

# Watch the Neighbors: A Unified K-Nearest Neighbor Contrastive Learning Framework for OOD Intent Discovery

Yutao Mou<sup>1\*</sup>, Keqing He<sup>2\*</sup>, Pei Wang<sup>1</sup>, Yanan Wu<sup>1</sup>  
Jingang Wang<sup>2</sup>, Wei Wu<sup>2</sup>, Weiran Xu<sup>1\*</sup>

<sup>1</sup>Beijing University of Posts and Telecommunications, Beijing, China

<sup>2</sup>Meituan, Beijing, China

{myt,wangpei,yanan.wu,xuweiran}@bupt.edu.cn

{hekeqing,wangjingang,wuwei}@meituan.com

## Abstract

Discovering out-of-domain (OOD) intent is important for developing new skills in task-oriented dialogue systems. The key challenges lie in how to transfer prior in-domain (IND) knowledge to OOD clustering, as well as jointly learn OOD representations and cluster assignments. Previous methods suffer from in-domain overfitting problem, and there is a natural gap between representation learning and clustering objectives. In this paper, we propose a unified K-nearest neighbor contrastive learning framework to discover OOD intents. Specifically, for IND pre-training stage, we propose a KCL objective to learn inter-class discriminative features, while maintaining intra-class diversity, which alleviates the in-domain overfitting problem. For OOD clustering stage, we propose a KCC method to form compact clusters by mining true hard negative samples, which bridges the gap between clustering and representation learning. Extensive experiments on three benchmark datasets show that our method achieves substantial improvements over the state-of-the-art methods.<sup>1</sup>

## 1 Introduction

Out-of-domain (OOD) intent discovery aims to group new unknown intents into different clusters, which helps identify potential development directions and develop new skills in a task-oriented dialogue system (Lin et al., 2020; Zhang et al., 2021; Mou et al., 2022). Different from traditional text clustering task, OOD discovery considers how to leverage the prior knowledge of known in-domain (IND) intents to enhance discovering unknown OOD intents, which makes it challenging to directly apply existing clustering algorithms (MacQueen, 1967; Xie et al., 2016; Chang et al., 2017; Caron et al., 2018) to the OOD discovery task.

\*The first two authors contribute equally. Weiran Xu is the corresponding author.

<sup>1</sup>We release our code at <https://github.com/myt517/KCOD>

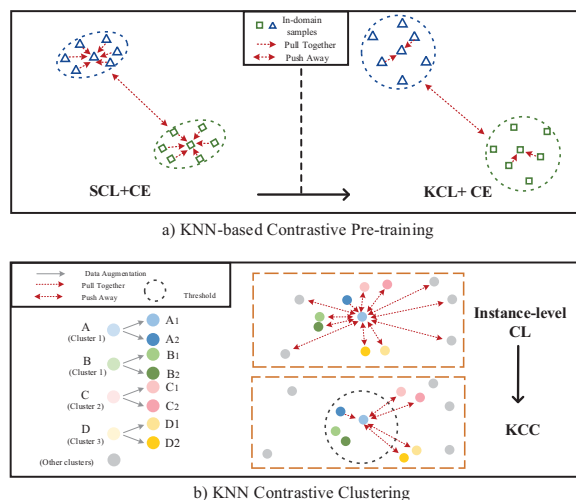


Figure 1: The high-level idea of our KCOD compared to baselines. Fig (a) shows KCL increases the intra-class variance by only taking the K nearest samples of the same class as positive samples. Fig (b) displays that KCC filters out false negative samples that belong to the same class as the anchor and selects K nearest neighbors as true hard negatives to form clear cluster boundaries.

The key challenges of OOD intent discovery come from two aspects: (1) **Knowledge Transferability**. It requires transferring in-domain prior knowledge to help downstream OOD clustering. Early unsupervised intent discovery methods (Hakkani-Tür et al., 2015; Padmasundari and Bangalore, 2018; Shi et al., 2018) only model unlabeled OOD data but ignore prior knowledge of labeled in-domain data thus suffer from poor performance. Then, recent work (Lin et al., 2020; Zhang et al., 2021; Mou et al., 2022) focus more on the semi-supervised setting where they firstly pre-train an in-domain intent classifier then perform clustering algorithms on extracted OOD intent representations by the pre-trained IND intent classifier. For example, Lin et al. (2020); Zhang et al. (2021) pre-train a BERT-based (Devlin et al., 2019) in-domain intent classifier using cross-entropy (CE) classification loss. Mou et al. (2022) further proposes

a supervised contrastive learning (SCL) (Khosla et al., 2020) objective to learn discriminative intent representations. (2) **Jointly Learning Representations and Cluster Assignments**. It’s important to learn OOD intent features while performing clustering. Lin et al. (2020) uses OOD representations to calculate the similarity of OOD sample pairs as weak supervised signals. The gap between pre-trained IND features and unseen OOD data makes it hard to generate high-quality pairwise pseudo labels. Then, Zhang et al. (2021) proposes an iterative clustering method, DeepAligned, to obtain pseudo cluster labels by K-means (MacQueen, 1967). It performs representation learning and cluster assignment in a pipeline way. Further, Mou et al. (2022) introduces a multi-head contrastive clustering framework to jointly learn representations and cluster assignments using contrastive learning.

However, all of these methods still suffer from two problems: (1) **In-domain Overfitting**: The state-of-the-art (SOTA) OOD intent discovery methods (Zhang et al., 2021; Mou et al., 2022) adopt general supervised pre-training objectives such as CE and SCL for IND pre-training, but none of them consider the following question: what kind of intent representation is more generalized to transfer to downstream OOD clustering? Although CE and SCL are effective for classifying known IND classes, such learned representations are poor for downstream transfer. Zhao et al. (2020); Feng et al. (2021) find larger intra-class diversity helps transfer. CE and SCL tend to pull all samples from the same class together to form a narrow intra-class distribution, thus ignore the intra-class diverse features, which makes the learned representations unfavorable to transfer to the downstream OOD clustering. (2) **Gap between Clustering Objectives and Representation Learning**. Learning OOD intent representations is key for achieving efficient clustering. Zhang et al. (2021) can’t align learning intent features with clustering because the two processes are iterative in a pipeline way. Mou et al. (2022) further proposes DKT to jointly learn representation and cluster assignment using contrastive learning (Gao et al., 2021; Yan et al., 2021). However, such a contrastive objective of learning intent features pushes apart representations of different samples in the same class, which impair the clustering goal where samples within the same class should be compact.

To solve the two problems, we propose a unified

**K-Nearest Neighbor Contrastive Learning** framework for **OOD Discovery (KCOD)**. For the in-domain overfitting issue, we propose a simple **K**-nearest neighbor Contrastive Learning objective (KCL) for IND pre-training in Fig 1 (a). Compared with SCL, we only take the **K** nearest samples of the same class as positive samples, which helps to increase the intra-class variance while maintaining a large inter-class variance. Larger intra-class diversity helps downstream transfer. For the gap between clustering and representation learning objectives, we propose a **K**-nearest neighbor Contrastive Clustering method (KCC) for OOD clustering in Fig 1 (b). Traditional instance-wise contrastive learning only regards an anchor and its augmented sample as positive pairs and pushes apart representations of different samples even in the same class. In contrast, KCC firstly filters out false negative samples that belong to the same class as the anchor and then selects the **K** nearest neighbors as true negatives. We aim to mine high-confident hard negatives and learn compact intent representations for OOD clustering to form clear cluster boundaries.

Our contributions are three-fold: (1) We propose a unified **K**-nearest neighbor contrastive learning (KCOD) framework for OOD discovery, which aims to achieve better knowledge transfer and learn better clustering representations. (2) We propose a **K**-nearest neighbor contrastive learning (KCL) objective for IND pre-training, and a **K**-nearest neighbor contrastive clustering (KCC) method for OOD clustering, which solve "in-domain overfitting" problem and bridge the gap between clustering objectives and representation learning. (3) Experiments and analysis demonstrate the effectiveness of our method for OOD discovery.

## 2 Approach

OOD discovery assumes there is a set of labeled in-domain data and unlabeled OOD data<sup>2</sup>. Our goal is to cluster OOD concepts from unlabeled OOD data using prior knowledge from labeled IND data. The overall architecture of our KCOD is shown in Fig 2, including KNN contrastive IND pre-training and KNN contrastive OOD clustering. IND pre-training firstly gets generalized intent representa-

<sup>2</sup>We notice there are two settings of OOD discovery: one is to cluster unlabeled OOD data and another is to cluster unlabeled mixed IND&OOD data. Here we adopt the first setting as Mou et al. (2022) because mixing IND&OOD makes it hard to fairly evaluate the capability of discovering new intent concepts.

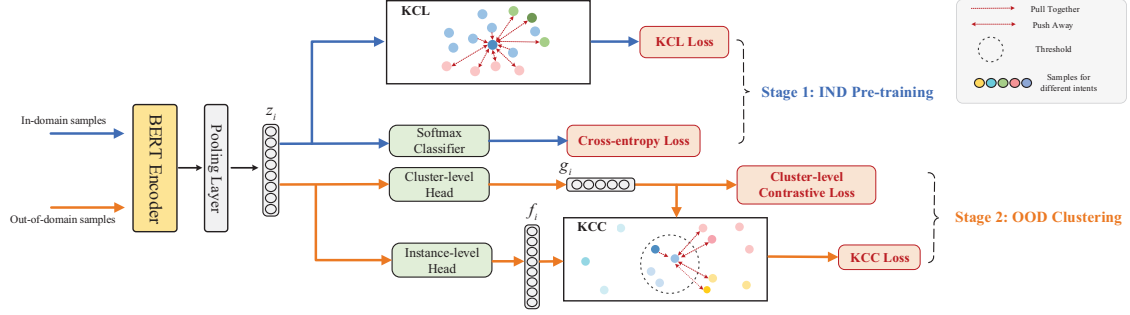


Figure 2: The overall architecture of our proposed unified K-nearest neighbor contrastive learning framework for OOD discovery, KCOD. Stage 1 denotes IND pre-training and Stage 2 denotes OOD clustering.

tions via our proposed KCL objective and then OOD clustering uses KCC to group OOD intents into different clusters.

## 2.1 KNN Contrastive Pre-training

K-nearest neighbor contrastive learning (KCL) aims to increase the intra-class variance to learn generalized intent representations for downstream clustering. Previous work (Zeng et al., 2021a; Mou et al., 2022) pull together IND samples belonging to the same class and push apart samples from different classes. However, such methods make all the instances of the same class collapse into a narrow area near the class center, which reduces the intra-class diversity. Zhao et al. (2020); Feng et al. (2021) find large intra-class diversity helps transfer knowledge to downstream tasks. Therefore, we relax the constraint by only limiting k-nearest neighbors close to each other. The KCL pre-training loss is as follows:

$$\mathcal{L}_{KCL} = \sum_{i=1}^N -\frac{1}{|K_i|} \sum_{j=1}^{K_i} \log \frac{\exp(f_i \cdot f_j / \tau)}{\sum_{k=1}^{|A_{ij}|} \exp(f_i \cdot f_k / \tau)} \quad (1)$$

where  $K_i$  is the set of k-nearest neighbors with the same class as  $i$ -th sample.  $A_{ij}$  denotes the union set of the positive  $f_j$  and negative samples whose classes are different from the  $i$ -th sample. Specifically, given an anchor sample, we firstly get all the samples belonging to the same class as the anchor from the batch, then select its k-nearest neighbors among these samples using extracted intent features. KCL aims to pull together the anchor and its neighbors in the same class and push apart samples from different classes. To support a large batch size, we employ a momentum queue (He et al., 2020) to update the intent features. The

queue decouples the size of contrastive samples from the batch size, resulting in a larger negative set. We perform joint training both using KCL and CE, and simply adding them gets the best performance. Section 4.1 proves our proposed KCL increases the intra-class variance and alleviates in-domain overfitting.

## 2.2 KNN Contrastive Clustering

After obtaining pre-trained intent features, we need to group OOD intents into different clusters. Existing work (Zhang et al., 2021; Mou et al., 2022) can't jointly learn representations and cluster assignments. Zhang et al. (2021) iteratively performs the two stages, leading to a suboptimal result. Mou et al. (2022) instead employs a contrastive clustering framework for joint learning but still has a gap between clustering objectives and representation learning. We first briefly introduce the contrastive clustering framework and then provide an analysis of how to bridge the gap.

Given an OOD example  $x_i$ , we firstly use the pre-trained BERT encoder to get an OOD intent feature  $z_i$ . Then, we use a cluster-level contrastive loss (CL)  $\ell_{i,j}^{clu}$  to learn cluster assignments. Specifically, we project  $z_i$  to a vector  $g_i$  with dimension  $C$  which equals to the pre-defined cluster number<sup>3</sup>. So we get a feature matrix of  $N \times C$  where  $N$  is the batch size. Following Li et al. (2021), we regard  $i$ -th column of the matrix as the  $i$ -th cluster representation  $y_i$  and construct cluster-level loss  $\ell_{i,j}^{clu}$  as follows:

$$\ell_{i,j}^{clu} = -\log \frac{\exp(\text{sim}(y_i, y_j) / \tau)}{\sum_{k=1}^{2C} 1_{[k \neq i]} \exp(\text{sim}(y_i, y_k) / \tau)} \quad (2)$$

<sup>3</sup>Estimating cluster number  $C$  is out of the scope of this paper. We provide a discussion in Section E.

where  $y_j$  is the dropout-augmented (Gao et al., 2021) cluster representation of  $y_i$  and  $\text{sim}$  denotes cosine distance. To learn intent representations, Li et al. (2021); Mou et al. (2022) use an instance-level contrastive learning loss  $\ell_{i,j}^{\text{ins}}$ :

$$\ell_{i,j}^{\text{ins}} = -\log \frac{\exp(\text{sim}(f_i, f_j)/\tau)}{\sum_{k=1}^{2N} \mathbf{1}_{[k \neq i]} \exp(\text{sim}(f_i, f_k)/\tau)} \quad (3)$$

where  $f_i$  is transformed from  $z_i$  by an instance-level head and  $f_j$  denotes its dropout augmentation.  $\tau$  is the temperature. However, Eq 3 only regards an anchor and its augmented sample as positive pair and even pushes apart different samples in the same class. The characteristic has a conflict with the clustering goal where the samples of the same class should be tight. This instance-level CL loss considers the relationship between instances instead of different types, which makes it hard to learn distinguished intent cluster representations.

Therefore, we propose a K-nearest neighbor contrastive clustering method (KCC) to form clear cluster boundaries. Firstly, we use the predicted logits from the cluster-level head  $g^4$  and compute the dot similarity of two samples to filter out false negative samples that belong to the same class as the anchor. Here, we find a simple similarity threshold can work well (see Section 4.4). So we select samples whose similarity scores are below the threshold as negatives. To further separate different clusters, we select the K nearest neighbors from the negative set as hard negatives. Our intuition is to push these hard negatives away from the anchor and form clear cluster boundaries. We formulate the KCC loss as follows:

$$\ell_{i,j}^{\text{KCC}} = -\log \frac{\exp(\text{sim}(f_i, f_j)/\tau)}{\sum_{k=1}^{|H_i|} \exp(\text{sim}(f_i, f_k)/\tau)} \quad (4)$$

where  $H_i$  is the union set of the augmented positive sample  $f_j$  and k-nearest hard negative set  $N_i$  of  $f_i$ . We give a theoretical explanation from the perspective of gradients. For convenience, we denote  $s_{i,i}$  as the positive pair and  $s_{i,j}, i \neq j$  as negative pairs. So original instance-level CL loss in Eq 3 is rewritten as:

$$\mathcal{L}(x_i) = -\log \left[ \frac{\exp(s_{i,i}/\tau)}{\sum_{k \neq i} \exp(s_{i,k}/\tau) + \exp(s_{i,i}/\tau)} \right] \quad (5)$$

<sup>4</sup>The output dim of projector  $g$  is equal to the cluster number  $C$ , so we can take the normalized output vector as the predicted probability on all clusters.

where  $s_{i,j} = \text{sim}(f(x_i), f(x_j))$ . We analyze the gradients with respect to different negative samples following Wang and Isola (2020):

$$\frac{\partial \mathcal{L}(x_i)}{\partial s_{i,j}} = \frac{1}{\tau} \frac{\exp(s_{i,j}/\tau)}{\sum_{k \neq i} \exp(s_{i,k}/\tau) + \exp(s_{i,i}/\tau)} \quad (6)$$

From Eq 5, we find that if easy negatives are filtered out, the gradient (penalty) to other hard negatives gets larger, thus pushing away these negatives from the anchor. It means our model can separate misleading samples near the cluster boundary (see Section 4.2). We simply add the cluster-level CL loss and KCC loss to jointly learn cluster assignments and intent representations. Following Li et al. (2021), we also add a regularization item to avoid the trivial solution that most instances are assigned to a single cluster. For inference, we only use the cluster-level head and compute the argmax to get the cluster results without additional K-means.

## 3 Experiments

### 3.1 Datasets

We conduct experiments on three benchmark datasets, Banking (Casanueva et al., 2020), HWU64 (Liu et al., 2021a) and CLINC (Larson et al., 2019). Banking contains 13,083 customer service queries with 77 intents in the banking domain. HWU64 includes 25,716 utterances with 64 intents across 21 domains. CLINC contains 22,500 queries covering 150 intents across 10 domains. Following Mou et al. (2022), we randomly sample a ratio of the intents as OOD (10%, 20%, 30% for Banking, 30% for CLINC and HWU64), and the rest as IND. Note that we only use the IND data for pre-training and use OOD data for clustering. To avoid randomness, we average results over three random runs.

### 3.2 Baselines

Similar with Mou et al. (2022), we mainly compare our method with semi-supervised baselines: PTK-means<sup>5</sup> (k-means with IND pre-training), DeepCluster (Caron et al., 2018), CDAC+ (Lin et al., 2020), DeepAligned (Zhang et al., 2021) and DKT (Mou et al., 2022), in which DKT is the current state-of-the-art method for OOD intent discovery.

<sup>5</sup>Here we conduct fair experiments with different IND pre-training objectives, in which PTK-means(SCL) adopts the same pre-training objective as DKT, and PTK-means(KCL) adopts the same pre-training objective as KCOD.



Method	Banking-10%			Banking-20%			Banking-30%			HWU64-30%			CLINC-30%		
	ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI
PTK-means(SCL)	55.00	36.18	53.75	51.68	35.65	56.77	45.06	32.12	57.93	56.95	43.79	61.94	61.63	40.96	75.90
PTK-means(KCL)	70.91	57.83	67.53	63.62	53.69	66.54	59.60	47.92	66.78	72.53	58.56	71.26	83.17	76.53	87.80
DeepCluster (Caron et al., 2018)	60.59	41.88	55.22	60.33	50.21	69.54	59.35	45.94	68.08	76.35	65.40	78.40	78.09	71.05	88.70
CDAC+ (Lin et al., 2020)	77.50	60.53	71.14	63.50	53.94	72.35	59.78	44.58	69.19	75.08	61.18	79.51	73.04	64.44	87.90
DeepAligned (Zhang et al., 2021)	77.78	66.95	76.91	67.01	58.79	76.06	63.86	52.84	73.66	82.04	76.13	86.35	91.56	86.58	94.91
DKT (Mou et al., 2022)	84.69	71.11	76.92	69.55	57.00	73.21	66.50	52.07	72.22	83.91	73.69	83.83	94.96	90.25	95.94
KCOD w/o KCC(ours)	85.21	71.67	78.40	71.07	59.45	74.71	69.35	54.78	73.74	84.55	76.54	84.91	95.62	91.61	96.67
KCOD(ours)	<b>86.67</b>	<b>74.05</b>	<b>79.89</b>	<b>73.09</b>	<b>60.96</b>	<b>75.67</b>	<b>71.09</b>	<b>57.73</b>	<b>75.79</b>	<b>86.28</b>	<b>77.07</b>	85.62	<b>96.48</b>	<b>92.46</b>	<b>96.89</b>

Table 1: Performance comparison on three datasets. For Banking, We randomly sample 10%, 20% and 30% of all classes as OOD types. For HWU64 and CLINC, we randomly sample 30% of all classes as OOD types. KCOD w/o KCC denotes we use the same clustering method as DKT and replace original SCL with KCL for IND pre-training. Results are averaged over three random runs. ( $p < 0.01$  under t-test)

We leave the details of the baselines in Appendix B. For fairness, all baselines use the same BERT backbone. We adopt three widely used metrics to evaluate the clustering results: Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI). ACC is the most important metric. Note that for Banking-10% and CLINC-30%, the results of all baselines except PTK-means(KCL) are retrieved from Mou et al. (2022), while for Banking-20%, Banking-30% and HWU64-30%, we rerun all baselines with the same dataset split for fair comparison. <sup>6</sup>

### 3.3 Main Results

Table 1 shows the main results of our proposed method compared to the baselines. In general, our method consistently outperforms all the previous baselines with a large margin. We analyze the results from four aspects:

**Our proposed KCL objective helps knowledge transfer.** We can see that KCOD w/o KCC has a significant improvement compared to DKT and DeepAligned. For example, KCOD w/o KCC outperforms previous state-of-the-art DKT by 2.85% (ACC), 2.71%(ARI), 1.52%(NMI) on Banking-30%. It is worth noting that KCOD w/o KCC adopts the same OOD clustering method as DKT, but uses our proposed KCL objective instead of SCL for IND pre-training. It proves that using the KCL pre-training objective learns generalized intent representations, which helps transfer in-domain prior knowledge for downstream OOD clustering. We also provide a deep analysis in Section 4.1 to explore the reasons.

**Our proposed KCC helps OOD clustering.** We can observe that KCOD further improves

<sup>6</sup>Mou et al. (2022) focuses on the multi-domain dataset CLINC, however, we find CLINC is relatively simple for its coarse-grained intent types. In this paper, we focus on the more challenging single-domain fine-grained dataset Banking.

	ACC	ARI	NMI
No-pretraining	32.99	16.59	33.47
CE	67.65	52.75	70.37
CE+SCL	69.55	57.00	73.21
CE+KCL	<b>71.07</b>	<b>59.45</b>	<b>74.71</b>

Table 2: Clustering performance comparison of different pre-training objectives using the same clustering method. We use Banking-20% for analysis.

1.74%(ACC), 2.95%(ARI) and 2.05%(NMI) compared to KCOD w/o KCC on Banking-30%. This proves that KCC can learn cluster-friendly representation, which is helpful for OOD clustering. We discuss the effect of KCC in detail in Section 4.2.

**Comparison of different datasets** We validate the effectiveness of our method on different datasets, where Banking is a single-domain fine-grained dataset, and CLINC and HWU64 are multi-domain datasets. We can see that all methods perform significantly worse on Banking than on CLINC and HWU64 datasets, which indicates that the single-domain fine-grained scenario is more challenging for OOD discovery. But our proposed KCOD achieves larger improvements of 3%-6% on the Banking-30% dataset compared to DKT, while only 2%-4% on CLINC-30% and HWU64-30%. It indicates that KCOD can better cope with the challenges in fine-grained intent scenarios, and has stronger transferability and generalization.

**Effect of different OOD ratios** We observe that all methods decrease significantly when the OOD ratio increases. Because the proportion of unlabeled OOD increases and labeled IND data decreases, making both knowledge transfer and clustering more difficult. However, our KCOD achieves more significant improvements with the increase of the OOD ratio. For example, compared to DKT, on Banking-10%, KCOD increases by 1.98% (ACC), on Banking-20%, KCOD in-

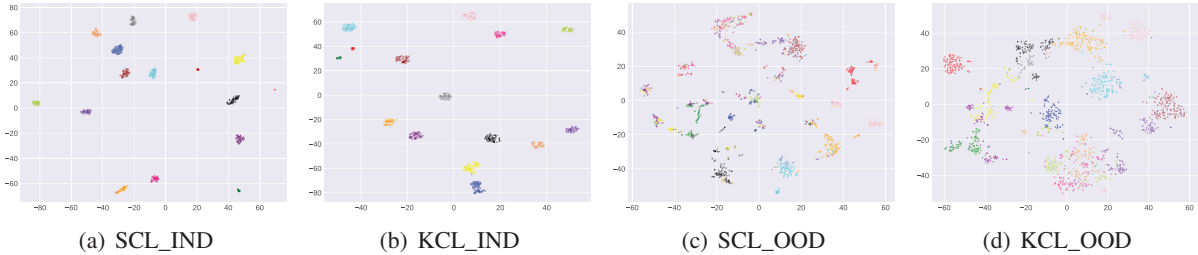


Figure 3: IND and OOD intent visualization for different IND pre-training objectives.

creases by 3.54% (ACC), and on Banking-30%, KCOD increases by 4.59% (ACC). This also reflects the strong generalization capability of the KCOD framework.

## 4 Qualitative Analysis

### 4.1 Effect of KCL

We analyze the effect of KCL from multiple perspectives.

**Ablation Study** We compare the OOD clustering performance under different IND pre-training objectives in Table 2. All models employ the same contrastive clustering method as DKT (Mou et al., 2022) for OOD clustering. We find that adding the KCL objective for IND pre-training significantly improves the performance for OOD clustering compared to SCL, which proves that KCL enhances knowledge transferability.

**Discussion of why KCL is effective** To explore why KCL is effective for knowledge transferability, we calculate the intra-class and inter-class distances following Feng et al. (2021). For the intra-class distance, we calculate the cosine similarity between each sample and its class center. For the inter-class distance, we calculate the cosine similarity between each class center and its 3 nearest class centers. We report the averaged  $1 - \cos(\cdot, \cdot)$  in Table 3. Results show that using KCL for IND pre-training can increase the intra-class distance of IND classes while maintaining a relatively large inter-class variance. Then we extract the representation of the OOD intents using the pre-trained model and perform K-means (MacQueen, 1967) (see K-means ACC in Table 3). We find KCL for IND pre-training helps OOD clustering and benefits knowledge transfer.

**Visualization** Fig 3 displays IND and OOD intent visualization for different IND pre-training methods SCL and KCL. We find KCL is beneficial to increasing the intra-class variance and helps build

	Intra-class $\uparrow$	Inter-class $\uparrow$	K-means ACC $\uparrow$
CE	0.04	0.24	58.11
CE+SCL	0.01	<b>0.68</b>	51.68
CE+KCL	<b>0.10</b>	0.43	<b>63.62</b>

Table 3: Representation distribution of different pre-training objectives.

Models	ACC	ARI	NMI
KCC	<b>73.09</b>	<b>60.96</b>	<b>75.67</b>
-w/o instance-level head	69.50	55.27	69.96
-w/o cluster-level head	63.50	43.42	66.63

Table 4: Ablation study of branches for KCC (We use KCL objective for IND pre-training).

the clear OOD boundary. We argue that too small intra-class variance or too small inter-class variance are not good for downstream transfer, and preserving intra-class diverse features is vital to generalization.

In summary, our proposed KCL increases the intra-class variance and preserves the features related to intra-class difference by selecting the top K positives, which alleviates the "in-domain overfitting" problem and helps knowledge generalization to OOD clustering.

### 4.2 Effect of KCC

To study the effect of KCC, we perform OOD visualization of DKT, KCOD w/o KCC, and KCOD in Fig 4. We see that KCOD can form clear cluster boundaries and separate different OOD clusters.

**Bridge the gap between clustering and representation learning** We also show the curve of the OOD SC value during the training process in Fig 5. The range of SC is between -1 and 1, and the higher score means the better clustering quality<sup>7</sup>. Results show that previous contrastive clustering methods (Li et al., 2021; Mou et al., 2022) for OOD clustering, such as DKT and KCOD w/o

<sup>7</sup>Please refer to more details about SC in Appendix D.

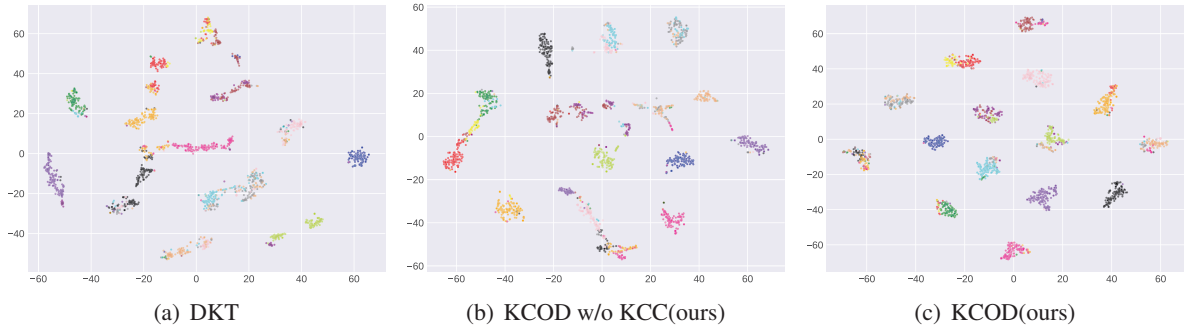


Figure 4: OOD intent visualization of different models. We use the same OOD test set of Banking-20%.

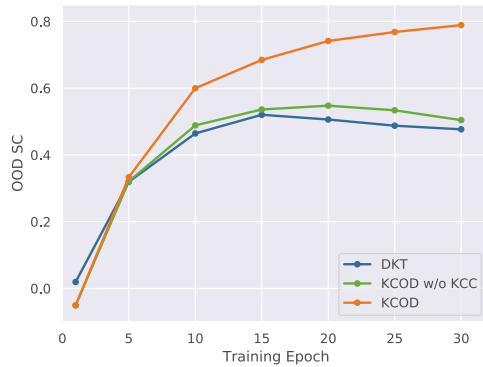


Figure 5: OOD SC curves in the training process.

KCC make the SC value first rise to a peak and then decrease to a certain extent. It’s because they use instance-level contrastive learning (CL) to learn intent features which violates the clustering objective. Instance-level CL pulls together an anchor and its augmented positives and pushes apart representations of different samples, which means OOD intents within the same class are still separated. But clustering requires compact clusters. Our proposed KCC bridges the gap between clustering objectives and representation learning.

**Form compact clusters** In order to more directly prove that KCC can effectively mine hard negative samples to form clear cluster boundaries and compact clusters, we randomly select five OOD classes in Banking-20%, and calculate the ratio of inter-class distance and intra-class distance for different clustering methods respectively, as shown in Fig 6. It can be seen that KCOD can form a more compact cluster for each class, which is beneficial to distinguishing these categories.

### 4.3 Ablation Study for KCC

K-nearest neighbor contrastive clustering (KCC) includes two branches, cluster-level head and instance-level head. In order to verify the effec-

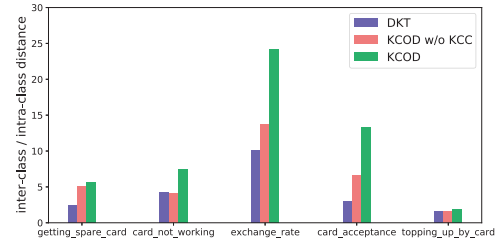


Figure 6: Cluster compactness of different OOD discovery methods for 5 OOD classes.

	ACC	ARI	NMI
DKT	69.55	57.00	73.21
+KCL (K=1)	71.00	56.91	73.36
+KCL (K=3)	<b>71.07</b>	<b>59.45</b>	<b>74.71</b>
+KCL (K=5)	70.37	56.99	72.22
+KCL (K=7)	69.33	56.69	73.03
+KCL (K=9)	68.17	57.05	73.37

Table 5: The effect of different K values of KCL.

tiveness of the two branches working together for clustering. We performed an ablation study and the results are shown in Table 4. For KCC w/o instance-level head, we remove the instance-level head and only use the cluster-level head for clustering. For KCC w/o cluster-level head, we remove the cluster-level head, and in the OOD clustering stage, instance-level contrastive learning is used for representation learning, and K-means is used for clustering. It can be seen that the clustering performance drops significantly when any head is removed, which indicates that jointly learning instance-level representation and cluster-level assignments is beneficial for improving clustering performance.

### 4.4 Hyper-parameter Analysis

**The effect of the K value for KCL** Table 5 shows the effect of different K values of KCL pre-training loss. For a fair comparison, we replace SCL of DKT with our proposed KCL and use the same clus-

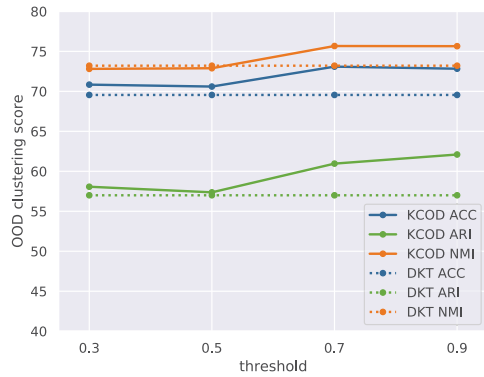


Figure 7: Effect of the threshold of KCC on Banking-20%. The range of  $t$  is between 0 and 1.

tering method as DKT. We find smaller  $K$  achieves superior performance for OOD discovery because smaller  $K$  makes the model learn more intra-class diversity. But when  $K=1$  KCL will be degraded to traditional instance-level contrastive learning and lose label information.

**The effect of KCC threshold** Our proposed KCC method needs to firstly filter out false negative samples whose similarity with the anchor is greater than the specified threshold  $t$ . These false negatives are considered as candidate positive samples and should not be pushed apart like conventional instance-level contrastive learning (Yan et al., 2021; Gao et al., 2021; Mou et al., 2022). We show the effect of different thresholds  $t$  on the performance of KCOD in Fig 7. We find that when the threshold  $t$  is in the range of [0.6, 0.9], the performance of KCOD is better than DKT. We argue that the KCC method needs to select a large threshold because a smaller threshold means that there may be more noise in the candidate positive sample set where some hard negative samples may be regarded as false positive samples.

**The effect of the K value of KCC** A core mechanism of KCC is to select  $K$  nearest neighbor samples in the candidate negative sample set to participate in the calculation of the contrastive learning loss. The main motivation is to mine hard negative samples to form clear cluster boundaries. We show the effect of different  $K$  values on KCOD performance in Fig 8. We find that our KCOD achieves consistent improvements under different  $K$  values, which proves the robustness of our method. And  $K \in [300, 400]$  gets the best metrics. Too large  $K$  brings a subtle drop because many easy negatives are considered and hard negatives near the cluster boundary can't be explicitly separated.

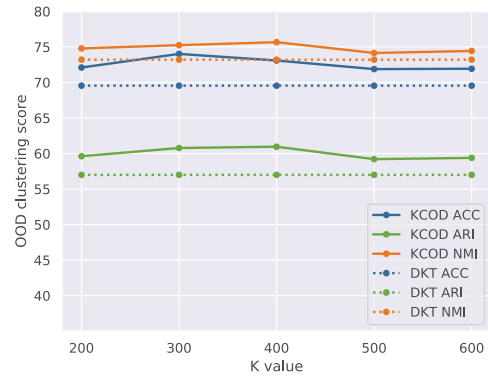


Figure 8: Effect of the K value of KCC on Banking-20%.

	ACC	ARI	NMI	C
DeepAligned	67.00	58.79	<b>76.06</b>	15
DKT	69.55	57.00	73.21	15
KCOD(ours)	<b>73.09</b>	<b>60.96</b>	75.67	15
DeepAligned	66.50	58.00	<b>75.00</b>	13
DKT	66.89	52.81	70.05	13
KCOD(ours)	<b>70.55</b>	<b>58.62</b>	74.12	13

Table 6: Estimate the number of OOD clusters.  $C=13$  is the estimated number compared to golden 15.

#### 4.5 Estimate the Number of Cluster C

All the results we showed so far assume that the number of OOD classes is pre-defined. However, in real-world applications, the number of clusters often needs to be estimated automatically. Table 6 shows the results using the same cluster number estimation strategy<sup>8</sup>. It can be seen that when the number of clusters is inaccurate, all methods have a certain decline, but our KCOD method still significantly outperforms all baselines, which also proves that KCOD is robust.

#### 4.6 Error Analysis

We further analyze the error cases of DKT and KCOD in Fig 9. We find that for similar OOD intents, DKT is probably confused but our KCOD can effectively distinguish them. For example, DKT incorrectly groups *getting\_spare\_card* intents into *card\_acceptance* (55% error rate) vs KCOD(0%), which proves KCOD helps separate semantically similar OOD intents.

## 5 Related Work

**OOD Discovery** Early methods (Xie et al., 2016; Caron et al., 2018) use unsupervised data for clus-

<sup>8</sup>Here we use the same estimation algorithm as Zhang et al. (2021). We leave the details in Appendix E.



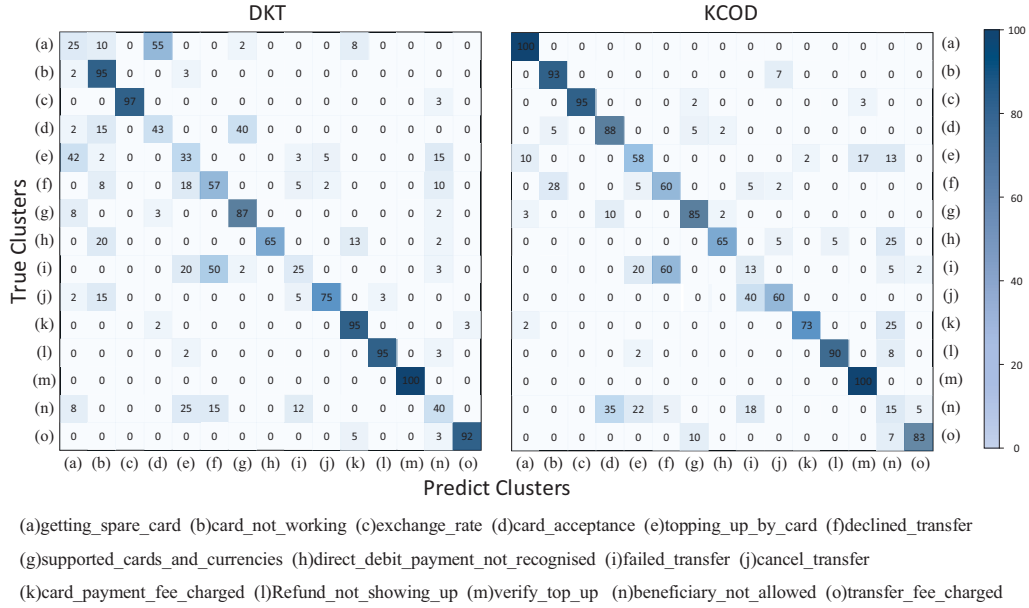


Figure 9: Confusion matrix for the clustering results of DKT and KCOD on Banking-20%. The percentage values along the diagonal represent how many samples are correctly clustered into the corresponding class. The larger the number, the deeper the color.

tering. Recent work (Lin et al., 2020; Zhang et al., 2021; Mou et al., 2022) performs semi-supervised clustering using labeled in-domain data. To transfer intent representation, Lin et al. (2020); Zhang et al. (2021) pre-train a BERT encoder using cross-entropy loss, then Mou et al. (2022) uses SCL (Khosla et al., 2020) to learn discriminative features. All the models face the challenge of in-domain overfitting issues where representations learned from IND data will degrade for OOD data. Thus, we propose a KCL loss to keep large inter-class variance and help downstream transfer. For OOD clustering, Zhang et al. (2021, 2022) use k-means to learn cluster assignments but ignore joint learning intent representations. Mou et al. (2022) uses contrastive clustering where the instance-level contrastive loss for learning intent features has a gap with the cluster-level loss for clustering. Therefore, we propose the KCC method to mine hard negatives to form clear cluster boundaries.

**Contrastive Learning** Contrastive learning (CL) is widely used in self-supervised learning (He et al., 2020; Gao et al., 2021; Khosla et al., 2020). Zeng et al. (2021a); Liu et al. (2021b); Zeng et al. (2021b) apply it to OOD detection. Zhou et al. (2022) proposes a KNN-contrastive learning method for OOD detection. It aims to learn discriminative semantic features that are more conducive to anomaly detection. In contrast, our method uses a unified

K-nearest neighbor contrastive Learning framework for OOD discovery where KCL increases intra-class diversity and helps downstream transfer, and KCC learns compact intent representations for OOD clustering to form clear cluster boundaries. Mou et al. (2022) uses contrastive clustering for OOD discovery. But original instance-level CL pushes apart different instances of the same intent which is against clustering. Thus, we use a simple k-nearest sampling mechanism to separate clusters and form clear boundaries.

## 6 Conclusion

In this paper, we propose a unified K-nearest neighbor contrastive learning (KCOD) framework for OOD intent discovery. We design a KCL objective for IND pre-training, and a KCC method for OOD clustering. Experiments on three benchmark datasets prove the effectiveness of our method. And extensive analyses demonstrate that KCL is helpful for learning intra-class diversity knowledge and alleviating the problem of intra-domain overfitting, and KCC is beneficial for forming compact clusters, effectively bridging the gap between clustering and representation learning. We hope to explore more self-supervised learning methods for OOD discovery in the future.

## Limitations

This paper mainly focuses on the out-of-domain (OOD) intent discovery task in task-oriented dialogue systems. We aim to leverage the prior knowledge of known in-domain (IND) intents to help OOD clustering. Our proposed KCOD method well addresses the two challenges of knowledge transferability and joint learning of representation and cluster assignment, and achieves SOTA performance on three intent recognition benchmark datasets. However, our method can also be used in broader fields, such as short text clustering, topic discovery, etc., which we did not explore further in this paper. We will try to apply this framework to a wider range of NLP topics in the future.

## Acknowledgements

We thank all anonymous reviewers for their helpful comments and suggestions. This work was partially supported by National Key R&D Program of China No. 2019YFF0303300 and Subject II No. 2019YFF0303302, DOCOMO Beijing Communications Laboratories Co., Ltd, MoE-CMCC "Artificial Intelligence" Project No. MCM20190701.

## References

- Mathilde Caron, Piotr Bojanowski, Armand Joulin, and M. Douze. 2018. Deep clustering for unsupervised learning of visual features. In *ECCV*.
- 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.
- Jianlong Chang, L. Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. 2017. Deep adaptive image clustering. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5880–5888.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yutong Feng, Jianwen Jiang, Mingqian Tang, Rong Jin, and Yue Gao. 2021. Rethinking supervised pre-training for better downstream transferring. *arXiv preprint arXiv:2110.06014*.
- Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. [Simcse: Simple contrastive learning of sentence embeddings](#). *ArXiv*, abs/2104.08821.
- Dilek Hakkani-Tür, Yun-Cheng Ju, Geoffrey Zweig, and Gokhan Tur. 2015. Clustering novel intents in a conversational interaction system with semantic parsing. In *Sixteenth Annual Conference of the International Speech Communication Association*.
- Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735.
- Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- 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)*.
- Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2021. Contrastive clustering. In *2021 AAAI Conference on Artificial Intelligence (AAAI)*.
- Ting-En Lin, Hua Xu, and Hanlei Zhang. 2020. Discovering new intents via constrained deep adaptive clustering with cluster refinement. In *AAAI*.
- Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021a. Benchmarking natural language understanding services for building conversational agents. In *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, pages 165–183. Springer.
- Zijun Liu, Yuanmeng Yan, Keqing He, Sihong Liu, Hong Xu, and Weiran Xu. 2021b. Self-training with masked supervised contrastive loss for unknown intents detection. *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7.
- J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations.
- Yutao Mou, Keqing He, Yanan Wu, Zhiyuan Zeng, Hong Xu, Huixing Jiang, Wei Wu, and Weiran Xu. 2022. [Disentangled knowledge transfer for OOD intent discovery with unified contrastive learning](#). In

*Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 46–53, Dublin, Ireland. Association for Computational Linguistics.

Padmasundari and S. Bangalore. 2018. Intent discovery through unsupervised semantic text clustering. In *INTERSPEECH*.

P. Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Chen Shi, Qi Chen, Lei Sha, Sujian Li, Xu Sun, Houfeng Wang, and Lintao Zhang. 2018. Auto-dialabel: Labeling dialogue data with unsupervised learning. In *EMNLP*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. *ArXiv*, abs/1511.06335.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. In *ACL/IJCNLP*.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021a. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 870–878, Online. Association for Computational Linguistics.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Hong Xu, and Weiran Xu. 2021b. Adversarial self-supervised learning for out-of-domain detection. In *NAACL*.

Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lv. 2021. Discovering new intents with deep aligned clustering. In *AAAI*.

Yuwei Zhang, Haode Zhang, Li-Ming Zhan, Xiao-Ming Wu, and Albert Lam. 2022. New intent discovery with pre-training and contrastive learning. *ArXiv*, abs/2205.12914.

Nanxuan Zhao, Zhirong Wu, Rynson WH Lau, and Stephen Lin. 2020. What makes instance discrimination good for transfer learning? *arXiv preprint arXiv:2006.06606*.

Yunhua Zhou, Peiju Liu, and Xipeng Qiu. 2022. Knn-contrastive learning for out-of-domain intent classification. In *ACL*.

## A Datasets

We show the detailed statistics of Banking, HWU64 and CLINC datasets in Table 7.

## B Baselines

The details of baselines are as follows:

- **PTK-means** This method pre-trains the encoder network with different IND pre-training objectives, and then performs OOD clustering with the K-means clustering algorithm. In this paper, we employ two different pre-training objectives: CE+SCL and CE+KCL.
- **DeepCluster** This is an iterative clustering method proposed by Caron et al. (2018). In each iteration, firstly, K-means is used to assign pseudo labels to all unlabeled samples, and then the cross-entropy objective is used for representation learning. Due to the randomness of the clustering index, the cluster header parameters need to be reinitialized during each iteration. In the semi-supervised setting, we use the same IND pre-training objective as Zhang et al. (2021)
- **CDAC+** This is the first work of new intent discovery (Lin et al., 2020), and also the first work to propose a two-stage framework for clustering new intents in a semi-supervised setting. Firstly, it pre-trains a BERT-based (Devlin et al., 2019) in-domain intent classifier then uses intent representations to calculate the similarity of OOD intent pairs as weak supervised signals.
- **DeepAligned** This is the second work of new intent discovery (Zhang et al., 2021). It is an advanced version of DeepCluster. The overall process of this method is basically the same as DeepCluster, and the only difference is that it designed a pseudo label alignment strategy to produce aligned cluster assignments for better representation learning. The method first performs K-means cluster assignments, and then performs representation learning. The two processes are iteratively performed in a pipeline manner, which results in the representation and cluster assignments not being updated simultaneously, leading to a suboptimal result.

Dataset	Classes	Training	Validation	Test	Vocabulary	Length (max / mean)
BANKING	77	9,003	1,000	3,080	5,028	79 / 11.91
CLINC	150	18,000	2,250	2,250	7,283	28 / 8.31
HWU64	64	8954	1076	1076	4,948	25 / 6.57

Table 7: Statistics of BANKING, CLINC and HWU64 datasets.

- **DKT** This is the current state-of-the-art method for OOD intent discovery (Mou et al., 2022). In the IND pre-training stage, the CE and SCL objective functions are jointly optimized, and in the OOD clustering stage, instance-level CL and cluster-level CL objectives are used to jointly learn representation and cluster assignment. The main motivation is to design a unified multi-head contrastive learning framework to match the IND pre-training objectives and the OOD clustering objectives.

## C Implementation Details

For a fair comparison with previous work, similar with Mou et al. (2022), we use the pre-trained BERT model (bert-base-uncased<sup>9</sup>, with 12-layer transformer) as our network backbone, and add a pooling layer to get intent representation (dimension=768). Moreover, we freeze all but the last transformer layer parameters to achieve better performance with BERT backbone, and speed up the training procedure as suggested in Zhang et al. (2021). We use two separate two-layer non-linear MLPs (ReLU as activation function) for instance-level head and cluster-level head. For the instance-level head, the output dimensionality is set to 128, and for the cluster-level head, the output dimensionality is set to the number of clusters.

In the IND pre-training stage, the training batch size is 128 and the learning rate is 5e-5; in the OOD clustering stage, the training batch size is 400 for Banking-10%, Banking-20%, Banking-30%, HWU64-30% and 512 for CLINC-30% and the learning rate is 0.0003. Similar with Mou et al. (2022), we use dropout (Gao et al., 2021) to construct augmented examples for contrastive learning in OOD clustering stage with dropout rate 0.1. The temperatures of KCL and KCC are 0.5, and the cluster-level temperature is 1.0. The augmented view is used as a new data point to participate in the KNN search process along with the original view.

<sup>9</sup><https://github.com/google-research/bert>

For the KCL objective function, in order to select K-nearest neighbors in a large enough search space and avoid using an excessively large batch size, we design an efficient dynamic queue mechanism. Specifically, in each iteration, for each sample in the batch, we randomly select 10 samples of the same type as it from the training set, and the queue length is maintained at 10\*batch size. The queue length is set to 1280 in our implementation, which is the maximum value that the current device can bear. For the KCC method, the augmented view is used as a new data point to participate in the KNN search process along with the original view, and we set the threshold to 0.7 and K value to 400, which has been discussed in section 4.4.

We use Adam optimizer (Kingma and Ba, 2015) to train our model, and use the SC value of OOD data in the validation set as the basis for selecting the best checkpoints. All experiments use a single Tesla T4 GPU (16 GB of memory). The pre-training stage of our model lasts about 1 minute per epoch and clustering runs for 0.11 minutes per epoch on Banking-20%<sup>10</sup>. The average value of the trainable model parameters is 17.34M, which is basically the same as DKT. It can be seen that our KCOD method has significantly improved performance compared to DKT, but the cost of time and space complexity is not large.

## D Silhouette Coefficient (SC)

Following Zhang et al. (2021), we use the cluster validity index (CVI) to evaluate the quality of clusters obtained during each training epoch after clustering. Specifically, we adopt an unsupervised metric Silhouette Coefficient (Rousseeuw, 1987) for evaluation:

$$SC = \frac{1}{N} \sum_{i=1}^N \frac{b(\mathbf{I}_i) - a(\mathbf{I}_i)}{\max\{a(\mathbf{I}_i), b(\mathbf{I}_i)\}} \quad (7)$$

where  $a(\mathbf{I}_i)$  is the average distance between  $\mathbf{I}_i$  and all other samples in the  $i$ -th cluster, which indicates

<sup>10</sup>DKT almost consumes 0.5 minutes per epoch for pre-training and 0.08 minutes per epoch for clustering.



the intra-class compactness.  $b(I_i)$  is the smallest distance between  $I_i$  and all samples not in the  $i$ -th cluster, which indicates the inter-class separation. The range of SC is between -1 and 1, and the higher score means the better clustering results.

## E Estimate Cluster C

Since we may not know the exact number of OOD clusters, we use the following estimation method (Zhang et al., 2021) to determine the number of clusters  $K$  before clustering. The method estimates  $C$  with the aid of the well-initialized intent features. We assign a big  $K'$  as the number of clusters at first. As a good feature initialization is helpful for partition-based methods (e.g., k-means), we use the pre-trained model to extract intent features. Then, we perform k-means with the extracted features. We suppose that real clusters tend to be dense even with  $K'$ , and the size of more confident clusters is larger than some threshold  $t$ . Therefore, we drop the low confidence cluster whose size is smaller than  $t$ , and calculate  $K$  with:

$$K = \sum_{i=1}^{K'} \delta(|S_i| \geq t) \quad (8)$$

where  $|S_i|$  is the size of the  $i^{th}$  produced cluster, and  $\delta(\cdot)$  is an indicator function. It outputs 1 if the condition is satisfied, and outputs 0 if not. Notably, we assign the threshold  $t$  as the expected cluster mean size  $\frac{N}{K'}$  in this formula.