On Collaborating Small and Large Models for Few-shot Intent Detection

Peng Chen[1], **Bang Wang**[2,*]
[1]School of Software Engineering,
Huazhong University of Science and Technology, Wuhan, China
[2]School of Electronic Information and Communications,
Huazhong University of Science and Technology, Wuhan, China
{hustchenpeng, wangbang}@hust.edu.cn

Abstract

Few-shot intent detection (FSID) targets the classification of user queries into in-scope intent categories or detecting them as out-of-scope, with only a few or even zero labeled examples per class. Existing PLM-based methods struggle in low-resource situations; while LLM-based methods face high inference cost and label interference. To harness their complementary strengths, we propose the **FCSLM**, a framework that collaborates a small prediction model with a large language model for the FSID task. During training, we leverage LLMs for data augmentation in self-supervised pretraining and supervised fine-tuning a task-specific prediction model. During inference, a multi-round reasoning process first applies the small prediction model to output candidate intents with uncertainty estimations, then invokes an LLM with enriched intent descriptions for refined prediction and OOS detection. Extensive experiments on three benchmark datasets demonstrate that our FCSLM outperforms strong competitors, achieving the new state-of-the-art performance in both intent classification and OOS detection. Our code is available at: https://github.com/hustchenpeng/FCSLM

1 Introduction

Intent detection is a core task in dialogue systems, aiming to interpret a user query and classify it into one of the predefined intent categories, or out-of-scope (OOS). In real-world scenarios, as manual annotation is costly, training data is scarce and few-shot intent detection (FSID) has recently become a central focus. The FSID task aims to classify a user query x into a predefined intent category $y \in \mathcal{Y}$, or detect it as out-of-scope (OOS), given only a limited number of labeled training examples per in-scope intent.

For the FSID task, fine-tuning *pre-trained language models* (PLMs) have long been a dominant approach (Zhang et al., 2021, 2023, 2025). Their semantic modeling capabilities have been further enhanced by recent advances in self-supervised pretraining and contrastive learning (Zhang et al., 2021; Yehudai et al., 2023; Singhal et al., 2023). However, the performance of PLM-based models is not satisfactory due to the data scarcity when training a model. This issue becomes particularly pronounced in OOS detection (Zhang et al., 2022), where a PLM-based model needs to first accurately classify in-scope queries, and then reject out-of-scope inputs.

With the rapid development of *large language models* (LLMs), recent studies have explored instruction-tuned LLMs for direct intent classification, typically adopting a full-label enumeration strategy in which all intent labels and examples are included into an input prompt (Zhang et al., 2023; Lin et al., 2023). While LLMs demonstrate strong detection capabilities under low-resource conditions, they also encounter several practical challenges. First, real-world applications often involve a large number of intent categories, leading to long prompt sequences and high inference costs (Yehudai and Bendel, 2024). Second, when semantic boundaries between intent categories are ambiguous, LLMs are prone to confusing similar intents, resulting in degraded discriminative capabilities even worse than PLM-based models. This issue is especially evident in OOS detection, where LLMs tend to misclassify OOS queries as in-scope (IS) when their semantics are close to a known intent category. Moreover, the full-label enumeration strategy inherently increases label-space interference, further raising the likelihood of an OOS query being incorrectly assigned to an IS intent category.

The existing studies indicate that PLMs and LLMs exhibit complementary strengths yet also respective limitations in the FSID task. Those su-

*Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 13937–13953
November 4-9, 2025

pervised PLM-based models retain high inference efficiency and stable performance even under large label spaces. In contrast, those LLM-based approaches excel in scenarios with fewer candidate intents, where their semantic reasoning capabilities can be well leveraged with lower inference cost and reduced label interference. This suggests that using a single technique, i.e., PLM or LLM, is often insufficient to balance the detection efficiency of a small model and the semantic inference of a large model. Furthermore, achieving reliable OOS detection remains a significant challenge for either PLM-based or LLM-based approaches. These considerations motivate us to reexamine the capabilities and roles of a small PLM-based model and a large language model for the FSID task, calling for an effective collaboration between a small and a large model in both offline training and online reasoning phase.

Motivated from the aforementioned considerations, we propose a framework for collaborating a small prediction model with a large language model for the FSID task, called **FCSLM**. The basic idea is to exploit the complementary capabilities of PLMs and LLMs in different task phases. The execution logic is to first train and use a task-specific *prediction model* (PM) to output a set of candidate intents for a query, and then employ an LLM to further discriminate those unconfident candidate intents for a final decision.

The workflow of our FCSLM is divided into the offline training and online reasoning phase. We propose an *offline augmentative training process* to train a task-specific PM with the help of an LLM for data augmentation. The training process includes three steps: self-supervised pre-training, supervised fine-tuning, and multi-predictor sampling. We propose an *online collaborative reasoning process* to output the predicted intent for an input query with the collaboration of the task-specific PM and an LLM. The reasoning process includes three rounds: The first round applies the PM to output a set of candidate intents. If they fail the confidence test in the first round, the second round employs an LLM to distinguish the most suitable candidate intent. If the prediction of the LLM differs from that of the PM, the third round adopts the LLM to perform a second-thought comparative reasoning for the final decision. Figure 1 presents the FCSLM workflow.

We conduct experiments on three benchmark intent detection datasets under 0/5/10-shot settings. Results show that our FCSLM achieves the new state-of-the-art performance in both intent classification and OOS detection for the FSID task. More importantly, our collaborative framework addresses the high costs of existing LLM-based methods by significantly reducing the number of LLM calls and token overhead.

A comprehensive discussion of related work is deferred to Appendix A.

2 Offline Training Phase

We propose an *offline augmentative training process* (OATP) to train a task-specific *prediction model* (PM) via LLM-assisted data augmentation. We note that although the FSID task is only with a few labeled samples, it is still possible to train a task-specific PM through data augmentation. Such a task-specific PM can itself execute intent detection and its detection outputs can also complement the reasoning process of an LLM for more confident results. Let $\mathcal{B}_{raw}$ denote an off-the-shelf PLM, such as the RoBERTa (Liu et al., 2019). First, we propose to use self-supervised pretraining to train $\mathcal{B}_{\text{raw}}$ into $\mathcal{B}_{\text{pre}}$, and then to use supervised fine-tuning to train $\mathcal{B}_{\text{pre}}$ into $\mathcal{B}_{\text{ft}}$. Based on $\mathcal{B}_{\text{ft}}$, we design a multi-predictor sampling strategy to obtain a set of prediction heads $\{\text{P}_c\}$.

2.1 Self-supervised Pretraining

We design a *self-supervised pretraining* (SSP) module to train $\mathcal{B}_{\text{raw}}$ into $\mathcal{B}_{\text{pre}}$ based on a public dataset $\mathcal{D}_{\text{pub}}$, such as SNIPS (Coucke et al., 2018). A sample $x_i \in \mathcal{D}_{\text{pub}}$ is a sentence, such as "add transmission to my found them first."

To improve the lexical and syntactic diversity of the public dataset, for each input sample $x_i \in \mathcal{D}_{\text{pub}}$, we use LLM-based paraphrasing (LP) to ask an LLM to generate its paraphrase x_i', maintaining its semantic consistency but differing in the expression. The paraphrase prompt is presented in Appendix F.1. The pairs (x_i, x_i') are then used to construct the pretraining dataset $\mathcal{D}_{\text{pt}} = \{(x_i, x_i')\}_{i=1}^{N}$. We first adopt the contrastive training for pretraining and design a contrastive loss as follows:

$$\mathcal{L}_{cl} = -\frac{1}{2N}\sum_{i=1}^{2N}\log\frac{\exp\left(\text{sim}(\mathbf{x}_i, \mathbf{x}_i')\right)}{\sum\limits_{\mathbf{x}_j \neq \mathbf{x}_i, \mathbf{x}_i'}\exp\left(\text{sim}(\mathbf{x}_i, \mathbf{x}_j)\right)},$$

where $\mathbf{x}_i$ is the encoded representation of x_i and $\text{sim}(\cdot)$ the cosine similarity function. This design encourages to train a language model to align the original sample with its paraphrase in the representation space while effectively distinguishing other

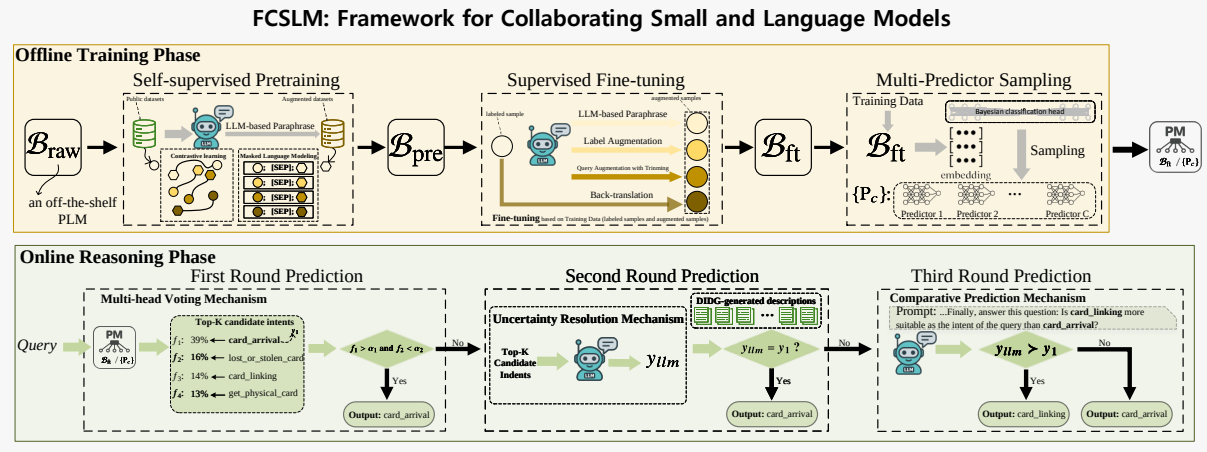

Figure 1: FCSLM Workflow: (a) Offline Training Phase: It leverages an LLM for data augmentation in training a task-specific prediction model; (b) Online Reasoning Phase: It applies a multi-round reasoning process to collaborate the small prediction model and a large language model to output target intent.

non-synonymous samples, thus improving its semantic consistency at the expression level.

In addition, to enhance the language understanding capability for a language model, we introduce a masked language modeling loss (Zhang et al., 2021; Yehudai et al., 2023). Specifically, we concatenate x_i and its paraphrased x_i' using a separator `[SEP]` to form an input sequence $(x_i; \texttt{[SEP]}; x_i')$, and perform random token masking on the concatenated sentence pair. This design leverages the semantic consistency between the original sentence and its paraphrase by using cross-sentence contextual information for the language model to predict the masked vocabulary, thereby strengthening token-level understanding while preserving semantic meaning. The MLM loss is defined by

$$\mathcal{L}_{\text{mlm}} = -\frac{1}{M}\sum_{m=1}^{M} \log P(\hat{x}_m \mid \tilde{x}_m),$$

where $\tilde{x}_m$ denotes the m-th masked token in the input sequence and $\hat{x}_m$ the corresponding true token.

The loss function of the SSP module is given by

$$\mathcal{L}_{\text{pre}} = \mathcal{L}_{\text{cl}} + \lambda \mathcal{L}_{\text{mlm}},$$

where $\lambda \in [0, 1]$ is a weighting coefficient balancing the contrastive loss and the MLM loss.

2.2 Supervised Fine-tuning

We design a *supervised fine-tuning* (SFT) module to train $\mathcal{B}_{\text{pre}}$ into $\mathcal{B}_{\text{ft}}$ based on the FSID task training dataset $\mathcal{D}_{\text{train}}$. We randomly select k samples with the same intent label from $\mathcal{D}_{\text{train}}$ to form a k-shot dataset $\mathcal{D}^y_{\text{few}} = \{(q, y)\}^k_1$, with q the query and y the label. We note that $\mathcal{D}^y_{\text{few}}$ could be an empty set for some intent label, i.e., zero-shot. We propose using an LLM to augment queries and labels for each few-shot $\mathcal{D}^y_{\text{few}}$ and present the augmentation approaches for zero-shot dataset in Appendix B.

We first design an LLM-based *query augmentation with trimming* (QAT) to augment queries for each $\mathcal{D}^y_{\text{few}}$ to obtain an augmented dataset $\mathcal{D}^y_{\text{aug}} = \{(q', y)\}$. For each intent y, we construct a prompt by combining it with the corresponding k-shot queries and input this prompt into the LLM to generate additional intent-consistent queries q'. The detailed settings and query augmentation prompt is presented in Appendix F.2.

We propose a trimming mechanism to control the semantic boundary of $\mathcal{D}^y_{\text{aug}}$ by filtering-out some q' with semantic drift as follows. (1) Encode each q_i and q'_j into a representation $\mathbf{q}_i$ and $\mathbf{q}'_j$, respectively, by a sentence encoder. (2) For each $\mathbf{q}_i$, first calculate a centroid vector excluding itself by $\mathbf{c}_i = \frac{1}{k-1}\sum_{k\neq i}\mathbf{q}_k$, and next compute its distance to the centroid by $s_i = \cos(\mathbf{q}_i, \mathbf{c}_i)$. (3) Compute $\theta_y = \min_i(s_i)$ as the filtering-out threshold, and compute the category centroid by $\mathbf{c}_y = \frac{1}{k}\sum_i \mathbf{q}_i$. (4) For each $\mathbf{q}'_j$, compute its distance to the category centroid by $s'_j = \cos(\mathbf{q}'_j, \mathbf{c}_y)$. If $s'_j < \theta_y$, then the generated query q'_j is regarded as semantically misaligned with the intent category, and is filtered-out from $\mathcal{D}^y_{\text{aug}}$.

We next design an LLM-based *Label Augmentation* (LA) to obtain a set of augmented intent labels $\mathcal{Y}_{\text{aug}}$ based on the original label set $\mathcal{Y}$. An intent label is often with a concise expression by one or a few words, which may not be enough to express

contextual semantics of the intent. To enhance the expressiveness of intent labels, we leverage an LLM to conduct label augmentation. For each intent y, we construct a prompt by combining it with the corresponding k-shot queries and input this prompt into the LLM to generate a more descriptive version of the label. Note that the augmented labels are used to enrich the fine-tuning data, but do not replace the original intent labels. The label augmentation prompt is presented in Appendix F.4.

We use the standard cross-entropy loss to fine-tune $\mathcal{B}_{\text{pre}}$ into $\mathcal{B}_{\text{ft}}$ based on the fine-tuning dataset $\mathcal{D}_{\text{ft}}$, with an additional linear classification layer on top of $\mathcal{B}_{\text{pre}}$. For few-shot cases, the queries of $\mathcal{D}_{\text{ft}}$ are from $\mathcal{Y}$, $\mathcal{Y}_{\text{aug}}$, $\mathcal{D}_{\text{few}}$, $\mathcal{D}_{\text{aug}}$, and the back-translated and paraphrased versions for queries in $\mathcal{D}_{few}$; while in zero-shot cases, they are from $\mathcal{Y}$ and $\mathcal{D}_{\text{aug}}$. Details of $\mathcal{D}_{\text{ft}}$ are presented in Appendix D.1.

2.3 Multi-Predictor Sampling

We propose a *multi-predictor sampling* (MPS) module to alleviate the prediction uncertainty of a single language model. This module leverages the $\mathcal{B}_{\text{ft}}$ as a feature extractor and constructs a Bayesian linear classifier with parameters $(\mathbf{W}, \mathbf{b})$, modeled as random variables, from which we draw C samples $\{(\mathbf{W}_c, \mathbf{b}_c)\}_{c=1}^{C}$ using the Hamiltonian Monte Carlo (HMC) method (Neal et al., 2011), conditioned on $\mathcal{D}_{\text{ft}}$. Each sampled pair defines a distinct prediction head P_c. Here, C is set equal to the number of intent classes as a design choice.

3 Online Reasoning Phase

We propose an *online collaborative reasoning process* (OCRP), which utilizes both the task-specific PM and a LLM to output the target intent y^* for an input query q. In the OCRP, we first use the trained PM to make the first round prediction based on a *multi-head voting mechanism* (MVM). If the voting is not confident, we next employ an LLM to make the second round prediction via an *uncertainty resolution mechanism* (URM). If the uncertainty still exists, we design a *comparative prediction mechanism* (CPM) to make the third round and final prediction. Furthermore, we also design a plug-in *null intent detection* (NID) module to deal with out-of-scope scenarios.

3.1 First Round Prediction

The first round prediction is based on the task-specific PM, namely, $\mathcal{B}_{\text{ft}}$ and $\{\text{P}_c\}$. For the input query q, the $\{\text{P}_c\}$ outputs a set of intent predictions $\{y_c\}$, each corresponding to a prediction head. Note that different prediction heads may output the same intent category. That is, it is possible $y_c = y_{c'}$ for $c \neq c'$. If an intent is predicted by more prediction heads, we regard it as of high confidence.

We sort the predicted intents $\{y_c\}$ according to the logit scores provided by $\mathcal{B}_{\text{ft}}$ in a decreasing order. Let $L_K = \{(y_k, f_k)\}_1^K$ denote the sorted list of top-K *candidate intents*, where f_k is the normalized occurrence frequency of the intent y_k. We set $K < C$, i.e., the number of candidate intents is smaller than the number of intent labels. Note that y_1 and y_2 are the predicted intents, respectively, with the highest and second-highest logits provided by $\mathcal{B}_{\text{ft}}$. The idea for multi-head voting is to check whether the top-1 predicted intent y_1 is far more confident than other predictions. To this end, we set two confidence thresholds, α_1 and α_2. If $f_1 > \alpha_1$ and $f_2 < \alpha_2$, then the top-1 intent y_1 is confirmed as the final prediction output y^*, and in this case, we do not perform the second and third round prediction. Otherwise, we proceed to the second round prediction with the top-K candidate intents.

3.2 Second Round Prediction

The second round utilizes an LLM to help resolving the uncertainty of the first round predictions. The basic idea is to check whether the output of an LLM is equivalent to the PM top-1 intent y_1. Let y_{llm} denote the intent predicted by the LLM. If $y_{\text{llm}} = y_1$, then the top-1 intent y_1 is confirmed as the final prediction output y^*, and in this case, we do not perform the third round prediction. Otherwise, we proceed to the third round prediction.

The key for an LLM to output an intent lies in the design of an appropriate input prompt. Note that in this round, the available information include the test query q, the top-K predicted intents L_K and $\mathcal{D}_{\text{few}}^{y}$ $(y \in L_K)$. The prompt instruction is to let the LLM select one intent from the L_K candidate indents. To further facilitate the LLM reasoning, we propose to first generate a kind of *discriminative intent descriptions* for all intents, and each candidate intent is attached its corresponding description to enrich the input prompt. The design objective of description generation is to establish clear semantic boundaries between one intent and others. Appendix C details the *discriminative intent description generation* (DIDG) module, and Appendix F.6 details the template for the input prompt in the second round.

3.3 Third Round Prediction

The third round is to further select which one of the y_{llm} and y_1 as the final prediction y^*. Note that although the y_{llm} is different from y_1 in the second round, we cannot consider y_1 a worse choice, as it holds the highest logit score among the candidate intents. We again utilize an LLM to make a kind of comparative prediction by evaluating some semantic evidences to prefer y_{llm} or y_1. Example semantic evidences include the prediction of the top-1 intent y_1 made by $\mathcal{B}_{\text{ft}}$ and the few-shot query examples corresponding to y_{llm} and y_1. The details of the input prompt for comparative prediction are provided in Appendix **F.7**.

3.4 Dealing with OOS scenarios

Our OCRF can be directly applied to out-of-scope scenarios, that is, an input query can be detected as with a *null intent* (none of the existing intents). Previous OOS detection approaches directly include a null label into the existing intent label set. As reported in (Wang et al., 2024; Arora et al., 2024), with the increase of existing intent labels, such approaches would degrade the OOS detection performance. In contrast, we propose to include the null intent together only with the top-K detected intents in the second round prediction, thereby avoiding the use of the full set of existing intent labels.

The proposed NID for OOS scenarios is a plug-in module to be used only in the second round prediction. Based on the PM output L_K in the first round, the first step is to append the null intent after the K candidate intents into the input prompt. If the LLM selects the null intent as its output, then the final prediction is made and the whole procedure ends. Otherwise, we introduce a *candidate intent re-ranking mechanism* (CIR), that is, the second step is to first reorder the K candidate intents in the increase order of their logits scores provided by the $\mathcal{B}_{\text{ft}}$ and then repeat the first step to obtain the LLM decision, so as to mitigate the LLM sensitivity to the intent order in the prompt and prevent it from misclassifying null intent queries as in-scope queries. If the LLM does not output the null intent, then this query is not regarded as OOS and the prediction proceeds to the third round as usual.

4 Experiment Settings

Datasets: We conduct experiments on three widely used datasets: **BANKING77** (Casanueva et al., 2020), **CLINC150** (Larson et al., 2019), and **HWU64** (Liu et al., 2021). They cover typical dialogue system scenarios, such as banking services, and voice assistants. Based on these datasets, we construct k-shot ($k \in \{0, 5, 10\}$) experiment settings as follows: k training samples are randomly selected from each intent class, and the original full test set is retained for evaluation.

Competitors: The first group of competitors contains pure PLMs (RoBERTa (Liu et al., 2019) and RoBERTa(SSP) (Liu et al., 2019)) and pure LLM Qwen2.5-72B (Yang et al., 2024)). The second group contains several the state-of-the-art methods, including QAID (Yehudai et al., 2023), CPFT (Zhang et al., 2021), PLE (Li et al., 2022), DFT++ (Zhang et al., 2023), INTENDD (Singhal et al., 2023), ICDA (Lin et al., 2023), ICL (Milios et al., 2023), ZeroGen (Ye et al., 2022), CoDa (Evuru et al., 2024), PromptMix (Sahu et al., 2023), and CUC (Zhang et al., 2025).

Metrics: Following (Zhang et al., 2025; Singhal et al., 2023; Zhang et al., 2022), we adopt the following performance metrics: Accuracy, OOS Precision, Recall and F1.

Appendix D provides all the details on all experiment settings.

5 Experiment Results

5.1 Main Results

We adopt the RoBERTa (Liu et al., 2019) for the PM component and Qwen2.5-72B (Yang et al., 2024) for the LLM component. In the first round prediction, we set uncertainty thresholds $\alpha_1 = 0.5$ and $\alpha_2 = 0.025$. For the 0-shot case, we use top-70% candidate intents, and top-10% for the 5-shot and 10-shot cases. Section 5.5 experiments the choices of the top-K candidate intents.

Table 1 presents the intent detection results on three benchmark datasets. Our FCSLM significantly outperforms those existing PLM-based and LLM-based competitors in all cases, achieving the new state-of-the-art results.

In the 0-shot scenario, the LLM baseline (Qwen2.5-72b) performs relatively well, mainly due to its powerful semantic comprehension capability acquired through large-scale pretraining. In contrast, those PLM-based models, lacking task-specific training examples, struggle to effectively distinguish semantic differences in between intents, resulting in significantly lower performance.

In the 5-shot and 10-shot scenarios, the performance of LLMs slightly declines as the number

Method	BANKING77			CLINC150			HWU64		
	0	5	10	0	5	10	0	5	10
RoBERTa (Liu et al., 2019)	1.14	64.58	81.59	2.29	81.89	90.51	1.86	69.24	79.83
RoBERTa(SSP) (Liu et al., 2019)	9.55	72.82	82.99	14.73	87.40	91.67	10.32	75.19	83.18
Qwen2.5-72b (Yang et al., 2024)	74.48	81.27	80.65	90.04	94.84	93.84	83.64	86.52	86.43
PLE (Li et al., 2022)	-	74.90	79.09	-	88.70	91.20	-	76.46	80.36
DFT++ (Zhang et al., 2023)	-	78.90	86.14	-	-	-	-	79.93	86.21
QAID (Yehudai et al., 2023)	-	85.25	88.83	-	93.41	94.64	-	85.52	87.98
INTENDD (Singhal et al., 2023)	-	85.34	89.62	-	93.52	94.71	-	84.11	88.37
ICDA-XL (Lin et al., 2023)	-	83.90	89.79	-	92.62	94.84	-	82.45	87.41
ICL (Milios et al., 2023)	-	85.29	88.18	-	95.18	95.58	-	86.15	88.57
CPFT (Zhang et al., 2021)	48.63	80.86	87.20	53.11	92.34	94.18	55.41	82.03	87.13
ZeroGen (Ye et al., 2022)	48.41	74.52	84.81	53.26	88.46	91.56	47.84	77.69	84.76
CoDa (Evuru et al., 2024)	58.12	79.72	85.98	66.08	90.41	92.08	59.34	78.69	85.02
PromptMix (Sahu et al., 2023)	75.37	81.43	86.13	74.27	91.68	92.10	74.55	81.91	85.20
CUC (Zhang et al., 2025)	75.26	84.41	89.67	75.63	93.20	94.75	75.05	82.54	87.51
Our FCSLM (RoBERTa+Qwen2.5-72b)	**76.98** (+1.61)	**87.66** (+2.32)	**90.23** (+0.44)	**90.18** (+0.14)	**97.00** (+2.16)	**97.09** (+2.25)	**83.83** (+0.19)	**87.73** (+1.21)	**89.03** (+0.66)

Table 1: Few-Shot Intent Detection Accuracy (%) under 0/5/10-shot settings for three benchmark datasets, The best results are highlighted in bold, and the second-best results are underlined. For FCSLM, the relative improvements over the second-best results are respectively reported in parentheses.

of training examples increases. This is attributed to the full-label enumeration strategy, which substantially lengthens the input prompt with all intent labels, resulting in more semantic redundancy and contextual noise in its inferences. This phenomenon highlights the importance of our proposed top-K candidate intent selection strategy: The small PM model significantly narrows the label space for the LLM, which effectively reduces token overhead and inference complexity for its inference. In contrast, those PLM-based models exhibit performance improvements with the increase of labeled training samples, showing their capabilities of capturing semantic boundaries with the increase of labeled examples.

In short, our FCSLM effectively leverages the complementary strengths of PLMs and LLMs, delivering stable performance gains under different data conditions. Specifically, our FCSLM utilizes the small task-specific PM to handle high-confidence predictions, while delegating only uncertain cases to the LLM for further jurisdiction. This collaborative design not only adapts to varying levels of data resources, but also achieves a favorable trade-off between performance and efficiency.

5.2 Ablation Study

Table 2 presents the ablation study results on the three datasets in the 5-shot scenario. Our FCSLM collaborates a small and a large model in both the training and reasoning phase. In the training phase, the SSP and QAT modules leverage an LLM to generate augmented data for training the PM component, by which the quality of candidate intents supplied by a PM can be improved, in turn helping the LLM for decision in the reasoning phase.

Method	BANKING77	CLINC150	HWU64
FCSLM	**87.66**	**97.00**	**87.73**
w/o SSP	86.69	96.42	86.90
w/o QAT	86.36	96.58	86.43
w/o CPM	86.46	96.87	86.71
w/o LLM	83.25	93.76	82.53
w/o SSP&LLM	81.10	92.71	83.09
w/o QAT&LLM	78.99	92.31	80.76

Table 2: Ablation study of our FCSLM on intent detection accuracy (%) in the 5-shot scenario.

In the reasoning phase, the use of CMP module enables the LLM to perform another round of more cautious reasoning when its initial prediction is inconsistent with that of the PM; while the CSLMP w/o CPM will take the LLM prediction as the output without proceeding to the third round, losing a second-thought chance.

We note that if no LLM is used in the reasoning phase (i.e., w/o LLM), and no LLM is used for data augmentation in the offline phase (i.e., w/o SSP&LLM and w/o QAT&LLM), the performance is significantly degraded. This again indicates the importance of collaborating a large model (LLM) with a small model (PM) in both the reasoning and training phase for the FSID task.

Dataset	Stage	Status	Count	Correct	Acc (%)
BANKING77 (3080)	Round 1 (3080)	Passed	1380	1327	96.16
		Left	1700	1237	72.76
		Stop	3080	2564	83.25
	Round 2 (1700)	Passed	1233	1095	88.81
		Left	467	241	51.61
		Stop	3080	2663	86.46
	Round 3 (467)	Passed	467	278	59.53
		Finish	3080	2700	87.66
CLINC150 (4500)	Round 1 (4500)	Passed	3338	3315	99.31
		Left	1162	904	77.80
		Stop	4500	4219	93.76
	Round 2 (1162)	Passed	905	862	95.25
		Left	257	182	70.82
		Stop	4500	4359	96.87
	Round 3 (257)	Passed	257	188	73.15
		Finish	4500	4365	97.00
HWU64 (1076)	Round 1 (1076)	Passed	496	491	98.99
		Left	580	397	68.45
		Stop	1076	888	82.53
	Round 2 (580)	Passed	410	358	87.32
		Left	170	84	49.41
		Stop	1076	933	86.71
	Round 3 (170)	Passed	170	95	55.88
		Finish	1076	944	87.73

Table 3: Prediction accuracy (%) at each inference round in the 5-shot setting using Qwen2.5-72B.

5.3 Online Reasoning Analysis

Table 3 presents the result-per-round statistics for the online reasoning procedure in the 5-shot scenario. We have some interesting observations. We take the BANKING77 datatset for the following discussions, and similar observations and discussions can be applied to the other two datasets.

The online reasoning of our FCSLM includes three consecutive rounds. It can reject the prediction for some testing queries but forward them to the 2nd round. In the BANKING77 datatset, there are in total 3080 testing queries, among which 1380 queries are made the final decision in the 1st round, and the rest 1700 queries are rejected for prediction. In contrast, we can also stop the whole online reasoning just by the first round via setting the top-1 candidate intent as the final decision. Similarly, the online reasoning can also stop just after the first two rounds, that is, take y_{llm} by the LLM as the final decision.

Those queries that have been rejected in the first two rounds are kind of hard ones. Observing the accuracy of the *'passed'* row in the table, the accuracies of predicted queries are 96.16%, 88.81% and 59.53%, respectively, in the 1st, 2nd and 3rd round. Insisting on predicting those hard queries in the current round results in lower accuracy. Observing the accuracy of the *'left'* row, the accuracy for those hard queries is 72.76%, lower than that of easy ones, i.e., 96.16% in the 1st round.

The multi-round collaborative reasoning of our FCSLM can effectively improve the overall prediction performance. We compare the accuracy 87.66% in the *'finish'* row with those in the two *'stop'* rows. The accuracy of 83.25% of the *'stop'* row in the 1st round represents only using the task-specific PM for prediction, and the 86.47% represents using the LLM to collaborate with the PM for further predicting those hard queries in the 1st round. The accuracy comparison of 87.66% over 86.46% and 86.46% over 83.25% clearly support our design objective of collaborating small and large model for improving prediction performance.

5.4 Effect of LLM Size on Performance

Model	BANKING77	CLINC150	HWU64
Qwen-14B	80.10	88.71	84.29
Qwen-32B	81.93	88.89	84.01
Qwen-72B	81.27	94.84	86.52
FCSLM (Qwen-14B)	85.81	96.49	**87.73**
FCSLM (Qwen-32B)	86.10	96.91	87.55
FCSLM (Qwen-72B)	**87.66**	**97.00**	**87.73**

Table 4: 5-shot accuracy (%) across different LLM sizes on three datasets.

To assess how LLM size affects reasoning performance, we used three versions of the Qwen2.5 model with different parameter scales (Qwen-14B (Yang et al., 2024), Qwen-32B (Yang et al., 2024), and Qwen-72B (Yang et al., 2024)) as the reasoning module within the FCSLM framework. As shown in Table 4, FCSLM consistently demonstrates strong performance across all model sizes and datasets. These results indicate that FCSLM exhibits good adaptability to LLMs of different sizes: even when using the smaller Qwen-14B model, the achieved performance significantly surpasses the corresponding standalone LLM baseline, and outperforms the currently published state-of-the-art methods on all three datasets (see Table 1). In addition, we observed that as the size of the LLM increases, the performance gains tend to plateau. This suggests that FCSLM effectively mitigates the reliance on extremely large models, enabling small-parameter LLMs to achieve a performance level comparable to that of large-parameter models. In summary, FCSLM achieves a favorable balance between performance and resource cost, and demonstrates strong potential for practical deployment.

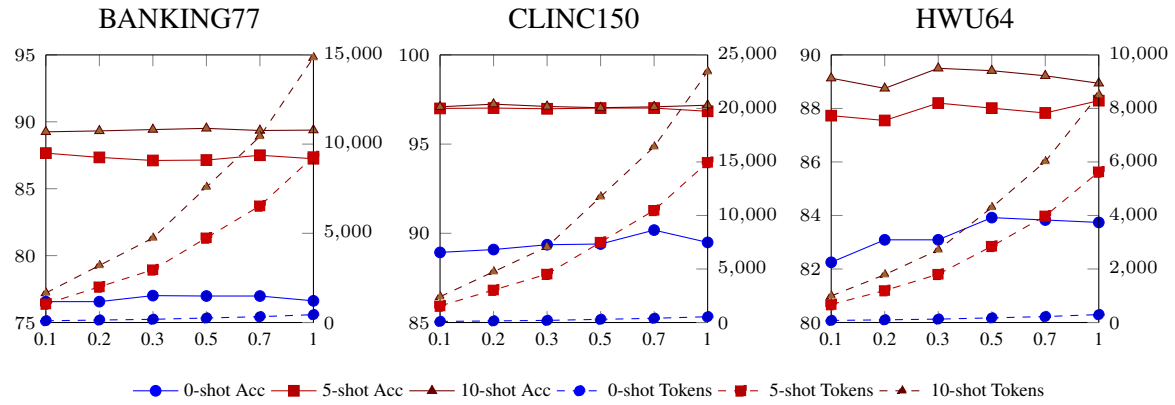

Figure 2: Impact of Top-k ratio (x-axis) on Accuracy (left y-axis, %) and Token Cost (right y-axis) across three datasets under 0-shot, 5-shot, and 10-shot settings.

5.5 Candidate Intent Analysis

Figure 2 plots the prediction accuracy and token cost against the choice of candidate intent percentage, i.e., top-K. Recall that the parameter K controls the number of intents to be integrated into the input prompt for the LLM in the 2rd round reasoning phase. From the results, it can be first observed that using all intents as candidates is not a wise choice. The prediction accuracy is not the best, yet the token cost is the highest. Recall that the prompt includes the descriptions of all candidate intents. So the more candidate intents, the more input tokens. We next can observe that a higher accuracy is often achieved for K in the range of $10\% - 50\%$, with moderate token costs. This is a good news. It indicates that using a subset of high-quality candidate intents can help reducing the inference difficulty for an LLM, as generally the more the choices, the more difficult of a selection.

5.6 Uncertainty Threshold Analysis

In the FCSLM framework, we employ an uncertainty mechanism based on MPS to enhance inference efficiency. To thoroughly investigate the impact of thresholds on model performance and efficiency, we conducted a comprehensive grid search experiment on the first candidate label threshold (f_1) and the second candidate label threshold (f_2), using Qwen2.5-72B as the LLM on the BANKING77 (5-shot) dataset. The experimental results are shown in Figure 3. The heatmap clearly reveals the relationship between accuracy and the LLM invocation rate. The dual-threshold mechanism performs well across a wide range of threshold values, enabling a flexible trade-off between performance and cost. More importantly, we observed a key phenomenon: the system's optimal performance is not achieved when relying solely on the LLM. When all samples are routed to the LLM (i.e., at a 100% invocation rate), the baseline accuracy is 87.37%. However, by setting the threshold combination to $f_1 = 30\%$ and $f_2 = 7.5\%$, our framework achieves a higher accuracy of 87.92%, while the LLM invocation rate drops sharply to 31.92%.

This result provides strong evidence that our framework design achieves a deep synergy between the small model and the LLM. It accurately filters samples that the small model can process with high confidence, routing only the most uncertain cases to the LLM, thereby avoiding unnecessary computational overhead. This strategy not only saves approximately 70% in LLM invocation costs but also optimizes the system's overall decision-making process by leveraging the respective strengths of each model, ultimately achieving performance that surpasses the single-LLM baseline.

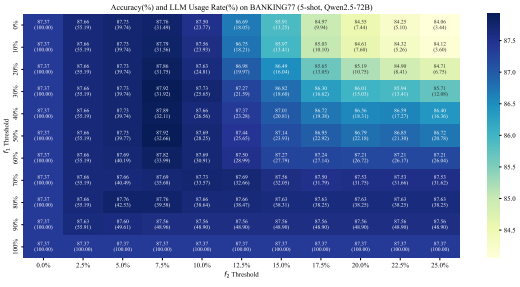

Figure 3: Sensitivity analysis of uncertainty thresholds on the BANKING77 (5-shot) dataset. Each cell displays the accuracy (%), with the corresponding LLM invocation rate (%) provided in parentheses.

Model		Banking			Credit Cards			BANKING77-OOS		
		R_{oos}	P_{oos}	$F1_{oos}$	R_{oos}	P_{oos}	$F1_{oos}$	R_{oos}	P_{oos}	$F1_{oos}$
Qwen2.5		62.57	97.33	76.06	42.57	99.33	59.70	46.02	85.10	<u>59.68</u>
CL(ALBERT) (Zhang et al., 2022)		86.3	57.9	69.87	75.9	55.8	64.54	89.5	39.8	55.00
CL(BERT) (Zhang et al., 2022)		81.8	70.8	76.00	76.5	68.1	72.12	90.9	41.3	56.58
CL(ELECTRA) (Zhang et al., 2022)		89.4	65.1	75.26	75.8	67.1	71.29	87.5	43.0	57.61
CL(RoBERTa) (Zhang et al., 2022)		78.4	78.6	<u>78.50</u>	86.8	63.3	<u>72.75</u>	83.1	46.3	59.52
CL(ToD-BERT) (Zhang et al., 2022)		75.8	69.4	72.45	72.3	61.1	66.45	82.7	43.8	57.34
FCSLM (Top-K)	20%	86.00	92.90	**89.33** (+10.83)	84.00	96.39	**89.41** (+16.66)	59.44	86.87	**70.58** (+10.90)
	50%	79.71	93.00	85.88	70.57	97.24	81.36	58.43	86.44	69.48
	100%	72.86	94.10	81.74	60.57	97.25	74.39	58.06	82.72	67.95

Table 5: Comparison of OOS detection results. We report OOS recall (R_{oos}), precision (P_{oos}), and F1 score ($F1_{oos}$). The best $F1_{oos}$ results are highlighted in bold, and the best competitors' results are underlined. For FCSLM@20%, the relative improvement over the best competitors' results are respectively reported in parentheses.

5.7 Out-of-Scope detection

Table 5 compares the OOS detection performance of our FCSLM with the existing competitors dealing with the OOS cases. The OOS datasets are detailed in Appendix D.6.

The pure LLM baseline (Qwen2.5) demonstrates relatively high OOS precision (P_{oos}) but significantly lower OOS recall (R_{oos}), primarily due to its use of a full-label enumeration strategy. Since the LLM is required to choose from all in-scope intent labels plus a null label, many OOS queries that are semantically close to existing intents tend to be misclassified as in-scope, thus weakening the model's rejection capability. Conversely, when a query is semantically distant from the in-scope space, the LLM is more likely to assign it to the null label, resulting in higher precision. In contrast, traditional fine-tuned PLM methods (Zhang et al., 2022) often achieve higher OOS recall at the expense of relative lower precision, resulting in overall lower F1 performance.

Our FCSLM achieves the best $F1_{oos}$ performance by the collaborative reasoning mechanism. Notably, OOS queries are more likely to trigger uncertain predictions of a task-specific PM. The PM will forward the uncertain predictions to an LLM, yet in most cases, with reduced candidate intents and labels for the LLM making decisions. This helps ensuring high detection precision while exhibiting strong recall performance. In addition, it can be observed that the $F1_{oos}$ of our FCSLM suffers from the increased number of top-K candidate intents. Notice that 100% candidate intents means to use all available intents, which often leads to much expanded label space for LLM inference in the second round. This observation further confirms the effectiveness of the top-K candidate mechanism in narrowing the label space for reducing LLM inference difficulty.

6 Conclusion

This paper has proposed a collaborative framework, called FCSLM, for the FSID task, which effectively enhances intent classification and out-of-scope detection by collaborating the strengths of a small prediction model and a large language model. In the training phase, the FCSLM leverages LLMs to generate for high-quality augmented data for self-supervised pretraining and supervised fine-tuning a task-specific prediction. In the inference phase, a multi-round collaborative reasoning mechanism is introduced to utilize the complementary inference capabilities of the small prediction model and a large language model. Extensive experiments on three benchmark datasets have shown that our FCSLM outperforms the state-of-the-art competitors in both intent classification and OOS detection performance. Further evaluations (see Appendix 5.4) reveal that our FCSLM maintains strong compatibility with LLMs of varying parameter sizes, highlighting both the effectiveness of its collaborative design and its potential for practical deployment.

Future work will focus on further improving OOS detection by deeply mining the capabilities of both small and large models, while preserving in-scope intent classification performance.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant 62172167.

Limitations

In zero-shot scenarios, the absence of real few-shot queries makes it difficult to clearly convey the semantic meaning of intents. As a result, the LLM lacks sufficient semantic grounding labels when generating more training data, which hinders the PLM's ability to learn the semantic distinctions between intents.

Ethics Statement

This paper has no particular ethic consideration.

References

Gaurav Arora, Shreya Jain, and Srujana Merugu. 2024. Intent detection in the age of LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1559–1570, Miami, Florida, US. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, and 1 others. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Chandra Kiran Evuru, Sreyan Ghosh, Sonal Kumar, Ramaneswaran S, Utkarsh Tyagi, and Dinesh Manocha. 2024. CoDa: Constrained generation based data augmentation for low-resource NLP. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3754–3769, Mexico City, Mexico. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Guodun Li, Yuchen Zhai, Qianglong Chen, Xing Gao, Ji Zhang, and Yin Zhang. 2022. Continual few-shot intent detection. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 333–343, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yen-Ting Lin, Alexandros Papangelis, Seokhwan Kim, Sungjin Lee, Devamanyu Hazarika, Mahdi Namazifar, Di Jin, Yang Liu, and Dilek Hakkani-Tur. 2023. Selective in-context data augmentation for intent detection using pointwise V-information. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1463–1476, Dubrovnik, Croatia. Association for Computational Linguistics.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021. Benchmarking natural language understanding services for building conversational agents. In *Increasing naturalness and flexibility in spoken dialogue interaction: 10th international workshop on spoken dialogue systems*, pages 165–183. Springer.

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*.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *arXiv preprint arXiv:2009.13570*.

Aristides Milios, Siva Reddy, and Dzmitry Bahdanau. 2023. In-context learning for text classification with many labels. In *Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP*, pages 173–184, Singapore. Association for Computational Linguistics.

Radford M Neal and 1 others. 2011. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

Iftitahu Nimah, Meng Fang, Vlado Menkovski, and Mykola Pechenizkiy. 2021. ProtoInfoMax: Prototypical networks with mutual information maximization for out-of-domain detection. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1606–1617, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Preprint*, arXiv:1908.10084.

Gaurav Sahu, Olga Vechtomova, Dzmitry Bahdanau, and Issam Laradji. 2023. PromptMix: A class boundary augmentation method for large language model distillation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5316–5327, Singapore. Association for Computational Linguistics.

Bhavuk Singhal, Ashim Gupta, V P Shivasankaran, and Amrith Krishna. 2023. IntenDD: A unified contrastive learning approach for intent detection and discovery. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14204–14216, Singapore. Association for Computational Linguistics.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in atis? In *2010 IEEE Spoken Language Technology Workshop*, pages 19–24. IEEE.

Pei Wang, Keqing He, Yejie Wang, Xiaoshuai Song, Yutao Mou, Jingang Wang, Yunsen Xian, Xunliang Cai, and Weiran Xu. 2024. Beyond the known: Investigating LLMs performance on out-of-domain intent detection. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2354–2364, Torino, Italia. ELRA and ICCL.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022. ZeroGen: Efficient zero-shot learning via dataset generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11653–11669, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Asaf Yehudai and Elron Bendel. 2024. When llms are unfit use fastfit: Fast and effective text classification with many classes. *Preprint*, arXiv:2404.12365.

Asaf Yehudai, Matan Vetzler, Yosi Mass, Koren Lazar, Doron Cohen, and Boaz Carmeli. 2023. Qaid: Question answering inspired few-shot intent detection. *arXiv preprint arXiv:2303.01593*.

Feng Zhang, Wei Chen, Fei Ding, Meng Gao, Tengjiao Wang, Jiahui Yao, and Jiabin Zheng. 2024a. From discrimination to generation: Low-resource intent detection with language model instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10167–10183, Bangkok, Thailand. Association for Computational Linguistics.

Feng Zhang, Wei Chen, Meng Gao, Fei Ding, Tengjiao Wang, Jiahui Yao, and Jiabin Zheng. 2025. Clear up confusion: Iterative differential generation for fine-grained intent detection with contrastive feedback. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2207–2221, Abu Dhabi, UAE. Association for Computational Linguistics.

Haode Zhang, Haowen Liang, Liming Zhan, Albert Y.S. Lam, and Xiao-Ming Wu. 2023. Revisit few-shot intent classification with PLMs: Direct fine-tuning vs. continual pre-training. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11105–11121, Toronto, Canada. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021. Few-shot intent detection via contrastive pre-training and fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1912, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5064–5082, Online. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Yao Wan, Zhiwei Liu, Ye Liu, Caiming Xiong, and Philip Yu. 2022. Are pre-trained transformers robust in intent classification? a missing ingredient in evaluation of out-of-scope intent detection. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 12–20, Dublin, Ireland. Association for Computational Linguistics.

Tianyi Zhang, Atta Norouzian, Aanchan Mohan, and Frederick Ducatelle. 2024b. A new approach for fine-tuning sentence transformers for intent classification and out-of-scope detection tasks. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 910–919, Miami, Florida, US. Association for Computational Linguistics.

Wenxuan Zhou, Fangyu Liu, and Muhao Chen. 2021. Contrastive out-of-distribution detection for pre-trained transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1100–1111, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Related Work

A.1 Few-Shot Intent Detection

In real-world applications, the frequent emergence of new intents and the scarcity of annotated data make few-shot intent detection a critical research

focus. Recent studies in this field have evolved along three primary directions. The first line of work explores how to fine-tune PLMs under low-resource settings, enhancing their semantic modeling capabilities through techniques such as unsupervised pretraining and contrastive learning (Zhang et al., 2021; Yehudai et al., 2023; Singhal et al., 2023). The second direction leverages large language models (LLMs) as data generators, utilizing their strong generalization ability to augment training samples, followed by downstream training on PLMs (Zhang et al., 2023; Lin et al., 2023; Zhang et al., 2025). The third and most recent approach introduces LLMs directly into the inference stage, exploiting their natural language reasoning capabilities for few-shot or zero-shot intent classification. These methods have demonstrated strong generalization (Zhang et al., 2024a; Arora et al., 2024). Although each of these approaches addresses different aspects of the task, they generally fail to fully integrate the complementary strengths of PLMs and LLMs. In particular, methods that employ LLMs for inference often rely on a full-label enumeration strategy, where the model is required to choose from the entire set of intent labels. This strategy incurs high computational costs and is susceptible to interference from the label space, thereby undermining inference stability and accuracy.

A.2 OOS Detection

Early OOS detection methods are typically trained based on traditional classifiers and use the maximum softmax probability or entropy as a rejection score (Larson et al., 2019). Subsequently, researchers proposed distance-based methods, constructing the semantic distribution of samples in embedding space and computing the distance between the input and known class centers or prototypes to determine whether the input is OOS (Nimah et al., 2021). These methods provide some geometric interpretability but remain insufficient in scenarios with fuzzy class boundaries and high semantic overlap. Further works (Zhou et al., 2021; Zhang et al., 2024b) introduced regularization designs to compress the distribution of In-Scope representations, making In-Scope samples form more compact clusters in the semantic space, thereby enhancing the ability to distinguish OOS inputs and improving inter-class discriminability.

However, most of the above methods assume access to abundant training samples, which limits their transferability to the low-resource setting commonly found in real-world applications—i.e., where only a few known intent samples are available and no OOS data is provided. Under this condition, achieving high-quality OOS detection in an unsupervised manner becomes a key challenge.

Moreover, Zhang et al. (Zhang et al., 2022) divide open-domain intent detection into two types: In-Domain OOS (ID-OOS) and Out-of-Domain OOS (OOD-OOS), and conduct few-shot experiments under the ID-OOS setting using a discriminative nearest-neighbor classification method combined with deep self-attention mechanisms (Zhang et al., 2020). Although this method improves OOS rejection capability, it comes at the cost of significantly reduced In-Scope intent recognition accuracy. The authors point out that even under the In-Domain OOS setting, few-shot OOS detection still faces considerable challenges.

B Boundary-Aware Data Augmentation

To further improve the model's discrimination ability under zero-shot conditions, we design a *boundary-aware data augmentation* (BADA) strategy. Traditional LLM-based augmentation typically generates synonyms based on original samples. However, in settings with insufficient guidance and fine-grained intent distinctions, the LLM may inadvertently cross semantic boundaries during generation, extending a sample from one category into a region semantically close to another, leading to confusion and degrading the performance of the pre-trained language model. To address this issue, we construct a confusion intent set for each category y in the vector space, based on semantic similarity across categories. Specifically, we compute vector representations for all intent labels and calculate cosine similarity between them. From each category's perspective, we select a set of non-matching labels whose similarity is below a threshold (cosine similarity < 0.6), forming the potential confusion set $\{y_1^c, y_2^c, ..., y_m^c\}$. This confusion set is then incorporated into the prompt to guide the LLM in focusing on semantic boundaries during generation, thus producing more discriminative samples. In addition to the original augmentation strategy, we design two types of augmented samples: one type aims to enhance inter-category distinctiveness by prompting the LLM to generate expressions that clearly distinguish confusing categories; the other type simulates samples that are semantically close to confusing categories but

still clearly belong to the current category, helping the model remain accurate in boundary-blurred situations.

All sentence representations in augmentation approaches are generated using the pre-trained sentence encoder all-MiniLM-L6-v2. We instruct the LLM to generate 20 examples for each intent class under each type of augmented sample. The augmentation prompt is presented in Appendix F.3.

C Discriminative Intent Description Generation

DIDG identifies confusion patterns between different intents and constructs differentiated expressions based on this, thereby generating more discriminative semantic descriptions to assist downstream LLMs in making more accurate judgments during the intent detection phase.

The core process of DIDG consists of two stages. First, we use the LLM to generate a general intent description for each intent by leveraging its example samples, which serves as a basic semantic understanding of the intent category. Then, we use the LLM to automatically predict intents for all samples in $\mathcal{D}^{y}_{\text{few}}$ by leveraging both the intent's example samples and its general description, and record the consistency between the model's outputs and the true labels. By analyzing all misclassified samples, we construct the corresponding confusion intent set $\{y^c_1, y^c_2, ..., y^c_m\}$ for each target intent y, i.e., the categories that are most easily confused during actual reasoning.

Next, we input the target intent and its confusion set into the LLM to construct a prompt containing contrastive context, asking the model to generate a highly differentiable intent description. The generated discriminative intent description directly serves as a complementary explanation to the general description. The final intent descriptions are cached and used as auxiliary semantic prompts in the LLM-based judgment stage. The DIDG prompt is presented in Appendix F.5.

D Detailed Experiment Settings

D.1 Datasets Used in the SFT Phase

Table 6 shows example queries for the `atm_support` intent in $\mathcal{D}_{\text{ft}}$. We use the MarianMT models (Junczys-Dowmunt et al., 2018) for back-translation.

Original Intent Label
`atm_support`

Label-Augmented Intent Label
`atm location and acceptance inquiry`

Few-shot Examples
where can i withdraw money from?
which atms accept this bill of fare ?
tell me what atms take this card.
tell me what atms use this card.
can i use my card to withdraw from my account?

Back-translated Examples
Where can I get the money?
Who accepted this fare?
Tell me what atms has on this card.
Tell me what atms is with this card.
Can I use my card to get it out of my account?

LLM-Paraphrase (LP) Examples
What's the closest place I can get cash from?
Which ATMs will take this type of currency?
Which ATMs are compatible with this card?
Where are the nearest ATMs that work with this card?
Can I withdraw funds from my account using this card?

Query Augmentation with Trimming (QAT) Examples
I need to find an ATM that works with my bank.
Which ATMs will let me withdraw cash using my card?
How do I find ATMs that accept my card?
Is it possible to use my card at all ATMs?
Can you help me find an ATM that accepts this card?
...

Table 6: Examples in $\mathcal{D}_{\text{ft}}$ for `atm_support`.

Dataset	Utterance	Intent	Domain
CLINC150 (Larson et al., 2019)	18200	150	10
BANKING77 (Casanueva et al., 2020)	10162	77	1
HWU64 (Liu et al., 2021)	10030	64	21
SNIPS (Coucke et al., 2018)	13084	5	-
ATIS (Tur et al., 2010)	4478	21	-

Table 7: Statistics of datasets used in SSP.

D.2 Datasets Used in the SSP Phase

Table 7 shows the statistics of the five datasets used in the SSP phase. To ensure fairness in the few-shot experiments, we remove the target dataset from the pre-training corpus in each evaluation task. For example, when evaluating on BANKING77, no training samples from BANKING77 are included in the pre-training stage. This setup ensures that CARS's generalization to target few-shot tasks stems entirely from the modeling capabilities of the framework itself, rather than from any leakage of extra training resources. In contrast, (Mehri et al., 2020; Zhang et al., 2021; Yehudai et al., 2023; Singhal et al., 2023) do not avoid such data overlaps when using the same corpus for both pre-training and evaluation.

D.3 Competitors Details

RoBERTa: This model builds on `roberta-base` (Liu et al., 2019), with an added linear classification layer. Supervised classification fine-tuning is performed on the label set $\mathcal{Y}$ and dataset $\mathcal{D}_{\text{few}}$, without any pre-training or data augmentation. During inference, the class with the highest logit is selected as the prediction.
RoBERTa (SSP): This model extends (1) by incorporating SSP module prior to supervised classification fine-tuning.
Qwen2.5-72B-Instruct: This model performs intent classification using a prompt-based few-shot inference approach with language model Qwen2.5-72B-Instruct (Yang et al., 2024), and does not depend on a separate PM encoder.
QAID (Yehudai et al., 2023) reformulates the intent detection task as a question-answer matching problem and introduces contrastive learning and domain alignment objectives during training to improve the model's generalization ability on new tasks.
CPFT (Zhang et al., 2021) combines self-supervised contrastive pre-training with supervised contrastive learning and intent classification fine-tuning to enhance few-shot intent detection performance.
PLE (Li et al., 2022) introduces a lightweight encoder, task transfer, pseudo-sample replay, and dynamic weighting mechanisms to build an end-to-end continual learning framework, enhancing cross-task transfer and anti-forgetting abilities in few-shot intent detection.
DFT++ (Zhang et al., 2023) is a few-shot intent detection method that does not rely on external corpora. It combines context augmentation based on generative models with a sequential self-distillation strategy to significantly improve the model's generalization and robustness under extremely low-resource conditions.
INTENDD (Singhal et al., 2023) leverages shared encoders and unsupervised contrastive learning for sentence representation, combined with two-stage post-processing on a graph structure (residual propagation and label smoothing) to improve multi-class intent detection performance.
ICDA (Lin et al., 2023) Fine-tunes a PLM to generate intent data, then filters irrelevant samples using PVI to improve intent classification.
ICL (Milios et al., 2023) employs a dense retrieval model to select the most relevant examples for a given input query, and then uses these retrieved examples to construct a prompt for in-context learning. We replaced the retrieval model from RoBERTa-base with roberta-base-nli-stsb-mean-tokens (Reimers and Gurevych, 2019), which is similar in scale but more specialized for sentence similarity tasks, and uniformly used Qwen2.5-72B-Instruct (Yang et al., 2024) as the large language model for in-context learning.
ZeroGen (Ye et al., 2022) A zero-shot learning method that first generates training data from scratch using large pre-trained language models in an unsupervised manner, and then trains a task model on the synthesized data.
CoDa (Evuru et al., 2024) A training-free data augmentation method that prompts large language models with simple, verbalized constraints to generate high-quality pseudo samples.
PromptMix (Sahu et al., 2023) Generates boundary examples and re-labels them to improve data quality for small model training.
CUC (Zhang et al., 2025) An iterative differential generation framework for fine-grained intent detection, incorporating contrastive feedback to guide large language models in generating high-quality pseudo samples. It distinguishes similar classes through differential prompts and refines samples using rubric-based evaluation.

D.4 Performance Metrics

Accuracy: This metric measures whether the predicted intent of each query matches the ground truth. It is used to evaluate model performance on known intent classes and serves as the standard metric for multi-class classification.
OOS Recall Measures the proportion of correctly identified OOS samples among all true OOS examples.
OOS Precision Measures the proportion of true OOS samples among all predicted OOS instances, indicating precision in detecting unknown intents.
OOS F1 The harmonic mean of OOS Precision and OOS Recall, providing a balanced view of the model's ability to reject out-of-scope queries.

D.5 Experiment Details

k-shot Dataset	# Intent	# Test	Domain
CLINC150	150	4500	Multi-domain
BANKING77	77	3080	Banking
HWU64	64	1076	Voice Assistant

Table 8: Statistics of datasets used in SFT.

Table 8 presents the k-shot datasets' statistics. All PM encoders in FCSLM are trained on an RTX 3090 GPU. The SSP phase follows the setup from (Zhang et al., 2021), using 20 training epochs, a batch size of 64, a learning rate of $1e^{-5}$, temperature parameter τ set to 0.07, and contrastive loss balancing factor λ set to 0.1. The SFT phase uses 10 training epochs, a batch size of 16, and a learning rate of $1e^{-5}$. The LLM inference module is accessed via the API interface provided by Alibaba Cloud Bailian platform, with all prompt templates fixed during inference.

All our reported results are the average of three independent runs. Some baseline results are from (Zhang et al., 2025). We reproduced ICL (Milios et al., 2023) with comparably scaled models. In additional, except for ICDA (Lin et al., 2023), which uses RoBERTa-Large (Liu et al., 2019), all other encoder-based baselines adopt RoBERTa-base (Liu et al., 2019).

D.6 OOS datasets

Dataset	IS	ID	Test (IS)	Test (OOS)
Banking	10	5	500	350
Credit Cards	10	5	500	350
BANKING77-OOS	50	27	2000	1080

Table 9: Dataset statistics used in OOS detection. "IS": in-scope intents, "OOS": out-of-scope intents.

We use the **"Banking"** and **"Credit Cards** domains (Zhang et al., 2022) from the CLINC150 dataset. Each domain contains 15 intent classes, from which five are held out as in-domain out-of-scope (ID-OOS) examples, while the remaining ten are used as in-scope (IS) classes. In addition, we adopt the **BANKING77-OOS** dataset (Zhang et al., 2022), constructed by selecting 27 intents from the original BANKING77 dataset as ID-OOS and using the remaining 50 as IS classes. Detailed dataset statistics are shown in Table 9. We include the results of ALBERT, BERT, ELECTRA, RoBERTa, and ToD-BERT trained with the CL method proposed in (Zhang et al., 2022), which serve as baselines for few-shot OOS detection. Our models are trained and evaluated under the same 5-shot IS-only setting, where no OOS examples are used during either training or inference.

Method		B-OOS	Banking	Credit Cards
FCSLM	20%	**89.33**	**89.41**	**70.58**
	50%	85.88	81.36	<u>69.48</u>
	100%	81.74	74.39	67.95
FCSLM w/o CIR	20%	<u>87.06</u>	<u>85.19</u>	66.68
	50%	84.79	78.15	65.53
	100%	80.61	72.67	62.40

Table 10: Ablation results (F1 score only) across different top-K values on three domains. **CARS w/o CIR** denotes the variant without the candidate intent re-ranking mechanism. Bold indicates the best, underline the second-best.

E Supplementary Experiments

E.1 Additional Experiments on CIR

Table 10 presents the ablation study on the candidate intent re-ranking mechanism (CIR). We observe that removing CIR leads to a decrease in $F1_{\text{OOS}}$, indicating that CIR effectively enhances the model's ability to identify OOS samples and improves the overall detection performance.

F Prompt Design

F.1 LLM-based Paraphrasing Prompt Design

Prompt: LLM-based Paraphrasing (LP)

sentence: add transmission to my found them first.
Task:
Please generate one alternative sentence that expresses the same meaning.
Please answer the new sentence directly and do not answer any other content.

F.2 Query Augmentation with Trimming Prompt Design

We instruct the LLM to generate 20 examples for each intent class.

Prompt: Query Augmentation with Trimming (QAT)

Intent Name: atm_support
Few-Shot Queries: where can i withdraw money from?...
Task:
You are tasked with augmenting training data for intent classification.
Based on the few-shot examples above, generate 20 new user queries that express the same intent.
Output:
Return exactly 20 new sentences, each on a new line, with no bullet points, numbering, or extra text.

F.3 BADA Prompt Design For Zero-shot

Prompt: Augmentation for zero-shot

Intent Name: atm_support
Task:
You are tasked with augmenting training data for intent classification.
Generate 20 new user queries that express the same intent.
Output:
Return exactly 20 new sentences, each on a new line, with no bullet points, numbering, or extra text.

Prompt: Discriminative Augmentation for zero-shot

Intent List: {Intent y and its set of confusing intents}.
Task:
There are the above-mentioned intents that are not easy to distinguish in dialogue system.
You are tasked with augmenting training data for intent Intent y.
Generate 20 new user queries that express the same intent.
The generated queries should be close to confusing intentions so that the model can obtain samples that can distinguish confusing relationships.
Output:
Return exactly 20 new sentences, each on a new line, with no bullet points, numbering, or extra text.

Prompt: Boundary Augmentation for zero-shot

Intent List: {Intent y and its set of confusing intents}.
Task:
There are the above-mentioned intents that are not easy to distinguish in my dialogue system.
You are tasked with augmenting training data for intent Intent y.
Generate 20 new user queries that express the same intent.
Please reconsider the differences between Intent y and other intent.
Output:
Return exactly 20 new sentences, each on a new line, with no bullet points, numbering, or extra text.

F.4 Label Augmentation Prompt Design

Prompt: Label Augmentation (LA)

Intent Name: atm_support
Few-Shot Queries: where can i withdraw money from?...
Task:
Rephrase the label based on the provided intent label and its example list to accurately reflect the user's intent.
You must retain all original label words and incorporate more specific description to ensure the label is clear, accurate, unambiguous, and specific, with words separated by spaces.
Do not directly add the words in the example to the newly generated intent.

F.5 DIDG Prompt Design

Prompt: Discriminative Intent Description Generation

Intent Name: {Intent y}
few-Shot Queries: ...
Intent Name: {the confusing intent1 of Intent y}
few-Shot Queries: ...
...
The above are several easily confused intents and their few-shot query examples.
You must create a unique description for each intent.
Make the description that encompasses the provided few-shot queries.
Also, don't use the given use cases examples of intent for the description.
Make the descriptions no longer than 15 words.
Return your output as a JSON object with this format:
{ "intent_name_1": "description", "intent_name_2": "description", ... }
DO NOT include any extra explanations or text outside the JSON format.

F.6 Second Round Prediction Prompt Design

Prompt: Second Round Prediction

Query: how do i locate my card?
Here are the intents with their descriptions and examples:
Intent: card_arrival
Description: Inquiries about the status or delay of a card's delivery.
Example: what is the expected delivery date of my card?...
Intent: lost_or_stolen_card
Description: Seeking assistance or reporting a missing or stolen financial card...
Example: can you help me retrieve my card?...
...
Please select the most suitable intent from the intent list based on the intent describe and intent query example.
Please answer the intent name directly

F.7 Third Round Prediction Prompt Design

Prompt: Third Round Prediction

Query: how do i locate my card?
A model has predicted that the query is likely related to intent 'card_arrival', but it could also belong to intent 'lost_or_stolen_card'.
Please do not decide based solely on the model's prediction.
Carefully analyze the semantics of the query.
Here are some example queries for each intent:
Intent: lost_or_stolen_card
Example:how do i set up my card pin?...
Intent: card_arrival
Example:what is the expected delivery date of my card?...
Step by step, compare the query with both intents and decide which one aligns better semantically.
Finally, answer this question: Is 'card_arrival' more suitable as the intent of the query than 'lost_or_stolen_card'?
Just reply with 'yes' or 'no', without any other text.