Zero-shot Cross-lingual Transfer of Prompt-based Tuning with a Unified Multilingual Prompt

Lianzhe Huang†,* Shuming Ma‡, Dongdong Zhang‡, Furu Wei‡ and Houfeng Wang†
†MOE Key Lab of Computational Linguistics, Peking University
‡Microsoft Research Asia
{hlz, wanghf}@pku.edu.cn
{shumma, dozhang, fuwei}@microsoft.com

Abstract

Prompt-based tuning has been proven effective for pretrained language models (PLMs). While most of the existing work focuses on the monolingual prompts, we study the multilingual prompts for multilingual PLMs, especially in the zero-shot cross-lingual setting. To alleviate the effort of designing different prompts for multiple languages, we propose a novel model that uses a unified prompt for all languages, called UniPrompt. Different from the discrete prompts and soft prompts, the unified prompt is model-based and language-agnostic. Specifically, the unified prompt is initialized by a multilingual PLM to produce language-independent representation, after which is fused with the text input. During inference, the prompts can be pre-computed so that no extra computation cost is needed. To collocate with the unified prompt, we propose a new initialization method for the target label word to further improve the model's transferability across languages. Extensive experiments show that our proposed methods can significantly outperform the strong baselines across different languages. We release data and code to facilitate future research[1].

1 Introduction

Pre-trained language models (PLMs) have been proven to be successful in various downstream tasks (Devlin et al., 2019; Yang et al., 2019; Conneau et al., 2020). Prompt-tuning is one of the effective ways to induce knowledge from PLMs to improve downstream task performance especially when the labeled data is not sufficient (Brown et al., 2020; Gao et al., 2021; Le Scao and Rush, 2021; Zhao and Schütze, 2021). The essence of prompt-tuning is to precisely design task input structure so that it can imitate the pre-training procedure of

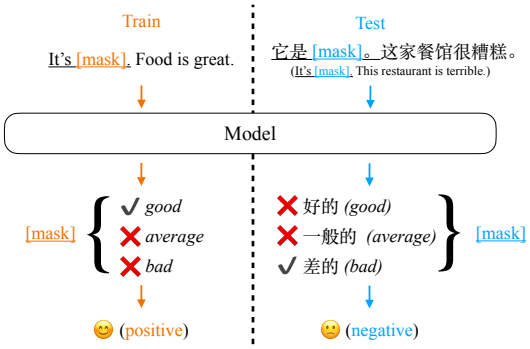

Figure 1: An example of zero-shot cross-lingual transfer of prompt-based tuning. The underline part of the input with `[mask]` token is template.

PLMs and better induce the knowledge from them. For example, to classify the sentiment polarity of the source sentence "Food is great", a template "It's [mask]." is constructed before the source input where the accurate label is masked. In this way, the sentiment-related words like 'good', 'bad', and 'average' is predicted at the masked position with probabilities on the target side, over which a verbalizer is leveraged to project the final sentiment labels.

Previously, most work on prompt-based tuning (Gao et al., 2021; Zhang et al., 2021) mainly considers the monolingual prompts. However, it is not straightforward when applying to multilingual tasks due to absent multilingual prompts that heavily rely on native language experts to design both the templates and label words. An alternative way to build multilingual prompts is to machine-translate source prompts into target languages. But it is still infeasible for low-resource languages as the translation quality is hard to be guaranteed. Other work also considers using soft prompts that consist of continuous vectors. Although it reduces the cost of building prompts for multiple languages, the mismatch between the pro-

*Contribution during internship at Microsoft.
[1]The data and the code of this paper are available at https://github.com/mojave-pku/UniPrompt.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 11488 - 11497
December 7-11, 2022

cedures of pre-training and prompt-tuning brings many obstacles to the desired tasks, because the soft prompts never occur in the model pre-training stage.

In this work, we focus on the zero-shot cross-lingual transfer of prompt-based tuning. As shown in Figure 1, the model is trained on the source language (English), while tested on the other language (Chinese). We explore the approaches to use a unified multilingual prompt that can transfer across languages. We propose a novel model, called UniPrompt, which takes the merits of both discrete prompts and soft prompts. UniPrompt is model-based and language-independent. It is initialized by a multilingual PLM that takes English prompts as input and produces language-agnostic representation benefit from the transferability of multilingual PLMs. During inference, the prompts can be pre-computed so that no extra computation cost is introduced. In this way, we can alleviate the effects of prompt engineering for different languages, while reserving the ability of PLMs. To better collocate with the unified prompt, we propose a new initialization method for the label words instead of using the language model head from the PLM. This proves to further improve the model's transferability across languages.

We conducted extensive experiments on 5 target languages with different scales of data. Experimental results prove that UniPrompt can significantly outperform the strong baselines across different settings. We summarize the contributions of this paper as follows:

- We propose a unified prompt for zero-shot cross-lingual language understanding, which is language-independent and reserve the ability of multilingual PLMs.
- We propose a novel label word initialization method to improve the transferability of prompts across languages.
- We conduct experiments in 5 languages to prove the effectiveness of the model, and design a detailed ablation experiment to analyze the role of each module.

2 UniPrompt

2.1 Overview

The major differences between UniPrompt and the existing prompt-based methods mainly lie in two parts: **template representation** and **label word initialization**.

For **template**, we use two independent encoder towers, which are the template tower and the context tower. The template tower is to encode the prompt's template, while the context tower is for the origin text input. Both towers are initialized by the bottom layers of the multilingual PLM. After that, the representations of the template and context are concatenated as the input of the fusion tower. The fusion tower is initialized by the top layers of multilingual PLMs. This is motivated by the previous studies (Sabet et al., 2020), which found that the lower layers of the pre-trained language model are related to language transfer, while the higher layers are related to the actual semantics. Therefore, it can get rid of the dependency of the template on the specific language, but also retain the ability of prompts to activate the potential knowledge of PLMs. Since the output of the prompt tower can be pre-computed before inference, the model will not introduce additional parameters or computation costs in the inference stage.

For **label words**, we use artificial tokens so that it is language-agnostic. Previous studies also have explored methods to use artificial tokens in label words (Hambardzumyan et al., 2021). Different from these works, we have a novel initialization method for the label words. Specifically, we minimize the distance between the label words and the sentence embeddings before fine-tuning. This is achieved by taking a simple average of the sentence embeddings in the same class as the label words. In this way, the label words not only have a good starting point but also are language-independent.

2.2 Two-tower Prompt Encoder

As a cross-lingual unified prompt, if it directly uses the existing tokens from the vocabulary, it will be biased towards some specific languages, which will harm the cross-lingual transfer due to the gap between languages. To alleviate this problem, the first goal of designing a template in this task is: *the template must not depend on any specific language*. An intuitive idea to achieve this goal is to use soft prompt, which is artificial tokens that have nothing to do with specific languages. However, these artificial tokens: *i)* will not be adequately trained due to little amount of data in few-shot scenarios; *ii)* do not appear in the pre-training stage. Therefore, the goal of the prompt, which is to activate the potential

	En	De	Es	Fr	Ja	Zh
Average characters per review	178.8	207.9	151.3	159.4	101.4	51.0
Number of reviews for training/development	$k \times 5$	-	-	-	-	-
Number of reviews for testing	-	5,000	5,000	5,000	5,000	5,000

Table 1: Statistics of MARC data used in our paper. k is the number of training samples per class.

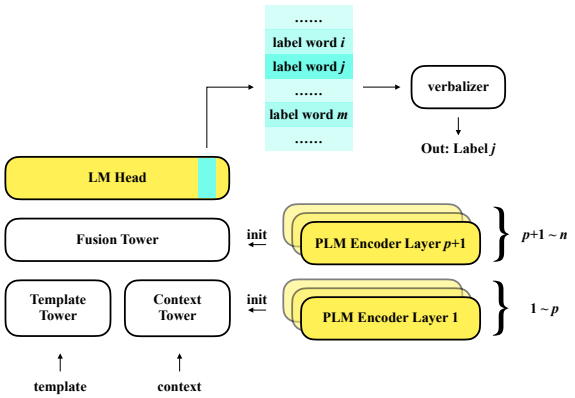

Figure 2: Overview of Two-tower Prompt Encoder. Template Tower and Context Tower are both initialized by $1 \sim p$ encoder layers of PLMs. Fusion Tower is initialized by $(p+1) \sim n$ encoder layers of PLMs. n means the total number of the PLMs encoder layers.

knowledge of PLMs, may not be achieved. Given the problems of soft prompt, the second goal of designing templates can be drawn: *to minimize the gaps between the pre-training and prompt-tuning.*

To achieve these goals, we now describe our method to model the prompts, called two-tower prompt encoder. The overview of the two-tower prompt encoder is shown in Figure 2. According to the previous work, the bottom layers of PLMs encode the information related to specific language tokens/grammar, while the top layers of PLMs model the semantic information. Therefore, we duplicate the bottom p layers of PLM encoders as two independent encoder towers to encode the template and context respectively. Formally, we can define them as:

$$H'_t = \text{TemplateTower}(X_t) \quad (1)$$
$$H'_s = \text{ContextTower}(X_s) \quad (2)$$

where X_t, X_s are embeddings of template and context.

Then we concatenate the outputs of the two encoders as the input of the fusion tower which is

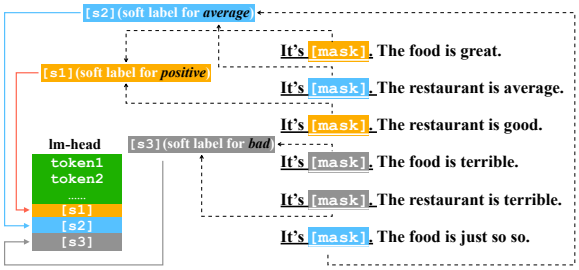

Figure 3: An example of our proposed soft label words initialization. We first encode sentences with prompts by the original PLM, and then use the average of the representations of `[mask]` with the same label as the initialization of the soft label word.

initialized with the top $n - p$ layers of PLM:

$$H' = [H'_t; H'_s] \quad (3)$$
$$H = \text{FusionTower}(H') \quad (4)$$

where n means the total number of the encoder layers and $[;]$ means the splicing operation. With the help of the multilingual PLM, the template tower can make the template easy to transfer across languages.

2.3 Initialization of Soft Label Words

With the two-tower prompt encoder, we are able to make the template more language-agnostic. As for label words, if we use the real tokens, they should correspond to some specific languages, which are difficult to transfer. Therefore, we use soft label words, i.e. artificial tokens, to achieve the goal of language independence.

To further reduce the gaps between the pre-training and fine-tuning of soft label words, we propose a novel initialization of the label words which is shown in Algorithm 1. And we also bring an example for this algorithm in Figure 3. If we regard the output projection matrix as the word embeddings of label words, the objective of fine-tuning is to minimize the distance between the encoder outputs and the corresponding label word embedding. Therefore, if the label word embeddings have already been close to the encoder outputs, it will be a good starting point for the models. Motivated by

Algorithm 1 Initialization of Soft Label Words

Input: original pre-trained language model $\boldsymbol{\theta}^0$, all training cases C_i with label i, prompt with `[mask]` token p
for each case c_j in C_i **do**
  form the prompt input c'_j for encoding: $c'_j \leftarrow p + c_j$
  encode the sequence c'_j without gradients: $H_j \leftarrow \boldsymbol{\theta}^0(c'_j)$
  get the representation of `[mask]` token h^m_j from H_j
end for
average all the h^m as the representation x_i of the soft label word for label i: $x_i \leftarrow \mathrm{Avg}(h^m_j), j \in C_i$
return x_i

this, we propose to compute the encoder outputs of all training samples, group them according to their labels, and then take a simple average of all encoder outputs in each group to initialize the label words. Note that for few-shot learning, the computation cost of pre-computing encoder outputs is small. In this way, the models will have good priors to the downstream tasks while reserving the knowledge from the PLMs.

Formally, we construct soft label word L_i for each label i, and group the training samples into C_i according to their labels. Then, we concatenate the training examples with the corresponding templates to compute the encoder outputs. We take the average of the `[mask]` representations h^m in the encoder outputs in each group to initialize the label words. The embedding x_i of the label word L_i can be defined as:

$$x_i = \mathrm{Avg}(h^m_c), c \in C_i \tag{5}$$

where Avg means average pooling, C_i is the set containing the training cases with label i.

2.4 Training

Similar to the previous prompt-based tuning method, we use the distribution probability of label words for classification:

$$\mathrm{logit}_{\mathrm{y}} = \frac{\exp(\mathbf{W}^{\mathbf{y}}_{\mathbf{lh}} h^m)}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{W}^{\mathbf{y}'}_{\mathbf{lh}} h^m)} \tag{6}$$

where $\mathcal{Y}$ is the set of all labels, $\mathbf{W}^{\mathbf{i}}_{\mathbf{lh}}$ is the parameters corresponding to the label i from the output projection matrix (i.e. label word embeddings).

Algorithm 2 Overall Workflow of UniPrompt

Input: pre-trained language model $\boldsymbol{\theta}^0$, prompt p, cases c with label
for each label i **do**
  group all the cases with label i as C_i
  initialize soft label word $x_i \leftarrow$ Algorithm 1 $(\boldsymbol{\theta}^0, p, C_i)$
end for
for each training case c_j **do**
  prompt and c_j are sent into the two-tower prompt encoder for encoding, respectively: $H^j_t \leftarrow \mathrm{TemplateTower}(p)$, $H^j_c \leftarrow \mathrm{ContextTower}(c_j)$
  The two vectors $H^{j\prime}_t, H^{j\prime}_t$ are stitched together and fed into the fusion tower to get the final representation $H^j \leftarrow \mathrm{FusionTower}(H^{j\prime}_t; H^{j\prime}_t)$
  get the label by the prediction result of maskLM task : $y \leftarrow \mathrm{maskLM}(h^m_j)$
end for

The loss function $\mathcal{L}$ in our model is the cross-entropy loss, which can be defined as:

$$\mathcal{L} = -g_y \log(\mathrm{logit}_{\mathrm{y}}), \tag{7}$$

where g_y is the "one-hot vector" of gold labels.

The overall workflow of our proposed UniPrompt is shown in Algorithm 2.

3 Experiments

3.1 Datasets

We choose the Multilingual Amazon Reviews Corpus (MARC)[2] (Keung et al., 2020) for experiments, which is a large-scale multilingual text classification dataset with the licence provided by Amazon[3]. The MARC dataset is available in 6 languages including English, German, French, Spanish, Japanese, and Chinese. The goal of this dataset is to predict the star rating given by the reviewer to the product based on their product reviews (from 1 to 5 stars, the higher the star rating, the more satisfied they are).

In the MARC dataset, the number of samples in each category is exactly the same, and we follow their settings to take the same samples for each category to form few-shot training and development

[2]https://registry.opendata.aws/amazon-reviews-ml/
[3]https://github.com/awslabs/open-data-docs/blob/main/docs/amazon-reviews-ml/license.txt

k	Model	De	Es	Fr	Ja	Zh	Average
4	Vanilla Finetune	27.94 ± 8.38	26.80 ± 5.96	27.38 ± 6.48	25.86 ± 5.08	25.95 ± 6.51	26.79
	Translation Prompt	30.76 ± 4.48	31.70 ± 3.46	26.77 ± 3.49	27.46 ± 4.52	21.56 ± 2.30	27.65
	English Prompt	33.52 ± 5.14	33.01 ± 3.61	33.32 ± 3.54	32.23 ± 2.25	30.97 ± 4.87	**32.61**
	Soft Prompt	29.26 ± 9.28	30.74 ± 3.22	31.10 ± 5.72	28.96 ± 0.76	28.26 ± 3.56	29.66
	UniPrompt	31.70 ± 5.42	30.79 ± 5.73	30.97 ± 6.29	30.21 ± 6.37	28.70 ± 4.40	30.47
8	Vanilla Finetune	31.95 ± 7.07	29.74 ± 7.38	31.79 ± 6.41	28.70 ± 5.30	29.67 ± 7.25	30.37
	Translation Prompt	33.04 ± 1.04	35.22 ± 1.16	28.93 ± 1.51	30.10 ± 2.38	23.62 ± 2.80	30.18
	English Prompt	36.96 ± 1.94	36.12 ± 0.98	36.70 ± 1.38	33.95 ± 4.21	32.97 ± 2.71	35.34
	Soft Prompt	33.30 ± 3.22	32.88 ± 2.34	33.28 ± 2.44	30.05 ± 3.87	29.46 ± 4.22	31.79
	UniPrompt	38.58 ± 2.96	37.68 ± 3.38	37.88 ± 3.80	35.72 ± 5.60	34.57 ± 5.69	**36.89**
16	Vanilla Finetune	40.10 ± 5.46	38.37 ± 3.55	38.73 ± 3.59	36.20 ± 4.82	35.94 ± 5.74	37.87
	Translation Prompt	36.62 ± 1.72	36.88 ± 1.14	32.48 ± 2.90	31.98 ± 1.42	24.20 ± 2.14	32.43
	English Prompt	39.13 ± 3.63	37.84 ± 3.34	38.58 ± 1.90	35.90 ± 4.44	35.05 ± 5.59	37.30
	Soft Prompt	37.25 ± 3.57	34.96 ± 2.74	35.18 ± 3.20	34.64 ± 2.76	34.20 ± 4.44	35.24
	UniPrompt	43.53 ± 5.11	41.43 ± 4.39	41.71 ± 5.21	39.55 ± 4.41	38.62 ± 2.82	**40.97**
32	Vanilla Finetune	45.48 ± 2.74	42.09 ± 4.21	43.14 ± 2.42	40.86 ± 4.74	41.39 ± 1.61	42.59
	Translation Prompt	39.29 ± 3.25	38.46 ± 1.96	34.75 ± 4.23	34.76 ± 2.52	26.88 ± 5.64	34.83
	English Prompt	42.04 ± 2.32	40.39 ± 1.51	41.33 ± 2.39	39.06 ± 2.34	37.72 ± 3.60	40.11
	Soft Prompt	40.58 ± 1.48	38.29 ± 2.05	39.50 ± 2.28	38.80 ± 1.90	35.90 ± 4.94	38.61
	UniPrompt	49.29 ± 2.11	46.74 ± 1.28	47.47 ± 1.03	45.94 ± 1.98	43.62 ± 1.16	**46.61**
64	Vanilla Finetune	49.85 ± 2.73	45.74 ± 3.18	47.72 ± 2.76	44.68 ± 3.80	43.84 ± 1.76	46.37
	Translation Prompt	41.70 ± 3.72	40.93 ± 1.85	37.24 ± 3.54	36.75 ± 1.03	29.72 ± 2.06	37.27
	English Prompt	45.63 ± 4.25	43.41 ± 4.33	44.18 ± 3.30	41.43 ± 3.65	39.63 ± 2.13	42.86
	Soft Prompt	44.49 ± 2.97	40.73 ± 3.35	42.15 ± 3.79	41.91 ± 3.09	40.36 ± 4.36	41.93
	UniPrompt	51.75 ± 1.57	47.64 ± 1.70	49.54 ± 0.98	45.99 ± 1.75	45.55 ± 3.13	**48.09**
128	Vanilla Finetune	51.84 ± 1.82	49.09 ± 1.17	49.52 ± 1.20	47.74 ± 3.12	45.81 ± 1.73	48.80
	Translation Prompt	42.99 ± 2.99	41.16 ± 1.52	36.78 ± 1.58	37.12 ± 1.92	29.46 ± 2.96	37.50
	English Prompt	49.36 ± 2.96	45.70 ± 2.18	45.98 ± 3.42	44.12 ± 2.84	43.97 ± 1.31	45.83
	Soft Prompt	48.06 ± 1.60	42.64 ± 3.94	43.41 ± 4.01	45.22 ± 1.64	44.19 ± 1.43	44.70
	UniPrompt	53.18 ± 1.28	49.74 ± 0.98	50.22 ± 0.68	48.48 ± 1.30	46.47 ± 1.31	**49.62**
256	Vanilla Finetune	53.06 ± 1.24	50.05 ± 1.57	50.47 ± 1.33	47.93 ± 3.63	46.40 ± 2.72	49.58
	Translation Prompt	46.44 ± 1.04	43.06 ± 1.30	38.76 ± 1.44	38.49 ± 1.79	29.40 ± 3.62	39.23
	English Prompt	51.66 ± 0.60	47.90 ± 2.52	48.18 ± 2.32	47.32 ± 1.70	44.77 ± 1.27	47.97
	Soft Prompt	50.81 ± 0.83	44.92 ± 1.76	45.29 ± 1.97	47.36 ± 1.56	44.24 ± 2.64	46.52
	UniPrompt	54.36 ± 1.16	51.13 ± 0.83	51.56 ± 0.86	49.66 ± 1.64	47.57 ± 1.29	**50.86**

Table 2: Main results. k is the number of training samples per class (i.e. k-shot).

sets. In our experiments, we randomly sample k cases from each category, $k \times 5$ cases in total, to form new training and development sets, and the test set remains unchanged. An overview of the dataset is shown in Table 1, and some statistics are directly taken from Keung et al. (2020). Our source language is English, which is the language used for the training and development sets. The target languages, which include the remaining 5 languages, are used for the test set.

The task and dataset we used are representative and challenging. Text classification is one of the fundamental problems for NLP. It also proves to be a good test bed for few-shot learning according to the previous work. The MARC dataset used in this work is challenging, especially for the multilingual few-shot scenarios. According to our experiments, vanilla fine-tuning only gets an average accuracy of 26.79 in the 4-shot setting. Therefore, we believe the benchmark is sound.

3.2 Experimental Setup

Our method is based on `XLM-RoBERTa-base` model (Conneau et al., 2020), which is a widely used multilingual pretrained language model. We implement our model with HuggingFace `Transformers` (Wolf et al., 2020) and code released by Gao et al. (2021). We optimize our models with a learning rate of 1e-5. The batch size is set to 8. We train each model for 1000 steps and

evaluate per 100 steps, the best checkpoint is used for the final prediction. The number of layers used for prompt and context towers is set to 9. The max sequence length of the model is set to 512. For each experiment reported in the paper, we use 5 different random seeds to sample 5 different few-shot training/development dataset from the original one. We run the model with the same random seeds as the one for dataset sampling and report the average results.

We also want to introduce the number of trainable parameters. Since we do not freeze the parameters during training, the trainable parameter number of the baseline is the total parameter number of `XLM-RoBERTa-base`. During the training of our model, there is an additional number of parameters from the template tower. The specific number is related to the number of layers (L) of the template tower, which will bring an additional parameter number of $L * p$, where p is the number of parameters for each layer. During inference, our model uses a template tower output cache, and the number of parameters is the same as the baseline.

3.3 Baselines

We compare the model with the following baseline models, all parameters in the baseline models are not frozen:

Vanilla Finetune add a task-dependent linear layer after the pretrained language model for classification (Devlin et al., 2019).

Translation Prompt proposed by Zhao and Schütze (2021), which uses the source language prompt for training and translates the prompt into the target language by machine translation model for testing.

English Prompt proposed by Lin et al. (2021), which trains and tests by prompts in the source language (English).

Soft Prompt uses artificial tokens instead of discrete tokens as templates, the label words are still in the source language.

The baseline models above are all implemented by us on the same codebase, initialized by the same pre-trained language model, and the hyper-parameters, including `max_steps`, `eval_steps`, `batch_size`, `learning_rate`, `max_seq_length`, and so on, are consistent. All experiments are performed on the same computing cluster with the same docker image.

3.4 Main Results

The main experimental results are shown in Table 2. As can be seen from the experimental results, our model outperforms all the listed baseline models at all data scales except slightly lower than **English Prompt** in the case of a very small amount of data ($k = 4$).

From the perspective of data scales, our model performs very well on medium data sizes ($k = 16, 32, 64$), with an average 2% higher accuracy than the strongest baseline. Especially when $k = 32$, the accuracy is more than 4% higher than the strongest baseline, which fully proves the ability of our model in few-shot cross-lingual transfer. As the size of the data continues to increase, the model leads by a smaller margin. But even if the data scale reaches $k = 256$, the accuracy of our model is still at least 1% higher than all other baselines.

Next, we discuss the comparison with each baseline separately. First, our model outperforms **Vanilla Finetune** models on all languages and data scales. We believe the reasons for the worse performance of **Vanilla Finetune** include: *i)* in the vanilla fine-tune model, a task-related linear layer is added on the top of PLMs. This layer is randomly initialized and requires more training data to be fully trained, which results in failure on low-resource tasks. *ii)* failing to exploit the latent knowledge in large-scale unlabeled corpus like prompt.

Second, our model also performs better than the **Translation Prompt** model (Zhao and Schütze, 2021). Converting the prompt directly using a machine translation model is indeed an intuitive and less expensive method. But there are also some problems. *i)* the model will be limited by the machine translation model, potentially causing error propagation. *ii)* since the translated prompt model has never been seen during training, the model cannot be properly fine-tuned according to the dataset situation, which may also lead to performance loss.

Next, we discuss **English Prompt**, which directly use the prompt from the source language. The **English Prompts** fits the training data which is also in English, so when the data scale is very small ($k = 4$), this method achieves the best results. But as the amount of data gets slightly larger ($k = 8$, which is still a very small scale), the performance of **English Prompt** is not as good as **UniPrompt**. The key point of the task in this paper is to enhance the cross-language transfer ability of the model. Since the PLMs are not trained by cross-

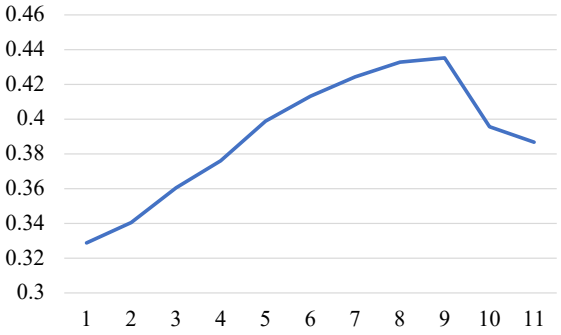

Figure 4: The effect of different number of template tower and context tower's layers. Except for the number of layers, the other settings are the same as the main experiment.

language splicing texts in the pre-training phase, so when an English prompt is combined with the context in another language in the testing phase of cross-lingual transfer, there will be gaps with the pre-training phase, which results in performance loss.

Finally, our model also has better performance than **Soft Prompt**. Although the soft prompt has nothing to do with the specific language and is consistent during training and testing. But *i)* it has not appeared in the pre-training stage, so it may be difficult to activate the latent knowledge in the pre-training stage. And *ii)* in low-resource scenarios, the completely randomly initialized soft prompt cannot be fully trained.

We note that the standard deviation of the experimental results is relatively large due to the small scale data of few-shot settings, which may lead to confusion about whether the performance gain is significant, especially the performance difference between the vanilla fine-tune and our model on some data scales. For this we selected the original experimental results of vanilla fine-tune and our UniPrompt in all 5 languages (de/es/fr/ja/zh) when $k = 16$ for the statistical test, the results indicate that the performance difference between our method and vanilla fine-tune is statistically significant.

4 Analysis

In this section, we will analyze the model in detail to verify the effectiveness of each module of the model. All experimental setups in this section are identical to the main experiments unless otherwise stated, and all experiments are based on 16-shot data.

4.1 Discussion on Two-tower Prompt Encoder

We first discuss the Two-tower Prompt Encoder. According to its setting, we discuss the effects of the number of layers and the pretrained models on model performance separately.

4.1.1 Number of Layers

As discussed above, the reason why the Two-tower Prompt Encoder works is that it splits the bottom encoders of PLMs, which are considered syntax-related, into two separate encoder towers, making the prompt free from language-specific dependencies. At the same time, when entering the top encoder of PLMs, which is considered to be semantically related, the two representations are fused, thereby stimulating the potential knowledge of PLMs in the pre-training stage.

A key question about the Two-tower Prompt Encoder is the **dividing line** between the top and bottom layers of PLMs. In response to this, we conducted experiments on the `en->de` data, and the experimental results are shown in the Figure 4. From the experimental results, we can see that the **dividing line** we expect is at 9 out of 12 layers, about 75% of the encoder layers of the PLMs. Before the dividing line, as the number of the independent lower and syntax-related encoder layers increases, the effect of decoupling the prompt from the specific language becomes better. Therefore, the performance of the model is gradually improved. After the dividing line, with the increase of the number of independent layers, although the ability to decouple with specific languages becomes stronger, the number of layers left for the fusion of template and context becomes less, and too little fusion limits the capacity prompt for activating the latent knowledge during the pre-training phase, so model performance gradually decreases.

4.1.2 Pretrained Models

Another point worth discussing about the Two-tower Prompt Encoder is the pretrained language models for template tower initialization. In our experiments, the template tower is initialized from the corresponding layers of the encoders of multilingual PLMs like the context/fusion tower. Therefore, we analyze whether the improvement is brought by the transferability from the multilingual PLM. Since in the original model, the prompt is initialized with the source language, that is, the English prompt. An intuitive idea is to use the English monolingual PLMs for template tower initializa-

Initialization	De	Es	Fr	Ja	Zh	Average
Random Initialization	40.24 ± 6.48	39.60 ± 3.32	40.25 ± 5.45	37.80 ± 4.34	37.51 ± 3.71	39.08
RoBERTa-base	38.96 ± 4.56	37.85 ± 3.75	37.90 ± 2.94	36.14 ± 3.10	36.05 ± 5.13	37.38
XLM-RoBERTa-base	43.53 ± 5.11	41.43 ± 4.39	41.71 ± 5.21	39.55 ± 4.41	38.62 ± 2.82	40.97

Table 3: Results of analysis on the template tower initialization.

Experiments	(1)	(2)	(3)	(4)
Actual Label	Label Words			
😊	good	[s1]		
😐	average	[s2]		
☹	bad	[s3]		
Init. By	`lm-head`	random	corresponding word in `lm-head`	dynamic

Table 4: The comparison of the settings of ablation study.

tion. Compared with multilingual PLMs, English PLMs do not have the ability of cross-lingual transfer, which helps us ablate the effects of transferability. In addition, we also compare with random initialization, which should not benefit from the PLMs. The results are shown in Table 3.

From the experimental results, it can be seen that using multilingual PLMs, which have cross-lingual transferability, achieve the best results. And there is a notable phenomenon that even if the template is based on the source language, initializing the template tower by the monolingual PLM RoBERTa (Liu et al., 2019) performs worse than random initialization. This indicates that the cross-lingual transferability is much more important than the effect of the PLMs itself for our method.

4.2 Discussion on Label Words

We also conduct the ablation study to verify the effect of our soft label words initialization method. The comparison of the settings of the 4 groups of experiments is shown in Table 4. The experimental results are shown in Table 5. First, we compare soft label words with discrete label words. The experimental results show that removing soft labels leads to a significant drop in performance. This illustrates the necessity of decoupling label words from specific languages in cross-lingual tasks. There will be gaps in the pre-training stage by using discrete tokens that depend on specific languages as label words during cross-language transfer, and the ability of prompts to activate knowledge in the pre-training stage will be weakened, resulting in performance loss.

Then, we compare the models with and without initialization. It shows that the initialization of label words results in a significant gain, proving our motivation that initialization is important for label words. We also compare our initialization method with the original PLM initialization. Compared (3) with (4), it proves that our initialization method is effective to improve cross-lingual performance. This is because our initialization method for the label words can reduce the gaps between the pre-training and the fine-tuning.

5 Related Work

5.1 Prompt-based tuning

The proposal of GPT-3 inspired the research on prompt (Brown et al., 2020). The key to prompt tuning is to reasonably imitate the pre-training process of the PLM model, so as to maximize the implicit knowledge learned by the model from the large-scale unlabeled corpus. Most of the existing template-based research focuses on how to design or search for the best template suitable for downstream tasks (Le Scao and Rush, 2021; Zhang et al., 2021; Li and Liang, 2021), but does not focus on optimization from aspects such as model parameters or structure. As for label words, almost all models are still using discrete tokens as label words. Hambardzumyan et al. (2021) proposed to use artificial tokens as label words, but they used randomly initialized label words and did not consider finding a better initialization for them, which may bring performance loss.

5.2 Zero-shot Cross-lingual Transfer

At a time when labeling resources are expensive, the research on Zero-shot Cross-lingual Text Classification is quite valuable. Past research in this area is usually based on cross-task transfer learning, that is, the model is first trained on a dataset of resource-rich tasks, and then fine-tuned on specific low-resource downstream tasks (Pruksachatkun et al., 2020; Zhao et al., 2021). As research on prompts

Label Words	De	Es	Fr	Ja	Zh	Average
(1) Discrete Labels	38.94 ± 2.46	37.84 ± 1.26	37.57 ± 1.87	36.80 ± 3.64	34.78 ± 4.02	37.19
(2) Soft Labels	40.64 ± 2.20	38.90 ± 0.98	39.97 ± 2.27	37.51 ± 2.79	37.24 ± 2.10	38.85
(3) (2) + PLM Init.	42.22 ± 4.28	40.51 ± 2.03	40.60 ± 3.02	37.94 ± 3.12	37.81 ± 2.51	39.82
(4) (2) + our Init.	43.53 ± 5.11	41.43 ± 4.39	41.71 ± 5.21	39.55 ± 4.41	38.62 ± 2.82	40.97

Table 5: Ablation study on the label words.

progresses, prompts have been found to perform well on low-resource tasks (Brown et al., 2020; Liu et al., 2021a). But most of the research on prompt-based text classification is monolingual (Gao et al., 2021; Liu et al., 2021b). And there are some problems with the few multilingual studies. Zhao and Schütze (2021) first uses a prompt-based approach on this task. They propose a hard prompt based on machine translation, but this approach relies on machine translation models and may introduce additional errors. They also proposed to use soft prompts. Although it can be decoupled from the specific language, there are still gaps between the randomly initialized soft prompt and the pre-training stage. Lin et al. (2021) proposes to use English prompts with non-English examples, but they did not consider decoupling the specific language from the view of the model structure, and still used discrete tokens as label words. Winata et al. (2021) performs few-shot multilingual multi-class classification without updating parameters by applying binary prediction and considering the confidence scores of boolean tokens. Although freezing parameters has the advantage of the model training cost, the model cannot be fine-tuned according to the actual task, which may lead to performance loss.

6 Conclusion

In this paper, we propose a new prompt-based tuning method for zero-shot cross-lingual text classification. For the two key elements of the prompt, we respectively give solutions under this task setting. For templates, we use a two-tower prompt encoder for encoding, which not only decouples specific languages but also preserves the ability of prompts to activate the latent knowledge of the language model. For label words, we use soft label words and dynamic initialization methods, which also achieve the goal of decoupling specific languages. The experimental results prove the utility of our model, and we also design experiments to carry out detailed analysis of the settings of our model.

Limitations

Our UniPrompt is more suitable for low resource scenarios. With the growth of the data scale, the advantages of UniPrompt become minor. This is also verified by the existing prompt-based methods. Prompt can stimulate the potential knowledge of PLM in the pre-training stage, which may be general and may not match the domain knowledge of downstream tasks. In the case of a small data scale, this general knowledge can greatly help the model to judge with limited domain knowledge. With the growth of the data scale, the model can summarize the corresponding knowledge which is adapted to the task from the domain data, and the importance of general knowledge brought by prompt decreases relatively.

Currently, our method is only applicable to natural language understanding tasks. This is determined by the selected PLM model, type of prompt, and model structure. We believe that some of the ideas in this paper can be used in natural language generation, which remains to be further investigated by subsequent research.

Acknowledgments

The work is supported by National Natural Science Foundation of China under Grant No.62036001 and PKU-Baidu Fund (No. 2020BD021). The corresponding author of this paper is Houfeng Wang.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised

cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

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.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A Smith. 2020. The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568.

Teven Le Scao and Alexander M Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, et al. 2021. Few-shot learning with multilingual language models. *arXiv preprint arXiv:2112.10668*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *ACL 2020*, pages 5231–5247.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. Simalign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Genta Indra Winata, Andrea Madotto, Zhaojiang Lin, Rosanne Liu, Jason Yosinski, and Pascale Fung. 2021. Language models are few-shot multilingual learners. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 1–15.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. 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.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

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. *arXiv preprint arXiv:2108.13161*.

Mengjie Zhao and Hinrich Schütze. 2021. Discrete and soft prompting for multilingual models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8547–8555.

Mengjie Zhao, Yi Zhu, Ehsan Shareghi, Ivan Vulić, Roi Reichart, Anna Korhonen, and Hinrich Schütze. 2021. A closer look at few-shot crosslingual transfer: The choice of shots matters. In *ACL 2021*, pages 5751–5767.