Semi-supervised New Slot Discovery with Incremental Clustering

Yuxia Wu
Xi'an Jiaotong University
wuyuxia@stu.xjtu.edu.cn

Lizi Liao
Singapore Management University
lzliao@smu.edu.sg

Xueming Qian
Xi'an Jiaotong University
qianxm@mail.xjtu.edu.cn

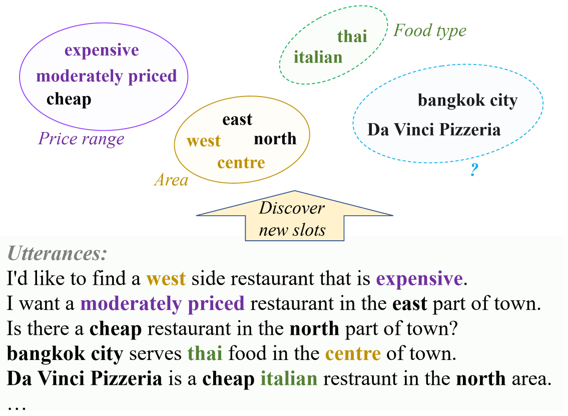
Tat-Seng Chua
Sea-NExT Joint Lab, NUS
dcscts@nus.edu.sg

Abstract

Discovering new slots is critical to the success of dialogue systems. Most existing methods rely on automatic slot induction in an unsupervised fashion or perform domain adaptation across zero or few-shot scenarios. They have difficulties in providing high-quality supervised signals to learn clustering-friendly features, and are limited in effectively transferring the prior knowledge from known slots to new slots. In this work, we propose a Semi-supervised Incremental Clustering method (**SIC**), to discover new slots with the aid of existing linguistic annotation models and limited known slot data. Specifically, we harvest slot value candidates with NLP model cues and innovatively formulate the slot discovery task under an incremental clustering framework. The model gradually calibrates slot representations under the supervision of generated pseudo-labels, and automatically learns to terminate when no more salient slot remains. Our thorough evaluation on five public datasets demonstrates that the proposed method significantly outperforms state-of-the-art models.

1 Introduction

Slot filling identifies contiguous word spans in an utterance based on *slots* to represent the meaning of user (Young et al., 2013; Zhang et al., 2019b). It is essential for the performance of dialogue systems (Fei et al., 2022a; Ye et al., 2022a; Liao et al., 2021b). Traditional supervised methods have shown remarkable performance in this task (Hakkani-Tür et al., 2016; Kurata et al., 2016; Goo et al., 2018; Qin et al., 2020). However, such approaches can only recognize pre-defined entity types from a limited slot set and require a significant amount of labeled training data, which is laborious and expensive to obtain. In practical settings, new unseen slots (as shown in Figure 1) may emerge after the deployment of the dialogue system, rendering these supervised models ineffective.



Utterances:
I'd like to find a **west** side restaurant that is **expensive**.
I want a **moderately priced** restaurant in the **east** part of town.
Is there a **cheap** restaurant in the **north** part of town?
bangkok city serves **thai** food in the **centre** of town.
Da Vinci Pizzeria is a **cheap** **italian** restraunt in the **north** area.
…

Figure 1: An illustration of the new slot discovery task. Those in bold font are extracted value candidates.

Hence, there are works trying to do automatic slot induction without human labels. Such methods often work in two steps: extract slot candidates and value first, then obtain slots via ranking. For example, Chen et al. (2014) combines semantic frame parsing with word embeddings for slot induction. Chen et al. (2015) further construct lexical knowledge graphs and perform a random walk to get slots. Hudeček et al. (2021) extend the ranking into an iterative process and build a slot tagger based on obtained slots for higher recall. Nonetheless, the ranking process needs deliberate human intervention and largely affects the final results (Zeng et al., 2021). Moreover, instead of entirely without labeled data, we often have access to a small or partial amount of that in real practice.

Therefore, another line of efforts resort to zero-shot (or few-shot) cross-domain adaptation, whose goal is to identify unseen slots in the target domain by leveraging evidence from labeled data in the source domain. These methods can be organized into two types. In one type, slot description or even example values are directly interacted with user utterances to conduct one-stage slot filling individually for each slot (Bapna et al., 2017; Shah et al., 2019; Lee and Jha, 2019; Hou et al., 2020; Oguz and Vu, 2021). In the other, the slot filling

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6207–6218
December 7-11, 2022 ©2022 Association for Computational Linguistics

task is decomposed into two or more stages (Liu et al., 2020; He et al., 2020; Siddique et al., 2021). First, they identify all slot values from utterances by the coarse-grained binary sequence labeling. Then these values are mapped to the representations of different slots in semantic space for slot assignment.

However, there are several drawbacks in these methods. Firstly, existing domain adaption efforts pay inadequate attention to slot value identification. They either ignore this process completely as in (Bapna et al., 2017) or oversimplify it as a coarse-grained sequence tagging (Liu et al., 2020). As evidenced in Hudeček et al. (2021), identifying proper slot values is of critical importance for discovering new slots. Secondly, existing methods tend to assume that new slot names, descriptions or even value examples are available, which is hardly true for slot discovery. Overemphasizing these auxiliary description information inhibits the model to learn from existing known slots. Also, they fail to provide proper guidance for the model to learn clustering-friendly features.

In this work, we thus propose a Semi-supervised Incremental Clustering method (**SIC**) to discover new slots. Instead of sequence labeling or span extraction, we design an iterative clustering and updating framework to gradually solicit evidence from both labeled and unlabeled data. As illustrated from Figure 2, relying on slot value candidates extracted via linguistic annotation models, we firstly pre-train a feature extractor with the limited labeled data under the supervision of the softmax loss. Then, we iteratively perform clustering and feature extractor training. The former provides high-quality self-supervised signals to guide the latter training stage, while the latter yields clustering-friendly features for the former. For each of the clustering stage, we estimate the cluster numbers and gradually expand for stable slot discovery. We evaluate the proposed model on five public datasets. It achieves new state-of-the-art performance across these datasets in various evaluation metrics.

The contributions are summarized as follows:

- We design a semi-supervised learning scheme for new slot discovery, which does not require any prior knowledge about new slots.
- We perform clustering and feature extractor training iteratively to harvest high-quality self-supervised signals and learn discriminative features for grouping values to slots.
- Experiments show that the proposed SIC model significantly outperforms state-of-the-art models across several public datasets.

2 Related Work

Our work is closely related to automatic slot induction works, cross-domain slot filling and semi-supervised Spoken Language Understanding (SLU) works. We briefly discuss their connections and analyze their differences.

2.1 Automatic Slot Induction

As an essential part of task-oriented dialogue systems, slot filling has been well studied in the supervised learning settings (Liao et al., 2021a,c). Been regarded as a sequence tagging or span extraction task, it has leveraged various sequential models such as recurrent neural network (RNN), conditional random field (CRF), or Transformers (Goo et al., 2018; Zhang et al., 2019a) for good performance. But such approaches are limited in real applications where unseen slots are common. Hence, automatic slot induction attracts much attention. The widely used pipeline is to first extract the candidate entities/slots via external semantic resources such as a frame-semantic parser. Then a ranking method is applied to select the slots. For example, Chen et al. (2014, 2015) parse the utterances using SEMAFOR to extract candidate values and slots, followed by a slot ranking method to obtain the final slots.

Recently, Hudeček et al. (2021) modify the ranking process into an iterative fashion and build a slot tagger based on obtained slots. Zeng et al. (2021) extend the induction into a three-step process with more arbitrary designs. Such process needs human intervention for good performance. More importantly, instead of entirely without slot labeled data, we often have access to a small or partial amount of that in real practice.

2.2 Cross-domain Slot Filling

To make use of slot labeled data at hand, a line of work focus on adapting such knowledge to new domain with unseen slots. There are both zero-shot methods (Bapna et al., 2017; Lee and Jha, 2019; Shah et al., 2019; Liu et al., 2020; He et al., 2020; Siddique et al., 2021; Wang et al., 2021b) and few-shot studies (Hou et al., 2020). These works mainly focus on incorporating textual descriptions of slots into sequence labeling models to handle the un-

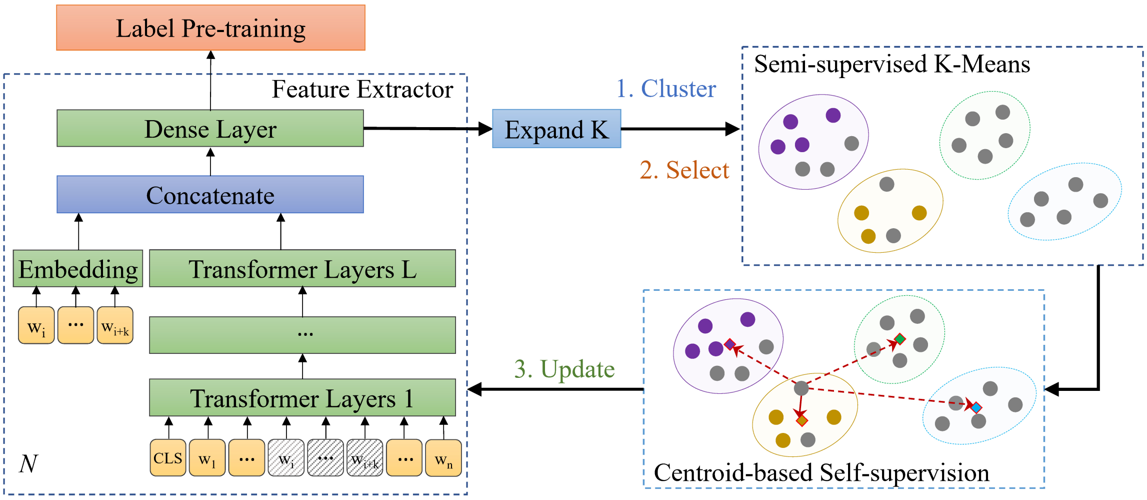


Figure 2: The SIC framework. Pre-trained under the supervision of few labeled samples, the feature extractor extracts representations for all samples to decide K expansion. Then the samples are clustered by semi-supervised K-Means, and the centroids provide self-supervision for further updating the feature extractor for next round.

seen slots (Bapna et al., 2017; Liu et al., 2020; He et al., 2020). They assume that slot descriptions or even some example values are available to use. However, it might not be realistic for slot discovery and the creation of slot descriptions needs qualified linguistic expertise. Therefore, Wu et al. (2021) introduces a novel slot detection task which detects the potential unknown slots without differentiating them. Some other researchers resort to meta learning instead of slot descriptions or names (Oguz and Vu, 2021; Wang et al., 2021a). They learn and transfer the meta-knowledge from the labeled examples and handle the unseen slots in the low-resource domain.

However, these existing domain adaption efforts pay little attention to the slot value identification process. For example, methods like (Bapna et al., 2017) ignore it completely, and methods such as (Liu et al., 2020) oversimplify it as a coarse-grained binary sequence tagging. Actually, the slot values are of critical importance for new slot discovery. More importantly, these methods heavily rely on auxiliary information such as new slot names, descriptions or even value examples, which inhibits the model to leverage prior knowledge from existing known slots.

2.3 Semi-supervised SLU

Our work is also connected to semi-supervised SLU studies, where intents and slots are predicted together (Shi et al., 2018; Zeng et al., 2021). Oftentimes, the intents are decided first, then the corresponding slots are determined. Due to the nature of the intent discovery task, semi-supervised clustering methods have been widely applied. They incorporate prior knowledge as constraints to guide the clustering process. The prior knowledge includes the pair-wise information (must-link and cannot-link constraints) and label information (Basu et al., 2004; Xie et al., 2016). A popular scheme is to follow a two-stage pipeline (Hsu et al., 2018, 2019; Han et al., 2019): firstly train a pair-wise similarity network based on the labeled data, then cluster unlabeled data with constraints to discover the unseen classes. For example, Lin et al. (2020) introduce constrained deep adaptive clustering with cluster refinement (CDAC+) for intent discovery in dialogue systems. They learn prior knowledge about the pair-wise similarity with the limited labeled data to guide clustering, but fail to provide specific supervision for unlabeled data or identify the number of novel intents. Recently, Zhang et al. (2021b) leverages the labeled data to pre-train a classifier model and perform clustering. An alignment strategy is designed to tackle the label inconsistency problem. However, these methods only focus on new intent discovery. The task of new slot discovery is more complicated in nature.

3 Method

In this section, we formally define the semi-supervised new slot discovery task first. Then we break into subsections to elaborate our proposed method as shown in Figure 2. Starting from the candidate value extraction and feature extractor pre-training, we gradually extend to the iterative clustering and updating scheme.

In the slot filling task, given an utterance $U = \langle w_1, ..., w_n \rangle$ with n tokens, the target is to predict a corresponding tag sequence $O = \langle o_1, ..., o_n \rangle$ in BIO format. Each tag o_i can take three types of values: B-slot type, I-slot type and O, where B- and I- indicate the start and inside token of one slot type, and O means the token does not belong to any slot type. Here, we work differently. Suppose there is a candidate value $x = \langle w_i, \cdots, w_{i+k} \rangle$ of length $|x| = k + 1$ identified from the utterance U, our setting is that in the whole dataset $\mathcal{D}$ with N candidate values, a limited amount are labeled $\mathcal{D}^{\mathcal{L}} = \{x_i, y_i\}_{i=1}^{M} \in \mathcal{X} \times \mathcal{Y}_{\mathcal{L}}$, while the rest of them are unlabeled $\mathcal{D}^{\mathcal{U}} = \{x_i, y_i\}_{i=1}^{N-M} \in \mathcal{X} \times \mathcal{Y}_{\mathcal{U}}$. We have $\mathcal{Y}_{\mathcal{L}} \mathcal{Y}_{\mathcal{U}}$ which indicates that we have a set of new slots to discover and the number of these unknown slots is not given. In our work, the candidate values are extracted via existing NLP models. We also experiment on ground truth values to compare the difference.

3.1 Candidate Value Extraction and Filtering

Candidate value extraction is an important part of the slot discovery task. The values can be a single word or span in users' utterances that convey essential semantic information about users' requirements. Inspired from (Hudeček et al., 2021), we use a frame semantic parser SEMAFOR (Das et al., 2010, 2014) and NER (named entities recognition) to extract the candidate values, but other models, such as SRL(semantic role labeling) (Palmer et al., 2010) or keyword extraction (Hulth, 2003) can be used in general. The SEMAFOR is a FrameNet-style semantic parsing tool developed based on Frame Semantics (Baker et al., 1998). It can automatically extract the semantic frame elements and lexical units from English sentences. Here, we use a simple union of results provided by all annotation models [1].

However, not all the extracted results should be used as the slot value candidates. We observe that there are still a certain amount of extracted spans conveying irrelevant information about users' queries. Therefore, we further conduct a simple filtering process to yield more suitable candidate values. In detail, we remove the stop words by the NLTK tool and set a threshold to remove these spans that appeared less than a certain number of times. Besides, we also delete these frequently appeared but meaningless values such as the word 'another', 'must', 'please', 'find' and so on.

3.2 Feature Extractor Pre-training

The aim of slot discovery is to group some salient candidate values in the utterances into coherent slot structures. The inherent semantics of the candidate value itself and the semantics of its context are both essential for the grouping process. As the example shown in Figure 1, the extracted candidate values (in bold) '*east*' and '*north*' are semantically similar by themselves (indicating direction) and by their context (find a restaurant in a certain part of the town), hence they are more likely be grouped together.

To capture these, we adopt the pre-trained BERT model as the backbone to obtain the candidate value representations. Given a candidate value $x = \langle w_i, \cdots, w_{i+k} \rangle$ inside the utterance U with n tokens ($0 \leq k \leq n$), we apply mean pooling of tokens' BERT embeddings to obtain the inner value representations:

$$\begin{aligned} \langle \mathbf{e}_i, \cdots, \mathbf{e}_{i+k} \rangle &= BERT(\langle w_i, \cdots, w_{i+k} \rangle), \\ \mathbf{v}^{inner} &= mean\_pooling(\langle \mathbf{e}_i, \cdots, \mathbf{e}_{i+k} \rangle), \end{aligned}$$

where $\mathbf{e}_i$ denotes the embedding vector of the token w_i in BERT model.

For the context representations, we deliberately replace the value span with the special mask token *[mask]* to remove the effect of the candidate value. We reconstruct the original utterance into $U' = \langle w_1, \cdots, [mask]_i, \cdots, [mask]_{i+k}, \cdots, w_n \rangle$[2]. As calculated via the self-attention of BERT, the *[mask]* tokens integrate the context information for the masked candidate value. Hence, we use mean pooling on the output of these *[mask]* tokens to obtain the context representation:

$$\begin{aligned} \langle \mathbf{h}_1, \cdots, \mathbf{h}_n \rangle &= BERT(U'), \\ \mathbf{v}^{context} &= mean\_pooling(\langle \mathbf{h}_i, \cdots, \mathbf{h}_{i+k} \rangle), \end{aligned}$$

where $\mathbf{h}_i$ denotes the embedding of the *[mask]* token $[mask]^i$ in the last hidden layer of BERT.

The final representation of candidate value is obtained via the concatenation of $\mathbf{v}^{inner}$ and $\mathbf{v}^{context}$:

$$\mathbf{v} = tanh(\mathbf{W}_1[\mathbf{v}^{inner}; \mathbf{v}^{context}]^T + \mathbf{b}_1),$$

[1] If the same token span is labeled multiple times by different annotation sources, the span is more likely to be considered as a candidate value. We only make use of the value span and the various labels from tools are discarded.

[2] Special tokens such as *[CLS] in beginning and [SEP]* at end are omitted for easy illustration.

where $\mathbf{W}_1$ and $\mathbf{b}_1$ represent the learnable weight matrix and bias in the dense layer. Then we feed the final representation $\mathbf{v}$ into the classifier layer to obtain the slot type prediction:

$$\mathbf{y} = Softmax(\mathbf{W}_2\mathbf{v}^T + \mathbf{b}_2),$$

where $\mathbf{W}_2$ and $\mathbf{b}_2$ are the learnable weight matrix and bias for the classifier layer.

As mentioned before, we have some limited labeled data with known slots in $\mathcal{Y}_{\mathcal{L}}$. To effectively utilize such prior knowledge, we pre-train the aforementioned classifier model with the labeled data under the supervision of the cross-entropy loss. It learns good initial network parameters. After pre-training, we remove the classifier layer and leverage the remaining parts as the feature extractor for the subsequent processing.

3.3 Incremental Clustering

After pre-training on the limited labeled data, we need to transfer knowledge about these known slots to the unknown ones and solicit evidence from both labeled and unlabeled data to discover new slots. Hence, we use the pre-trained feature extractor to extract features for all candidate values. To discover semantically coherent and well separated slots, there are two critical issues to address: firstly, how to find the correct number of new slots and obtain good cluster results; secondly, how to further make use of the cluster results and gradually update the feature extractor towards better cluster-friendly features. Therefore, we design an iterative cluster and updating scheme which performs the following two steps (Section 3.3.1 and Section 3.3.2) alternately.

3.3.1 Expand K and Clustering

We refine the traditional K-means to group candidate values with similar representations. A key hyper-parameter, the number of clusters K, is often unknown in practice due to the lack of information about the corpus. Therefore, suppose the labeled data contains $K_0 = |\mathcal{Y}_{\mathcal{L}}|$ slots, we propose a simple method to gradually expand K in each iteration. Specifically, we first double K_{t-1} in the last iteration as $K_{max} = 2 \times K_{t-1}$ [3] to get the largest possible cluster number in the current iteration t. Then, we perform k-means with the extracted features. We assume that real clusters tend to be dense even with K_{max}, and the size of more confident clusters is larger than some threshold ϵ (Zhang et al., 2021b). Hence, we drop the low confidence clusters with a size smaller than ϵ, and calculate K_t:

$$K_t = \sum_{i=1}^{K_{max}} \mathbb{1}(|C_i| > \epsilon),$$

where $|C_i|$ is the size of the i-th produced cluster.

For semi-supervised K-means, we initialize the K_t centroids from two parts. For the labeled data, we compute the centroids as the average representations of the candidate values belonging to each slot. For these unlabeled data, we adopt k-means++(Arthur and Vassilvitskii, 2006) to initialize the remaining centroids. For each cluster iteration, we modify the assignments of the labeled data to be the original labels. Then we update the centroids with the new assignments. In this way, we force the assignments of the labeled data unchanged and each unlabeled sample is assigned to one cluster based on its distances to the centroids.

3.3.2 Centroid-based Self-Supervision

In the current iteration t, we will obtain cluster assignments on the K_t clusters after the clustering process. As we have a large amount of unlabeled data and the feature extractor is trained on partial data, the cluster assignments will contain noise. To alleviate this issue, we conduct sample selection to choose the data with high confidence. Suppose there are K_t centroids, we calculate the similarity score of each value sample $\mathbf{v}_i$ to its corresponding cluster centroid as s_i. We set threshold γ on the score to select high confidence samples $\mathcal{D}^{\mathcal{S}} = \{x_i, y_i : s_i \geq \gamma\}$. Then we can expand the labeled set as:

$$\mathcal{D}^{\mathcal{L}} = \mathcal{D}^{\mathcal{L}} \cup \mathcal{D}^{\mathcal{S}},$$

where the slot label set also expands to $|\mathcal{Y}_{\mathcal{L}}| = K_t$ at the same time.

With the updated labeled set $\mathcal{D}^{\mathcal{L}}$, we update the corresponding centroids for each cluster as $\{c_1, \cdots, c_{K_t}\}$. We then make use of these centroids to update the feature extractor using centroid-based self-supervision. The intuition behind this is that each sample should be close to its cluster center while be far away from other cluster centers. Therefore, we have the objective function for

[3] Many different ways can be used to estimate the K such as Thorndike (1953); Ben-David et al. (2007), but we empirically find this simple and effective on all five datasets.

updating the feature extractor network as

$$L_s = -\sum_{i=1}^{|\mathcal{D}^{\mathcal{L}}|} \log \frac{exp(\mathbf{v}_i \cdot \mathbf{c}_i/\tau)}{\sum\limits_{\mathbf{c}_j \neq \mathbf{c}_i} exp(\mathbf{v}_i \cdot \mathbf{c}_j/\tau)},$$

where $\mathbf{c}_i$ is the corresponding cluster centroid rerpesentation for the sample x_i and τ is the temperature hyper parameter.

By updating the feature extractor with the above learning objective, it tends to learn cluster-friendly features for next round clustering. We terminate the iteration when K_t stops increasing.

4 Experiments

4.1 Datasets

We conduct experiments on the following datasets:

- CamRest676 (CR) (Wen et al., 2017) is a task-oriented dialogue corpus in the restaurant domain with 2,744 utterances and 4 slots.
- MultiWOZ (Bojanowski et al., 2017; Eric et al., 2020) is a multi-domain dialogue corpus. We choose two domains: WOZ-hotel(WH) in the hotel domain with 14,435 utterances and 9 slots; WOZ-attr (WA) in the attraction domain with 7524 utterances and 8 slots.
- Cambridge SLU (CS) (Henderson et al., 2012) is also a dialogue corpus in the restaurant domain contains 10,569 utterances and 5 slots.
- ATIS (AT) (Hemphill et al., 1990) is a dialogue corpus in flights domain with 4,978 utterances and 79 slots.

4.2 Training Details

We randomly select 75% of all slots as the known slots and choose 10% data for each slot as labeled data. We apply the pre-trained BERT model (bert-base-uncased, with 12 transformer layers) as our backbone to pre-train the feature extractor by the labeled data. Most of the hyper-parameters are the same as the default parameter of the BERT model. The batch size is set to 64 and the learning rate is set to 5e-5. The dimension of the token representation is 768. During training, we freeze all but the last transformer layer parameters. To define the threshold when selecting the high confident unlabeled samples, we apply a simple greedy search method to validate the performance based on the candidate thresholds from 0.5 to 0.95. All the parameters are tuned on the validation set. We implement our method using a public TEXTOIR toolkit which contains standard and unified interfaces to ensure fair comparison on different baselines (Zhang et al., 2021a). We extend the original interfaces to the open slot discovery task on our own datasets.

4.3 Evaluation Metrics

The goal of slot discovery is to identify the slot type of each value in the utterance. We adopt the popular classification metric F1 to measure the performance. A mapping between the discovered slots and the ground truth slots is constructed.

We also adopt the clustering based metrics similar to (Zhang et al., 2021b): Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and Accuracy (ACC) to measure the clustering performance of our method and the clustering-based baselines. NMI computes the mutual information of the predicted clusters and the true classes. ARI measures the pair-wise accuracy about whether any two samples belong to the same cluster. ACC is used to compare the obtained labels with the ground truth labels, where the mapping is also needed similar to F1.

4.4 Baselines

The baselines we used can be divided into four categories: unsupervised, supervised, weakly supervised and semi-supervised methods. For fair comparison, we use the same BERT model as the backbone for different methods.

- **Supervised**: We use the same set of baselines in (Hudeček et al., 2021) including *Tag-supervised* and *Dict-supervised* methods.
- **Unsupervised**: Chen et al. (2014) combines the FrameNet semantic parsing and the pre-trained word embeddings for candidate slot ranking. We also compare with *WeakS-notag*, the variant of (Hudeček et al., 2021) with only slot merging and selection step.
- **Weakly supervised**: *WeakS-full* (Hudeček et al., 2021) is the state-of-the-art weakly supervised method for slot discovery. They use existing tools to yield slots and further train a slot tagger model with extracted results.
- **Semi-supervised**: We compare our method with several semi-supervised methods including *BERT-KCL* (Hsu et al., 2018), *BERT-MCL* (Hsu et al., 2019), *BERT-DTC* (Han et al., 2019), *CDAC+* (Lin et al., 2020) and *DeepAligned* (Zhang et al., 2021b). All of these methods need to know the slot numbers

of the unlabeled data except for *DeepAligned*, which predicts the number of slots at the very beginning.

4.5 Main Results

We report the main results for all compared methods in Table 1. Generally speaking, when the full supervision is not available, the proposed method *SIC* performs consistently better than all the baseline methods on nearly all five datasets, which validates our design of the semi-supervised incremental clustering scheme. Although the *Tag-supervised* and *Dict-supervised* method achieve high performance results, they require full supervision. For example, *Dict-supervised* uses a dictionary of labels covering all slots. However, we still observe that *SIC* performs slightly better than them on the CS and WA datasets. This might be because CS and WA contain more variant values for each slot. *SIC* manages to capture the relationships between these values but *Dict-supervised* and *Tag-supervised* methods are limited on this.

Compared with unsupervised methods (Unsup.) and weakly supervised (Weakly-sup.) method, *SIC* maintains a large performance gap. Although all these methods make good use of NLP tools such as SEMAFOR, the former methods only focus on merging and selecting frames as slots, and the latter uses obtained results to train a neural model to increase generability. In *SIC*, our special design for feature extractor captures both the inherent and contextual semantics for candidate values and makes use of these to group them into slots. Moreover, *SIC* also manages to learn from the patterns of known slots and transfer them to unknown slots.

Under the semi-supervised setting (Semi-sup.), *SIC* constantly outperforms *BERT-KCL*, *BERT-MCL*, *BERT-DTC*, *CDAC+* and *DeepAligned* in all datasets using extracted candidate values. Note that the first four methods all use the ground truth slot number. Still, we observe a big performance drop on the CR, CS, WH datasets for these four methods. This is probably because these methods overemphasize pairwise similarity as prior knowledge and these datasets have very imbalanced class-wise sample distributions. We observe that they tend to assign most samples to some specific slots. On the WA and AT datasets, the performance gaps are narrower. Among methods in this semi-supervised group, the *DeepAligned* method performs the second best. As it also conducts gradual clustering on the whole dataset, this suggests the validity of gradually leveraging evidence from both labeled and unlabeled data. Our method SIC outperforms DeepAligned method on all datasets. That's because the DeepAligned method uses fixed cluster number K, which will affect the performance if the predicted K is inaccurate. Besides, we gradually select the samples with high confidence to make the model learn cluster-friendly features for slot discovery.

As the proposed method is based on a clustering scheme. We further evaluate via cluster-based metrics and compare with these cluster-based methods. Table 2 shows the performance results of these semi-supervised methods under the popular clustering metrics *NMI* and *ARI*.

We can observe that our method *SIC* achieves the best performance on all datasets, indicating that our feature extractor training and incremental clustering process have the advantage to obtain better cluster-friendly features, hence resulting in advanced new slot discovery performance. Among the baselines, the *DeepAligned* method shows comparable clustering performance on most of the datasets. They gradually cluster data samples and conduct alignment of clustering assignments among different epochs. The clustering results provide pseudo-labels for further training. Such an incremental clustering scheme is similar to ours. However, the fixed cluster number and representations of clusters hinder its adaptability during the learning process.

4.6 In-depth Analysis

4.6.1 Effect of Candidate Values

As shown in Table 1, when ground truth candidate values are applied, all semi-supervised methods perform better. This is as expected because existing language tools still bring in noise even after the filtering process. However, the better performance of *SIC* still shows that making use of such prior knowledge is advantageous.

To further investigate the effect of the candidate value representation strategy, we further experiment on two variant models: *w/o inner* and *w/o context* which indicate the model with only context representation and inner representation, respectively. The results are shown in Table 3. We can observe that the performance is decreased when we only consider either of them. Generally speaking, the inner representations are more important, and

		CR		CS		WH		WA		AT	
		Extr	*GT*	*Extr*	*GT*	*Extr*	*GT*	*Extr*	*GT*	*Extr*	*GT*
Sup.	*Tag-supervised*	**0.778**	-	0.724	-	**0.742**	-	0.731	-	**0.848**	-
	Dict-supervised	0.705	-	0.753	-	0.750	-	0.665	-	0.678	-
Unsup.	*Chen et al.*	0.535	-	0.590	-	0.382	-	0.375	-	0.616	-
	WeakS-notag	0.552	-	0.664	-	0.388	-	0.383	-	0.648	-
Weakly-sup.	*WeakS-full*	0.665	-	0.692	-	0.548	-	0.439	-	0.710	-
Semi-sup.	*BERT-KCL**	0.189	0.224	0.131	0.188	0.178	0.346	0.560	0.731	0.492	0.584
	*BERT-MCL**	0.188	0.321	0.129	0.210	0.179	0.332	0.532	0.729	0.504	0.591
	*BERT-DTC**	0.131	0.303	0.138	0.206	0.170	0.334	0.545	0.670	0.543	0.578
	*CDAC+**	0.204	0.270	0.178	0.221	0.174	0.332	0.552	0.641	0.582	0.588
	DeepAligned	0.663	0.901	0.633	0.899	0.378	0.750	0.644	0.719	0.629	0.676
	SIC(Ours)	**0.706**	**0.913**	**0.770**	**0.969**	**0.588**	**0.824**	**0.761**	**0.851**	**0.638**	**0.721**

Table 1: Results compared with baselines on F1. * indicates that the method uses the ground truth slot number. *Extr* and *GT* represent that we use extracted candidate values or ground truth values respectively.

	CR		CS		WH		WA		AT	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
*BERT-KCL**	22.06	12.48	12.56	6.57	12.10	8.99	63.27	61.27	29.02	54.42
*BERT-MCL**	64.21	63.03	10.60	3.77	11.49	9.19	63.25	61.20	30.65	55.43
*BERT-DTC**	64.08	34.25	11.35	2.61	11.67	8.83	64.64	65.51	27.61	52.43
*CDAC+**	21.22	13.55	31.12	26.27	11.71	9.05	69.07	71.04	30.69	55.90
DeepAligned	82.05	80.01	88.77	90.20	81.53	76.21	70.84	68.59	71.44	78.78
SIC(Ours)	**82.61**	**81.23**	**90.62**	**92.88**	**87.71**	**86.86**	**71.36**	**72.87**	**73.70**	**78.85**

Table 2: Results compared with baselines via cluster-based metrics. * indicates that the method uses the ground truth slot number.

	CR	CS	WH	WA	AT
SIC	0.706	0.770	0.588	0.761	0.638
w/o inner	0.661	0.514	0.530	0.573	0.619
w/o context	0.686	0.688	0.449	0.64	0.620

Table 3: Effect of the candidate value representation.

the context representations are indispensable too. Still, we see that the context is more useful in the WH dataset as evidenced by a larger performance drop for *w/o context* than *w/o inner*, because there are some slots containing numeric values such as "people" , "stay", "star". For example, in the utterance "Book it for 1 people and 5 nights starting from Sunday" in WH, the values '1' and '5' belong to the slot "people" and "stay", respectively. If we only consider the value itself, it will make the model difficult to recognize their slots.

4.6.2 Threshold for Sample Selection

To explore the influence of the threshold γ for sample selection during centroid-based self-supervision, we vary the threshold in the range from 0.5 to 0.95 with an interval of 0.05. Due to space limitation, we show the results on the AT dataset in Figure 3 and the results on the remaining datasets in Figure A.2 (See in Appendix A.2). We observe that the threshold indeed affects the final performance. Generally speaking, the optimal threshold for most datasets is between $0.7 \sim 0.95$, which is reasonable. Because a small threshold will encourage the model to select more unlabeled data at the early iterations. This will bring more noise since the model is not well-trained yet. Also, we show that the threshold works differently in differ-

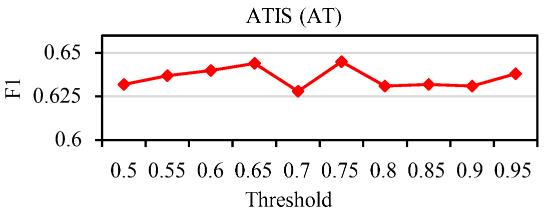


Figure 3: Effect of sample selection threshold on AT.

ent datasets. For example, the results are relatively stable on AT but more fluctuate on CS and WA. These patterns might be decided by the inherent nature of these datasets, such as the sizes of different slots, closeness among values inside slots and the relations among different slots *etc*.

4.6.3 Effect of Labeled Data

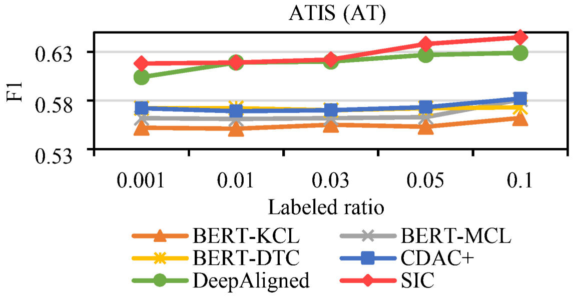
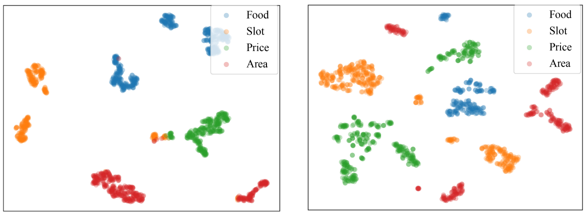


Figure 4: Effect of labeled data ratio on AT.

We also vary the ratio of labeled data in the training set in the range of $0.001, 0.01, 0.03, 0.05$ and 0.1 to test the effect of these ratios. The results on the AT dataset are shown in Figure 4 and the results on the remaining datasets in Figure A.1 (See in Appendix A.1). We observe that even if the ratio of labeled data is much lower than 0.1, the proposed *SIC* still performs better than most baselines. This demonstrates its strength in learning from labeled data and discovering inherent patterns from unlabeled data. At the same time, we see that the performance increases as more labeled data is leveraged. This is as expected. Some methods show performance drop such as *BERT-KCL* and *DeepAligned* on WA. This might be because we randomly sample labeled data for each ratio independently during our experiments. These methods might be more sensitive to certain groups of labeled data.

4.6.4 Visualizing Learned Features

In Figure 5, we further visualize the candidate value representations via t-SNE (Van der Maaten and Hinton, 2008) for the CR dataset on the trained feature extractor model. Dots with different colors represent candidate values in different slot clusters. Evidently, there is a clear margin between clusters captured by the 2D representations learned by our feature extractor. This indicates that our model learns cluster-friendly features for slot discovery. Also, we observe that the clusters obtained by our method are generally more clear and well-separated than those obtained by *DeepAligned*. This shows the superiority of our method *SIC* in the same semi-supervised slot discovery setting.



(a) SIC

(b) DeepAligned

Figure 5: t-SNE visualization of learned features.

5 Conclusion

We presented a novel approach for semi-supervised new slot discovery in dialogue systems that detects new slots without any prior knowledge such as slot name, description or new slot number. Our method removes the heavy reliance on large-scale annotated data and shows great potential in handling unseen situations for robust system deployment. It leverages off-the-shelf linguistic annotation models to extract candidate values, then builds an incremental clustering scheme to gradually solicit evidence from both labeled and unlabeled data to discover slot structures from the corpus. Experiments on five datasets in four domains mark significant improvements over various groups of baselines on different evaluation metrics.

Limitations

Our work has the following potential limitations. Firstly, our approach relies on existing linguistic annotation models. We show that the method is able to combine multiple annotation sources, working better than the original annotation by gradually selecting good ones. Nevertheless, the results are still limited by the input annotation quality. Secondly, this work only focuses on discovering new slots. As new intents and slots often are closely intertwined, we plan to investigate these from a joint perspective for better performance and practicality in the future.

Acknowledgments

This research was supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant, and the Sea-NExT Joint Lab.

References

David Arthur and Sergei Vassilvitskii. 2006. k-means++: The advantages of careful seeding. Technical report, Stanford.

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *COLING*, pages 86–90.

Ankur Bapna, Gokhan Tür, Dilek Hakkani-Tür, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *INTERSPEECH*, pages 2476–2480.

Sugato Basu, Mikhail Bilenko, and Raymond J Mooney. 2004. A probabilistic framework for semi-supervised clustering. In *SIGKDD*, pages 59–68.

Shai Ben-David, Dávid Pál, and Hans Ulrich Simon. 2007. Stability of k-means clustering. In *COLT*, pages 20–34.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

Yun-Nung Chen, William Yang Wang, and Alexander Rudnicky. 2015. Jointly modeling inter-slot relations by random walk on knowledge graphs for unsupervised spoken language understanding. In *NAACL*, pages 619–629.

Yun-Nung Chen, William Yang Wang, and Alexander I Rudnicky. 2014. Leveraging frame semantics and distributional semantics for unsupervised semantic slot induction in spoken dialogue systems. In *SLT*, pages 584–589.

Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. 2014. Frame-semantic parsing. *Computational linguistics*, 40(1):9–56.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A Smith. 2010. Probabilistic frame-semantic parsing. In *NAACL*, pages 948–956.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *LREC*, pages 422–428.

Hao Fei, Jingye Li, Shengqiong Wu, Chenliang Li, Donghong Ji, and Fei Li. 2022a. Global inference with explicit syntactic and discourse structures for dialogue-level relation extraction. In *IJCAI*, pages 4107–4113.

Hao Fei, Shengqiong Wu, Meishan Zhang, Yafeng Ren, and Donghong Ji. 2022b. Conversational semantic role labeling with predicate-oriented latent graph. In *IJCAI*, pages 4114–4120.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *NAACL*, pages 753–757.

Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Interspeech*, pages 715–719.

Kai Han, Andrea Vedaldi, and Andrew Zisserman. 2019. Learning to discover novel visual categories via deep transfer clustering. In *ICCV*, pages 8401–8409.

Keqing He, Jinchao Zhang, Yuanmeng Yan, Weiran Xu, Cheng Niu, and Jie Zhou. 2020. Contrastive zero-shot learning for cross-domain slot filling with adversarial attack. In *COLING*, pages 1461–1467.

Yingxu He, Lizi Liao, Zheng Zhang, and Tat-Seng Chua. 2021. Towards enriching responses with crowd-sourced knowledge for task-oriented dialogue. In *ACM MM Workshop on MuCAI*, pages 3–11.

Charles T Hemphill, John J Godfrey, and George R Doddington. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Workshop*, pages 96–101.

Matthew Henderson, Milica Gašić, Blaise Thomson, Pirros Tsiakoulis, Kai Yu, and Steve Young. 2012. Discriminative spoken language understanding using word confusion networks. In *SLT*, pages 176–181.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *ACL*, pages 1381–1393.

Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. 2018. Learning to cluster in order to transfer across domains and tasks. In *ICLR*.

Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. 2019. Multi-class classification without multi-class labels. In *ICLR*.

Vojtěch Hudeček, Ondřej Dušek, and Zhou Yu. 2021. Discovering dialogue slots with weak supervision. In *ACL-IJCNLP*, pages 2430–2442.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *EMNLP*, pages 216–223.

Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. 2016. Leveraging sentence-level information with encoder lstm for semantic slot filling. In *EMNLP*, pages 2077–2083.

Sungjin Lee and Rahul Jha. 2019. Zero-shot adaptive transfer for conversational language understanding. In *AAAI*, pages 6642–6649.

Lizi Liao, Le Hong Long, Yunshan Ma, Wenqiang Lei, and Tat-Seng Chua. 2021a. Dialogue state tracking with incremental reasoning. *TACL*, pages 557–569.

Lizi Liao, Le Hong Long, Zheng Zhang, Minlie Huang, and Tat-Seng Chua. 2021b. Mmconv: an environment for multimodal conversational search across multiple domains. In *SIGIR*, pages 675–684.

Lizi Liao, Yunshan Ma, Xiangnan He, Richang Hong, and Tat-seng Chua. 2018. Knowledge-aware multimodal dialogue systems. In *ACM MM*, pages 801–809.

Lizi Liao, Tongyao Zhu, Le Hong Long, and Tat Seng Chua. 2021c. Multi-domain dialogue state tracking with recursive inference. In *WWW*, pages 2568–2577.

Ting-En Lin, Hua Xu, and Hanlei Zhang. 2020. Discovering new intents via constrained deep adaptive clustering with cluster refinement. In *AAAI*, pages 8360–8367.

Zihan Liu, Genta Indra Winata, Peng Xu, and Pascale Fung. 2020. Coach: A coarse-to-fine approach for cross-domain slot filling. In *ACL*, pages 19–25.

Cennet Oguz and Ngoc Thang Vu. 2021. Few-shot learning for slot tagging with attentive relational network. In *EACL*, pages 1566–1572.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103.

Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020. Agif: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In *Findings of EMNLP*, pages 1807–1816.

Darsh Shah, Raghav Gupta, Amir Fayazi, and Dilek Hakkani-Tur. 2019. Robust zero-shot cross-domain slot filling with example values. In *ACL*, pages 5484–5490.

Chen Shi, Qi Chen, Lei Sha, Sujian Li, Xu Sun, Houfeng Wang, and Lintao Zhang. 2018. Autodialabel: Labeling dialogue data with unsupervised learning. In *EMNLP*, pages 684–689.

AB Siddique, Fuad Jamour, and Vagelis Hristidis. 2021. Linguistically-enriched and context-awarezero-shot slot filling. In *WWW*, pages 3279–3290.

Robert L Thorndike. 1953. Who belongs in the family. In *Psychometrika*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *JMLR*, 9(11).

Hongru Wang, Zezhong Wang, Gabriel Pui Cheong Fung, and Kam-Fai Wong. 2021a. Mcml: A novel memory-based contrastive meta-learning method for few shot slot tagging. *arXiv preprint arXiv:2108.11635*.

Liwen Wang, Xuefeng Li, Jiachi Liu, Keqing He, Yuanmeng Yan, and Weiran Xu. 2021b. Bridge to target domain by prototypical contrastive learning and label confusion: Re-explore zero-shot learning for slot filling. In *EMNLP*, pages 9474–9480.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*, pages 438–449.

Yanan Wu, Zhiyuan Zeng, Keqing He, Hong Xu, Yuanmeng Yan, Huixing Jiang, and Weiran Xu. 2021. Novel slot detection: A benchmark for discovering unknown slot types in the task-oriented dialogue system. In *ACL-IJCNLP*, pages 3484–3494.

Yuxia Wu, Lizi Liao, Gangyi Zhang, Wenqiang Lei, Guoshuai Zhao, Xueming Qian, and Tat-Seng Chua. 2022. State graph reasoning for multimodal conversational recommendation. *TMM*.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*, pages 478–487.

Chenchen Ye, Lizi Liao, Fuli Feng, Wei Ji, and Tat-Seng Chua. 2022a. Structured and natural responses co-generation for conversational search. In *SIGIR*, pages 155–164.

Chenchen Ye, Lizi Liao, Suyu Liu, and Tat-Seng Chua. 2022b. Reflecting on experiences for response generation. In *ACM MM*, pages 5265–5273.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, pages 1160–1179.

Zengfeng Zeng, Dan Ma, Haiqin Yang, Zhen Gou, and Jianping Shen. 2021. Automatic intent-slot induction for dialogue systems. In *WWW*, pages 2578–2589.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and S Yu Philip. 2019a. Joint slot filling and intent detection via capsule neural networks. In *ACL*, pages 5259–5267.

Hanlei Zhang, Xiaoteng Li, Hua Xu, Panpan Zhang, Kang Zhao, and Kai Gao. 2021a. Textoir: An integrated and visualized platform for text open intent recognition. In *ACL-IJCNLP*, pages 167–174.

Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lyu. 2021b. Discovering new intents with deep aligned clustering. In *AAAI*, pages 14365–14373.

Zheng Zhang, Lizi Liao, Minlie Huang, Xiaoyan Zhu, and Tat-Seng Chua. 2019b. Neural multimodal belief tracker with adaptive attention for dialogue systems. In *WWW*, pages 2401–2412.

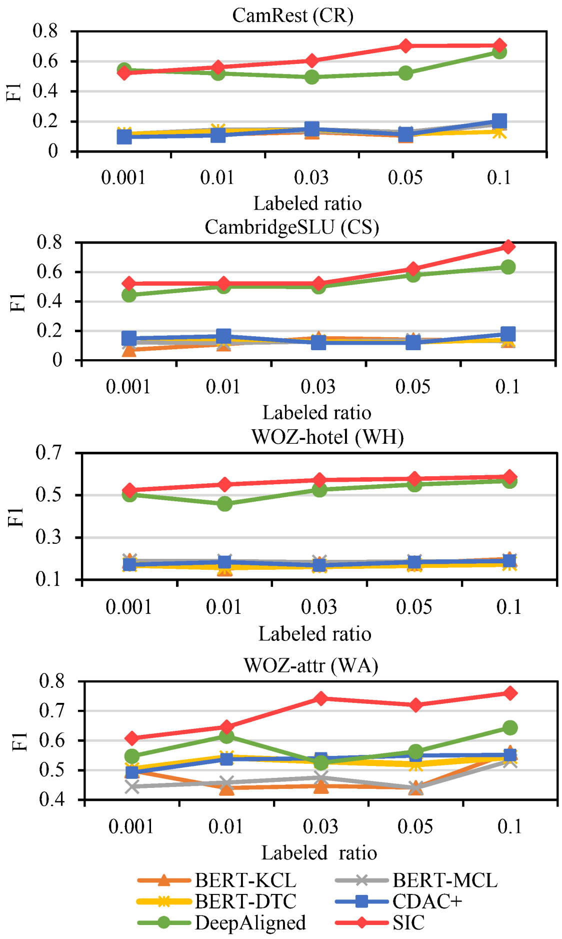


Figure A.1: Effect of labeled data on remaining datasets.

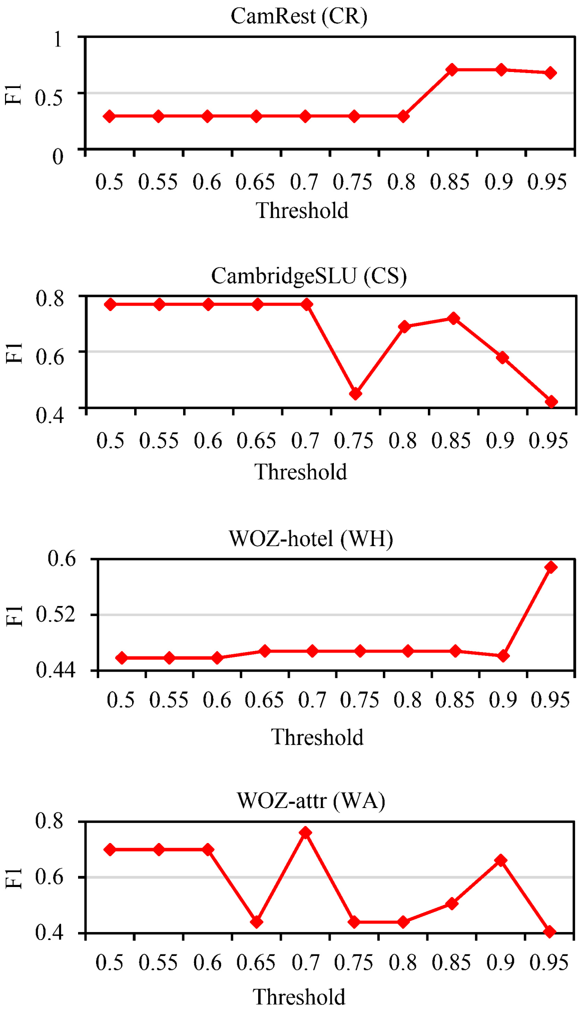


Figure A.2: Effect of the threshold for sample selection on remaining datasets.

A Additional Results

Due to space limitation, we show more results on the remaining datasets here. We first illustrate the effect of different labeled data ratio, then show the effect of sample selection threshold.

A.1 More Results on Labeled Data Ratio

Figure A.1 shows the performance results of different labeled data ratios on the CR, CS, WH and WA datasets for various semi-supervised methods. Generally speaking, we observe that the performance results of *BERT-KCL*, *BERT-MCL*, *BERT-DTC* and *CDAC+* change rather slowly as the ratio of labeled data increases. We suspect that this is because these methods struggle to learn enough evidence when the labeled data is very insufficient. By looking into the cluster level information, the *DeepAligned* method performs better and its performance increases faster than other baselines when the ratio of labeled data increases. However, it is still more sensitive to certain groups of labeled data than our proposed *SIC*, as evidenced by some sudden decrease trend on WH and WA datasets.

A.2 More Results on Selection Threshold

Here we show more performance results of our proposed method *SIC* with different threshold for sample selection in Figure A.2. The results are for the CR, CS, WH and WA datasets. We observe that the best threshold for CR starts from 0.85. The best threshold for CS resides in the range from 0.5 to 0.7. Also, we see that the best threshold for WH is rather large (near 0.95), while the best one for WA points in several value segments. This signals that it is important for the performance our method. This might be decided by the inherent nature of these datasets, such as the sizes of different slots, relations among values and slots *etc*. As these are important for further adaptation for new dialogue applications (Liao et al., 2018; He et al., 2021; Fei et al., 2022b; Ye et al., 2022b; Wu et al., 2022), more investigation can be devoted to this issue in future.