 Methodology

Many existing prompt tuning methods, including P-tuning, have the drawback of occupying an unnecessarily large portion of the input token space owing to their long prompts. Inspired by knowledge distillation methods, we propose a novel prompt compression methodology called P-Distill. This approach aims to compress the prompts while maintaining the performance, thereby increasing the available space for input tokens and enhancing the overall model efficiency. To this end, the proposed P-Distill comprises the following two methods: prompt initialization and prompt distillation. Figure 2 shows the learning and compression processes of P-Distill. Our approach involves two main steps where the first step is training a teacher model using P-tuning, and the second step focuses on distilling knowledge to a student model with shorter prompts, effectively reducing the length of the prompts.

3.1 Prompt-Based Teacher Learning

When solving downstream tasks using P-tuning, the performance is only influenced by the continuous prompts, because the pre-trained weights of the language model remain fixed. Furthermore, the optimal prompt length varies with task complexity. For simple sentence classification tasks, the optimal length is approximately 20, whereas for more difficult sequence-labeling tasks, it can extend up to 100 (Liu et al., 2022). Understanding the variation in prompt length is crucial, as longer prompts inherently limit the maximum sequence length that the model can handle.

We train a teacher model on various tasks using the P-tuning methodology, which fixes the pre-trained weights of the PLM. This model tokenizes input data x and embeds it into text embeddings $\overline{x}$. Subsequently, the continuous prompts P_k^T, $P_v^T \in R^{n_t \times d}$ of the teacher model are randomly initialized and concatenated with the key vectors $K \in R^{n_x \times d}$ and value vectors $V \in R^{n_x \times d}$ of each layer. Here, d is the dimensionality of the hidden representations, n_t is the prompt length of the teacher model, and n_x is the length of token embeddings. The teacher model, which utilizes attention heads incorporating continuous prompts, is trained to take the text embedding $\overline{x}$ as input and generate the final logits y^T. The parameter optimization of the teacher model is guided by the cross-entropy loss, which is formalized as follows:

$$L_{CE}^T = -\frac{1}{|B|}\sum_{i=1}^{|B|}\log(softmax(y_i^T)[c_i]), \quad (5)$$

where $|B|$ is the number of data points in the current batch, y_i^T is the logits output by the teacher model for the i-th data point in the batch, $softmax(y_i^T)$ is the softmax-transformed probability distribution over the classes, and c_i is the true class index for the i-th data point.

3.2 Prompt-enhanced Distillation (P-Distill)

We initiate the training of a student model which employs shorter continuous prompts, rather than the teacher model, using the same prompt attachment methodology. During the initial training phase, we initialize the continuous prompts of the student model P_k^S and $P_v^S \in R^{n_s \times d}$ based on the teacher model's prompts P_k^T and P_v^T. Subsequently, student prompts are also attached to the key and value vectors across all layers to compute the attention heads. The length of the student model

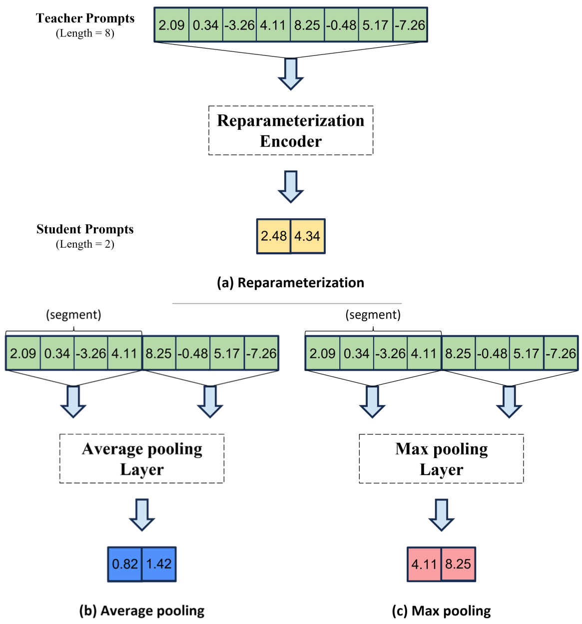

Figure 3: Illustration of various prompt initialization methods.

prompts, represented by n_s, is shorter than that of the teacher model prompts n_t. The student model, denoted by f^S, takes the text embedding $\overline{x}$ as input and generates the output logits y^S. The teacher and student models share the same underlying language model architecture, differing only in the length and content of their respective prompts. In this context, we focus on distilling the knowledge from the more extensive teacher model prompts into the shorter student model prompts. To enhance the effectiveness of knowledge transfer, we propose two novel methods for knowledge distillation.

3.2.1 Prompt Initialization

For solving downstream tasks, the model utilizes the attached prompts to generate answers. Starting with the randomly initialized prompts for the model can result in an unstable training process (Lester et al., 2021). To mitigate this challenge, the study (Vu et al., 2022) employed a method for transferring the prompts learned in one task to another task. We aim to stabilize the training by initializing the student model prompts P_k^S and P_v^S based on the teacher model prompts P_k^T, P_v^T. We experiment with various prompt initialization methods, including reparameterization, average pooling, and max pooling, as illustrated in Figure 3. In reparameterization, we employ a reparameterization encoder to adjust the length of the teacher model prompts to that of the student model prompts. For average pooling, we divide the teacher model's prompts into smaller segments and compute their averages to initialize the student prompts. In max pooling, we focus on the most prominent features by obtaining the maximum value from each segment of the teacher model's prompts. Based on the experimental results, we apply the reparameterization encoder to the teacher model's prompts to construct the student model's prompts as follows:

$$P_k^S = (P_k^T \cdot W_k^T) + b_k^T, \tag{6}$$

$$P_v^S = (P_v^T \cdot W_v^T) + b_v^T, \tag{7}$$

where W_k^T and W_v^T are the learnable weight matrices used to construct the student's prompts, and b_k^T and b_v^T are the corresponding bias terms used. The results of various prompt initialization experiments are shown in Section 4.5.

3.2.2 Prompt Distillation

In this section, we focus on prompt distillation, a key aspect of the proposed approach. Recognizing the influence of soft prompts on both the hidden states and the prediction layer outputs within the model, we employ the following two distillation techniques: prediction layer and hidden state distillations. These techniques focus on different aspects of the teacher model's output to ensure comprehensive knowledge transfer.

Prediction layer distillation In this method, a student model learns to emulate the preidctions of a teacher model. This process involves the student model utilizing soft labels from the teacher model's output, which encapsulate the teacher model's understanding of the data. Particularly, a loss function is used to minimize the difference between the logits y^S and y^T produced by the student and teacher models, respectively. The distillation loss L_{pred} is formulated as follows:

$$L_{pred} = KL(softmax(y_i^S/\theta), softmax(y_i^T/\theta)), \tag{8}$$

where y_i^S and y_i^T are the logits vectors predicted by the student and teacher, respectively, and KL denotes the Kullback-Leibler divergence, which measures the difference between the probability distributions of the two models. θ is a temperature hyperparameter that adjusts the smoothness of these distributions, enabling a more nuanced transfer of knowledge from the teacher to student model. The distillation loss L_{pred} is then used in the optimization process to update the parameters of the student model, thereby aligning its predictive behavior more closely with that of the teacher model.

Hidden state distillation Additionally, we also distill knowledge from the intermediate representations of the teacher model. The concept of distilling knowledge through intermediate representations was initially introduced by Fitnets (Romero et al., 2015), with the aim of enhancing the training process of the student model. Based on the provided prompts and inputs, we extract knowledge from the transformer layers of the teacher model and distill into the student model. This process is formalized using the loss function L_{hidden}, which is calculated as the MSE between the hidden states H_S and H_T of the student and teacher models, respectively, as follows:

$$L_{hidden} = MSE(H_S, H_T), \tag{9}$$

where the matrices $H_S, H_T \in R^{n \times d}$ represent the hidden states, n is the sequence length, and d is the hidden state dimensionality of the two models.

3.3 Distillation-based Student Learning

While training the student model, the cross-entropy loss is computed similar as that of the teacher model. This loss serves as a measure of the student model's accuracy in predicting the true class labels as follows:

$$L_{CE}^S = -\frac{1}{|B|}\sum_{i=1}^{|B|} \log(softmax(y_i^S)[c_i]). \tag{10}$$

Subsequently, the overall loss function L_{total} for the student model is then a weighted combination of the cross-entropy loss and the distillation losses as follows:

$$L_{total} = \lambda_1 \cdot L_{CE}^S + \lambda_2 \cdot L_{pred} + \lambda_3 \cdot L_{hidden}, \tag{11}$$

where λ_1, λ_2, and λ_3 are the learnable weighted coefficients with the constraint that their combined sum equals 1. During the training, the teacher model parameters are fixed to serve as the sources of prior knowledge.

4 Experiments

This section presents the datasets employed in our experiments, baseline models for comparison, results of these datasets, and analyses from our additional studies.

Table 1: Experimental results for each model on the SuperGLUE validation dataset. For P-Distill, training was performed using a teacher model with the prompt length exhibiting the best performance for P-tuning. The numbers in parentheses indicate the lengths of the prompt attached to the model. (Acc.: Accuracy; **bold**: the best; underline: the second best)

	BoolQ Acc.	CB Acc.	COPA Acc.	MultiRC F1a	ReCoRD F1	RTE Acc.	WiC Acc.	WSC Acc.
Fine-tuning	**0.777**	0.946	0.710	0.705	0.706	0.762	0.746	0.683
P-tuning	0.764$_{(8)}$	0.946$_{(32)}$	**0.810**$_{(4)}$	0.711$_{(16)}$	**0.728**$_{(16)}$	0.794$_{(4)}$	0.756$_{(4)}$	**0.731**$_{(16)}$
	0.738$_{(1)}$	0.929$_{(4)}$	0.790$_{(1)}$	0.707$_{(2)}$	0.721$_{(2)}$	0.783$_{(1)}$	0.745$_{(1)}$	0.692$_{(2)}$
P-Distill	0.776$_{(1)}$	**0.964**$_{(4)}$	**0.810**$_{(1)}$	**0.718**$_{(2)}$	0.726$_{(2)}$	**0.798**$_{(1)}$	**0.759**$_{(1)}$	0.721$_{(2)}$

4.1 Datasets

Our evaluation of the proposed P-Distill method encompasses a comprehensive range of natural language understanding tasks, utilizing datasets that are well-established benchmarks within the field.

For named entity recognition, we use the CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003), CoNLL-2004 (Carreras and Màrquez, 2004), CoNLL-2005 (Carreras and Màrquez, 2005), CoNLL-2012 (Pradhan et al., 2012b), and OntoNotes 5.0 datasets (Weischedel et al., 2013), each providing richly annotated text for entity classification. The SQuAD dataset, in its versions 1.1 (Rajpurkar et al., 2016) and 2.0 (Rajpurkar et al., 2018), facilitate testing reading comprehension, requiring the model to parse passages and answer questions with a high degree of understanding. Furthermore, we include various tasks from SuperGLUE benchmark (Wang et al., 2019), which assesses a model's understanding and reasoning abilities across different contexts, including BoolQ (Clark et al., 2019), CB (De Marneffe et al., 2019), COPA (Roemmele et al., 2011), MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018), RTE (Dagan et al., 2006; Bar Haim et al., 2006), WiC (Pilehvar and Camacho-Collados, 2019) and WSC (Levesque et al., 2011). All these datasets are English, open-source, and utilized solely for academic research purposes. For accurate comparisons, we follow the train, validation, and test set splits as specified in the referenced work (Liu et al., 2022).

4.2 Baselines

We compare P-Distill against the following methods to validate its competitive performance, with all methods utilizing BERT$_{large}$ with 335M parameters as the backbone architecture.

Fine-tuning All parameters of a PLM are updated to the downstream task, thereby adapting the weights of the entire model to the task-specific data.

P-tuning (Liu et al., 2022) It appends trainable continuous prompts to the key and value matrices of a model, enabling task-specific learning while keeping the model's pre-trained weights fixed.

P-Distill Our proposed method, P-Distill, employs a P-tuning approach to train continuous prompts for each task. Subsequently, the optimally trained continuous prompts are distilled into a student model with shorter prompts using two distinct knowledge distillation techniques.

4.3 Experimental Details

In our training process, we exclusively focus on continuous prompts while keeping the backbone parameters of the model fixed. The model is trained with a batch size of 16, and the learning rate is individually optimized for each task. Furthermore, we employ the AdamW optimizer for training. For the temperature hyperparameter θ used in the distillation process, we experimentally determine the optimal setting by sweeping across [1, 5, 10]. For the learnable parameter λ_2, we explore the initial values of [0.1, 0.5, 0.9]. Considering the significant impact of the hidden state loss, we experiment with the initial values of [1e-3, 1e-4, 1e-5] for λ_3. All experiments were performed using PyTorch [1] and HuggingFace Transformers (Wolf et al., 2020) on three NVIDIA A100 GPUs, and to ensure consistency in our results, each task was conducted using the same random seed.

4.4 Results

Tables 1 and 2 present the experimental results of Fine-tuning, P-tuning, and P-Distill. In P-Distill,

[1]https://pytorch.org/

Table 2: Experimental results for each method on named entity recognition, question answering, and semantic role labeling. For P-Distill, training was performed using a teacher model with the prompt length exhibiting the best performance for P-tuning. The numbers in parentheses indicate the lengths of the prompts attached to the model. All metrics are reported as micro-f1 scores. (**bold**: the best; underline: the second best)

	CoNLL03	CoNLL04	CoNLL05 WSJ	CoNLL05 Brown	OntoNotes 5.0	SQuAD 1.1 dev	SQuAD 2.0 dev
Fine-tuning	**0.928**	<u>0.882</u>	<u>0.885</u>	<u>0.827</u>	**0.890**	**0.911**	**0.819**
P-tuning	<u>0.919</u>$_{(64)}$	0.880$_{(128)}$	**0.890**$_{(128)}$	**0.837**$_{(32)}$	0.885$_{(128)}$	<u>0.902</u>$_{(64)}$	<u>0.782</u>$_{(128)}$
	0.914$_{(8)}$	0.866$_{(16)}$	0.877$_{(16)}$	0.807$_{(4)}$	0.881$_{(16)}$	0.891$_{(8)}$	0.771$_{(16)}$
P-Distill	<u>0.919</u>$_{(8)}$	**0.888**$_{(16)}$	<u>0.885</u>$_{(16)}$	0.817$_{(4)}$	<u>0.886</u>$_{(16)}$	0.896$_{(8)}$	0.775$_{(16)}$

the prompt length is compressed to one-eighth of that of the teacher model prompts. For fewer than eight teacher model prompts, the length is compressed to 1. Experimentally, the proposed P-Distill method exhibits a comparable or superior performance than those of the other methods while using shorter prompts.

Results on SuperGLUE Table 1 shows the performance of each approach on the SuperGLUE benchmark. Despite using shorter prompts, P-Distill matches and exceeds the performances of the baseline methods, particularly P-tuning.

When applying the P-tuning, we observed comparable or superior performance in all tasks, except for BoolQ, when compared to the Fine-tuning. However, a limitation of P-tuning is that the length of tokens that can be input into the model is reduced by the length of the prompt used. For instance, setting the prompt length to 32 in the CB task resulted in a performance comparable to that of Fine-tuning. However, this also resulted in a reduction of 32 tokens in the input length. Applying the proposed P-Distill compresses the prompt length by eightfold, resulting in a better performance even when the prompt length was reduced to 4. Particularly, P-Distill exhibited a 1.90% higher performance than that of the teacher model and 2.73% improvement over the same-length prompt trained using P-tuning. This indicates that knowledge learned from a teacher model with longer prompts can be effectively transferred to a student model with eight times shorter prompts.

Results on Across Tasks Based on Table 2, achieving optimal performance via P-tuning requires training with longer prompts for question-answering, named entity recognition, and semantic role labeling tasks. Similar to the SuperGLUE benchmark and compared to the existing methods, P-Distill achieved comparable or better performance using shorter prompts even for tasks requiring longer prompts. This is evident in the CoNLL04 task, where the optimal prompt length for the model trained using P-tuning is 128. Despite occupying a significant portion of the input token space with a length of 128, the performance was lower than that achieved with Fine-tuning. However, applying prompt compression using P-Distill reduced the prompt length to 16 while outperforming Fine-tuning. Notably, P-Distill achieves a performance improvement of 2.54% over the same-length prompt trained using P-tuning and a 0.90% improvement over the teacher model.

4.5 Additional Study

To further verify the effectiveness of the proposed method, we conduct experiments using the following P-Distill variants:

P-Distill$_{-init}$ Instead of training without prompt initialization, it focuses exclusively on leveraging the two types of distillation losses designed to transfer the knowledge from the teacher to student model in different ways.

P-Distill$_{-pred}$ This approach does not implement the prediction layer distillation loss. Following the application of the prompt initialization method, it trains the student model based on the hidden state distillation loss. This method aligns the internal representations of the student model with those of the teacher model without focusing on the final output predictions.

P-Distill$_{-hidden}$ This variant does not consider the differences between the hidden state outputs of the teacher and student models. Instead, it focuses on training based on differences in the prediction layer output. This approach aligns the final predictions of the student model closely with those of the teacher model without directly focusing on their

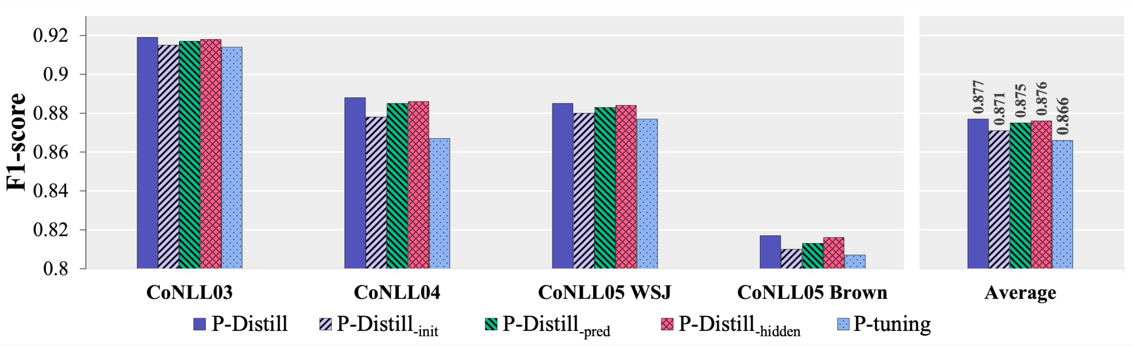

Figure 4: Comparison of ablation study results across various tasks, with different colors and bar styles representing the distinct variants of P-Distill.

Table 3: Comparison of additional experiment results across various tasks based on prompt initialization methods. All metrics are reported as micro-f1 scores. (**bold**: the best)

	CoNLL03	CoNLL04	CoNLL05 WSJ	CoNLL05 Brown
P-Distill	**0.919**	**0.888**	**0.885**	**0.817**
P-Distill$_{mean}$	0.915	0.875	0.878	0.809
P-Distill$_{max}$	0.912	0.872	0.872	0.803

internal representations.

Figure 4 shows that all three variants of P-Distill cause performance degradation, which is evident in downstream tasks and overall averages. However, the extent of degradation varies among different variants. First, P-Distill$_{-init}$ exhibited the most significant performance degradation across various tasks. Even without prompt initialization, conducting prediction layer distillation and hidden state distillation led to performance improvements over P-tuning. However, we observed lower performance when prompts were randomly initialized compared to when prompt initialization is applied. This indicates that prompt initialization, based on the teacher model's prompt, is crucial in prompt-based knowledge distillation. Second, P-Distill$_{-hidden}$ and P-Distill$_{-pred}$ exhibited decreased prediction performance. This demonstrates that integrating prompt initialization with the hidden state or prediction layer distillation techniques enhances the stability and effectiveness of knowledge distillation. Therefore, combining these methods is the most effective approach for prompt-based knowledge distillation, resulting in the best performance.

To inspect the effectiveness of different prompt initialization methods within P-Distill, we conduct experiments to compare the performance of P-Distill with two variants: P-Distill$_{mean}$, which initializes the teacher model prompts using an average pooling layer, and P-Distill$_{max}$, which uses a max pooling layer for the same purpose. The results, as detailed in Table 3, demonstrated that both P-Distill$_{mean}$ and P-Distill$_{max}$ underperformed in comparison to the P-Distill. We assume that the use of average pooling and max pooling leads to an excessive simplification of the teacher model's prompts, resulting in the loss of crucial nuances and complexities. Conversely, the reparameterization encoder for prompt initialization effectively captures and transfers the complex knowledge of the teacher model prompts, thereby enhancing the predictive performance. This suggests that the reparameterization encoder is a more suitable method for prompt initialization in P-Distill, contributing significantly to the overall effectiveness of the knowledge distillation process.

5 Conclusion

In this paper, we introduce P-Distill, a novel approach in NLP that utilizes two knowledge distillation techniques to enhance performance by compressing unnecessary prompt length. This approach combines prompt initialization, two types of prompt distillation to effectively transfer knowledge from a teacher model with longer prompts to a student model with prompts that are eight times shorter. To evaluate the efficacy of our proposed method, we conduct experiments across various NLP tasks. Our results demonstrate that using prompts of the same length, the proposed method achieves an average improvement of 2.73% over the existing prompt-tuning methods across the SuperGLUE benchmark. Furthermore, P-Distill exhibits competitive performance even against models trained with prompts that are eight times longer.

Limitations

One limitation of this study is that we evaluated our method only on the BERT architecture. Conducting additional experiments on other architectures could be beneficial to determine the generalizability of our findings. Additionally, while our model improves performance through the process of training a teacher model and transferring its knowledge, it incurs more time and cost compared to previous methods. In future work, we plan to develop an approach that integrates the training of the teacher model and the knowledge distillation process in an end-to-end manner.

References

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. 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 NAACL-HLT 2019*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. To appear in proceedings of Sinn und Bedeutung 23. Data can be found at https://github.com/mcdm/CommitmentBank/.

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.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

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.

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.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

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*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In

AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning, volume 46, page 47.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. Gpt understands, too. *AI Open*.

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.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: The word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of NAACL-HLT*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012a. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint conference on EMNLP and CoNLL-shared task*, pages 1–40.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012b. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

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.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *CoRR*, abs/2010.15980.

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*, pages 2158–2170, Online. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

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)*, pages 5039–5059, Dublin, Ireland. 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. *CoRR*, abs/1905.00537.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2020. Superglue: A stickier benchmark for general-purpose language understanding systems.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23:170.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension.