A Multi-cluster Boundary Learning Method for Out-of-Scope Intent Detection via MiniLM Embedding

Anonymous ACL submission

Abstract

Intent detection is a critical task that bridges human intents and system actions in human-machine interaction systems. However, there still exist challenges for detecting out-of-scope (OOS) intents. (i) The traditional methods view the OOS intent detection as a multi-class classification, then the detection accuracy decreases as the class number of the known intents increases; (ii) LLM-embedding methods require large parameters, that makes them difficult to train and practically deploy. Thus, this work proposes a multi-cluster boundary learning method to detect OOS intents via MiniLM embedding (i.e., `all-MiniLM-L6-v2`) in an one-class classification workflow. The method learns the boundaries of multi-cluster embeddings generated by MiniLM from the training utterances, and then rejects the out-of-domain utterances as OOS intents. Experiments are conducted on public CLINC150, StackOverflow and Banking77 datasets. The results show that the method achieves the state-of-the-art OOS intent detection performance compared the other baselines. Ablation studies are also conducted and the results show that the used MiniLM can better adapt to the workflow and utterance embedding requirements. The code is available at supplementary materials.

1 Introduction

Intent detection is a critical yet challenging task for human-machine interaction systems. It maps user utterances to system actions, which bridges human intents and system behaviors (Tur et al., 2010; Louvan and Magnini, 2020; Wölflein et al., 2025). However, human intents are complex and cannot be fully enumerated in a closed-set intent detection system. That means if the system receives an out-of-scope (OOS) intent utterance but fails to reject it, that will trigger incorrect actions and induce harmful consequences, as shown in figure 1. Thus, a more important and difficult task

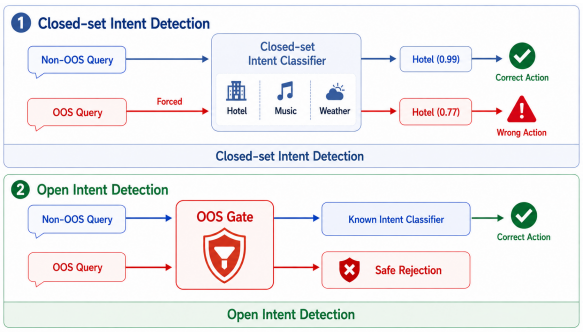

Figure 1: A diagram of OOS intent detection

is to detect the OOS intents and reject them in a timely manner, which is actually an open intent detection task (Hoffman et al., 2024; Muzahid et al., 2024).

There are three types of intent detection methods, including statistical methods, deep-learning methods and LLM-embedding methods. Among them, the statistical methods formulate intent detection as a supervised text classification problem. These methods combine manually designed lexical features with traditional statistical classifiers, such as support vector machine (Haffner et al., 2003) and naive Bayes classifier (Schuurmans and Frasincar, 2020). But the deep-learning methods use deep neural networks, e.g., convolutional neural network (CNN) (Kim, 2014), recurrent neural network (RNN) (Liu and Lane, 2016) and transformer network (Vaswani et al., 2017), to perform intent detection via semantic feature extraction. Early deep-learning classifiers are trained on a closed-set mode, and then identify unknown out-of-domain utterances as known intents (Hendrycks and Gimpel, 2017; Guo et al., 2017). Thus, some types of open intent detection are further proposed to conduct safe rejection for the unknown utterances with OOS intents. They are class-wise open classification method (Shu et al., 2017), distance-based scor-

ing method (Lee et al., 2018), contrastive representation learning method (Zeng et al., 2021) and adaptive boundary learning method (Zhang et al., 2021). After that, with the strong semantic representation ability of LLMs, the LLM-embedding methods are proposed to extract utterance features by semantic embedding, prompt-based reasoning and uncertainty-aware agent routing (Arora et al., 2024).

However, the existing methods still face two limitations. First, the OOS queries with similar semantic meaning as known intents are easily absorbed into the decision regions of the known intents (Li et al., 2025), but in fact they are OOS. Thus, if mix OOS labels into known intent labels, and simply view the OOS intent detection as a multi-class classification, the accuracy of OOS rejection would decrease as the class number increases. Second, although LLM-embedding methods improve the ability of semantic understanding, they still require very large parameters, that induces remarkable computational costs and high sensitivity of prompt designs (Zaera et al., 2025). Thus, they are difficult to train and deploy in real-time resource-constrained dialogue systems.

To address these limitations, this work proposes a multi-cluster boundary learning method for OOS intent detection with a cascade workflow. The workflow decouples the complex open intent detection problem into multiple simpler stages. This allows the workflow to use a lightweight MiniLM instead of LLM wih large-scale parameters. Moreover, it reduces the OOS intent detection to a one-class classification, instead of a complex multi-class classification. Moreover, we found that MiniLM can embed the observation data into multiple clusters. Thus, if the boundaries of clusters are learned, the out-of-domain utterances can be identified as OOS intents.

Thus, the main contributions are as follows:

1. A multi-cluster boundary learning method is proposed. This method learns the boundaries of multi-cluster embeddings generated by MiniLM from the training utterances, and then rejects the out-of-domain utterances as OOS intents.

2. A cascade workflow is proposed to first conduct OOS intent detection as an one-class classification, and then conduct known intent detection as a close-set multi-class classification. This means the OOS intent detection task can be addressed independently.

3 Experiments are conducted on public real-world datasets for OOS intent detection task. The results show that the method achieves the state-of-the-art performance for OOS intent detection. Moreover, ablation studies are also conducted, and the results show that the used MiniLM, `all-MiniLM-L6-v2`, can better adapt to the workflow and utterance embedding requirements.

2 Related Work

The open intent detection methods can be mainly categorized into three types, i.e., statistical methods, deep-learning methods and LLM-embedding methods, according to the problem settings and the types of classifiers.

2.1 Statistical Methods for Intent Detection

The statistical methods use hand-crafted lexical features with statistical classifiers to detect the intent classes. Among them, Wang et al. (2002) proposes a method combining rule-based grammars with statistical classifiers, but the grammar must be hand-crafted for each domain. To reduce the dependence, Haffner et al. (2003) use support vector machine as discriminative model, which shows strong performance on intent classification without rule-based components. But these methods still rely on manually designed features. Thus, Tur et al. (2010) proposes an n-gram method to model local contextual dependencies, but feature sparsity limits its effectiveness on diverse intent expressions. Then, Schuurmans and Frasincar (2020) compares these methods, and shows that dense word embeddings outperform the sparse features for intent classification.

2.2 Deep-Learning Methods for Intent Detection

Then, the deep-learning methods use artificial neural networks to parametrize intent classifiers and learn dense semantic representations. E.g., CNN is used to extract the local semantic patterns from word embeddings (Kim, 2014), and RNN is used to capture the sequential dependencies for joint intent detection and slot filling (Liu and Lane, 2016). Moreover, Transformer model is used to directly capture the global semantic dependencies through self-attention mechanism (Vaswani et al., 2017).

These methods improve the in-distribution accuracy, but the classification mode is still closed-set, and the classifier can only identify unknown utterances with known intent labels, and cannot reject the OOS intents (Hendrycks and Gimpel, 2017; Guo et al., 2017).

Thus, to reject the OOS intents, there are several methods, that view the OOS intent detection as a open-set classification task. E.g., Hendrycks and Gimpel (2017) set an extra OOS class, and transform the open-set classification into a closed-set one. Then, the method assigns regular scores to each class through softmax probability. Also, Lee et al. (2018) propose a Mahalanobis-based scoring method, that models the score as class-conditional Gaussian distribution. But these score assumptions usually violate the real-world class distributions. Moreover, with the increasing class number, the OOS labels couple with many known intent labels, then the score of OOS class is diluted and the accuracy decreases. To improve the representation separability, Zeng et al. (2021) propose a supervised contrastive learning, that learns discriminative embeddings to better separate intent categories. But this still yields limited improvements.

Then, Shu et al. (2017) drop the transformation and propose a open classification method. The method replaces the softmax probability with one-vs-rest sigmoid classifiers, and then calculates a class-wise rejection threshold. If the softmax score is over the threshold, the input utterance would be rejected. Then, Zhang et al. (2021) propose an decision boundary learning method to learn a boundary from all embedding known intents, and the external region is OOS. But the method can only learn a single centroid for each intent. Actually, the embeddings usually have multiple centroids based on our observation. Then, Li et al. (2025) propose a multi-granularity boundary learning method to learning multiple boundaries for each intent.

2.3 LLM-Embedding Methods for Intent Detection

Then, with the LLM technique, the LLM-embedding methods are proposed to strengthen the semantic representations and feature embeddings for user utterance by multi-agent routing mode. Among them, Arora et al. (2024) proposes a hybrid method, that uses a Sentence Transformer to route uncertain intent predictions to a designated LLM. But this LLM is only designed for uncertain queries and is not trained for OOS queries. Then, Zaera et al. (2025) propose an uncertainty-aware routing method, that only triggers a fine-tuned LLM according to the Monte Carlo dropout criterion. But the routing decisions are still unstable if there exists distribution drift. And, the multi-agent LLMs require large computation cost, that make it difficult to deploy for real-world applications. Moreover, Chen and Wang (2025) propose a small-large model collaboration method for few-shot intent detection. The method uses LLM to generate OOS utterance and conduct data augmentation during pretraining. And then, it uses the smaller deep-learning models to realize intent detection.

Thus, the existing methods still face two problems. (i) the OOS intent detection is usually viewed as a multi-class classification and hardly separated from the known intent detection. (ii) The multi-agent LLMs require remarkably large computation cost. And due to the reason (i), the current routing mechanism is not accurate.

3 Method

3.1 Cascade Workflow for Intent Detection

The proposed method uses a three-stage cascade: gate $\rightarrow$ router $\rightarrow$ expert, as shown in figure 2. Given an input utterance x, the gate module first determines the OOS rejection. If x is rejected, the gate module outputs an OOS label, otherwise, x is inputted into the following modules. Then, the router module discriminates the coarse-grained class domain for the remained known intents. Then the utterance x is delivered to an expert module w.r.t the class domain for fine-grained discrimination of a few classes. Note that, due to the fact that the known intent classification is close-set, all intents for training is known, thus the router and experts are all parametrized by SmolLM-135M, that is supervisedly fine-tuned through low-rank adaptation (LoRA) (Hu et al., 2022) technique on the observation data of known intents.

For an input utterance x, the gate $G(\cdot)$ first checks whether x is OOS:

$$z = G(x), \quad z \in \{\text{ID}, \text{OOS}\}, \tag{1}$$

where z is the binary decision, ID means the input is in-distribution, and OOS means the input is out-of-scope. If $z = \text{OOS}$, an OOS label is returned. Otherwise, the router $R(\cdot)$ predicts the domain:

$$d = R(x), \quad d \in \mathcal{D}, \tag{2}$$

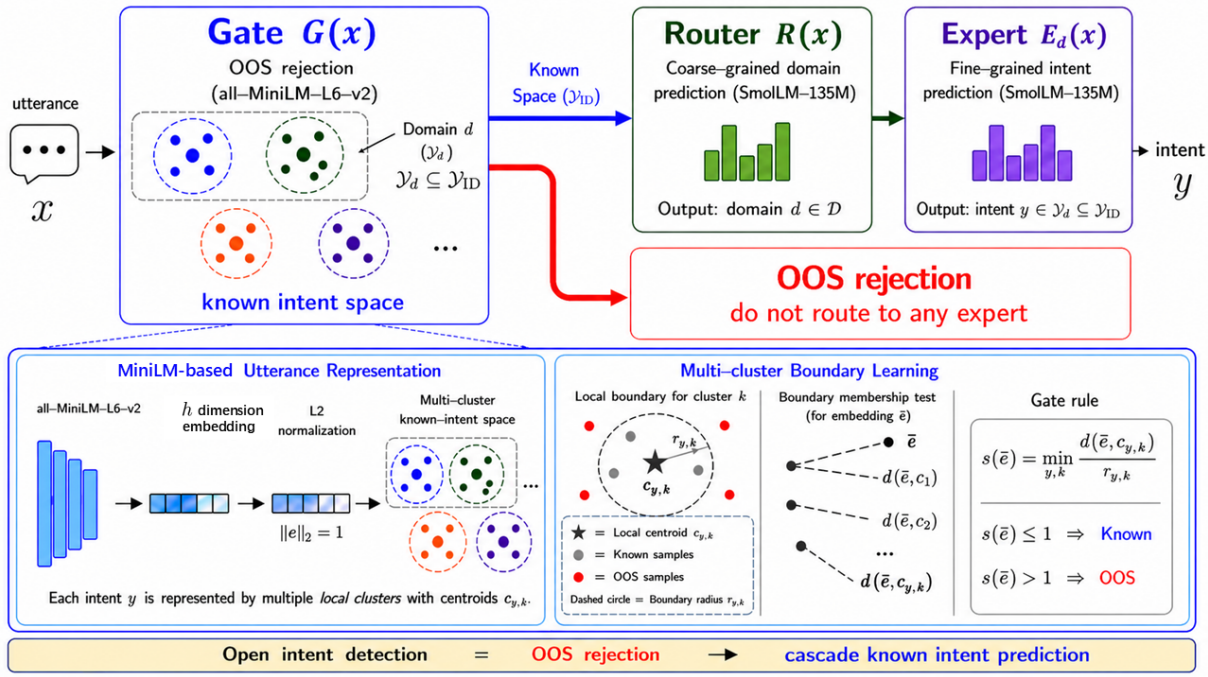

Figure 2: An Overview of the cascaded workflow via MiniLM-based multi-cluster boundary learning.

where $\mathcal{D}$ is the domain set. Then the expert $E_d(\cdot)$ classifies the intent:

$$y = E_d(x), \quad y \in \mathcal{Y}_d \subseteq \mathcal{Y}_{\text{ID}}, \tag{3}$$

where $\mathcal{Y}_{\text{ID}}$ is the known intent label set and $\mathcal{Y}_d$ is the intent subset for domain d.

The proposed cascade workflow separates OOS detection from known-intent classification. Each module handles a simpler sub-task to reduces the complexity. As a result, the Gate only needs to determine whether an input belongs to the known-intent space. Thus, a lightweight model is sufficient.

3.2 MiniLM-Based Intent Embedding

Since the gate only needs to distinguish OOS from known-intent samples, a lightweight MiniLM encoder, `all-MiniLM-L6-v2`, is used to construct the semantic representation space. The MiniLM maps each input utterance x to an embedding with fixed dimension:

$$e = f_\theta(x), \quad e \in \mathbb{R}^h, \tag{4}$$

where $f_\theta(\cdot)$ denotes the MiniLM-based encoder parameterized by θ, and h denotes the embedding dimension. Note that, the MiniLM with parameter θ has no need to train in this work, that implicitly assumes the MiniLM can directly embed the utterance into multiple clusters, and in fact, that is indeed the case through empirical experiments. Then, the gate applies L_2 normalization before computing distances to the centroids:

$$\bar{e} = \frac{e}{\|e\|_2}. \tag{5}$$

The normalized embedding $\bar{e}$ is then used to construct the known-intent geometry for OOS detection.

The embedding space is not used solely as a generic feature space. It also defines the local structure of known intents.But a single-centroid assumption can be too restrictive.Because utterances with the same intent label may differ in wording, expression, and semantic focus. Thus, forcing a known intent into a single compact region can produce an inaccurate boundary. This increases the risk of rejecting valid in-domain samples or accepting OOS samples. Thus, each known intent is represented by multiple local clusters, and each cluster has a centroid.

For intent y, its centroid set is defined as:

$$\mathcal{C}_y = \{c_{y,1}, c_{y,2}, \ldots, c_{y,K_y}\}, \tag{6}$$

where $c_{y,k}, k = 1, \ldots, K_y$, denotes the k-th centroid of known intent y, and K_y is the number of centroids. The centroids are constructed via K-means clustering within each intent. The number of centroids is controlled by a global setting and

can be overridden for specific intents. If the number of known intents is M and each intent has K centroids, the gate maintains $M \times K$ centroids in total. These centroids are local geometric representatives of the known intents. Thus the representation provides the basis for boundary learning.

3.3 Multi-cluster Boundary Learning for OOS Intent Detection

The multi-cluster representation defines local semantic regions for known intents, and OOS detection is therefore formulated as a boundary learning problem in the embedding space. An input is accepted if it falls inside at least one learned known-intent cluster region and is rejected if it lies outside all known-intent regions. To achieve this, the gate learns a local boundary around the centroid of each cluster to separate supported inputs from OOS inputs.

For the centroid of each cluster $c_{y,k}$, the gate estimates a local radius $r_{y,k}$ from the training samples assigned to that specific cluster. In the default configuration, the radius is computed as follows:

$$r_{y,k} = \mu_{y,k} + \lambda \sigma_{y,k}, \tag{7}$$

where $\mu_{y,k}$ and $\sigma_{y,k}$ are the mean and standard deviation of the distances from the assigned samples to $c_{y,k}$, and λ controls the boundary width.

The distance function is a diagonal Mahalanobis distance. For an embedding $\bar{e}$ and a centroid c, the distance is computed as:

$$d(\bar{e}, c) = \sqrt{\sum_j (\bar{e}_j - c_j)^2 \frac{1}{\sigma_j^2 + \epsilon}}, \tag{8}$$

where $\bar{e}_j$ is the j-th component of $\bar{e}$ and c_j is the same componenet of c. Moreover, σ_j^2 is the corresponding feature-wise variance estimated from samples assigned to the corresponding cluster, and ϵ is a small regularization constant. This distance weights each embedding dimension by its estimated variance, which reduces the influence of dimensions with large natural variation.

For an incoming utterance, the gate compares its normalized embedding with all known-intent centroids and computes the normalized nearest-boundary score:

$$s(\bar{e}) = \min_{y \in \mathcal{Y}_{\text{ID}},\, 1 \le k \le K_y} \frac{d(\bar{e}, c_{y,k})}{r_{y,k}}, \tag{9}$$

where $c_{y,k}$ denotes the k-th local centroid of known intent y, and $r_{y,k}$ denotes the corresponding local boundary radius. The ratio of Eq. (9) measures the distance from $\bar{e}$ to this centroid relative to the learned local boundary size.

The gate decision is:

$$G(x) = \begin{cases} \text{ID}, & s(\bar{e}) \le 1 \\ \text{OOS}, & s(\bar{e}) > 1 \end{cases} \tag{10}$$

The threshold is one because each distance is normalized by its radius. If $s(\bar{e}) \le 1$, the input falls inside at least one known-intent region and is accepted. If $s(\bar{e}) > 1$, the input lies outside all known-intent regions and is rejected as OOS. The gate only decides whether an input belongs to the known intent space. It does not assign a specific intent label.

4 Experiments

4.1 Datasets

Three datasets are used for performance evaluation, as shown in Table 1. For each dataset, they are split into training, validation and test sets under 6:1:3.

CLINC150 is a large-scale multi-domain dataset with 150 in-scope intents and explicit OOS samples (Larson et al., 2019). The data source is available at `https://github.com/clinc/oos-eval`.

StackOverflow is a technical-domain short-text dataset with 20 intent classes and substantial semantic overlap across classes. The data source is available at `https://github.com/jacoxu/StackOverflow`.

Banking77 is a fine-grained, single-domain dataset with 77 intent classes. It focuses on high-granularity classification in the financial domain. The data source is available at `https://github.com/PolyAI-LDN/task-specific-datasets`.

Dataset	#Intents	#Samples	Type
CLINC150	150	22500	Multiple domains
StackOverflow	20	20000	Technical
Banking77	77	13083	Banking

Table 1: Dataset statistics.

4.2 Baselines and Experimental Settings

This work defines Known Intent Ratio (KIR) to describe the class number of all intents that split into the known intent class from datasets, as follows:

$$\text{KIR} = \frac{|\mathcal{Y}_{\text{ID}}|}{|\mathcal{Y}|}, \tag{11}$$

where $\mathcal{Y}$ is the full intent label set and $\mathcal{Y}_{\text{ID}}$ is the subset of intents treated as known during training.

Moreover, the following state-of-the-art methods are selected as baselines:

MSP (Hendrycks and Gimpel, 2017) uses the maximum softmax probability as the confidence score for detecting misclassified or out-of-distribution samples.

OpenMax (Bendale and Boult, 2016) recalibrates activation scores and estimates the probability that an input belongs to an unknown class.

DOC (Shu et al., 2017) replaces the softmax layer with independent sigmoid classifiers and performs class-wise open-set rejection.

DeepUnk (Lin and Xu, 2019) uses margin loss to learn discriminative intent features and applies density-based novelty detection for unknown intent detection.

KNNCL (Zhou et al., 2022) combines KNN-based contrastive representation learning with density-based outlier detection for OOD intent classification.

ADB (Zhang et al., 2021) learns adaptive class-specific decision boundaries for open intent detection.

DA-ADB (Zhang et al., 2023) improves ADB with distance-aware representation learning and adaptive boundary learning.

Moreover, for the proposed cascade workflow, the model in gate module is 22M-parameter `all-MiniLM-L6-v2`. The models in router and expert modules are both 135M-parameter `SmolLM-135M`. The rank of LoRA (see section 2.1) is set as 32 for router, and set as 16 for expert. The scaling factor of LoRA is set as 64 for router, and set as 32 for expert. Due to the usage of LoRA, the parameter scaling is about 160M, which is close to that of BERT-base (Zeng et al., 2021; Zhang et al., 2021; Li et al., 2025). The models are all trained on NVIDIA RTX 5070 GPU. The optimizer is AdamW with default settings. The batch size is 32. The learning rate is 2×10^{-4}. The sub-centroid number for each intent y in Eq. (6) is set as $K_y = 2$. The boundary scaling factor in Eq. (7) is set as $\lambda = 0.5$ for CLINC150 and $\lambda = 1$ for the other datasets. The embedding dimension in Eq. (4) is set as $h = 384$.

4.3 Evaluation Metrics

To evaluate and compare the performances of all methods, Known F1 score, OOS F1 score and Acc are selected as metrics.

Known F1 score (Liu et al., 2021; Zawbaa et al., 2024) is calculated by averaging F1 scores over all known intent classes, as follows:

$$\text{Known F1} = \frac{1}{|\mathcal{Y}_{\text{ID}}|} \sum_{y \in \mathcal{Y}_{\text{ID}}} \frac{2P_y R_y}{P_y + R_y}, \tag{12}$$

where $\mathcal{Y}_{\text{ID}}$ denotes the known-intent label set. P_y and R_y denote the precision and recall of class y, respectively.

OOS F1 score (Liu et al., 2021; Zawbaa et al., 2024) is calculated for evaluating OOS detection performance, as follows:

$$\text{OOS F1} = \frac{2P_{\text{OOS}} R_{\text{OOS}}}{P_{\text{OOS}} + R_{\text{OOS}}}, \tag{13}$$

where

$$\begin{aligned} P_{\text{OOS}} &= \frac{\text{TP}_{\text{OOS}}}{\text{TP}_{\text{OOS}} + \text{FP}_{\text{OOS}}}, \\ R_{\text{OOS}} &= \frac{\text{TP}_{\text{OOS}}}{\text{TP}_{\text{OOS}} + \text{FN}_{\text{OOS}}}. \end{aligned} \tag{14}$$

TP_{OOS}, FP_{OOS} and FN_{OOS} denote true positives, false positives and false negatives w.r.t. the OOS class, respectively.

Acc is the overall accuracy over all test samples, as follows:

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i), \tag{15}$$

where N is the size of test set, y_i and $\hat{y}_i$ are the ground truth and the predicted labels of the i-th sample, respectively. And $\mathbb{I}(\cdot)$ is the indicator function.

4.4 Performance on OOS Intent Detection

The performance on OOS intent detection is presented in table 2. The results show that the proposed method achieves the state-of-the-art OOS F1 scores across all settings. Compared to the baselines, the proposed method achieves 0.85%∼17.12% improvements on OOS F1 scores.

As shown in table 2, the OOS F1 scores of all methods decrease as KIR increases. That means the dense known intent distributions intensify the semantic overlap between in-domain and OOS queries.

But the proposed method performs stability on OOS F1 scores under different KIRs. A possible reason is that our method maps user utterance into multiple clusters, that helps draw a clearer boundary compared to single cluster (Zhang et al., 2021,

KIR	Method	CLINC150			StackOverflow			Banking77		
		Known F1	OOS F1	Acc	Known F1	OOS F1	Acc	Known F1	OOS F1	Acc
0.25	MSP	51.02	59.26	53.38	42.66	11.66	27.91	50.47	39.42	42.19
	OpenMax	62.65	77.51	70.27	47.51	34.52	38.97	53.42	48.52	47.76
	DOC	75.46	90.78	86.08	56.30	62.50	57.75	65.16	76.64	70.31
	DeepUnk	76.95	91.61	87.18	47.39	36.87	40.03	64.97	76.98	70.68
	KNNCL	78.85	93.56	89.87	41.79	15.26	28.65	65.54	79.34	73.01
	ADB	77.85	92.36	88.30	77.62	90.96	86.75	70.92	85.05	79.33
	DA-ADB	**79.57**	93.20	89.48	**80.87**	92.65	89.07	73.05	86.57	81.19
	Ours	71.75	**95.01**	**90.45**	73.61	**94.47**	**91.04**	**75.83**	**93.99**	**89.07**
0.50	MSP	72.82	63.71	66.68	66.28	26.94	53.23	73.20	46.29	61.67
	OpenMax	79.83	82.15	80.22	69.88	46.11	60.27	75.16	55.03	65.53
	DOC	83.84	87.45	85.19	77.37	71.18	73.88	78.38	72.66	74.60
	DeepUnk	83.30	87.48	84.95	67.67	35.80	55.46	75.61	67.80	71.01
	KNNCL	83.25	87.85	85.32	61.50	8.50	45.38	75.16	67.21	70.41
	ADB	85.12	88.60	86.54	85.32	87.70	86.49	81.39	79.43	79.61
	DA-ADB	**85.58**	90.10	**87.93**	**86.71**	88.86	**87.78**	**82.54**	79.93	**81.51**
	Ours	79.95	**91.96**	86.78	75.48	**89.71**	85.54	74.90	**88.23**	78.98
0.75	MSP	83.65	63.86	76.19	81.42	37.86	73.20	84.99	46.05	77.08
	OpenMax	71.14	75.18	75.36	82.98	49.69	75.78	85.50	53.02	78.32
	DOC	87.91	83.87	85.93	85.64	65.32	80.55	84.14	63.51	78.94
	DeepUnk	86.57	82.67	84.61	80.51	34.38	71.56	81.65	50.57	74.73
	KNNCL	86.14	82.05	84.12	76.16	7.19	65.01	81.76	51.42	74.78
	ADB	**88.97**	84.85	86.99	86.91	74.10	82.89	**86.44**	67.34	**81.39**
	DA-ADB	88.43	86.00	**87.39**	**87.66**	74.55	83.56	85.93	69.37	81.12
	Ours	66.32	**87.10**	79.83	80.18	**75.57**	81.32	70.28	**86.49**	77.84

Table 2: Performance comparison for all methods under different KIRs.

2023). Moreover, the proposed method achieves outstanding performance on Banking77 dataset when KIR=0.25, but ranks second or third on known intent detection task. That means the supervised fine-tuning of MiniLMs used in router and experts yields unsatisfactory performance. But we think it still provides some insights for known intent detection task, though it is not the main contribution of this work. And the cascade workflow actually separates the OOS rejection and known intent detection process. Thus, if the other baseline methods can perform better than ours, it is easy to replace the router-experts pipeline with that method.

4.5 Interpretability Analysis for Gate Stage

Figure 3 provides an example, that shows the MiniLM embedding results on two dimensions. It is clear that the embeddings of known intents generate multiple clusters, and the utterance embedding with the same intent class would concentrate into the same cluster. This makes it easy to discover the boundary for all known intents, especially with higher dimension, and the outside space is out-of-scope. A clearer evidence to show the separation is presented in figure 4. According to our method, the threshold score strictly separates the two distributions of the known intent samples and OOS intent samples.

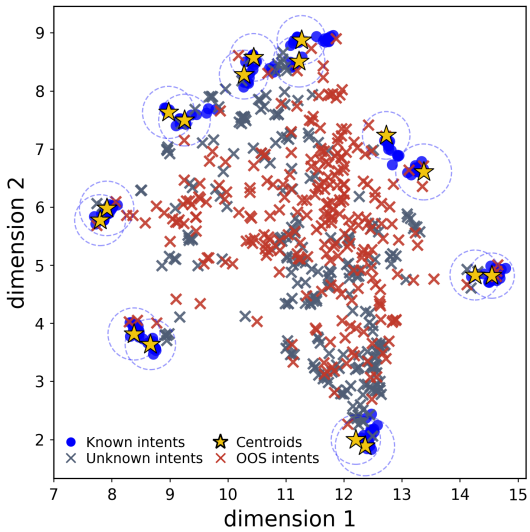

Figure 3: An example of MiniLM embeddings from CLINC150 dataset when KIR = 0.50.

Moreover, some OOS intent samples are used

KIR	Variant	CLINC150		StackOverflow		Banking77	
		Acc	OOS F1	Acc	OOS F1	Acc	OOS F1
0.25	**Ours**	**90.45**	**95.01**	**91.04**	**94.47**	**89.07**	**93.99**
	Without Gate	67.49	76.88	81.80	88.30	84.90	91.27
	Cascade-MiniLM	90.07	94.14	79.73	86.85	85.10	91.41
	Cascade-SmolLM	84.76	90.98	40.28	45.00	51.20	64.43
0.50	**Ours**	**86.78**	**91.96**	**85.54**	**89.71**	**78.98**	**88.23**
	Without Gate	67.90	70.24	80.28	82.79	77.15	82.88
	Cascade-MiniLM	85.89	90.67	77.46	79.41	77.11	84.63
	Cascade-SmolLM	73.58	78.69	50.10	55.68	65.83	77.61
0.75	**Ours**	**79.83**	**87.10**	**81.32**	**75.57**	**77.84**	**86.49**
	Without Gate	73.40	66.50	77.36	65.83	76.50	84.20
	Cascade-MiniLM	79.10	81.00	75.91	62.50	77.23	83.69
	Cascade-SmolLM	74.31	71.79	58.91	28.72	52.48	55.41

Table 3: Ablation results under different KIR settings.

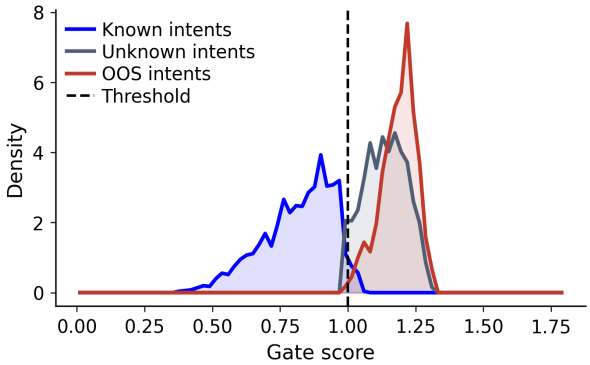

Figure 4: Gate score distribution on CLINC150 under KIR $= 0.50$. The dashed line marks $s(\bar{e}) = 1$.

to help learn the parameter λ in Eq. (7). They are viewed as unknown intent samples in the validation set. But actually, although there is a bit of overlap between samples of the unknown intent samples and the OOS intent test samples as shown in figure 3, the distributions of them are different, as shown in figure 4. That means, the distribution of the unknown intent samples do not influence the learning performances.

4.6 MiniLM Selection for Gate stage

The ablation studies for MiniLM selection of gate stage are also conducted, as shown in table 3. In the table, **Without Gate** means to cancel the gate stage in our full pipeline, and instead conduct OOS rejection in the router module. **Cascade-MiniLM** and **Cascade-SmolLM** means to use the MiniLMs in all stages.

The results that the full pipeline of ours achieves the state-of-the-art performance, compared the ablation variants. That means, (i) the gate stage is essential for OOS rejection, (ii) hybrid MiniLMs in ours have more advantages.

Moreover, we first use the 135M-parameter `SmolLM-135M` for all stages actually. But after that, we replace it with the 22M-parameter `all-MiniLM-L6-v2` for gate stage, and the performance is improved. That means, it is not essential to use models with large parameters, which refers to our insight on LLMs in section Related Work.

5 Conclusion

This work proposes a multi-cluster boundary learning method for OOS intent detection via a cascade MiniLM workflow. That workflow separates the OOS intent detection from the multi-class classification of known intents, and views it as a one-class classification problem. Then, multi-cluster boundary learning is conducted to learn the boundary of MiniLM embeddings from the input user utterances. That boundary separates the embedding domain of OOS intents and known intents. Extensive experiments are conducted on public datasets. The results show that the proposed method achieve the stete-of-the-art OOS intent detection performance. Ablation studies also show that the MiniLM settings are reasonable at present. Moreover, the performance of the method is not always the best one on the task of known intent detection, but the cascade workflow completely decouples the two detection process for OOS and known intents. Thus, the result does not influence the aforementioned conclusion. And moreover, the lightweight models have outstanding advantages for real-world applications compared to the large models. The more advanced MiniLMs perhaps improve the detection performance by using this framework in the future.

Limitations

The proposed method in this work achieves the state-of-the-art performance on OOS intent detection task, but does not generalize well on the task of known intent detection. Thus our primary problem setting is OOS intent detection in the title.

Ethics Statement

This paper does not involve text reuse from previously published work. We do not use any AI-assisted tools for writing this paper.

References

Gaurav Arora, Shreya Jain, and Srujana Merugu. 2024. Intent detection in the age of LLMs. In *EMNLP*. https://aclanthology.org/2024.emnlp-industry.114/.

Abhijit Bendale and Terrance E. Boult. 2016. Towards open set deep networks. In *CVPR*. https://openaccess.thecvf.com/content_cvpr_2016/html/Bendale_Towards_Open_Set_CVPR_2016_paper.html.

Peng Chen and Bang Wang. 2025. On collaborating small and large models for few-shot intent detection. In *EMNLP Findings*. https://aclanthology.org/2025.findings-emnlp.749/.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *ICML*. https://proceedings.mlr.press/v70/guo17a.html.

Patrick Haffner, Gokhan Tur, and Jerome H. Wright. 2003. Optimizing SVMs for complex call classification. In *ICASSP*. https://doi.org/10.1109/ICASSP.2003.1198860.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*. https://openreview.net/forum?id=Hkg4TI9xl.

Guy Hoffman, Tapomayukh Bhattacharjee, and Stefanos Nikolaidis. 2024. Inferring human intent and predicting human action in human-robot collaboration. *Annual Review of Control, Robotics, and Autonomous Systems*, 7:73–95. https://doi.org/10.1146/annurev-control-071223-105834.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *ICLR*. https://openreview.net/forum?id=nZeVKeeFYf9.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*. https://aclanthology.org/D14-1181/.

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 *EMNLP*. https://aclanthology.org/D19-1131/.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*. https://papers.nips.cc/paper_files/paper/2018/hash/abdeb6f575ac5c6676b747bca8d09cc2-Abstract.html.

Yanhua Li, Xiaocao Ouyang, Chaofan Pan, Jie Zhang, Sen Zhao, Shuyin Xia, Xin Yang, Guoyin Wang, and Tianrui Li. 2025. Multi-granularity open intent classification via adaptive granular-ball decision boundary. In *AAAI*. https://ojs.aaai.org/index.php/AAAI/article/view/34630.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *ACL*. https://aclanthology.org/P19-1548/.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *INTERSPEECH*. https://www.isca-archive.org/interspeech_2016/liu16c_interspeech.html.

Lemao Liu, Haisong Zhang, Haiyun Jiang, Yangming Li, Enbo Zhao, Kun Xu, Linfeng Song, Suncong Zheng, Botong Zhou, Dick Zhu, Xiao Feng, Tao Chen, Tao Yang, Dong Yu, Feng Zhang, Zhanhui Kang, and Shuming Shi. 2021. Texsmart: A system for enhanced natural language understanding. In *ACL Demo*. https://aclanthology.org/2021.acl-demo.20/.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *COLING*, pages 480–496. https://aclanthology.org/2020.coling-main.42/.

Abu Jafar Md Muzahid, Xiaopeng Zhao, and Zhenbo Wang. 2024. Survey on human-vehicle interactions and AI collaboration for optimal decision-making in automated driving. *arXiv*. https://arxiv.org/abs/2412.08005.

Jetze Schuurmans and Flavius Frasincar. 2020. Intent classification for dialogue utterances. *IEEE Intelligent Systems*, 35(1):82–88. https://doi.org/10.1109/MIS.2019.2954966.

Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: Deep open classification of text documents. In *EMNLP*. https://aclanthology.org/D17-1314/.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in ATIS? In *IEEE Spoken Language Technology Workshop*. https://doi.org/10.1109/SLT.2010.5700816.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Ye-Yi Wang, Alex Acero, Ciprian Chelba, Brendan Frey, and Leon Wong. 2002. Combination of statistical and rule-based approaches for spoken language understanding. In *ICSLP*, pages 609–612.

Georg Wölflein, Dyke Ferber, Daniel Truhn, Ognjen Arandjelović, and Jakob Nikolas Kather. 2025. LLM agents making agent tools. In *ACL*. https://aclanthology.org/2025.acl-long.1266/.

Álvaro Zaera, Diana Nicoleta Popa, Ivan Sekulic, and Paolo Rosso. 2025. Efficient out-of-scope detection in dialogue systems via uncertainty-driven LLM routing. In *ACL*. https://aclanthology.org/2025.acl-industry.25/.

Hossam M. Zawbaa, Wael Rashwan, Sourav Dutta, and Haytham Assem. 2024. Improved out-of-scope intent classification with dual encoding and threshold-based re-classification. In *LREC and COLING*. https://aclanthology.org/2024.lrec-main.763/.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *ACL and IJCNLP*. https://aclanthology.org/2021.acl-short.110/.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021. Deep open intent classification with adaptive decision boundary. In *AAAI*. https://ojs.aaai.org/index.php/AAAI/article/view/17690.

Hanlei Zhang, Hua Xu, Shaojie Zhao, and Qianrui Zhou. 2023. Learning discriminative representations and decision boundaries for open intent detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1611–1623. https://ieeexplore.ieee.org/document/10102404.

Yunhua Zhou, Peiju Liu, and Xipeng Qiu. 2022. KNN-contrastive learning for out-of-domain intent classification. In *ACL*. https://aclanthology.org/2022.acl-long.352/.