Prompt-Based Meta-Learning For Few-shot Text Classification

Haoxing Zhang **Xiaofeng Zhang** **Haibo Huang** **Lei Yu***
Sino-French Engineer School, Beihang University, Beijing, China
{mickael97, xiaofeng_z, huanghaibo, yulei}@buaa.edu.cn

Abstract

Few-shot Text Classification predicts the semantic label of a given text with a handful of supporting instances. Current meta-learning methods have achieved satisfying results in various few-shot situations. Still, they often require a large amount of data to construct many few-shot tasks for meta-training, which is not practical in real-world few-shot scenarios. Prompt-tuning has recently proved to be another effective few-shot learner by bridging the gap between pre-train and downstream tasks. In this work, we closely combine the two promising few-shot learning methodologies in structure and propose a **P**rompt-**B**ased **M**eta-**L**earning (PBML) model to overcome the above meta-learning problem by adding the prompting mechanism. PBML assigns label word learning to base-learners and template learning to meta-learner, respectively. Experimental results show state-of-the-art performance on four text classification datasets under few-shot settings, with higher accuracy and good robustness. We demonstrate through low-resource experiments that our method alleviates the shortcoming that meta-learning requires too much data for meta-training. In the end, we use the visualization to interpret and verify that the meta-learning framework can help the prompting method converge better. We release our code to reproduce our experiments[1].

1 Introduction

Humans can quickly learn new knowledge from a few examples, reflecting a high degree of intelligence. The meta-learning method was proposed to implement such human-like intelligence and achieved promising results in various few-shot situations. It lifts the training unit from data point to task and trains a meta-learner over many tasks (called episodes) to grasp the few-shot learning (FSL) pattern. In this paper, we tackle the few-shot text classification, which aims to predict a given text's label with only a handful of support instances.

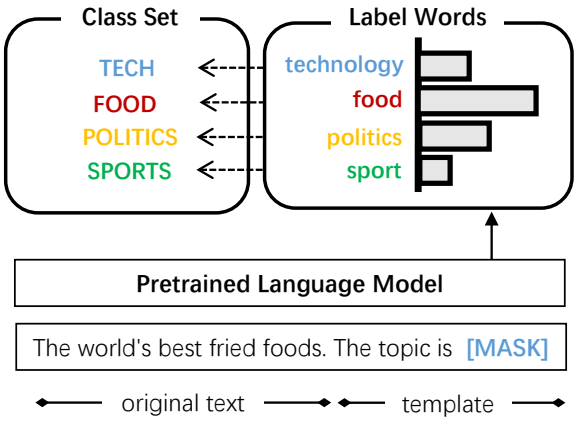


Figure 1: An example of prompting. We concatenate the original text with a template. Then a pre-trained language model (PLM) will fill in [MASK] with words from a predefined label word set. A verbalizer will further map the predicted label word into the corresponding task label.

Current methods to solve the few-shot text classification problem are based on meta-learning (Finn et al., 2017; Dong et al., 2020; Bao et al., 2020; Geng et al., 2020) or prompt-tuning (Brown et al., 2020; Gao et al., 2021; Zhao et al., 2021; Liu et al., 2021; Schick and Schütze, 2021). These two approaches deal with "few-shot" from different perspectives. On the one hand, the meta-learning algorithm aims to train a meta-learner across many episodes to extract transferable meta-knowledge. For each episode, the meta-learner further trains an individual base-learner for task-specific adaptation. While existing meta-learning algorithms allow models to rapidly learn new concepts based on previous experience, they often require a large amount of meta-training data to construct meta-tasks. This bottleneck makes meta-learning methods less practical in real-world few-shot scenarios

*Corresponding author.
[1]Code at https://github.com/MGHZHANG/PBML

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 1342 - 1357
December 7-11, 2022

where diverse and sufficient meta-tasks may not be available. Prompt-tuning, on the other hand, has recently emerged as another effective few-shot learning methodology. It transforms the original problem into a cloze-test to bridge the gap between pre-train and downstream tasks (Figure 1). By doing so, prompt-tuning is more likely to stimulate the knowledge from the pre-training stage and can efficiently adapt the masked language model (MLM) to downstream tasks with fewer training data.

Given that the above two approaches both achieved satisfying results in various few-shot situations, we hope to alleviate the shortcomings of meta-learning methods by adding a prompting mechanism. In this work, we use an elegant way to closely combine these two approaches in structure and propose Prompt-Based Meta-Learning (**PBML**) to push the performance on few-shot text classification. In terms of prompts, we adopt the 'soft' strategy (i.e., use continuously differentiable label words and templates). Regarding meta-learning, our meta-learner mainly learns soft template embeddings and an MLM-based encoder. The core idea of our combination of the two approaches is to assign template&encoder learning to the meta-learner and label word learning to base-learners, respectively. The intuition is that different tasks may involve different classes, and label words need to consider specific classes, so the learning of label words is handled over to base-learners for task-specific adaptation. Correspondingly, the output embedding at `[MASK]` position by the prompting method reflects the model's understanding of the text. Various tasks should share this ability of natural language understanding (NLU), which is why the learning of template and encoder is assigned to the meta-learner.

Experimental results demonstrate the good compatibility between meta-learning and prompt-tuning. Our **PBML** (1) achieved a new state-of-the-art performance on four datasets under few-shot settings, with higher accuracy and good robustness. (2) Through low-resource experiments, we can clearly see that the prompting mechanism helps **PBML** maintain good performance when the available meta-training data is significantly reduced. (3) Comparison with prompt-tuning baseline demonstrates that meta-learning enables prompt-tuning to adapt more efficiently to new tasks. (4) The visualization results verify that the "learning-to-learn" framework could result in better convergence for prompts.

2 Preliminaries

2.1 Few-Shot Text Classification

Given a text x, we aim to predict its label y with a few annotated examples (x_i, y_i). We follow the commonly adopted N-way K-shot setting, where N is the number of classes and K is the number of examples per class. Each task provides a support set $\mathcal{S}$ containing $N \times K$ support instances and a query set $\mathcal{Q}$. We train a classifier on $\mathcal{S}$ and evaluate it on $\mathcal{Q}$. Higher N and lower K indicate a more challenging task. An example of a 5-way 3-shot scenario is given in Appendix A.

2.2 Meta-learning

The purpose of meta-learning is to train a meta-learner through diverse meta-tasks such that the meta-learner can quickly obtain a task-specific base-learner on a small support set.

Formally, we consider two phases: meta-training and meta-testing. During meta-training, we sample a batch of tasks $\{\mathcal{T}_b\}_{b=1}^{B}$ from a task distribution $p(\mathcal{T})$. Then the meta-learner trains a base-learner for each task $\mathcal{T}_b$ using the loss on the support set $\mathcal{S}_b$. The base-learner is then tested on the query set $\mathcal{Q}_b$, and we optimize the meta-learner by minimizing the query loss. During meta-testing, new tasks are sampled, and the accuracy of query instances will be measured. The two phases share no overlapping classes, so we can estimate the model's ability to handle new classes.

2.3 Prompt

Prompt aims to bridge the gap between the pre-train and downstream tasks. It concatenates the original text x with a template containing at least one `[MASK]` token, converting the problem into a cloze-test. We note the concatenated text x_{prompt}. A pre-trained language model $\mathcal{M}$ is applied to predict the label word at `[MASK]` position. Then, a function called verbalizer $\phi : w \in \mathcal{W} \mapsto y \in Y$ maps the set of label words $\mathcal{W}$ to the set of task labels Y. For instance, we can set $\phi(\text{"technology"}) = \text{TECH}$. We formulate the probability of instance x predicting class y as follows:

$$P(y|x) = P(\texttt{[MASK]} = w_y|x_{prompt}) = \frac{\exp(\mathbf{w}_y \cdot \mathbf{h})}{\sum_{\mathbf{w} \in \mathbf{W}} \exp(\mathbf{w} \cdot \mathbf{h})} \quad (1)$$

Softly vs hardly prompts refers to the way of designing the template tokens that are concatenated to the original text. Hardly prompts use discrete tokens that are fixed and not learnable, such as “The topic is [MASK]”. Softly prompts use continuous embeddings that are initialized with real words but can be optimized during training, such as “This article discusses [MASK]”. Continuous prompts are more flexible and can better adapt to different tasks.

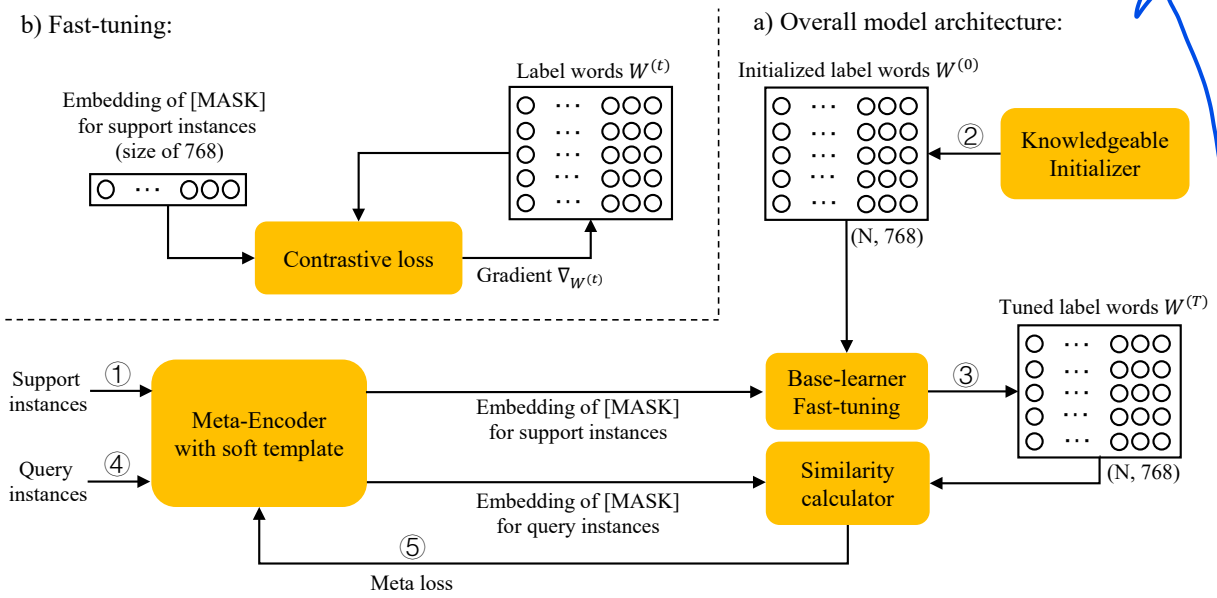


Figure 2: a) Overall model architecture: the work flow is numbered from 1 to 5. Meta-encoder (subsection 3.1) encodes instances using MLM and obtains embeddings at the [MASK] position (steps 1, 4); Label word initialization (subsection 3.2) explores an external knowledge graph to initialize label word embeddings $\mathbf{W}^{(0)}$ (step 2); Fast-tuning (subsection 3.3): A base-learner continuously tunes the label word embeddings based on the support set (step 3). Meta-optimization (subsection 3.5): optimizing meta-parameters with the loss on query set (step 5); b) Fast-tuning: using contrastive loss to update label word embeddings.

where $\phi(w_y) = y$, $\mathbf{w}_y$ is the label word embedding of w_y and $\mathbf{h}$ is the output hidden state of [MASK].

3 Methodology

This section presents the details of our proposed framework, PBML (Figure 2). PBML consists of 3 parts: Firstly, the meta-learner (Meta-Encoder) encodes instances and gets the embedding of the [MASK] token for each instance. Secondly, we explore an external knowledge graph for continuous label word initialization. Then a base-learner will update the label word embeddings using predicted embeddings of support instances. Inference for queries is based on the adapted label word embeddings, and we use the loss on the query set $\mathcal{Q}$ for meta-optimization.

3.1 Meta-encoder and Template Design

Given a statement x, we first concatenate a template to x and obtain x_{prompt}. If we take the topic classification task as an example, the prompted text can be given as:

$$x_{prompt} = x \text{ The topic is [MASK]}.$$

where “The topic is [MASK]” are template tokens (details of template designs are in Appendix C). Then a masked language model, noted $\mathcal{M}$, serving as the meta-encoder, takes x_{prompt} as input and outputs $\mathbf{h}$, the hidden state of [MASK] as the predicted answer representation.

Though we manually select template tokens from vocabulary items, we treat them “softly” (i.e., continuous template). They are replaced by learnable embeddings, initialized with the exact embeddings as manually selected real words. This soft strategy allows templates to be optimized continuously instead of being limited by discrete tokens. We note the encoder parameters as θ_e and soft template embeddings as θ_s. The meta-encoder can be formulated as follows:

$$\mathbf{h} = \mathcal{M}(x_{prompt}; \theta_e, \theta_s) \quad (2)$$

3.2 Label word Initialization

Given an N-way K-shot episode, we denote $\{\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_N\}$ the N categories. Though it is intuitive to directly apply class names as label words, the semantic meaning of class names is sometimes too conceptional without sufficient semantic information. Following the idea of Knowledgeable Prompt (Hu et al., 2022), we introduce Related Words [2], an external knowledge graph to expand rich label words for each class $\mathcal{C}_i$ from their class name. Specifically, we explore the knowledge graph to obtain the top $N_{KG} = 20$ relevant

[2] https://relatedwords.org

words with the class name as candidate words. The obtained candidate word set $\{w_k^i\}_{k=1}^{N_{KG}}$ includes synonyms and words highly related to the class name. For example, the candidate words associated with "Politics" are "policy", "government", "law", "diplomatic", etc.

Then we merge the candidate words into one prototype for each class by averaging candidate word embeddings$\{\mathbf{w}_k^i\}_{k=1}^{N_{KG}}$. In the end, we obtain N synthesized continuous label word embeddings, and we note $\mathbf{W}^{(0)} \in \mathbb{R}^{N\times D}$, the matrix containing the initial N label word embeddings. $\mathbf{W}^{(0)}$ represents roughly the semantic meaning of N classes and will be further tuned in the next module.

3.3 Label word Fast-tuning

In this section, we introduce the fast-tuning module for label word adaptation. A base-learner will continuously tune the initialized label word embeddings $\mathbf{W}^{(0)}$, using support instances. Our objective is to make these label word embeddings more discriminative by incorporating contextual information from the support set.

Specifically, we impose two goals we need to achieve by fast-tuning. (a) For each support instance s_i^j from class $\mathcal{C}_i$, we note $\mathbf{h}_i^j$ its hidden state at the `[MASK]` position. We expect that the similarity between $\mathbf{h}_i^j$ and $\mathbf{w}_i$ (label word embedding of $\mathcal{C}_i$) is larger than the similarity between $\mathbf{h}_i^j$ and other label words; (b) For each class $\mathcal{C}_i$, the similarity between its label word $\mathbf{w}_i$ and the support instances belonging to $\mathcal{C}_i$ should be larger than the similarity between $\mathbf{w}_i$ and instances from other classes. To realize these two goals, we define the following two contrastive losses:

$$\mathcal{L}_{s2w}^{att} = -\frac{1}{NK}\sum_{\substack{1\le i\le N\\ 1\le j\le K}} \alpha_i^j \log \frac{\exp\left(\mathbf{h}_i^j \cdot \mathbf{w}_i^{(t)}\right)}{\sum_{\mathbf{w}\in\mathbf{W}^{(t)}} \exp\left(\mathbf{h}_i^j \cdot \mathbf{w}\right)} \quad (3)$$

$$\mathcal{L}_{w2s}^{att} = -\frac{1}{N}\sum_{1\le i\le N} \log \frac{\exp\left(\mathbf{w}_i^{(t)} \cdot \mathbf{h}_i\right)}{\sum_{1\le i'\le N} \exp\left(\mathbf{w}_i^{(t)} \cdot \mathbf{h}_{i'}\right)} \quad (4)$$

$$\mathbf{h}_i = \sum_j \alpha_i^j \mathbf{h}_i^j$$

where $\mathbf{W}^{(t)} \in \mathbb{R}^{N\times D}$ is the label word matrix at iteration t. $\mathbf{w}_i^{(t)}$ is the i^{th} row of $\mathbf{W}^{(t)}$, representing the label word embedding of class $\mathcal{C}_i$. To make the adaptation more robust, we add an instance-level attention mechanism via coefficient α_i^j. Such attention score is to measure the informative degree of each support instance. According to previous works (Gao et al., 2019a; Dong et al., 2020), instances are not equally informative, and we should make those informative (resp. noisy instances) contribute more (resp. less) during fast-tuning to improve the robustness. We calculate α_i^j as shown below:

$$\alpha_i^j = \frac{\exp(\mathbf{h}_i^j \cdot \mathbf{w}_i^{(0)}/\gamma)}{\sum_{j'=1}^{K} \exp(\mathbf{h}_i^{j'} \cdot \mathbf{w}_i^{(0)}/\gamma)} \quad (5)$$

where γ is a temperature hyper-parameter set to 3. Equation 5 indicates that if $\mathbf{h}_i^j$ is more similar to the initial label word embedding, it is considered more informative and assigned with higher attention α_i^j. In contrast, we assign noisy instances with little attention in $\mathcal{L}_{s2w}^{att}$ and $\mathcal{L}_{w2s}^{att}$, producing smaller gradient steps and more robust adaptation trajectories.

At each iteration of fast-tuning, we apply gradient descendent as follows:

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} - \beta_{task}\nabla_{\mathbf{W}^{(t)}}\mathcal{L}^{att}(\mathcal{S}, \mathbf{W}^{(t)}) \quad (6)$$

where β_{task} is base-learner's task-adaptation learning rate and $\mathcal{L}^{att} = \mathcal{L}_{s2w}^{att} + \mathcal{L}_{w2s}^{att}$. The fast-tuning process will iterate T steps and output $\mathbf{W}^{(T)}$.

We clarify here that we only update label words $\mathbf{W}^{(t)}$ during fast-tuning, which corresponds to the task-specific adaptation of the base-learner. The meta-learner $\mathcal{M}(\theta_e, \theta_s)$ is unchanged during fast-tuning as it contains task-agnostic parameters and learns across tasks. Since we only update $\mathbf{W}^{(t)}$, the fast-tuning process saves lots of computational costs, with $O(N^2 \times K \times D \times T)$ time complexity. (More details in Appendix G)

3.4 Inference for query

We calculate the inner product between the query embedding $\mathbf{h}_q$ and task-adapted label word embeddings $\mathbf{W}^{(T)}$ to predict query instances' labels. The probability score for class $\mathcal{C}_i$ is

$$P(\mathcal{C}_i|q) = \frac{\exp(\mathbf{h}_q \cdot \mathbf{w}_i^{(T)})}{\sum_{\mathbf{w}\in\mathbf{W}^{(T)}} \exp(\mathbf{h}_q \cdot \mathbf{w})} \quad (7)$$

Then we use the argmax function for the prediction.

$$\widetilde{\mathcal{C}}_q = \underset{i}{\text{argmax}}\, P(\mathcal{C}_i|q) \quad (8)$$

3.5 Meta-Optimization

During meta-training, we randomly construct many few-shot meta-tasks from the meta-training set. The base-learner of each episode learns task-specific label word embeddings $\mathbf{W}^{(T)}$ as shown in subsection 3.3. Then, to meta-learn the template and encoder from tasks, we update our meta-learner $\mathcal{M}(\theta_e, \theta_s)$ considering the loss on the query set $\mathcal{Q}$. Hence, the optimization rule of the meta-learner can be formulated as follows:

$$\theta = \theta - \beta_{meta} \nabla_{\theta} \mathcal{L}(\mathcal{Q}, \mathbf{W}^{(T)}, \theta) \quad (9)$$

where β_{meta} is the meta-learning rate for meta-parameters $\theta = (\theta_e, \theta_s)$ and $\mathcal{L}$ is the cross-entropy loss on $P(\mathcal{C}_i|q)$. The overall algorithm of PBML is summarized in Appendix D. In this way, we combine prompt-tuning with meta-learning such that both task-specific and task-agnostic knowledge can be learned. Specifically, 1) the meta-learner learns at a lower speed β_{meta} to reach appropriate parameters for soft template embeddings and the encoder layers 2) base-learners learn continuous label words at a higher speed β_{task} for fast adaptation.

4 Experiments

This section evaluates the performance of our proposed **PBML**. We conduct extensive experiments on four widely-used text classification datasets under few-shot settings and make a full-scale comparison with existing state-of-the-art baselines. We report our implementation details in Appendix E.

4.1 Dataset

Following Bao et al. (2020), we adopt the following four text classification datasets for experiments: **FewRel** (Han et al., 2018) (relation classification), **HuffPost headlines** (Misra, 2018) (news headlines classification), **Reuters** (Lewis, 1997) (articles classification) and **Amazon product data** (He and McAuley, 2016) (reviews classification). These datasets provide diverse benchmarks that vary in domain and text length. We use the same setting as previous work for splitting training, validation, and testing sets. More details are in Appendix A.

4.2 Baseline Models

We compare our model with previous strong baselines for few-shot text classification. **BERT-PAIR** (Gao et al., 2019b) pairs query and support instance and apply BERT classification head to obtain the similarity score. **MTB** (Soares et al., 2019) proposes a pre-training task for Relation Extraction. The pre-trained relation extractor is further fine-tuned on FewRel. **Frog-GNN** (Xu and Xiang, 2021) is a graph neural network learning query embeddings after the multi-aggregation of neighbor nodes. We also compare various meta-learning models. **1-NN** and **Prototypical Network** (Snell et al., 2017) are metric-based meta-learning. The latter averages support instance embeddings as prototypes and predict the query label based on the distance with prototypes. (**PROTO-IDF**, **PROTO-BERT** stands for two encoders). **DS** (Bao et al., 2020), **MAML** (Finn et al., 2017) and **MIML** (Dong et al., 2020) are optimization-based meta-learning methods, showing excellent few-shot learning performance. They apply different meta-learners and train them across episodes.

We also implement the baseline **Prompt-tuning** without meta-learning to assess the impact of meta-learning in our PBML. **Prompt-tuning** no longer constructs $\mathcal{S}$ and $\mathcal{Q}$ to meta-train through episodes but directly uses label words (from knowledge graph) to pre-train the encoder and soft template. To test the model's ability to solve new N-way K-shot tasks (unseen classes), we further fine-tune it on the small support set before query inference.

4.3 Results and Analysis

4.3.1 Main Results

Table 1 shows the results of different models on four benchmark test sets under 5-way 1-shot and 5-way 5-shot settings. **PBML** significantly and consistently outperforms previous models, especially under the more challenging one-shot configuration. (Similar observations under 10-way k-shot settings in Appendix H). Notably, on the relation classification task, our method exceeds the accuracy of human performance reported from the FewRel leader-board[3]. It also outperforms the state-of-the-art method **MTB**, which utilizes larger PLM and requires external corpus for relation extraction pre-training.

Meanwhile, we have the following observations: Our model outperforms previous meta-learning baselines (**PROTO-BERT**, **MAML**, and **MIML**) by a large margin. We also observed that **PBML** converges faster and smoother (Ap-

[3] https://www.zhuhao.me/fewrel/

Method	FewRel		HuffPost		Reuters		Amazon		Average	
	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
1-NN	46.8	60.6	31.5	42.3	57.8	82.9	51.4	67.1	46.9	63.2
PROTO-IDF	43.0	61.9	34.8	50.2	61.0	72.1	41.9	59.2	45.2	60.9
DS	67.1	83.5	43.0	63.5	81.8	96.0	62.6	81.1	63.6	81.0
PROTO-BERT	86.5	95.0	52.9	69.3	86.7	93.9	68.1	82.5	73.6	85.2
BERT-PAIR	88.3	93.2	53.5	69.1	88.6	94.7	69.8	82.8	75.1	84.9
FROG-GNN	88.9	94.3	54.1	69.6	-	-	71.5	83.6	-	-
MAML	87.5	94.4	43.7	54.3	84.8	94.0	75.7	85.2	72.9	82.0
MIML	92.6	96.0	53.9	57.1	86.6	95.2	76.1	84.4	77.3	83.2
Prompt-Tuning	93.8	96.1	71.8	74.0	92.9	96.7	76.5	81.8	83.8	87.2
Human	92.2	-	-	-	-	-	-	-	-	-
MTB $\star$	93.9	97.1	-	-	-	-	-	-	-	-
PBML (OUR)	**96.6**	**97.4**	**74.9**	**78.0**	**96.4**	**97.6**	**81.8**	88.0	**87.4**	**90.3**
OUR w/o KG	95.1	97.2	71.3	73.7	95.9	97.2	78.6	86.1	85.2	88.3
OUR w/o $\mathcal{L}_{w2s}^{att}$	95.9	97.1	74.0	76.5	94.7	96.9	80.6	**88.1**	86.3	89.7
OUR w/o $\mathcal{L}_{s2w}^{att}$	95.2	96.2	73.8	76.0	93.1	94.9	79.4	84.0	85.4	87.8

Table 1: Results of 5-way 1-shot and 5-way 5-shot text classification accuracies (%) on four benchmark test sets. We report the results of baseline models as published. Several baselines were not tested on all four datasets when published, and we re-run their public code (if available) to supplement the results.

pendix F). We infer that the improvements are mainly from prompting, which effectively bridges the gap between pre-train and downstream tasks. Then, compared to the **Prompt-tuning** baseline, the improvements demonstrate that meta-learning can help prompting adapt to new tasks more efficiently. These two comparisons show the excellent compatibility between prompt-tuning and meta-learning. In subsubsection 4.3.4, we will further explain and visualize why such a combination is good.

4.3.2 Ablation Study

At the bottom of Table 1, we present the results of ablations on the external knowledge graph and the two contrastive losses. We observe that the performance drops consistently if we directly use class names to initialize $\mathbf{W}^{(0)}$ without introducing external KG. This suggests that our knowledgeable label word initialization does provide rich and essential information. The ablation study on loss functions shows that both $\mathcal{L}_{s2w}^{att}$ and $\mathcal{L}_{w2s}^{att}$ contribute to fast-tuning, though the former has a more significant impact.

We also conducted the ablation study on the instance-level attention α_i^j designed for robustness (i.e., allowing informative instances to contribute more to the loss). To evaluate the robustness, we follow the idea of Dong et al. (2020) by intentionally replacing a portion of support instances with randomly sampled instances from different categories. Table 2 displays the accuracy and the performance drop scale under various noised settings.

noise rate	model	Accuracy (%)	$\lvert\Delta\rvert$
0%	**PBML w/o** att	94.98	-
	PBML	**95.23**	-
20%	**PBML w/o** att	93.27	1.71
	PBML	**94.63**	**0.60**
40%	**PBML w/o** att	91.26	3.72
	PBML	**93.16**	**2.07**
60%	**PBML w/o** att	89.14	5.84
	PBML	**91.85**	**3.38**

Table 2: 5-way 5-shot model robustness on FewRel validation set under different noised setting.

The results show that our attentional loss does help to improve the robustness. More obvious advantages can be seen in a noisier setting.

4.3.3 Dependence on Training Data

In this section, we investigate our method's performance when lacking sufficient meta-training data. As mentioned previously, existing meta-learning methods often require a sufficiently large dataset to build diverse episodes for meta-training. Otherwise, the performance will drop seriously.

To investigate the effect of the amount of meta-training data, we follow previous works (Soares et al., 2019; Ding et al., 2021b) and limit the size of the meta-training set in two ways: 1) by decreasing the number of instances per class; 2) by decreasing the number of available classes. We evaluate the 5-way 1-shot accuracy under various low-resource configurations on FewRel. Figure 3 and Figure 4 show respectively the results under the two size con-

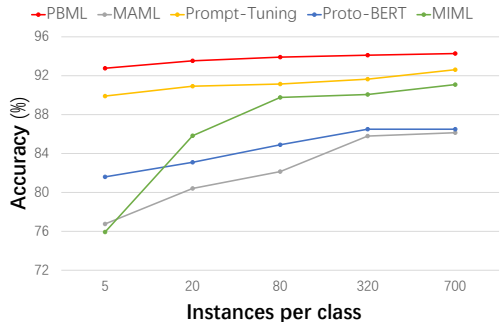


Figure 3: Impact of number of instances per type

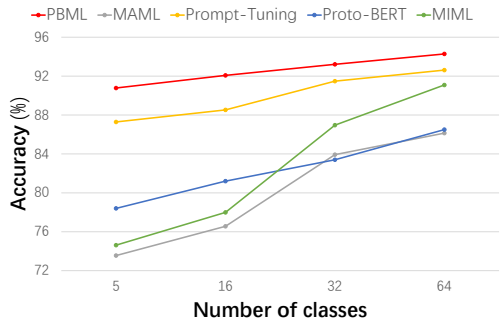


Figure 4: Impact of number of classes

straints. **Prompt-tuning** and our proposed **PBML** suffered little from low resource data under both constraints. For example, when we limit the meta-training data per class to 5, **PBML** could still maintain reasonable accuracy while the performance of other meta-learning methods (i.e., **MAML**, **MIML**, **PROTO-BERT**) drops significantly. We infer that such improvement is also because prompting can narrow the gap between pre-train and downstream tasks, thus making meta-learning less dependent on meta-training data. The results demonstrate that our PBML framework can not only learn from a few supporting instances during meta-testing (benefits of meta-learning) but also demand less training data during meta-training (benefits of prompting), showing a comprehensive few-shot learning ability.

4.3.4 Visualization

Previous sections show that prompting makes meta-learning better few-shot learners. In this section, we emphasize the benefits of the meta-learning side. We will visualize and analyze how "learning to learn" contributes to prompting.

We made a PCA visualization of different encoders' output representation at the `[MASK]` position to demonstrate that meta-learning can help the encoder converge. We have selected five categories from FewRel that are difficult to distinguish (according to Brody et al. (2021)). As shown in Figure 5, the encoder with open-source pre-trained parameters (a) presents poor separation. Classical prompt-tuning (b, c) directly uses label words to tune the encoder and can improve the performance. In comparison, the encoder meta-trained by the PBML algorithm (d) produces the best separation. The results suggest that meta-learning can help the prompting method learn a better encoder and soft template, showing a stronger ability to distinguish different concepts.

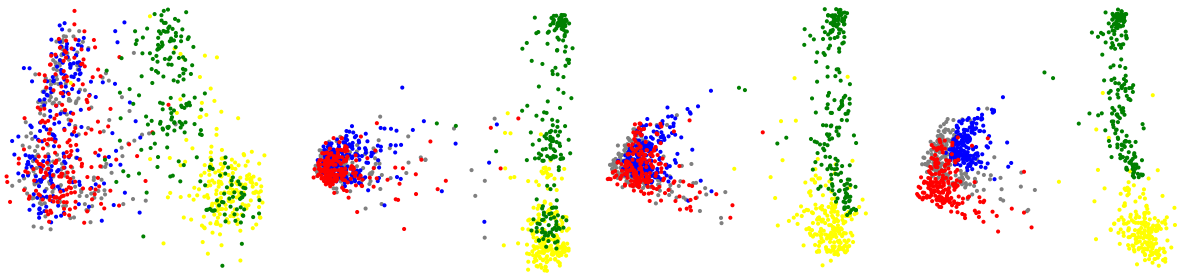
Classical prompt-tuning tunes the encoder using cross-entropy loss on distribution shown in Equation 1. Since label word embeddings (real word embeddings or learnable embeddings) contribute directly to the loss, the convergence of the encoder and soft template highly depends on the quality of the label word embeddings. By comparison, our **PBML** does not directly use label words to tune the encoder. Instead, our method has two loops: In the outer loop, we randomly sample meta-tasks, and during the inner loop of each meta-task, the initialized label words $\mathbf{W}^{(0)}$ are fast-tuned on the support set within T iterations. At each iteration, the contrastive loss enhances the discrimination of different classes. By considering support instances, the base-learner can incorporate contextual information into label word embeddings and produces $\mathbf{W}^{(T)}$ of richer semantic information. When the inner loop is finished, our encoder and soft template will be optimized using Equation 9 where the more ideal label word embeddings $\mathbf{W}^{(T)}$ act. Once the meta-learner is optimized, it can produce better support instance embeddings and reinforce the quality of $\mathbf{W}^{(T)}$. In this learning-to-learn framework with two loops, the meta-learner (soft template and encoder) and the base-learner (label words) can promote each other and enhance their coordination, resulting in better performance and a smooth convergence.

5 Related Work

5.1 Meta-Learning

Current promising meta-learning methods are mainly metric-based or optimization-based.

Metric-based meta-learning learns a metric space where representations of instances from the same class can get closer. Han et al. (2018) applied popular metric-based few-shot-learning methods



(a) Encoder with pretrained parameters

(b) Prompt-tuned encoder using learnable label words

(c) Prompt-tuned encoder using real label words from KG

(d) Encoder meta-trained by the PBML algorithm using fast-tuned label words

Figure 5: PCA visualization of encoders' output representation at `[MASK]` for 1000 instances in FewRel (200 instances per class). The correspondence between colors and categories is as follows: child, spouse, mother, member of, part of. The five categories we chose are more difficult to distinguish apart.

(Koch et al., 2015; Vinyals et al., 2016; Snell et al., 2017) to few-shot relation classification but the results suggest that there is still an ample space to improve. Gao et al. (2019a) introduced a hybrid attention mechanism into Prototypical Network, which can enhance the model's robustness. Ye and Ling (2019) proposed a Multi-Level Matching and Aggregation Network to learn representations through multiple rounds of interaction. Geng et al. (2019) proposed Induction Network, applying the dynamic routing algorithm to build class-aware representation. Xiao et al. (2021) integrated label information into features, providing vital guidance for the prototypical network. Soares et al. (2019) and Ding et al. (2021b) proposed novel ways to pre-train a metric space for relation extraction.

Optimization-based meta-learning aims to improve the optimization procedure so that the base-learner can learn from a few examples without dramatic over-fitting. Strong frameworks such as MAML (Finn et al., 2017) meta-learns task-sensitive parameters, which serve as a good initialization. Inspired by their work, Dong et al. (2020) proposed meta-information guided meta-learning (MIML), where they introduce information from class names to realize a class-aware initialization. Ravi and Larochelle (2017) studied optimization rules and proposed an LSTM-based meta-optimizer. Li et al. (2017) proposed Meta-SGD, making the learning rate a meta-parameter to be learned. Bao et al. (2020) proposed to meta-learn an attention generator across all episodes while training an individual ridge regressor for each episode.

5.2 Prompt-tuning

The gap between pre-train and downstream tasks often causes difficulty in fine-tuning of PLMs like GPT (Brown et al., 2020), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), etc. Inspired by GPT-3, which presents the strong potential of FSL by prompting, numerous work has applied prompt-tuning for Relation Classification (Han et al., 2021; Chen et al., 2021), Named Entity Recognition (Ma et al., 2021; Ding et al., 2021a), Topic Classification (Cui et al., 2022; Zhao et al., 2021), etc. As for the choice of template, some works applied continuous template (Li and Liang, 2021; Zhang et al., 2021; Liu et al., 2021) and some other works employed generative model like T5 (Raffel et al., 2020) to generate templates (Gao et al., 2021). Concerning the choice of label words, Chen et al. (2021) used learnable soft label words. Cui et al. (2022) applied contrastive loss to learn prototypical label words. Hu et al. (2022) explored external knowledge graphs to enrich semantic information.

6 Conclusion

In conclusion, we propose prompt-based meta-learning for few-shot text classification. Our method combines prompt-tuning with meta-learning closely, where the base learners tune task-specific label words (learn) and the meta-learner tunes the task-agnostic template and encoder (learning-to-learn). Experimental results demonstrate the excellent compatibility of meta-learning and prompt-tuning, with state-of-the-art performance in various few-shot text classification tasks and good robustness under noisy settings. Additionally, We show that the prompting mech-

anism helps **PBML** maintain good performance with significantly reduced meta-training data. The visualization results further demonstrate that meta-learning enables prompt-tuning to distinguish semantics better.

Limitations

We summarize the limitations of our method as follows: 1) We employ a simple average pooling to process the candidate label words for label word initialization. However, candidate words sometimes contain noise, and average pooling is vulnerable to extreme outliers, which may reduce essential information. In our experiment, we consider a fixed selection of 20 candidate words, but this choice is not always optimal because, for some categories, the obtained candidate words are less relevant to the class, thus presenting more noise. Even though the simple average pooling achieves good results, we think an attention generator is needed to calculate the contribution of different words dynamically; 2) Although our model outperforms previous baselines in various few-shot settings. The improvement of our method from 1-shot to 5-shot setting is not particularly large (about 3 percent while prototypical network can improve more than 10 percent). Therefore, we believe the model still has enough room for improvement in multi-shot scenarios.

References

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *Proceedings of the 8th International Conference on Learning Representations*. OpenReview.net.

Sam Brody, Sichao Wu, and Adrian Benton. 2021. Towards realistic few-shot relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5338–5345. Association for Computational Linguistics.

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

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. *CoRR*, abs/2104.07650.

Ganqu Cui, Shengding Hu, Ning Ding, Longtao Huang, and Zhiyuan Liu. 2022. Prototypical verbalizer for prompt-based few-shot tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7024. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021a. Prompt-learning for fine-grained entity typing. *CoRR*, abs/2108.10604.

Ning Ding, Xiaobin Wang, Yao Fu, Guangwei Xu, Rui Wang, Pengjun Xie, Ying Shen, Fei Huang, Hai-Tao Zheng, and Rui Zhang. 2021b. Prototypical representation learning for relation extraction. In *Proceedings of the 9th International Conference on Learning Representations*. OpenReview.net.

Bowen Dong, Yuan Yao, Ruobing Xie, Tianyu Gao, Xu Han, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2020. Meta-information guided meta-learning for few-shot relation classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1594–1605. International Committee on Computational Linguistics.

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

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 3816–3830. Association for Computational Linguistics.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019a. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 6407–6414. AAAI Press.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019b. Fewrel 2.0:

Towards more challenging few-shot relation classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 6249–6254. Association for Computational Linguistics.

Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2020. Dynamic memory induction networks for few-shot text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1087–1094. Association for Computational Linguistics.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3902–3911. Association for Computational Linguistics.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. PTR: prompt tuning with rules for text classification. *CoRR*, abs/2105.11259.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809. Association for Computational Linguistics.

Ruining He and Julian J. McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 507–517. ACM.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. 2022. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 2225–2240. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomás Mikolov. 2016. Fasttext.zip: Compressing text classification models. *CoRR*, abs/1612.03651.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille.

D. Lewis. 1997. Reuters-21578 text categorization test collection, distribution 1.0. *http://www.research/.att.com*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4582–4597. Association for Computational Linguistics.

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. 2017. Meta-sgd: Learning to learn quickly for few shot learning. *CoRR*, abs/1707.09835.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *CoRR*, abs/2103.10385.

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.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot NER. *CoRR*, abs/2109.13532.

R. Misra. 2018. News headlines dataset for sarcasm detection.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543. ACL.

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.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *Proceedings of the 5th International Conference on Learning Representations*. OpenReview.net.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 4077–4087.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 2895–2905. Association for Computational Linguistics.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, pages 3630–3638.

Yan Xiao, Yaochu Jin, and Kuangrong Hao. 2021. Adaptive prototypical networks with label words and joint representation learning for few-shot relation classification. *CoRR*, abs/2101.03526.

Shiyao Xu and Yang Xiang. 2021. Frog-gnn: Multi-perspective aggregation based graph neural network for few-shot text classification. *Expert Syst. Appl.*, 176:114795.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-level matching and aggregation network for few-shot relation classification. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 2872–2881. Association for Computational Linguistics.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable prompt makes pre-trained language models better few-shot learners. *CoRR*, abs/2108.13161.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

A Dataset Description

FewRel is a dataset for few-shot relation classification, containing 100 relations (Han et al., 2018). Each statement has an entity pair and is annotated with the corresponding relation. The position of the entity pair is given, and the goal is to predict the correct relation based on the context. The 100 relations are split into 64, 16, and 20 for training, validation, and test, respectively. We apply the same pre-process method as Soares et al. (2019) by adding special entity markers to highlight the position of the entity pair. Besides, FewRel provides each relationship with additional rich information (i.e., alias, descriptions, etc.), which we used for our label word initialization.

HuffPost headlines is a dataset for topic classification. It contains news headlines published on HuffPost between 2012 and 2018 (Misra, 2018). The 41 topics are split into 20, 5, 16 for training, validation and test respectively. These headlines are shorter and more colloquial texts.

Reuters-2157 is a dataset of Reuters articles over 31 classes (Lewis, 1997), which are split into 15, 5, 11 for training, validation and test respectively. These articles are longer and more grammatical texts.

Amazon product data contains customer reviews from 24 product categories (He and McAuley, 2016). Our goal is to predict the product category based on the content of the review. The 24 classes are split into 10, 5, 9 for training, validation and test respectively.

We summarize the details of the four datasets in Table 4. In this work, we focus on few-shot text classification under N-way K-shot setting. Table 3 is an example of a 5-way 3-shot topic classification.

Support set
Class 1 *SPORTS*:
Instance 1 *Everything you need to know about the 2022 NBA Finals.*
Instance 2 *Chelsea legend Didier Drogba speaks to CNN about club's plights.*
Instance 3 *International Skating Union proposes raising minimum age to 17 for competitions.*
Class 2 *POLITICS*: ...
Class 3 *TRAVEL*: ...
Class 4 *TECH*: ...
Class 5 *FOOD&DRINK*: ...
Query instance
Why Olympic figure skaters don't get dizzy.

Table 3: An example of 5-way 3-shot text classification. We omit instances for class 2-5 for simplicity. The given query instance belongs to class 1: *SPORTS*

B Baseline details

1-NN is a 1-nearest neighbor classifier under Euclidean distance. The encoder is implemented with IDF, which represents each statement as a weighted average of word embeddings from pre-trained fastText (Joulin et al., 2016). The weights are calculated based on inverse document frequency on the training set.

DS (Bao et al., 2020) is a meta-learning algorithm aiming to meta-train an attention generator based on distribution signatures (DS). **DS** considers two signatures: general word importance and class-specific word importance. The attention generator then uses these signatures to generate attention scores, serving as each word's weight for sentence representation.

MAML (Finn et al., 2017) is a classic meta-learning framework aiming to learn initialization for few-shot tasks. Dong et al. (2020) implement **MAML** for few-shot text classification, which meta-learns initialization parameters of a general multilayer perceptron and a BERT-based meta-encoder.

MIML (Dong et al., 2020) proposed a novel meta-learning framework, which uses class-aware semantic information from GloVe (Pennington et al., 2014) to provide strong guidance for meta-learning. **MIML** incorporates class names as meta-information to realize a class-aware initialization. The meta-initializer module is implemented via a fully connected layer.

Prototypical network (Snell et al., 2017) averages the embedding of support instances as class prototypes and makes a metric-based prediction for query instances. (Bao et al., 2020) use IDF to implement the instance encoder **PROTO-IDF**. We add the performance of **PROTO-BERT** which uses BERT as the encoder backbone.

C Template design

FewRel

$$x_{prompt} = x \texttt{ [SEP] } h \text{ is } \texttt{[MASK]} \text{ of } t$$

HuffPost

$$x_{prompt} = x \texttt{ [SEP] Topic is about [MASK]}$$

Reuters

$$x_{prompt} = \text{Article is about } \texttt{[MASK] [SEP]} \, x$$

Amazon

$$x_{prompt} = \text{Review is about } \texttt{[MASK] [SEP]} \, x$$

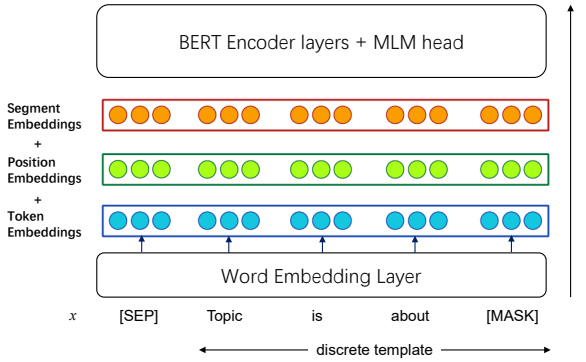


Figure 6: Discrete template tokens.

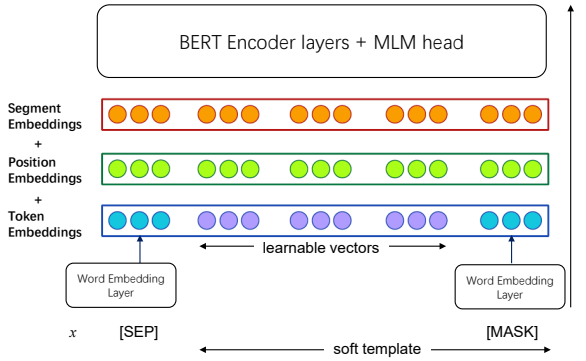


Figure 7: Soft template with learnable vectors.

As mentioned in subsection 3.1, we adopt a soft template strategy. Specifically, we use learnable vectors to replace discrete template tokens and use their word embeddings for vector initialization. We show the comparison comparison between the discrete template and the soft template in Figure 6 and Figure 7.

Our soft approach first maps discrete token-ids from the original text into word embeddings, then we concatenate the learnable vectors directly to these word embeddings. We note that we freeze the word embedding layer of $\mathcal{M}$ and only update soft template embeddings and parameters from encoder layers.

During our experiments, we found that if we update the word embedding layer of the meta-learner, the latter is very likely to overfit the label words of training episodes, showing lower meta-train loss and worse meta-test accuracy. We infer that this is because the number of categories in the dataset is far less than the number of samples. Since the label word embedding appears close to the meta-loss, the word embedding layer could easily overfit meta-training labels. The phenomenon suggests that our meta-learner will tend to memorize the matching relation between label words and training samples from the meta-training set and fails to understand the real meaning of the sentence, ending up with a poor performance on novel tasks. To avoid this, we decided to freeze the word embedding layer. The initialization of label word embeddings is always copied from the exact BERT-pre-trained embeddings (In subsection 3.2 , we call this a rough estimation of the class), then we let base-learners learn better label representations, allowing the meta-learner to only focus on the learning of encoder&soft-template for NLU ability.

D Algorithm of PBML

We present here the overall process of PBML in Algorithm 1.

E Implementation Details

We select $\text{BERT}_{\text{base}}$ (110M) (Devlin et al., 2019) as our meta-encoder and choose AdamW (Loshchilov and Hutter, 2019) for optimizing. Meanwhile, the warmup mechanism is used during meta-training. We implement PBML with PyTorch (Paszke et al., 2019). All the experiments run on one NVIDIA RTX 3090 .We report the hyper-parameters in Table 5. The model was meta-trained with 500-2000 iterations, depending on the dataset and the approximate per-iteration cost is 6-8 seconds. During meta-testing, the per-episode cost for 1-shot and 5-shot scenarios is approximately 180 and 540 milliseconds.

The choice of fast-tuning iteration steps T and task learning rate β_{task} greatly impacts the perfor-

Dataset	text length (avg.)	examples/cls	train cls	val cls	test cls
FewRel	24	700	64	16	20
HuffPost	11	900	20	5	16
Reuters	168	20	15	5	11
Amazon	140	1000	10	5	9

Table 4: Dataset statistics

Algorithm 1 Prompt-based Meta-learning

Require: $p(\mathcal{T})$: distribution over mete-training tasks
$\mathcal{T}_{test}$: meta-testing tasks
$\beta_{task}, \beta_{meta}$: learning rates
T: fast-tuning steps

Initialize meta-encoder θ_e, soft template embeddings θ_s
while not done **do**
  Sample batch of tasks $\mathcal{T}_b \sim p(\mathcal{T})$
  for all $\mathcal{T}_b$ **do**
    Initialize label word embeddings $\mathbf{W}_b^{(0)}$
    for $t = 0, ..., T-1$ **do**
      Evaluate $\mathcal{L}^{att}(\mathbf{W}_b^{(t)}, \mathcal{S}_b, \theta_e, \theta_s)$
      Update $\mathbf{W}_b^{(t)}$ using Equation 6
    end for
    Evaluate $\mathcal{L}(\mathbf{W}_b^{(T)}, \mathcal{Q}_b, \theta_e, \theta_s)$
  end for
  Update (θ_e,θ_s) using Equation 9
end while

mance. We choose these two hyper-parameters as follows: First we set (T, β_{task}) to (10, 5e-2). After meta-training, we modify the hyper-parameters pair (T, β_{task}) to search for more suitable values during meta-testing. Once we find better hyper-parameters (T^*, β^*_{task}), we perform meta-training again with (T^*, β^*_{task}).

We also report in Table 6, the corresponding validation performance for our model's test results. The FewRel testing set is not publicly available, so we visit their benchmark website[4] to get our test performance.

F Convergence speed

We plot the loss on FewRel training set and the 5-way 1-shot accuracy on FewRel validation set as functions of the number of iterations. As we can see from Figure 8, **PBML** training loss descends very fast while the validation accuracy grows to 93% within only hundreds of steps. In contrast, the convergence of **MIML** (state-of-the-art meta-learning baseline) is much slower, and the performance upper bound is also limited.

[4]https://thunlp.github.io/fewrel.html

Datasets	FewRel	HuffPost	Reuters	Amazon
Train iters	1000	1000	500	2000
Batch size	32	32	4	16
Encoder lr	5e-5	1e-5	2e-5	3e-6
Task lr	1e-2	1e-3	5e-3	1e-2
FT-steps	30	50	20	20

Table 5: Hyper-parameters of PBML architecture

Figure 8: Convergence speed of **PBML** and **MIML**

G Fast-tuning deployment

During meta-testing, **PBML** uses support instances to iteratively fast-tune label word embeddings, which corresponds to the task-adaptation process of the meta-learning algorithm. This means that our method still requires some "training" during the testing phase. However, executing any loops during inference is not recommended in real-world neural network deployment. Fortunately, our fast-tuning process only updates N label word embeddings while leaving the encoder unchanged. Therefore, we suggest manually computing the gradients of the proposed contrastive loss (Equation 3 and Equation 4) and using T identical layers to replace the

Evaluation set	FewRel		HuffPost		Reuters		Amazon	
	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
Validation	94.1	95.2	67.4	76.3	90.1	93.6	75.2	81.7
Test	96.6	97.4	74.9	78.0	96.4	97.6	81.8	88.0

Table 6: 5-way 1-shot and 5-way 5-shot text classification accuracies (%) on four benchmark test sets and the corresponding validation performance.

For Loop of T gradient steps.

$$\nabla_{\mathbf{w}_i^{(t)}} \mathcal{L}_{s2w}^{att}(\mathcal{S}, \mathbf{W}^{(t)}) \tag{10}$$
$$= \frac{1}{|\mathcal{S}|} \sum_{\substack{1 \le i' \le N \\ 1 \le j \le K}} \alpha_{i'}^{j} \left(\frac{\exp\left(\mathbf{h}_{i'}^{j} \cdot \mathbf{w}_i^{(t)}\right)}{\sum_{\mathbf{w} \in \mathbf{W}^{(t)}} \exp\left(\mathbf{h}_{i'}^{j} \cdot \mathbf{w}\right)} - \delta(i, i') \right) \mathbf{h}_{i'}^{j}$$

$$\nabla_{\mathbf{w}_i^{(t)}} \mathcal{L}_{w2s}^{att}(\mathcal{S}, \mathbf{W}^{(t)}) \tag{11}$$
$$= \frac{1}{N} \sum_{i'=1}^{N} \left(\frac{\exp\left(\mathbf{w}_i^{(t)} \cdot \mathbf{h}_{i'}\right)}{\sum_{i''=1}^{N} \exp\left(\mathbf{w}_i^{(t)} \cdot \mathbf{h}_{i''}\right)} - \delta(i, i') \right) \mathbf{h}_{i'}$$

$$\mathbf{h}_i = \sum_{j=1}^{K} \alpha_i^j \mathbf{h}_i^j$$

The similarity computation of all $(\mathbf{h}, \mathbf{w})$ pairs is of $O(N^2 \times K \times D)$ time complexity. Then we use the similarity table to compute gradients $\nabla_{\mathbf{w}_i^{(t)}}$ for each class i and the overall computation cost is also of $O(N^2 \times K \times D)$.

We can define the gradient descent layer L_{gd} as follows:

$$\mathbf{W}^{(t)} \mapsto \mathbf{W}^{(t)} - \beta_{task} \nabla_{\mathbf{W}^{(t)}} \mathcal{L}^{att}(\mathbf{W}^{(t)}, \mathbf{h}) \tag{12}$$

The initialized label word embeddings $\mathbf{W}^{(0)}$ will pass through T identical L_{gd} layers consecutively to obtain $\mathbf{W}^{(T)}$

H Discussion on the N-way effect

We also discuss the effects on the number of categories. Table 7 compares different models under 10-way 1-shot and 10-way 5-shot settings. We draw similar conclusions from this table as in Table 1 that our method still outperforms previous state-of-the-art baselines.

Method	FewRel		HuffPost		Reuters		Amazon		Average	
	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
PROTO-BERT	81.9	90.1	39.9	59.1	84.2	92.0	58.9	75.8	66.2	79.3
BERT-PAIR	80.6	87.0	41.2	63.6	85.5	93.3	61.7	77.4	67.3	80.3
MAML	78.9	89.1	40.3	57.1	83.5	93.7	61.3	75.4	66.0	78.8
MIML	87.5	93.2	46.5	57.3	83.2	96.4	60.1	77.6	69.3	81.1
Prompt-Tuning	91.4	93.5	60.7	67.0	90.3	94.3	68.1	75.2	77.6	82.5
Human	85.9	-	-	-	-	-	-	-	-	-
MTB $\star$	89.2	94.3	-	-	-	-	-	-	-	-
PBML (OUR)	**93.2**	**94.5**	**64.6**	**68.6**	**93.7**	**96.1**	**70.8**	**79.9**	**80.6**	**84.8**

Table 7: Results of 10-way 1-shot and 10-way 5-shot text classification accuracies (%) on four benchmark test sets. The test set of Amazon product data only contains nine categories. Thus we test the 9-way 1-shot and 9-way 5-shot performance instead.