Episode-based Prompt Learning for Any-shot Intent Detection

Pengfei Sun*, Dingjie Song*, Yawen Ouyang, Zhen Wu(✉), and Xinyu Dai

National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
{spf, songdj, ouyangyw}@smail.nju.edu.cn
{wuz, daixinyu}@nju.edu.cn

Abstract. Emerging intents may have zero or a few labeled samples in realistic dialog systems. Therefore, models need to be capable of performing both zero-shot and few-shot intent detection. However, existing zero-shot intent detection models do not generalize well to few-shot settings and vice versa. To this end, we explore a novel and realistic setting, namely, any-shot intent detection. Based on this new paradigm, we propose **E**pisode-based **P**rompt **L**earning (EPL) framework. The framework first reformulates the intent detection task as a sentence-pair classification task using prompt templates and unifies the different settings. Then, it introduces two training mechanisms, which alleviate the impact of different prompt templates on performance and simulate any-shot settings in the training phase, effectively improving the model's performance. Experimental results on four datasets show that EPL outperforms strong baselines by a large margin on zero-shot and any-shot intent detection and achieves competitive results on few-shot intent detection.

Keywords: Intent detection · Any-shot learning · Prompt learning.

1 Introduction

Intent detection is an essential component of task-oriented dialogue systems, which can be treated as a text classification problem to predict the intent of users' text input [3]. Although supervised deep learning methods [16, 20] have achieved promising performance for intent detection, their success depends on large-scale annotated data. The models' generalization ability is limited in low-resource settings [1]. Therefore, understanding how to detect users' intent in low-resource settings effectively has become an emerging research topic.

Zero-shot learning (ZSL) and few-shot learning (FSL) have recently provided feasible solutions for low-resource intent detection. Although both paradigms have similar goals - using limited annotated data (including zero or a few annotated data) to detect intent - they still require different models designed for different settings. Specifically, zero-shot intent detection (ZSID)[1] models are

* Equal contribution.

[1] Zero-shot intent detection is a setup in which a model can learn to detect intents that it hasn't explicitly seen before in training [21].

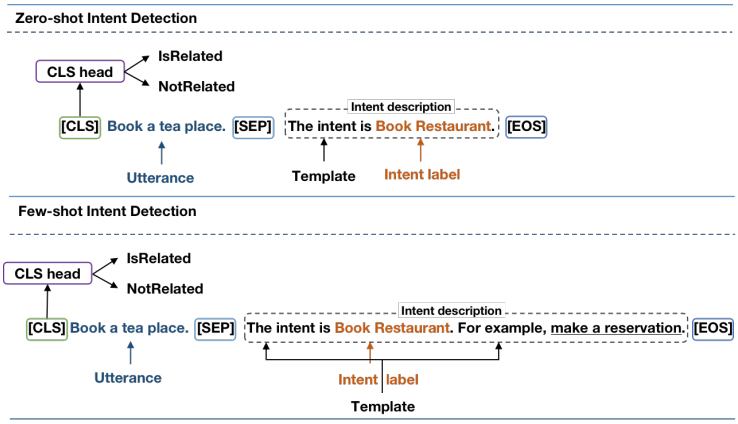


Fig. 1: Different settings use different templates to combine with each intent label as intent descriptions. Notably, in the few-shot intent detection, the template also includes the annotation data, which is underlined.

built on the semantic matching ability of labels. In contrast, few-shot intent detection (FSID)[2] models are built on the semantic summarization ability of a few examples [13]. This means that ZSID models are ineffective when applied to FSID and vice versa. However, in realistic scenarios, some emerging intents may have no available examples (i.e., zero-shot intents), while others may have a few (i.e., few-shot intents). For this reason, there is a need for a more realistic setting where it is possible for both zero-shot and few-shot intents can co-occur during inference. This paper refers to this setting as *any-shot intent detection* (ASID). ASID aims to develop a model that can unify ZSID and FSID in a single framework.

To achieve this goal, we intend to model around the common information (intent labels) of ZSID and FSID for unified modeling across different settings. Specifically, as illustrated in Figure 1, we use different prompt templates for different settings to incorporate each intent label into sentences, regarded as the intent descriptions. Afterward, the intent detection task can be reformulated into a sentence-pair classification task to determine if the utterance relates to the intent description. Although this approach achieves uniform modeling across different settings, it still faces the following challenges: (1) we have empirically found that this approach relies on prompt templates' design, and (2) how the model efficiently utilizes the limited annotated data for training in the ASID task.

To address the above challenges, we propose a novel and unified **E**pisode-based **P**rompt **L**earning (EPL) framework for ASID. The framework first reconstructs the intent detection task into a sentence-pair classification task using different prompt templates to unify the different settings. After that, as shown in Figure 2, two-stage training mechanisms are proposed to alleviate the impact

[2] Few-shot intent detection is a setup in which a model can learn to detect intents that only a few annotated examples are available. [22]

of different prompt templates on performance and improve the model's generalization ability. In stage 1, we propose a prompt-based pre-training mechanism to quickly adapt the model to different prompt templates and learn transferable task-specific knowledge. Specifically, we design the supervised and self-supervised losses for continued pre-training [8] on the reconstructed dataset. In stage 2, we introduce an episodic training mechanism[3] [19] to train the model within a collection of episodes, each designed to simulate an any-shot setting. The model gradually learns to adapt to emerging intents by training in multiple episodes and improves its generalization ability. Extensive analytical experiments have shown that two-stage training mechanisms can effectively mitigate the effects of different prompt templates on performance, helping the model achieve competitive results in the any-shot setting.

The main contributions of the method presented are below:

• For the first time, we present the any-shot intent detection task where both zero-shot and few-shot intents simultaneously co-occur during inference.

• We propose a unified **E**pisode-based **P**rompt **L**earning (EPL) framework for ASID. EPL models different settings uniformly by reformulating the intent detection task as a sentence-pair classification task and proposing a two-stage training mechanism to train the model. The mechanism effectively alleviates the model's reliance on prompt templates and improves its generalization ability.

• We comprehensively evaluate our approach on four popular benchmark datasets and demonstrate the superiority of our approach in zero-shot, few-shot, and any-shot intent detection.

2 Preliminaries

2.1 Problem Statement

Suppose the intents set is $\mathcal{Y} = \mathcal{Y}_s \cup \mathcal{Y}_u, \mathcal{Y}_s \cap \mathcal{Y}_u = \varnothing$, where $\mathcal{Y}_s$ is the set of seen intents, $\mathcal{Y}_u$ is the set of unseen (emerging) intents. We take the dataset of seen intents as the training set, which is defined as $\mathcal{D}_{\text{tr}} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}_{\text{tr}}|}$, where $y_i \in \mathcal{Y}_s$ and $x_i \in \mathcal{X}_s$, $\mathcal{X}_s$ is the set of seen utterances. Similarly, we take the dataset corresponding to the unseen intents as the test set, which is defined as $\mathcal{D}_{\text{te}} = \{(x_j, y_j)\}_{j=1}^{|\mathcal{D}_{\text{te}}|}$, where $y_j \in \mathcal{Y}_u$ and $x_j \in \mathcal{X}_u$, $\mathcal{X}_u$ is the set of unseen utterances. In ZSID, no utterances for each $y_j \in \mathcal{Y}_u$ are available during the inference phase. The goal is to train a model based on the $\mathcal{D}_{\text{tr}}$ so that it can predict the label of $\mathcal{X}_u$ that belongs to unseen intents. Moreover, in FSID, only a few randomly chosen utterances for each $y_j \in \mathcal{Y}_u$ are available during the inference phase, and the goal is the same as the zero-shot setting. While in ASID, a few or no utterances for each $y_j \in \mathcal{Y}_u$ are available during the inference phase, and the goal is to predict both zero-shot and few-shot intents.

[3] Episodic training mechanism attempts to simulate a realistic setting by generating a small set of artificial tasks from a larger set of training tasks for training and proceeds similarly for testing.

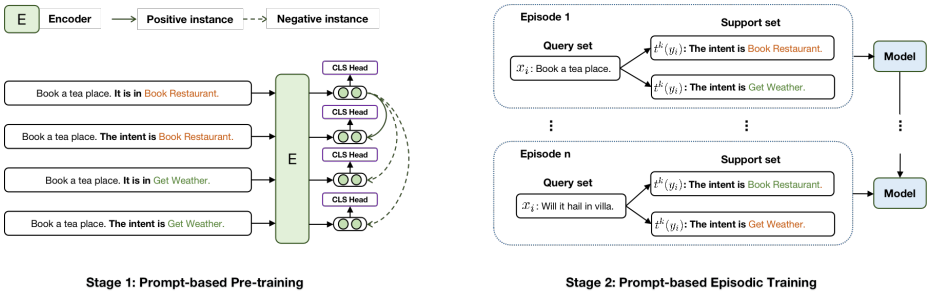


Fig. 2: The framework of the multi-stage training mechanism. Stage 1 is the prompt-based pre-training mechanism, which quickly adapts the model to different prompt templates and learns transferable task-specific knowledge. Stage 2 is the prompt-based episodic training mechanism, which enhances the model's generalization ability as the episodes progress. Note that an intent label with orange fonts indicates the instance is related (i.e., $y^{i,k}_{\text{output}} = 0$), and an intent label with green fonts indicates the instance is unrelated (i.e., $y^{i,k}_{\text{output}} = 1$). The bold text indicates the prompt template.

2.2 Episodic Training Mechanism

In the episodic training mechanism [19], an episode is similar to a batch in the traditional training method. One episode consists of two parts: support set S and query set Q. If the support set S contains C classes, and each class includes K labeled samples, such as $K = 0$, we call this task C-way K-shot. The episodic training mechanism keeps the training conditions consistent with the test conditions. For example, if the test task is a 5-way 1-shot intent detection task, our model is trained using the same 5-way 1-shot task during training. The advantage is that if a model can fit these large numbers of episodes when faced with a new, similar target task, it can also generalize to this target task.

3 Episode-based Prompt Learning

3.1 Reformulation and Definition

Task Reformulation. We reformulate the intent detection task as a sentence-pair classification task using prompt templates. A set of prompt templates is defined as $T = \{t^k\}_{k=1}^{M}$, where M is the number of prompt templates. For each prompt template t^k, we take the intent $y_i \in \mathcal{Y}_s$ as input and output a sentence $t^k(y_i)$ which is regarded as the intent description. Based on the above definition, we use the prompt template t^k to reconstruct dataset $\mathcal{D}_{\text{tr}}$ into a sentence-pair classification dataset $\hat{\mathcal{D}}^k_{\text{tr}} = \{(x^{i,k}_{\text{input}}, y^{i,k}_{\text{output}})\}_{i=1}^{|\mathcal{D}_{\text{tr}}|}$, where $x^{i,k}_{\text{input}} = (x_i, t^k(y_i))$. The output space $y^{i,k}_{\text{output}}$ is composed of two labels, when the label of x_i is y_i, $y^{i,k}_{\text{output}} = 0$. Otherwise, $y^{i,k}_{\text{output}} = 1$. To adapt the sentence-pair classification training, we reconstructed the training dataset as $\hat{\mathcal{D}}_{\text{tr}} = \{(\hat{\mathcal{D}}^k_{\text{tr}}, -\hat{\mathcal{D}}^k_{\text{tr}})\}_{k=1}^{M}$, where $-\hat{\mathcal{D}}^k_{\text{tr}}$ denotes negative samples, i.e., the label of x_i is not y_i. The dataset $\mathcal{D}_{\text{te}}$ is similarly reconstructed to obtain $\hat{\mathcal{D}}_{\text{te}}$.

Definition. We denote the representations of $x^{i,k}_{\text{input}}$ as $\mathbf{h}^{i,k} \in \mathbb{R}^H$, which is the hidden vector of [CLS]. The output probability of the model's [CLS] head is defined as follows:

$$p_{\text{cls}}(y^{i,k}_{\text{output}}|x^{i,k}_{\text{input}}) = \text{softmax}(\mathbf{W}\mathbf{h}^{i,k}), \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{2\times H}$ is a learnable matrix.

3.2 Prompt-based Pre-training

We propose a prompt-based pre-training mechanism to quickly adapt the model to different prompt templates and learn transferable task-specific knowledge. Specifically, we use the reconstructed dataset $\hat{\mathcal{D}}_{\text{tr}}$ to pre-train the model with the supervised loss $\mathcal{L}_{\text{ce}}$ and the self-supervised loss $\mathcal{L}_{\text{con}}$.

The supervised loss $\mathcal{L}_{\text{ce}}$ is designed to narrow the gap between the pre-trained language model and the downstream tasks and to speed up the learning of transferable task-specific knowledge. We thus define the supervised loss as follows:

$$\mathcal{L}_{\text{ce}} = -y^{i,k}_{\text{output}}\log\, p_{\text{cls}}(y^{i,k}_{\text{output}}|x^{i,k}_{\text{input}}) - (1 - y^{i,k}_{\text{output}})\log(1 - p_{\text{cls}}(y^{i,k}_{\text{output}}|x^{i,k}_{\text{input}})). \tag{2}$$

Furthermore, we use different semantic information generated by different prompt templates as self-supervised signals to design the self-supervised loss function $\mathcal{L}_{\text{con}}$. It can more effectively stimulate the language model to learn to adapt to different semantic information, thus alleviating the performance difference caused by different prompt templates. Accordingly, we follow [5] and define the self-supervised loss function $\mathcal{L}_{\text{con}}$ as the normalized temperature-scaled cross-entropy loss. Concretely, for each sample $x^{i,k}_{\text{input}}$ in a batch, we treat $x^{i,k}_{\text{input}}$ as an anchor sample. We use samples $\{x^{i,n}_{\text{input}}\}^M_{n=1}(n \neq k)$ constructed from different prompt templates as positive examples, and other samples within the same batch as negative examples. The self-supervised loss function is:

$$\mathcal{L}_{\text{con}} = -\frac{1}{M-1}\log\frac{\sum^M_{n=1}\mathbb{I}_{[n\neq k]}\exp(\mathbf{h}^{i,k}\cdot\mathbf{h}^{i,n}/\tau)}{\sum^B_{j=1}\sum^M_{p=1}\exp(\mathbf{h}^{i,k}\cdot\mathbf{h}^{j,p}/\tau)}, \tag{3}$$

where B is the number of samples in a batch, and τ is the temperature hyper-parameter. The overall loss function is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{ce}} + \lambda\mathcal{L}_{\text{con}}, \tag{4}$$

where λ is a hyperparameter that balances the supervised and self-supervised losses.

3.3 Prompt-based Episodic Training

To effectively mimic low-resource settings, we introduce the prompt-based episodic training mechanism to train the ASID model. Specifically, this mechanism simulates any-shot settings by constructing a series of episodes (each episode aims

to mimic an ASID task), gradually accumulating a wealth of experience in predicting unseen intents, enhancing the model's adaptability, and improving its generalization ability. In addition, we empirically find that this training mechanism further alleviates the performance differences caused by different prompt templates.

As shown in Stage 2 of Figure 2, we first divide the training process into a series of episodes, in which each episode includes the support set S and the query set Q. We construct the support set S by designing a few prompt templates and then using them in both ZSID and FSID to eliminate the differences between the two settings. Specifically, in ZSID, the intent label y_i is mapped to the prompt template t^k as the intent description $t^k(y_i)$. Afterward, we define the intent description $t^k(y_i)$ and the corresponding output $y^{i,k}_{\text{output}}$ as a support set sample, i.e., $S_i = \{(t^k(y_i), y^{i,k}_{\text{output}})\}^M_{k=1}$. In FSID, we concatenate support set samples $\{x_j\}^K_{j=1}$ with the original prompt templates as new prompt templates $T_{\text{new}} = \{t^k\}^{M\times K}_{k=1}$. After that, same as ZSID, the intent label y_i is mapped to the new prompt template t^k as the intent description $t^k(y_i)$, so that we can get a support set that contains both the intent label and a few labeled samples, i.e., $S_i = \{(t^k(y_i), y^{i,k}_{\text{output}})\}^{M\times K}_{k=1}$. For the query set Q, the settings of $x_i \in \mathcal{X}_u$ are the same in ZSID and FSID. Since ASID sits on a continuum between ZSID and FSID, the above settings are still used for ASID. In addition, the probability of the model is defined as follows:

$$p(y_i|x_i, t^k) = \frac{\exp\, p_{\text{cls}}(y^{i,k}_{\text{output}} = 0|x_i, t^k(y_i))}{\sum_{y_j\in\mathcal{Y}} \exp\, p_{\text{cls}}(y^{i,k}_{\text{output}} = 0|x_i, t^k(y_j))}. \tag{5}$$

Finally, we define the loss $\mathcal{L}_{\text{et}}$ as follows:

$$\mathcal{L}_{\text{et}} = -\log\, p(y_i|x_i, t^k). \tag{6}$$

Notably, during inference, given an input sentence x, for each intent label $y_i \in \mathcal{Y}_s$, the model generates a set of rendered prompts $t^k(y_i)$ using different prompt templates. The model then computes a similarity score between x and each $t^k(y_i)$ using the similarity function $p_{\text{cls}}(y^{i,k}_{\text{output}} = 0|x_i, t^k(y_i))$. Finally, the model predicts the intent label with the highest similarity score as the output.

4 Experiment

4.1 Datasets

We evaluate our models on four intent detection datasets: BANKING77 [2], SNIPS [6], HWU64 [14] and CLINC150 [11]. **BANKING77** is a fine-grained intent detection dataset specific to the banking domain and is composed of 77 intents with approximately 13k samples. **SNIPS** is a personal voice assistant dataset comprising approximately 14k samples of 7 intents. **HWU64** was collected from Amazon Mechanical Turk, a fine-grained intent detection dataset. It contains approximately 11k samples of 64 intents. **CLINC150** is also a voice

Dataset	#sents	#cls train/valid/test	#sents/cls	Prompt Templates
SNIPS	14, 484	2/3/2	2069.14	t^1 = **[s]**.
CLINC150	22, 500	50/50/50	150.00	t^2 = This is in **[s]**.
BANKING77	13, 083	25/25/27	169.91	t^3 = It is about **[s]**.
HWU64	11, 036	23/16/25	172.44	t^4 = It is about the intent of **[s]**.

Table 1: Statistics of four datasets (left) and templates used in our experiment (right). The *#sents* denotes sentence number, *#cls* denotes class number and *#sents/cls* denotes average sentences per class. [s] is the placeholder of the intent label.

assistant dataset and supports both in-scope and out-of-scope data. We follow [12] for the partitioning method, using only 150 intents from 10 domains with approximately 22k samples. Zero-shot, few-shot, and any-shot settings are constructed based on these four datasets. The training, validation, and test sets are partitioned according to the intent labels without intersections. The detailed data statistics are presented in Table 1 (left).

4.2 Experimental Settings

For each experiment, we construct different episodes (i.e., C-way K-shot tasks) to evaluate the model's performance. To make a fair comparison, we complete experiments with C set to different values. For ZSID, we set C to the number of intents in the test dataset and set K to 0, i.e., 0-shot. For FSID, we set K to 1 and 5 to construct 1-shot and 5-shot scenarios, respectively. C is set to 2 on the SNIPS dataset. For the other datasets, we follow the evaluation settings from [4] and set C to 5. For ASID, we select one 0-shot class and one 1-shot class in one episode for SNIPS, one 0-shot class, two 1-shot classes and two 5-shot classes for the other datasets. Also, as the different choices of training and validation sets affect the performance of the test, we perform a 5-fold cross-validation and report the average results. In addition, in our experiments, to evaluate the impact of different templates, we manually designed four templates for the dataset and reported the average results of the four templates and the results of the best template. The detailed templates are presented in Table 1 (right).

We implement all methods using Pytorch. EPL is built on top of the BERT-base model. For the training phases of EPL, we optimize the model using Adam [10], with the stage 1 learning rate set to 5e-6 and the stage 2 learning rate set to 2e-5. For prompt-based pre-training, similar to [23], we first select the target dataset and then use the remaining three out of four datasets as the pre-training corpus. We split the pre-training corpus as the training set and validation set, where the ratio of classes is 9:1. The grid search mechanism is utilized to select optimal hyperparameter combinations on each split (the hyperparameter λ from {0.1,0.5,1,5}, the negative samples' number from {1,3,5}). Finally, we select the hyperparameter λ and negative samples' numbers as 0.5 and 3, respectively. The temperature coefficient τ is set to 0.7, and the batch size is 8. For prompt-based episodic training, we follow the experimental setup in [7] and report the average accuracy of over 500 episodes sampled from the test set.

4.3 Baseline Models

We compare our proposed EPL model with three types of baseline models. We first compare the EPL model with the following state-of-the-art approaches for ZSL. **CTIR**: [17] propose a class transformation framework that encourages models to learn the difference between seen and unseen intents through multi-task learning objectives and presents similarity scorers to correlate associations between intents. **NSP-BERT**: [18] propose a sentence-level prompt learning method that determines whether the two sentences are adjacent by reformulating the downstream task as a binary classification problem and directly using a BERT-based pre-trained language model.

For FSL, we compare the proposed method with several state-of-the-art FSL models. **ProtAugment**: [7] propose a meta-learning-based intent detection method that avoids the overfitting problem during meta-learning by extending the prototype network. **ContrastNet**: [4] propose a contrastive learning framework for solving the task-level and instance-level overfitting problems in the few-shot text classification task.

In addition, we also compare our approach with the models which apply in both settings. **NLI**: [15] reformulate the intent detection task as a natural language inference task and fine-tune the BERT model using the reconstructed dataset. To ensure fairness, we adapt NLI to fit ASID. **KPT**: [9] incorporate external knowledge into the verbalizers. Two strategies are designed to refine the verbalizers for the zero-shot and few-shot settings, respectively, but are not applicable for the any-shot setting.

5 Results and Analysis

5.1 Main Results

Results for ZSID In Table 2, we report the results of the ZSID experiments. EPL outperforms other methods on all four datasets compared to state-of-the-art models. Specifically, EPL improves by 2.98%, 0.82%, 3.75%, and 0.68% on SNIPS, CLINC150, HWU64, and BANKING77, respectively. The result indicates that EPL is effective in ZSID. Notably, we also find that the EPL and NLI models are better than NSP-BERT. We speculate that retraining the [CLS] head based on the reformulated task can effectively learn transferable and task-specific knowledge, thus improving the model's performance.

Results for FSID As shown in Table 2, we can observe that in the 1-shot setting of the FSID task, our EPL model outperforms the best baseline results (in most cases, ContrastNet). Similarly, in the 5-shot setting of the FSID task, our EPL model outperforms the best baseline results on the SNIPS, CLINC150, and HWU64 datasets and is close to the best baseline results for BANKING77. Although ContrastNet and ProtAugment are better than EPL in some datasets in the 5-shot setting, their methods only apply to FSID, which limits their application in the any-shot setting. Furthermore, we observe that KPT improves in the

Setting	Method	SNIPS	CLINC150	HWU64	BANKING77	$\textbf{Avg}_{\text{Std}}$
0-shot	CTIR	86.59	27.20	21.44	20.76	$39.00_{31.86}$
	NSP-BERT	82.88	60.99	45.09	58.47	$61.86_{15.66}$
	NLI	92.20	81.51	61.01	75.45	$77.54_{13.02}$
	KPT	68.17	18.58	16.69	18.54	$30.49_{25.13}$
	EPL	**95.18**	**82.33**	**64.76**	**76.13**	$\mathbf{79.60}_{12.68}$
1-shot	ProtAugment	86.33	96.49	84.34	89.56	$89.18_{5.33}$
	ContrastNet	90.44	96.59	86.56	91.18	$91.19_{4.13}$
	NLI	94.00	95.59	83.11	90.84	$90.88_{5.55}$
	KPT	75.36	51.78	46.84	45.38	$54.84_{13.95}$
	EPL	**95.39**	**97.55**	**89.96**	**93.31**	$\mathbf{94.05}_{3.23}$
5-shot	ProtAugment	92.52	**98.74**	92.55	94.71	$94.63_{2.93}$
	ContrastNet	**96.21**	98.46	92.57	**96.40**	$\mathbf{95.91}_{2.45}$
	NLI	94.50	97.24	89.51	93.44	$93.67_{3.20}$
	KPT	80.41	62.61	50.49	47.71	$60.30_{14.88}$
	EPL	**96.21**	98.69	**92.91**	95.30	$95.78_{2.39}$
any-shot	NLI	94.00	95.50	83.34	90.53	$90.84_{5.42}$
	EPL	**95.15**	**97.47**	**88.44**	**92.78**	$\mathbf{93.46}_{3.86}$

Table 2: Comparison of mean accuracy (%) on four datasets in ZSID, FSID, and ASID settings. For each setting, the best and the second-best results are highlighted. The $\textbf{Avg}_{\text{Std}}$ denotes the averaged mean and standard deviation over four datasets of each model.

	Method	SNIPS	CLINC150	HWU64	BANKING77	Average
NSP	BERT	82.88	60.99	45.09	58.47	61.86
	Pre-training	**95.85**	75.41	59.60	70.91	75.44
EPL	BERT	94.92	81.06	59.78	74.95	77.68
	Pre-training	95.18	**82.33**	**64.76**	**76.13**	**79.60**

Table 3: Impact of prompt-based pre-training on performance in ZSID. Note that the best results are highlighted.

1-shot and 5-shot settings compared to the zero-shot setting but still performs poorly. The results indicate that the approach resembling KPT, which relies on verbalizer construction, is less applicable to fine-grained intent detection tasks.

Results for ASID The results of the ASID experiments are displayed in Table 2. Compared with NLI, EPL achieves substantial improvements on the four datasets. This result validates the effectiveness of our proposed method in the any-shot setting.

5.2 Experimental Analysis

Impact of Prompt-based Pre-training. The following experiments are conducted on four datasets to evaluate the impact of the prompt-based pre-training

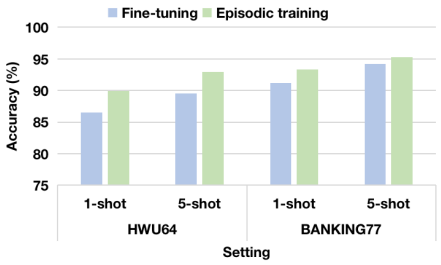
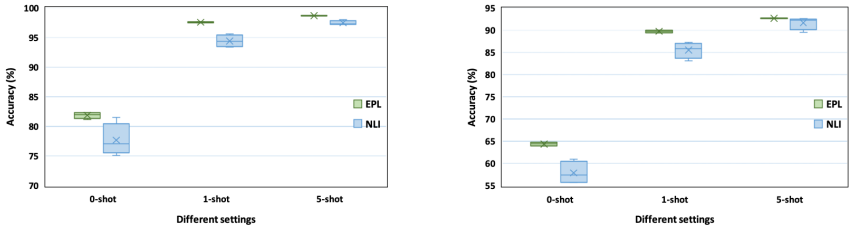


Fig. 3: Effect of different training mechanisms on two datasets.



(a) CLINC150 **(b)** HWU64

Fig. 4: Fluctuations in performance of different prompt templates on two datasets. The size of the box reflects the volatility of the data. The smaller box indicates more concentrated data. The cross represents the mean value.

mechanism on performance. Concretely, we interchange the prompt-based pre-trained model and the vanilla BERT model. For NSP-BERT, we replace the vanilla BERT model with the prompt-based pre-trained model and refer to it as NSP-PreTraining. We replace the prompt-based pre-trained model in EPL with a vanilla BERT model and refer to it as EPL-BERT. Table 3 shows that using the prompt-based pre-trained model improves accuracy by an average of at least 1.9% on the zero-shot setting compared to the model using BERT. These results suggest that the prompt-based pre-training mechanism is effective. This is because continued pre-training on the reconstructed data enables the [CLS] head to adapt to downstream tasks, learn transferable task-specific knowledge, and improve the model's generalization. In addition, we note that although NSP-PreTraining achieves more competitive results on the SNIPS dataset, its accuracy is much lower than EPL on other datasets. Meanwhile, in Table 3, we can also see that EPL-BERT outperforms NSP-BERT on all datasets, which indicates that the EPL model still obtains good results even without prompt-based pre-training. This result further demonstrates that EPL is effective and superior.

Impact of Prompt-based Episodic Training Mechanism. To evaluate the impact of the prompt-based episodic training mechanism, we compare fine-tuning with the prompt-based episodic training mechanism on the HWU64 and BANKING77 datasets with different settings in FSID. As shown in Figure 3, the model's performance based on the prompt-based episodic training mechanism outperforms that of the fine-tuning training mechanism on both the HWU64

and BANKING77, especially in the 1-shot setting, where the improvement of prompt-based episodic training is more prominent. We conjecture the prompt-based episodic training mechanism can simulate low-resource settings, and the model gradually learns to adapt to unseen intents, leading to better results. It also indicates that the prompt-based episodic training mechanism is effective.

Impact of Prompt Template. We conduct experiments to verify the effect of different prompt templates (Table 1 right) on the performance of the EPL and NLI. The results are shown in Figure 4. Compared with NLI, EPL effectively alleviates the performance difference caused by different prompt templates, with the performance of EPL remaining stable between different prompt templates, particularly for the FSID settings. One reason could be that a few samples in the support set can be used to effectively fine-tune the model and mitigate the effects of semantic differences between different prompt templates. For the ZSID settings, the performance of both EPL and NLI fluctuates between different prompt templates, but EPL is more stable. This implies that the lack of available samples makes the model more dependent on the prompt templates, and the performance fluctuates slightly. The relative stability of EPL further suggests that EPL can mitigate the effect of differences between prompt templates.

6 Conclusion

We introduce any-shot learning to intent detection for the first time and propose a novel, unified EPL model. The EPL model first reformulates the intent detection task as a sentence-pair classification task and unifies the different settings using prompt templates. Then we propose a two-stage training mechanism (prompt-based pre-training and prompt-based episodic training). The mechanisms not only effectively mitigate the impact of different templates on performance but also simulates a low-resource setting and improve the model's generalization capability. Finally, extensive experiments have shown that the proposed model achieves state-of-the-art performance on four publicly available datasets.

Acknowledgements. The authors would like to thank the anonymous reviewers for their helpful comments. Zhen Wu is the corresponding author. This research is supported by the National Natural Science Foundation of China (No. 61936012, 62206126 and 61976114).

References

1. Bhathiya, H.S., Thayasivam, U.: Meta learning for few-shot joint intent detection and slot-filling. In: ICMLT. p. 86–92 (2020)
2. Casanueva, I., Temčinas, T., Gerz, D., Henderson, M., Vulić, I.: Efficient intent detection with dual sentence encoders. In: NLP4ConvAI. pp. 38–45 (2020)
3. Celikyilmaz, A., Hakkani-Tur, D., Tur, G., Fidler, A., Hillard, D.: Exploiting distance based similarity in topic models for user intent detection. In: ASRU. pp. 425–430 (2011)

4. Chen, J., Zhang, R., Mao, Y., Xue, J.: Contrastnet: A contrastive learning framework for few-shot text classification. In: AAAI. pp. 10492–10500 (2022)
5. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607 (2020)
6. Coucke, A., Saade, A., Ball, A., Bluche, T., Caulier, A., Leroy, D., Doumouro, C., Gisselbrecht, T., Caltagirone, F., Lavril, T., et al.: Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. arXiv preprint arXiv:1805.10190 (2018)
7. Dopierre, T., Gravier, C., Logerais, W.: Protaugment: Intent detection meta-learning through unsupervised diverse paraphrasing. In: ACL/IJCNLP (2021)
8. Gururangan, S., Marasovic, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., Smith, N.A.: Don't stop pretraining: Adapt language models to domains and tasks. In: ACL. pp. 8342–8360 (2020)
9. Hu, S., Ding, N., Wang, H., Liu, Z., Li, J.Z., Sun, M.: Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. ArXiv **abs/2108.02035** (2021)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
11. Larson, S., Mahendran, A., Peper, J.J., Clarke, C., Lee, A., Hill, P., Kummerfeld, J.K., Leach, K., Laurenzano, M.A., Tang, L., Mars, J.: An evaluation dataset for intent classification and out-of-scope prediction. In: EMNLP-IJCNLP. pp. 1311–1316 (2019)
12. Li, J.Y., Zhang, J.: Semi-supervised meta-learning for cross-domain few-shot intent classification. In: MetaNLP (2021)
13. Liu, F., Lin, H., Han, X., Cao, B., Sun, L.: Pre-training to match for unified low-shot relation extraction. arXiv preprint arXiv:2203.12274 (2022)
14. Liu, X., Eshghi, A., Swietojanski, P., Rieser, V.: Benchmarking natural language understanding services for building conversational agents. In: IWSDS (2019)
15. Malik, V., Kumar, A., Vepa, J.: Exploring the limits of natural language inference based setup for few-shot intent detection. ArXiv **abs/2112.07434** (2021)
16. Qin, L., Liu, T., Che, W., Kang, B., Zhao, S., Liu, T.: A co-interactive transformer for joint slot filling and intent detection. In: ICASSP. pp. 8193–8197 (2021)
17. Si, Q., Liu, Y., Fu, P., Lin, Z., Li, J., Wang, W.: Learning class-transductive intent representations for zero-shot intent detection. In: IJCAI (2021)
18. Sun, Y., Zheng, Y., Hao, C., Qiu, H.: Nsp-bert: A prompt-based zero-shot learner through an original pre-training task-next sentence prediction. ArXiv **abs/2109.03564** (2021)
19. Vinyals, O., Blundell, C., Lillicrap, T.P., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NIPS (2016)
20. Wang, J., Wei, K., Radfar, M., Zhang, W., Chung, C.: Encoding syntactic knowledge in transformer encoder for intent detection and slot filling. In: AAAI. vol. 35, pp. 13943–13951 (2021)
21. Xia, C., Zhang, C., Yan, X., Chang, Y., Philip, S.Y.: Zero-shot user intent detection via capsule neural networks. In: EMNLP. pp. 3090–3099 (2018)
22. Xu, W., Zhou, P., You, C., Zou, Y.: Semantic transportation prototypical network for few-shot intent detection. In: Interspeech. pp. 251–255 (2021)
23. Zhang, H., Zhang, Y., Zhan, L.M., Chen, J., Shi, G., Wu, X.M., Lam, A.Y.: Effectiveness of pre-training for few-shot intent classification. In: EMNLP. pp. 1114–1120 (2021)