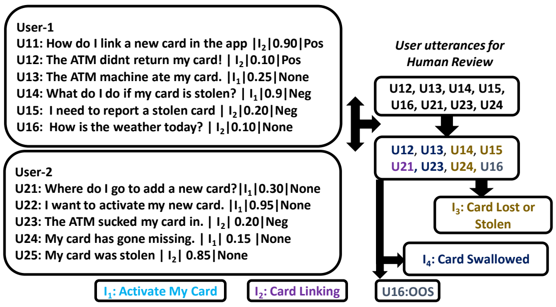
Intent Detection and Discovery from User Logs via Deep Semi-Supervised Contrastive Clustering

Rajat Kumar, Mayur Patidar, Vaibhav Varshney, Lovekesh Vig, Gautam Shroff
TCS Research, New Delhi, India
{k.rajat2, patidar.mayur, varshney.v,
lovekesh.vig, gautam.shroff}@tcs.com

Abstract

Intent Detection is a crucial component of Dialogue Systems wherein the objective is to classify a user utterance into one of the multiple pre-defined intents. A pre-requisite for developing an effective intent identifier is a training dataset labeled with all possible user intents. However, even skilled domain experts are often unable to foresee all possible user intents at design time and for practical applications, novel intents may have to be inferred incrementally on-the-fly from user utterances. Therefore, for any real-world dialogue system, the number of intents increases over time and new intents have to be discovered by analyzing the utterances outside the existing set of intents. In this paper, our objective is to i) detect known intent utterances from a large number of unlabeled utterance samples given a few labeled samples and ii) discover new unknown intents from the remaining unlabeled samples. Existing SOTA approaches address this problem via alternate representation learning and clustering wherein pseudo labels are used for updating the representations and clustering is used for generating the pseudo labels. Unlike existing approaches that rely on epoch-wise cluster alignment, we propose an end-to-end deep contrastive clustering algorithm that jointly updates model parameters and cluster centers via supervised and self-supervised learning and optimally utilizes both labeled and unlabeled data. Our proposed approach outperforms competitive baselines on five public datasets for both settings: (i) where the number of undiscovered intents is known in advance, and (ii) where the number of intents is estimated by an algorithm. We also propose a human-in-the-loop variant of our approach for practical deployment which does not require an estimate of new intents and outperforms the end-to-end approach.



User-1
U11: How do I link a new card in the app |I_2|0.90|Pos
U12: The ATM didnt return my card! | I_2|0.10|Pos
U13: The ATM machine ate my card. |I_1|0.25|None
U14: What do I do if my card is stolen? | I_1|0.9|Neg
U15: I need to report a stolen card | I_2|0.20|Neg
U16: How is the weather today? | I_2|0.10|None

User-2
U21: Where do I go to add a new card?|I_1|0.30|None
U22: I want to activate my new card. |I_1|0.95|None
U23: The ATM sucked my card in. | I_2| 0.20|Neg
U24: My card has gone missing. | I_1| 0.15 |None
U25: My card was stolen | I_2| 0.85|None

User utterances for Human Review
U12, U13, U14, U15, U16, U21, U23, U24
U12, U13, U14, U15 U21, U23, U24, U16
I3: Card Lost or Stolen
I4: Card Swallowed
I1: Activate My Card
I2: Card Linking
U16:OOS

Figure 1: An instance of user logs, where the intent detection model is trained on two known intents, i.e., i_1 and i_2. After manual analysis of user logs a human reviewer has discovered two new intents i_3 and i_4 and assigned utterance $u21$ to an existing intent, i.e., i_2.

1 Introduction

Modern dialogue systems (Louvan and Magnini, 2020) are increasingly reliant on intent detection to classify a user utterance into one of the multiple known user intents. Intent detection is typically modeled as a multi-class classification problem where labeled data comprising of utterances for each known intent is manually created by domain experts. However, most real-world applications have to cope with evolving user needs and new functionality is routinely introduced into the dialogue system resulting in a continuously increasing number of intents over time. Even for seasoned domain experts estimating future user requirements at design time is challenging and these often have to be discovered from recent user logs which contain information corresponding to past user utterances, model response (i.e., predicted intent), implicit (confidence or softmax probability), and explicit (user clicks on a thumbs up or thumbs down icon) feedback as shown in Fig 1. The intent detection model presented in Fig. 1 is trained on two initial intents (I_1, I_2) from the banking domain using labeled data created by domain experts. Filtered user logs containing implicit and explicit feedback were shared with domain experts, who, discovered two new intents (I_3, I_4) and mapped the filtered utterances to these new intents. Additionally, experts also have to identify and discard utterances that are outside the domain of the dialog system.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1836 - 1853
July 10-15, 2022 ©2022 Association for Computational Linguistics

The remaining user logs are the primary source of evolving user needs and the process of identifying utterances belonging to “new intents”/“unknown intents” from user logs is referred to as *Intent Discovery/Intent Mining* (Chatterjee and Sengupta, 2020; Zhang et al., 2021b). However, manually monitoring user logs is not scalable and our objective in this paper is to present novel SOTA techniques to perform automated and semi-automated intent detection and discovery over user logs given only a few labeled utterances from known intents in addition to unlabeled utterances from both known and unknown intents.

Several classical (MacQueen, 1967; Chidananda Gowda and Krishna, 1978; Ester et al., 1996) and deep learning (Xie et al., 2016a; Zhang et al., 2021a) based clustering methods have been used for intent discovery. Chatterjee and Sengupta (2020) modeled intent discovery from unlabeled utterances as an unsupervised clustering problem and proposed a variant of DBSCAN (Ester et al., 1996) for clustering but do not employ any representation learning and rely heavily on manual evaluation. Zhang et al. (2021a) use a contrastive learning (Chen et al., 2020) based unsupervised approach for joint representation learning and clustering where performance largely depends on the quality of an auxiliary target distribution (Xie et al., 2016a). Lin et al. (2020) and Zhang et al. (2021c) model intent detection and discovery as a semi-supervised learning problem where the objective is to detect known intents and discover new intents given 1) a few labeled utterances from known intents along with 2) unlabeled utterances from known and new intents. This is similar to our problem of intent detection and discovery from user logs. Deep-Aligned (Zhang et al., 2021c) is the current SOTA approach for intent detection and discovery alternately performing representation learning and clustering by utilizing pseudo-labeled data obtained from clustering for representation learning. Deep-Aligned uses k-means (MacQueen, 1967) as the clustering algorithm of choice and updates a BERT (Devlin et al., 2019) backbone's parameters in the process. As k-means may assign different cluster ids to the same set of data points over different iterations the authors propose an alignment algorithm to align clusters obtained in consecutive epochs. Thus, an incorrect cluster alignment over epochs may lead to a significant drop in clustering accuracy. Additionally, they make the unrealistic assumption of a uniform distribution over intents to estimate the number of intents.

In this paper, we propose a novel end-to-end **D**eep **S**emi-**S**upervised **C**ontrastive **C**lustering (DSSCC-E2E) algorithm for intent detection and discovery from user logs. *DSSCC-E2E* is motivated by recent advances in self-supervised (Chen et al., 2020; Wu et al., 2020; Li et al., 2021) and supervised contrastive learning (Khosla et al., 2020; Gunel et al., 2021) applied to Computer vision and natural language processing. We model intent detection and discovery as a form of semi-supervised contrastive clustering wherein we jointly update backbone representations and cluster centers by minimizing the distance between the distribution over clusters of similar utterances and maximizing the same for dissimilar utterances via a semi-supervised variant of supervised contrastive (SupCon) loss(Khosla et al., 2020). For contrastive learning, we use the contextual augmenter (Ma, 2019) to create pairs of augmentations (positive pairs/ similar utterances) corresponding to each unlabeled utterance from known and unknown intents. For labeled utterances, we use pairs of utterances with the same intent to create positive pairs. To improve the accuracy of intent detection, we update representations based on labeled utterances by minimizing the cross-entropy loss. To avoid the trivial solution that assigns all utterances to the same cluster (Hu et al., 2017), similar to Van Gansbeke et al. (2020); Li et al. (2021) we also add an entropy term to the loss function which distributes utterances uniformly across the clusters.

Existing semi-supervised approaches(Lin et al., 2020; Zhang et al., 2021c) including DSSCC-E2E and some unsupervised approaches (MacQueen, 1967; Zhang et al., 2021a) for intent discovery require an estimate of the number of new intents (m) present in the user logs. Incorrect estimates for m can lead to noisy clusters (i.e., a cluster which contains utterances from multiple intents), which then require substantial manual effort to split cleanly. Unsupervised approaches (Ester et al., 1996; Chatterjee and Sengupta, 2020) often lead to a large number of clusters due to poor semantic utterance representations. For practical deployment, we propose a human-in-loop variant of *DSSCC-E2E* called DSSCC-HIL which does not estimate m and instead creates multiple dense clusters via DBSCAN. Domain experts can then merge these clusters with minimal manual effort such that each merged cluster represents an intent.

Our key contributions are: (1) We propose a novel deep semi-supervised contrastive clustering-based approach for intent detection and discovery from user logs. (2) *DSSCC-E2E* does not require epoch-wise cluster alignment, is end-to-end trainable, and fully utilizes both the labeled and unlabeled utterances for novel intent discovery. (3) *DSSCC-E2E* outperforms competitive baselines on five public datasets in the following settings: (3.1) Number of intents are known in advance. (3.2) Number of intents is estimated by an algorithm. (4) For realistic deployment, we propose a human in the loop intent detector *DSSCC-HIL*, which does not need to estimate the number of new intents. (5) With minimal manual effort, *DSSCC-HIL* outperforms existing approaches on the intent discovery task.

2 Related Work

2.1 Self-supervised and Supervised Representation Learning

Different self-supervised pre-training tasks have been proposed to pre-train PLMs, such as Masked language modeling (MLM) (Devlin et al., 2019), and MAsked Sequence to Sequence pre-training (MASS)(Song et al., 2019). Reimers and Gurevych (2019); Gao et al. (2021) fine-tune BERT on a supervised contrastive learning objective to learn better sentence embeddings. Zhang et al. (2021a) use SBERT (Reimers and Gurevych, 2019) as a PLM to learn clustering friendly representations in an unsupervised scenario by using instance-level contrastive representation learning (Chen et al., 2020) and clustering in a joint-fashion. We propose a deep semi-supervised contrastive learning approach for intent detection and discovery where we jointly update PLM representations and cluster centers. We leverage the idea of unsupervised contrastive clustering (Li et al., 2021) for images to semi-supervised contrastive clustering of text and use a modified version of the supervised contrastive loss function (Khosla et al., 2020) to jointly update model parameters and cluster centers.

We also compare the proposed approach with existing unsupervised and semi-supervised approaches for clustering in Appendix A.1.

3 Problem Description

Consider a set of n known intents with a few labeled utterances per intent $I_{known} = \{I_1, I_2, ..., I_n\}$ where $I_i = \{\bigcup_{k=1}^{r} (u_{ik}, y_i)\}$, $\mathcal{D}_l = \bigcup_{i=1}^{n} I_i$ and user logs $\mathcal{D}_{user\_logs} = \{\bigcup_{v=1}^{S} (u_v)\}$ which contain unlabeled user utterances from both known and unknown intents. The objective is to detect and discover existing and new/unknown intents from a test set, $\mathcal{D}_{test}$ given a training set $\mathcal{D}_{train} = \mathcal{D}_l \cup \mathcal{D}_{user\_logs}$ (and $\mathcal{D}_{val}$) which contains utterances from both known and unknown intents. $\mathcal{D}_{test} = \{I'_{known} \cup I_{new}\}$ and $I_{new} = \{I_{n+1}, I_{n+2}, ..., I_m\}$ where m represents the number of new intents, $I = I_{known} \cup I_{new}$ and $|I| = n+m$ represents the total number of known and unknown intents. I'_{known} is similar to I_{known} except it only contains the new set of utterances for known intents from user logs. In a realistic scenario the number of "new intents" m present in user logs are not known apriori. We perform experiments in two scenarios (i) The number of known and new intents is known in advance and (ii) The number of new intents has to be inferred from the user logs.

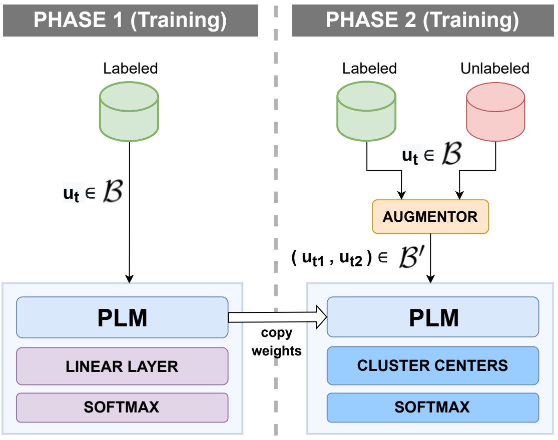
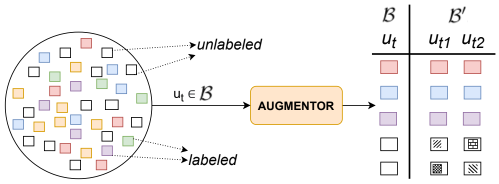


Figure 2: Proposed Approach - Deep Semi-Supervised Contrastive Clustering (DSSCC)

4 Proposed Approach

As shown in Fig. 2, we propose a two-phase algorithm for intent detection and discovery from user logs. In the first phase of *DSSCC*, the parameters of a pre-trained language model (PLM) are updated based on labeled data from known intents. In the second phase, we perform joint representation learning and clustering by updating the cluster centers and PLM parameters via semi-supervised contrastive learning. In addition to contrastive learning, *DSSCC* also uses labeled utterances to update PLM representations via cross-entropy loss. Further, the entropy of the intent distribution is maximized to



For labeled utterance, u_t: *u_{t1} and u_{t2} are utterances from the same intent.*
For unlabeled utterance, u_t: *u_{t1} and u_{t2} are generated by contextual augmentor.*

Figure 3: Utterance augmentations for semi-supervised contrastive learning.

.

alleviate the issue of empty clusters, i.e., when all utterances become part of a single cluster.

In a realistic scenario, the number of unknown intents m present in user logs is not known in advance so we use the approach proposed by Zhang et al. (2021c) to estimate m and use *DSSCC-E2E* for intent detection and discovery. The approach uses a PLM to get utterance representations and run K-means with a very high value of $k = K'$ and only count clusters with cluster cardinality greater than $|\mathcal{D}_{train}|/K'$ to estimate k. However in practice, we find that even this estimation method is often incorrect, and in section 4.5, we describe how we derive *DSSCC-HIL* from *DSSCC-E2E* to address this problem with a small amount of manual supervision. We use the Hungarian (Kuhn and Yaw, 1955) algorithm to align clusters with known and unknown intents and report performance. In the rest of the paper, we use DSSCC and DSSCC-E2E interchangeably and use DSSCC-HIL to refer to the human-in-the-loop variant of our approach.

4.1 Phase-1: Fine-tuning of PLM using labeled utterances from known intents

To leverage the labeled utterances from known intents for intent detection and discovery, in the first phase of *DSSCC*, we use these to update the parameters of the PLM, as shown in Phase-1 of Fig. 2. We fine-tune the PLM by minimizing the cross-entropy loss over a batch $\mathcal{B}$ of size N consisting of labeled utterances from known intents, as shown in Eq. 1 and 2.

$$p(I_{known} \mid u_t) = \text{softmax}(\boldsymbol{h_t} * \boldsymbol{W} + b) \quad (1)$$

$$\mathcal{L}_{CE} = -\frac{1}{N} \sum_{t \in \mathcal{B}} \sum_{i=1}^{n} y \cdot log(p(I^i_{known} \mid u_t)) \quad (2)$$

In Eq. 1, $\boldsymbol{h_t}$ denotes a d-dimensional representation of the t^{th} utterance (u_t) in a batch $\mathcal{B}$ obtained from the PLM and $\boldsymbol{W} \in \mathbb{R}^{d*m}$, b represent the weights and bias of a linear layer respectively. In Eq. 2, $p(I^i_{known} \mid u_t)$ denotes the probability of assigning u_t to the i^{th} known intent and y is 1 only for the true intent and zero otherwise. After fine-tuning the PLM on labeled utterances, we discard the linear layer and use the PLM with updated weights in the second phase of *DSSCC*.

4.2 Phase-2: Deep Clustering

In the second phase, we use both labeled and unlabeled utterances to perform representation learning and clustering jointly via semi-supervised contrastive learning. To maintain intent detection accuracy on known intents, we also update representations and cluster centers by minimizing cross-entropy loss on labeled utterances from known intent.

4.2.1 Semi-supervised Representation Learning and Clustering

In addition to labeled utterances from known intents, we also use unlabeled utterances from both known and unknown intents to improve performance on intent detection and discovery, as shown in Phase-2 of Fig. 2. To learn better cluster representations, instead of minimizing the distance between utterances belonging to the same intent, we minimize the distance between their corresponding cluster distributions. Conversely, we maximize the distance between cluster distributions for clusters corresponding to different intents. In contrast to self-supervised (Chen et al., 2020) or supervised contrastive (Khosla et al., 2020) learning, for semi-supervised learning a batch $\mathcal{B}$ of size N may contain both labeled and unlabeled utterances. As shown in Fig. 3, similar to self-supervised contrastive learning, we create a pair of augmentations (u_{t1}, u_{t2}) or positive pairs corresponding to the t^{th} or anchor utterance (u_t) in $\mathcal{B}$ to obtain $\mathcal{B}'$, which contains two augmented utterances corresponding to each utterance in $\mathcal{B}$. To generate augmentations for a labeled utterance, we randomly sample two utterances from the same intent and use them as augmentations whereas for an unlabeled utterance, we generate two augmented pairs by performing contextual augmentation [1] (Kobayashi, 2018; Ma, 2019), as shown in Fig. 3. In contextual augmentation, given an utterance, we randomly mask a few words and use BERT's masked-language modeling (MLM) objective to generate words corresponding to masked positions. If u_{t1}, u_{t2} are augmentations of a labeled utterance u_t then $P(u_{t1})$ is defined as

[1] https://github.com/makcedward/nlpaug

the set of utterances belonging to the same intent as u_{t1} in $\mathcal{B}'$ whereas $N(u_{t1})$ will contain the all $2N$-1 utterances excluding u_{t1} (note that $N(u_t)$ and $P(u_t)$) may have utterances in common). If u_{t1}, u_{t2} are augmentations of an unlabeled utterance u_t then $P(u_{t1})$ will only contain u_{t2} and $N(u_{t1})$ will contain all $2N$-1 utterances excluding u_{t1}. We update PLM parameters and cluster centers by minimizing the Semi-Supervised Contrastive (SSC) loss as shown in Eq. 3.

$$\mathcal{L}_{ssc} = \sum_{t\in\mathcal{B}'} \frac{-1}{|P(u_t)|} \sum_{p'\in P(u_t)} \log \frac{\exp\left(p(I \mid u_t) \cdot p(I \mid u_{p'})/\tau\right)}{\sum\limits_{a\in N(t)} \exp\left(p(I \mid u_t) \cdot p(I \mid u_a)/\tau\right)}; \tag{3}$$

In Eq. 3, $(\cdot)$ symbol and τ denotes dot product and scalar temperature parameter respectively. To get the distribution over intents/clusters ($I/\boldsymbol{C}$), i.e., $p(I \mid u_t)$ for the t^{th} utterance in $\mathcal{B}'$ we apply a linear transformation over $\boldsymbol{h_t}$ and normalize via softmax, $p(I \mid u_t) = \text{softmax}(\boldsymbol{h_t} * \boldsymbol{C} + b)$. Each column $\boldsymbol{C}_i$ of a linear layer $\boldsymbol{C} \in \mathbb{R}^{d\times(n+m)}$ acts as a cluster center, where $\boldsymbol{C_i}$ is the d-dimensional representation of the i^{th} cluster and $(m+n)$ represents the total number of known and unknown intents.

To avoid the trivial solution that assigns all utterances to the same cluster Hu et al. (2017), similar to Van Gansbeke et al. (2020); Li et al. (2021) we also add an entropy term to the loss function and maximize it which distributes utterances uniformly across the clusters, as shown in Eq. 4 and Eq. 6.

$$\mathcal{L}_{em} = -\sum_{i\in\mathcal{T}} p(I^{'i}) * \log p(I^{'i}) \tag{4}$$

$p(I^{'i}) = \frac{1}{|\mathcal{B}|}\sum_{t\in\mathcal{B}} p(I^i \mid u_t)$, where $p(I^i \mid u_t)$ denotes the probability of an utterance u_t being assigned to the i^{th} intent.

4.2.2 Supervised Representation Learning

To maintain intent detection accuracy on known intents, we also update the cluster centers and PLM parameters by minimizing cross-entropy loss over labeled utterances from known intents, as shown in Eq. 5 where y is 1 only for the target intent and zero otherwise. Unlike in phase-1, $\mathcal{B}$ can contain labeled and unlabeled utterances from known intents and unlabeled utterances from unknown intents but we ignore the unlabeled utterances during backpropagation.

$$\mathcal{L}_{srl} = -\frac{1}{N}\sum_{t\in\mathcal{B}}\sum_{i=1}^{n} y \cdot log(p(I^i \mid u_t)) \tag{5}$$

4.3 Training

We train *DSSCC* in two phases, in the first phase we fine-tune the PLM using labeled utterances from known intents as mentioned in Eq. 2. In the second phase, we do representation learning and clustering jointly by minimizing a combination of SSC loss, and cross-entropy loss. Further, to avoid the trivial solution of all the utterances getting assigned to a single cluster, the entropy over the intent distribution is maximized, as shown in Eq. 6. λ is a scalar hyper-parameter which controls the contribution of $\mathcal{L}_{em}$ in $\mathcal{L}$.

$$\mathcal{L} = \mathcal{L}_{ssc} + \mathcal{L}_{srl} - \lambda * \mathcal{L}_{em} \tag{6}$$

4.4 Inference

We propose two ways of utilizing representations learned by *DSSCC* for intent detection and discovery.

Clustering via learned cluster centers (*DSSCC-CH*) Cluster assignment is based on similarity between cluster and utterance representations, i.e., $\underset{1\le i\le n+m}{\text{argmax}}\ p(I^i \mid u_t)$.

K-means over representations (*DSSCC-KM*) We get the representations ($\boldsymbol{h_t}$) for each utterance in $\mathcal{D}_{test}$ from the PLM and use K-means for clustering.

4.5 Intent discovery with Human-in-the-loop (*DSSCC-HIL*)

Existing approaches including *DSSCC* assume knowledge of the number of unknown intents (m) to achieve good performance. This is not a realistic assumption and despite SOTA performance, significant manual effort is still needed to denoise the discovered clusters. However, the manual effort can be drastically reduced if we can generate dense clusters with high purity and assign a natural language description to each cluster. These descriptions can be used by a domain expert to merge similar clusters (i.e., a set of clusters with similar descriptions) or split a noisy cluster into multiple sub-clusters. Merging clusters requires much less manual effort than splitting as merging does not require examining every utterance in the cluster. Thus, if we can obtain

a set of pure clusters (i.e., a cluster where the majority of the utterances belong to a single intent) then the domain expert only needs to examine a few representative utterances per cluster before merging similar clusters.
To obtain better representations for clustering, we use phase-1 of *DSSCC* without any modification. To obtain pure clusters, unlike phase-2 of *DSSCC*, we perform representation learning and clustering in an alternate fashion and use DBSCAN for generating a large number of pure clusters. We assume that the utterances which are part of the same cluster belong to the same intent and use this fact to create a pair of augmentations for a given utterance along with contextual augmentation. Due to the unknown value of m, we perform semi-supervised contrastive representation learning at the utterance level, as shown in Eq. 7. The model parameters are updated according to $\mathcal{L} = \mathcal{L}^{'}_{ssc} + \mathcal{L}_{srl}$, where $\mathcal{L}_{srl}$ and $\mathcal{L}^{'}_{ssc}$ is defined in Eq. 5 and 7 respectively.

Cluster Merger Algorithm After phase-2, we run DBSCAN and obtain a set of clusters including the outlier cluster. For each cluster (except the outlier) we randomly sample p utterances and use them as cluster descriptions. The cluster representation is obtained as the mean of these utterance representations. So, now each cluster has its own description and representation. Now we randomly pick one cluster as the query cluster (q) and get its s nearest neighbors based on cosine-similarity and ask "Which of these s clusters should be merged with q"? to a domain expert. Based on the domain expert's response we merge similar clusters, recalculate the cluster representations, and assign a cluster description of q to this newly created cluster. We repeat this process till the domain expert finds no candidate for thirty consecutive query clusters. One iteration of the cluster merging algorithm is illustrated in Fig 4.

Now we treat each cluster as an intent and train a logistic classifier to label utterances that belong to the outlier cluster. We use the same classifier for intent detection on the test set.

$$\mathcal{L}^{'}_{ssc} = \sum_{t \in \mathcal{B}'} \frac{-1}{|P(u_t)|} \sum_{p' \in P(u_t)} \log \frac{\exp\left(u_t \cdot u_{p'})/\tau\right)}{\sum\limits_{a \in N(t)} \exp\left(u_t \cdot u_a)/\tau\right)}; \tag{7}$$

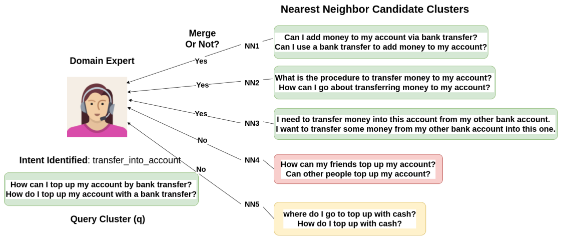

Nearest Neighbor Candidate Clusters
Merge Or Not?
Domain Expert
Yes
Yes
Yes
No
No
NN1
NN2
NN3
NN4
NN5
Can I add money to my account via bank transfer?
Can I use a bank transfer to add money to my account?
What is the procedure to transfer money to my account?
How can I go about transferring money to my account?
I need to transfer money into this account from my other bank account.
I want to transfer some money from my other bank account into this one.
How can my friends top up my account?
Can other people top up my account?
where do I go to top up with cash?
How do I top up with cash?
Intent Identified: transfer_into_account
How can I top up my account by bank transfer?
How do I top up my account with a bank transfer?
Query Cluster (q)

Figure 4: **Cluster Merger Algorithm:** In each iteration, a Domain Expert is shown (s = 5) Candidate Clusters which are nearest neighbors of Query Cluster (q), where per cluster, only (p = 2) utterances are shown to the expert. And based on the reply from the domain expert, some candidate clusters are merged into the query cluster and the cluster definition is updated.

5 Experimental Setup

In this section, we describe the various baselines, datasets, and evaluation metrics used in our experiments.

5.1 Baseline Approaches

We use *unsupervised K-means* (MacQueen, 1967), *Agglomerative Clustering (AG)* (Chidananda Gowda and Krishna, 1978), *DEC* (Xie et al., 2016b), *SAE-KM* (Xie et al., 2016b), *DCN* (Yang et al., 2017), *DAC* (Chang et al., 2017), *DeepCluster* (Caron et al., 2018), *SCCL* (Zhang et al., 2021a) and *semi-supervised PCK-means* (Basu et al., 2004), *BERT-KCL* (Hsu et al., 2018), *BERT-MCL* (Hsu et al., 2019), *BERT-DTC* (Han et al., 2019), *CDAC+* (Lin et al., 2020), *DeepAligned* (Zhang et al., 2021c) clustering approaches as baselines. *SCCL* and *DeepAligned* are the state-of-the-art approaches for unsupervised and semi-supervised clustering respectively. Details about unsupervised and semi-supervised baselines are included in Appendix A.1.

5.2 Dataset Description

We evalaute *DSSCC* on five datasets with a varying number of intents. We use *BANKING77* (Casanueva et al., 2020), *CLINC150 (and CLINC150$_{OOS}$)*, (Larson et al., 2019) *SNIPS* (Coucke et al., 2018), *StackOverflow* Xu et al. (2015) and, *DBPedia* (Zhang and LeCun, 2015). For *BANKING77* and *CLINC150* we use the same train, val and test split as Zhang et al. (2021c) and for *SNIPS*, *StackOverflow* and *DBPedia* we follow the same split as Lin et al. (2020). For more details, please refer to Appendix A.3

5.3 Evaluation Metrics

Similar to previously reported results (Lin et al., 2020; Zhang et al., 2021c), we use *Clustering Accuracy* (**ACC**) (Yang et al., 2010), *Normalized Mutual Information* (**NMI**) (Strehl and Ghosh, 2002) and *Adjusted Rand Index* (**ARI**) (Hubert and Arabie, 1985) as evaluation metrics. All metrics range from 0 to 100 and higher values of a metric indicate superior clustering results. Further details of the experimental setup are provided in Appendix A.4. Due to space constraints, we report ACC, NMI only but in Appendix, we use all three metrics to report results.

6 Results And Discussion

We present results comparing *DSSCC* with unsupervised and semi-supervised clustering approaches for scenarios where 1) the system is aware of the number of unknown intents, 2) the system is unaware of the number of unknown intents. We further report results of *DSSCC-HIL* and compare it with *DSSCC-E2E.*

6.1 Intent detection and discovery when the number of new intents m is provided

For the semi-supervised scenario, we assume $x\%$ of the total intents are known apriori, also referred to as the Known Intent Ratio (KIR). For each known intent, 10% of the utterances are labeled and the rest are unlabeled. For a fair comparison, we also use the same BERT-based PLM as other baselines. *DSSCC* outperforms all unsupervised baselines on both BANKING77 and CLINC150 datasets by significant margins suggesting that for known intents, supervision in the form of a few labeled utterances leads to better intent representations, as shown in Table 1. *DSSCC* also outperforms semi-supervised baselines on BANKING77 and CLINC150 for all cases except for the NMI metric on the CLINC150 dataset for KIRs 50% and 75%.

For K-means$_{SBERT}$, we use SBERT Reimers and Gurevych (2019) as our PLM instead of BERT to obtain utterance representations and find that it outperforms K-means$_{BERT}$ by a significant margin, as shown in Table 1. This suggests that utterance representations from SBERT are more suitable for clustering than BERT-based representations. Motivated by these results we compared BERT and SBERT representations in Table 2 and found that *DSSCC* (Ours)$_{SBERT}$ outperforms DSSCC (Ours)$_{BERT}$, DeepAligned$_{BERT}$ and DeepAligned$_{SBERT}$ by a significant margin on both datasets for all evaluated intent ratios. DeepAligned$_{SBERT}$ outperforms DeepAligned$_{BERT}$ on BANKING77 by a significant margin but we observe the opposite result on CLINC150. For the remaining experiments, we use SBERT as our PLM in *DSSCC* and compare it with DeepAligned$_{BERT}$ and DeepAligned$_{SBERT}$.

		CLINC150		BANKING77	
KIR	Approach	ACC	NMI	ACC	NMI
0%	**K-means$_{BERT}$**	45.06	70.89	29.55	54.57
	K-means$_{SBERT}$	61.04	82.22	55.72	74.68
	AG	44.03	73.07	31.58	57.07
	SAE-KM	46.75	73.13	38.92	63.79
	DEC	46.89	74.83	41.29	67.78
	DCN	49.29	75.66	41.99	67.54
	DAC	55.94	78.40	27.41	47.35
	DeepCluster	35.70	65.58	20.69	41.77
	SCCL	33.52	66.63	13.41	34.14
25%	**PCK-means**	54.51	68.71	32.66	48.22
	BERT-KCL	24.72	65.74	22.11	52.42
	BERT-MCL	24.35	65.06	22.07	51.96
	BERT-DTC	49.1	74.17	25.24	48.58
	CDAC+	64.64	84.25	48.71	69.78
	DeepAligned	73.71	88.71	48.88	70.45
	DSSCC$_{BERT}$	**75.72**	**89.12**	**55.52**	**72.73**
50%	**PCK-means**	54.51	68.62	32.26	48.11
	BERT-KCL	46.91	78.45	40.97	65.22
	BERT-MCL	47.21	78.39	41.43	65.68
	BERT-DTC	71.68	86.20	53.59	71.40
	CDAC+	69.02	86.18	53.34	71.53
	DeepAligned	80.22	**91.63**	59.23	76.52
	DSSCC$_{BERT}$	**81.46**	91.39	**63.08**	**77.60**
75%	**PCK-means**	54.61	68.70	32.66	48.22
	BERT-KCL	68.86	86.82	60.15	75.21
	BERT-MCL	69.66	87.72	61.14	75.68
	BERT-DTC	80.73	90.41	56.51	76.55
	CDAC+	69.89	86.65	53.83	72.25
	DeepAligned	86.01	**94.03**	64.90	79.56
	DSSCC$_{BERT}$	**87.91**	93.87	**69.82**	**81.24**

Table 1: We report ACC and NMI on CLINC150 and BANKING77 datatsets in the semi-supervised scenario for three different known intent ratios (KIR). Except **K-means$_{SBERT}$** and **SCCL**, we take all baseline results from Zhang et al. (2021c).

		CLINC150		BANKING77	
KIR	Approch	ACC	NMI	ACC	NMI
25%	**DA$_{BERT}$**	73.71	88.71	48.88	70.45
	DA$_{SBERT}$	67.78	86.50	57.0	75.0
	DSSCC$_{BERT}$	75.72	89.12	55.52	72.73
	DSSCC$_{SBERT}$	**80.36**	**91.43**	**64.93**	**80.17**
50%	**DA$_{BERT}$**	80.22	91.63	59.23	76.52
	DA$_{SBERT}$	77.69	91.40	64.14	79.30
	DSSCC$_{BERT}$	81.46	91.39	63.08	77.60
	DSSCC$_{SBERT}$	**83.49**	**92.78**	**69.38**	**82.68**
75%	**DA$_{BERT}$**	86.01	94.03	64.90	79.56
	DA$_{SBERT}$	85.89	94.20	74.08	83.80
	DSSCC$_{BERT}$	87.91	93.87	69.82	81.24
	DSSCC$_{SBERT}$	**88.47**	**94.50**	**75.15**	**85.04**

Table 2: *DA* vs *DSSCC* with *BERT* and *SBERT* as PLM

We evaluated *DSSCC* on three other public datasets for intent detection and discovery, and found that *DSSCC* outperforms DeepAligned on all three datasets except for NMI on DBPedia for known intent ratios of 75% , as shown in Table 10. Results of *DSSCC$_{SBERT}$* on intent detection and

	CLINC150 ($\mathcal{T}$=150, n=112, $m = 38$)				**BANKING77** ($\mathcal{T}$=77, n=58, m=19)			
Approach	K'	K_{Pred}	**ACC**	**NMI**	K'	K_{Pred}	**ACC**	**NMI**
DA$_{BERT}$	300	130	77.18	92.5	154	65.1	62.49	78.88
DA$_{SBERT}$	300	129.6	76.87	92.61	154	66.9	63.53	80.84
DSSCC$_{SBERT}$			**81.37**	**92.97**			**71.77**	**84.29**
DA$_{BERT}$	450	189.2	83.81	93.57	231	99.3	63.98	79.93
DA$_{SBERT}$	450	190.1	82.57	**93.85**	231	96.8	66.20	82.11
DSSCC$_{SBERT}$			**84.59**	93.64			**72.93**	**84.97**
DA$_{BERT}$	600	258.6	72.22	91.8	308	121.9	61.05	79.95
DA$_{SBERT}$	600	255.9	72.29	**92.18**	308	118.1	62.67	82.05
DSSCC$_{SBERT}$			**80.83**	92.04			**67.56**	**84.43**
DSSCC$_{SBERT}$	150	150	**88.47**	**94.50**	77	77	**75.15**	**85.04**

Table 3: Intent detection and discovery with unknown value of m for KIR=75%. We obtain results corresponding to DeepAligned$_{BERT}$ (DA$_{BERT}$) (Zhang et al., 2021c) and report results for DeepAligned$_{SBERT}$ (DA$_{SBERT}$) using code provided by the authors of Zhang et al. (2021c) with SBERT as the PLM.

discovery are separately reported in Table 11.

6.2 Intent detection and discovery with unknown number of new intents m

We evaluate *DSSCC* for the realistic scenario when the number of new (m) intents present in user logs is not provided to the system apriori, i.e., the total number of intents ($\mathcal{T} = n + m$) is not known in advance and the system has to infer the number of clusters present in user logs. We use an existing algorithm proposed by Zhang et al. (2021c) which refines an initial guess K' to arrive at the final estimate K_{pred}. As shown in the Table 3, *DSSCC* outperforms DeepAligned for $K' \in \{2 * \mathcal{T}, 3 * \mathcal{T}, 4 * \mathcal{T}\}$ except for the NMI metric on the CLINC150 dataset for K'=450 and K'=600 although results are competitive. As compared to the scenario where the total number of intents $\mathcal{T}$ are known (last row in Table 3), on an average there is drop of 6.20% in ACC, 1.61% in NMI on CLINC150 and a drop of 4.39% in ACC, 0.47% in NMI on BANKING77. These results suggest that the performance of *DSSCC* does not drop significantly even when the total number of intents in user logs are not known in advance.

6.3 DSSCC-HIL

We evaluate *DSSCC-HIL* in a realistic scenario where the number of new intents is not known and KIR=75%. As shown in Table 4, we get 333 and 523.7 clusters after phase-2 of *DSSCC-HIL* with average cluster purity of 96.96% and 98.30% corresponding to B77 and C150 respectively. Average purity refers to average clustering accuracy where we use ground-truth labels and based on majority voting, assign an intent label to the predicted cluster. For merging similar clusters, we show (s=5) candidate clusters per query to the domain expert who is asked to choose clusters that are similar to the query cluster. Here, we have used oracle ground truth cluster labels instead of a domain expert to answer these queries. For B77 and C150, 259.3 and 349.5 queries are required (avg over 10 runs) to merge similar clusters respectively where the domain expert has to read 12 utterances (2 per cluster) per query. As a result, we obtain 81 and 152.59 clusters (intents) for B77 and C150 which are close to the actual number of intents i.e., 77 and 150 respectively. Then a classifier is trained with these intents and prediction is done on the test set. *DSSCC-HIL* achieved an ACC of 81.21% and 88.93% on B77 and C150 respectively which is significantly higher than the ACC of *DSSCC-E2E*. Also, DSSCC-HIL is able to discover all intents in the ground truth. We also employ a cluster merging strategy with *DSSCC-E2E* i.e., *DSSCC-E2E+HIL* and got an improvement of 2% for C150 but negative results for B77. This is due to better initial cluster purity (P) of C150 (i.e. 87.73%) versus B77 (i.e. 79.92%), as the merging of noisy clusters intuitively leads to a decrease in ACC. This observation supports the fact that, for merging clusters by a domain expert, a good initial cluster purity is required. Due to comparatively low purity initial clusters, HIL does not help much in *DSSCC-E2E*.

A	**Q**	**P**	K'	K_{pred}	**ACC**	**NMI**
E2E	0	NA	231	96.8	72.93	84.97
E2E+HIL	144.8	79.92	96.8	67	71.14	84.00
HIL	259.3	96.96	333	81	**81.21**	**87.35**
E2E	0	NA	450	190.1	84.59	93.64
E2E+HIL	218.2	87.73	190.1	148.6	86.10	93.83
HIL	349.5	98.30	523.7	152.6	**88.93**	**95.19**

Table 4: We compare *DSSCC-E2E*, *DSSCC-E2E+HIL* and *DSSCC-HIL*. P, Q refers to average cluster purity before merging and number of queries respectivley. For *DSSCC-E2E*, we pick best results from Table 3. First row and second row of the table contains results on B77 and C150 respectively.

6.4 Ablation Study

As part of the ablation study, we answer the following questions "Do we need both phase-1 and phase-2 in DSSCC ?”, “Does joint training on multiple loss functions in phase-2 affect clustering accuracy ?”, “Do we need different values of Entropy Weight (λ) (Eq. 6) for different datasets?”, “How does the presence of out-of-scope (oos) utterances in user logs affect DSSCC performance”? and present our observations. To answer these questions we perform ablation studies on CLINC150 and BANKING77.

“**Do we need both phase-1 and phase-2 in DSSCC?"** As shown in Table 5, KIR=75%, finetuning of the PLM on labeled utterances from known intents in phase-1 is more vital for BANKING77 as compared to CLINC150 because without phase-1 there is a significant drop in performance on the BANKING77 dataset (*DSSCC w/o phase-1* vs *DSSCC*). Whereas there is an improvement of 4.39% in ACC, 1.15% in NMI and 4.86% due to phase-2 on CLINC150 (*DSSCC w/o phase-1* vs *DSSCC*), which suggests that both phase-1 and phase-2 of *DSSCC* are important for a generic solution. In the case of *BANKING77*, performance largely depends on labeled utterances from known intents and there is not much improvement due to phase-2, which may be attributed to its complexity as all intents are semantically closer and belong to the Banking domain as compared to *CLINC150* where utterances belong to 10 different domains (utility, travel, etc.). But when we perform the same ablation with fewer known intents i.e., KIR=25%, then both phase-1 and phase-2 are equally important to achieve good performance. This suggests that DSSCC phase-2 is more important when there are fewer known intents and fewer labeled utterances per known intent.

“Does joint training on multiple loss functions in phase-2 affect clustering accuracy"? As shown in Table 5, we also perform an ablation on different loss functions used in phase-2 and found that Semi-supervised contrastive (*ssc*) loss (*DSSCC w/o ssc* **vs** *DSSCC*), entropy maximization (*em*) (*DSSCC w/o em* **vs** *DSSCC*) and supervised-representation learning (*srl*) loss (*DSSCC w/o srl* **vs** *DSSCC*) all affect the clustering performance on CLINC150 and BANKING77. The reason for the most drop in ACC and NMI in the case of (*DSSCC w/o em*) is that, when entropy maximization is not done, the probability distribution over clusters for unlabeled utterances is decided only by the supervised-representation learning (*srl*) loss. Therefore, all the utterances become part of known intent clusters and the clusters for unknown intents remain empty.

“Do we need different values of Entropy Weight (λ) (Eq. 6) for different datasets?” In Table 7, we report *DSSCC-E2E* results on *CLINC150* and *BANKING77* with different values of λ. And we observe that a value of $\lambda \in [10, 14]$ yields the best results across datasets.

“How does the presence of out-of-scope (oos) utterances in user logs affect *DSSCC* performance”? We use oos utterances as part of user logs in CLINC150 dataset and evaluate DSSCC. As shown in Table 6, there is a small drop in performance because a few oos utterances are classified to belong to the set of actual intents. This observation shows that even with presence of oos utterances, *DSSCC* is able to maintain it’s performance. Manual intervention may be required in practice to filter clusters containing oos queries.

		CLINC150		BANKING77	
KIR	**Approach**	**ACC**	**NMI**	**ACC**	**NMI**
75%	**DSSCC w/o srl**	85.99	93.68	70.24	81.78
	DSSCC w/o ssc	73.87	87.07	67.97	78.81
	DSSCC w/o em	71.05	91.88	64.55	81.56
	DSSCC w/o phase-1	88.13	94.04	72.39	81.76
	DSSCC w/o phase-2	84.08	93.35	74.09	*84.56*
	DSSCC	**88.47**	**94.50**	**75.17**	**83.69**
25%	**DSSCC w/o srl**	78.60	90.24	54.93	73.05
	DSSCC w/o ssc	40.32	71.09	37.17	61.15
	DSSCC w/o em	24.86	75.02	23.36	64.10
	DSSCC w/o phase-1	79.95	90.44	58.85	74.74
	DSSCC w/o phase-2	72.51	88.38	58.62	76.40
	DSSCC	**80.36**	**91.43**	**61.91**	**77.44**

Table 5: Ablation with KIR (75%) and (25%) on CLINC150 and BANKING77 with $DSSCC_{SBERT}$ where we use *DSSCC-CH* for inference.

7 Conclusion

In this work, we propose a semi-supervised contrastive learning approach for intent detection and discovery from user logs. The proposed approach optimally utilizes both labeled and unlabeled utterances to outperform SOTA approaches for both scenarios where the total number of intents is either known in advance or has to be estimated. We also propose a variant of our approach which does not need to estimate the number of new intents and yields pure clusters which are merged by domain experts based on the cluster descriptions. Future work will focus on, (i) “How to get better cluster descriptions?” (ii) “How to optimally select query and corresponding candidate clusters?” and (iii) “How to discover new intents with minimum human effort with a long-tail distribution over new intents in user logs?”

References

Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. 2004. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the Fourth SIAM International Conference on Data Mining*, pages 333–344.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. 2017. Deep adaptive image clustering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Ajay Chatterjee and Shubhashis Sengupta. 2020. Intent mining from past conversations for conversational agent. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4140–4152, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

K. Chidananda Gowda and G. Krishna. 1978. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern Recognition*, 10(2):105–112.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

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.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Amir Hadifar, Lucas Sterckx, Thomas Demeester, and Chris Develder. 2019. A self-training approach for short text clustering. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 194–199, Florence, Italy. Association for Computational Linguistics.

Kai Han, Andrea Vedaldi, and Andrew Zisserman. 2019. Learning to discover novel visual categories via deep transfer clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

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*.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. 2017. Learning discrete representations via information maximizing self-augmented training. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1558–1567. JMLR.org.

Lawrence J. Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification*, 2:193–218.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

H. W. Kuhn and Bryn Yaw. 1955. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97.

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)*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2021. Contrastive clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(10):8547–8555.

Ting-En Lin, Hua Xu, and Hanlei Zhang. 2020. Discovering new intents via constrained deep adaptive clustering with cluster refinement. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8360–8367.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 480–496, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

J. B. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Md Rashadul Hasan Rakib, Norbert Zeh, Magdalena Jankowska, and Evangelos Milios. 2020. Enhancement of short text clustering by iterative classification. In *Natural Language Processing and Information Systems*, pages 105–117, Cham. Springer International Publishing.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

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

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617.

Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. 2020. Scan: Learning to classify images without labels. In *Proceedings of the European Conference on Computer Vision*.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Semi-supervised clustering for short text via deep representation learning. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 31–39, Berlin, Germany. Association for Computational Linguistics.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *ArXiv*, abs/2012.15466.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016a. Unsupervised deep embedding for clustering analysis. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 478–487, New York, New York, USA. PMLR.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016b. Unsupervised deep embedding for clustering analysis. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 478–487, New York, New York, USA. PMLR.

Jiaming Xu, Peng Wang, Guanhua Tian, Bo Xu, Jun Zhao, Fangyuan Wang, and Hongwei Hao. 2015. Short text clustering via convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 62–69, Denver, Colorado. Association for Computational Linguistics.

Jiaming Xu, Bo Xu, Peng Wang, Suncong Zheng, Guanhua Tian, and Jun Zhao. 2017. Self-taught convolutional neural networks for short text clustering. *Neural networks : the official journal of the International Neural Network Society*, 88:22–31.

Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3861–3870. JMLR.org.

Yi Yang, Dong Xu, Feiping Nie, Shuicheng Yan, and Yueting Zhuang. 2010. Image clustering using local discriminant models and global integration. *IEEE Transactions on Image Processing*, 19(10):2761–2773.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shang-Wen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang. 2021a. Supporting clustering with contrastive learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5419–5430, Online. Association for Computational Linguistics.

Hanlei Zhang, Xiaoteng Li, Hua Xu, Panpan Zhang, Kang Zhao, and Kai Gao. 2021b. TEXTOIR: An integrated and visualized platform for text open intent recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 167–174, Online. Association for Computational Linguistics.

Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lyu. 2021c. Discovering new intents with deep aligned clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14365–14373.

Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *CoRR*, abs/1502.01710.

A Appendix

A.1 Unsupervised and Semi-supervised Clustering

A.1.1 Unsupervised Clustering

Building upon classical clustering techniques such as **K-means** (MacQueen, 1967) and **Agglomerative Clustering (AG)** (Chidananda Gowda and Krishna, 1978) several deep learning-based clustering techniques have recently been proposed in the literature. **DEC** (Xie et al., 2016b) is a two-step deep unsupervised clustering algorithm where the first step involves training a stacked autoencoder (SAE) and the second step involves updating the SAE encoder and the cluster centers based on high confidence assignment of an utterance to cluster while using an auxiliary target distribution. In **SAE-KM** (Xie et al., 2016b), k-means is used to cluster the representations obtained from the encoder of the (SAE). **DCN** (Yang et al., 2017) is a joint deep unsupervised clustering algorithm performing joint representation learning and clustering. **DAC** (Chang et al., 2017) recasts the clustering problem into a binary pairwise classification framework to determine whether pairs of samples belong to the same cluster or not and the cosine similarity between pairs of samples is used to create pseudo-training data. **DeepCluster** (Caron et al., 2018) jointly learns the parameters of a neural network and the cluster assignments of the resulting features. DeepCluster iteratively groups the features with a standard clustering algorithm such as k-means and uses the subsequent assignments as supervision to update the weights of the network. **STC** (Xu et al., 2017) is an approach for short-text clustering where learned representations are clustered using k-means. **Self-Train** (Hadifar et al., 2019) extends **DEC** for short-text clustering and uses weighted average of pre-trained word embeddings (Mikolov et al., 2013) to get text representations. Rakib et al. (2020) alternately use classification and outlier detection to improve the accuracy of existing short-text clustering algorithms. **SCCL** Zhang et al. (2021a) jointly perform representation learning and clustering via contrastive representation learning and minimize a modified version of the clustering loss proposed by Xie et al. (2016b).

A.1.2 Semi-supervised Clustering

PCK-means (Basu et al., 2004) is a semi-supervised clustering algorithm where labeled samples are used as pairwise constraints to improve clustering performance. **KCL** (Hsu et al., 2018) is a two-stage image clustering algorithm that uses a binary classification model trained on labeled data in the first phase to measure pair-wise image similarity and in the second stage, a clustering model is trained on unlabeled data by using the output of the binary classification model for supervision. The network is trained using a Kullback-Leibler divergence-based contrastive loss (KCL). Meta Classification Likelihood **MCL** (Hsu et al., 2019) leverages pairwise similarity between samples and optimizes a binary classifier for pairwise similarity prediction and through this process learns a multi-class classifier as a submodule. **DTC**

	CLINC150 ($\mathcal{T}$=150, n=112, m = 38)				CLINC150$_{OOS}$ ($\mathcal{T}$=150, n=112, m=38)			
Approach	K'	K_{Pred}	**ACC**	**NMI**	K'	K_{Pred}	**ACC**	**NMI**
DSSCC$_{SBERT}$	300	129.6	81.37	92.97	300	130.5	80.24	91.95
	450	190.1	84.59	93.64	450	182.2	86.23	93.84
	600	255.9	80.83	92.04	600	248.1	79.96	91.20
	150	150	**88.47**	**94.50**	150	150	**87.23**	**94.39**

Table 6: *DSSCC*$_{SBERT}$ with and without out-of-scope (OOS) utterances in user logs, where value of m is not known.

		CLINC150			BANKING77		
KIR	λ	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
75%	2	71.20	91.95	66.09	67.34	79.46	55.26
	5	80.44	93.48	75.47	70.29	79.46	57.65
	7	83.78	94.0	78.31	70.78	79.39	58.11
	10	87.20	94.15	80.57	70.16	78.93	57.11
	14	89.51	94.61	82.52	67.99	77.82	54.95
	15	89.73	94.61	82.75	68.99	77.91	55.37
	17	88.93	94.10	81.60	68.70	77.51	54.72
	20	89.42	94.17	82.23	67.70	76.92	53.97
	25	89.91	94.26	82.77	66.82	76.16	52.77
	30	88.76	93.44	80.77	66.33	76.02	52.37

Table 7: Ablation with KIR (75%) on *CLINC150* and *BANKING77* where we try different values of λ. We use *DSSCC*$_{SBERT}$ for ablation.

(Han et al., 2019) extend *DEC* to a semi-supervised scenario and also improve upon DEC by enforcing a representation bottleneck, temporal ensembling, and consistency. **CDAC+** (Lin et al., 2020) is an end-to-end clustering method that incorporates pairwise constraints obtained from labeled utterances as prior knowledge to guide the clustering process and clusters are further refined by forcing the model to learn from high confidence assignments. **DeepAligned** (Zhang et al., 2021c) use labeled utterances from known intents to update BERT parameters and in the second step perform representation learning and K-means clustering alternately by minimizing cross-entropy loss over labeled and pseudo-labeled utterances treating each k-means cluster as one intent.

A.2 Representation of an utterance from PLM

We get the representation $\boldsymbol{h_t}$ = mean-pooling($[CLS, T_1, T_2, ..., T_l]$) of an utterance u_t consisting of l tokens from BERT/SBERT by applying mean-pooling over representations of all tokens including CLS, where T_j denotes representation corresponding to j^{th} token.

Dataset	Avg utterance length	$z\%$
CLINC150	8.31	10
BANKING77	11.91	20
SNIPS	9.03	10
StackOverflow	9.18	10
DBPedia	29.97	30

Table 8: Data Augmentation Details.

A.3 Dataset Description and Details

We evaluate *DSSCC* on five datasets with a varying number of intents. And all of them are available in english language and released under creative Commons licences.

BANKING77 (Casanueva et al., 2020) is a fine-grained intent detection dataset from the banking domain comprising of 13,083 customer queries labelled with 77 intents.

CLINC150 (Larson et al., 2019) is a crowdsourced multi-domain (10 domains such as utility, travel etc.) intent detection dataset comprised of 23,700 queries with 22,500 in-scope queries labelled with 150 intents and 1,200 out-of-scope queries. For our experiments, we use both sets-CLINC150 which contains only in-scope queries, and CLINC-150$_{OOS}$ which contains both in-scope and out-of-scope queries and we use the balanced version of the dataset.

SNIPS (Coucke et al., 2018) consists of 16000 crowd-sourced user utterances distributed across 7 intents. Out of 16k, 14484 utterances has been used for experimental purpose in the past.

StackOverflow: This dataset was originally released as part of a kaggle competition[2]. Xu et al. (2015) used a subset of this dataset for short-text clustering. The dataset consists of 20, 000 technical question titles distributed across 20 intents with 1k questions per intent.

DBPedia (Zhang and LeCun, 2015) is an ontology classification dataset constructed by picking 14

[2] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow

non-overlapping classes from DBpedia 2014 Lehmann et al. (2015). Wang et al. (2016) used a subset of this dataset consisting of 14000 samples distributed across 14 classes.

We mention dataset statistics in table 9. We mention the train, validation, and test split details, that were used while performing the experiments for each dataset. Based, on the len and $\mathcal{T}(n+m)$ attributes, we can see the variation and complexity of these datasets.

A.4 Training Details

For the first phase, we follow the same pre-training steps outlined by Zhang et al. (2021c). For the second phase *of DSSCC-E2E*, an embedding of dimension ($d*(n+m)$), is used for the cluster centers where a d dimensional representation of an utterance u is obtained from the PLM and $n+m$ represents the total number of clusters corresponding to known and new intents. For the *HIL* approach, two different linear heads/layers are used - one with dimension ($d*(128)$) for Instance-Level Contrastive Learning and second with dimension ($d*(n)$) for Supervised Representation Learning. For simulating a real-world problem of intent detection and discovery from user logs, we follow the experimental setting similar to Zhang et al. (2021c) where they assume that x% $\in$ {25%, 50%, 75%} of the total intents for a given dataset is known (we denote this number by n) where x is also referred to as the known intent ratio (KIR). The remaining number of intents (m) are considered novel. Accordingly, each dataset is divided into a $\mathcal{D}_{train}$, $\mathcal{D}_{val}$ and $\mathcal{D}_{test}$ where $\mathcal{D}_{train}$ contains 10% of labeled utterances per known intent and unlabeled utterances from both known and unknown intents. $\mathcal{D}_{val}$, $\mathcal{D}_{test}$ consists of utterances from known and new intents. We do two sets of experiments – one with a known value of $n+m$ (number of total intents) and another one where the total number of intents is not known in advance. For a given dataset and KIR, we run the same experiment on ten different seeds and report the average ACC, NMI, and ARI on $\mathcal{D}_{test}$. *For DSSCC-E2E,* we use *DSSCC-CH*, *DSSCC-KM* for intent detection and discovery and report the best results based on majority voting over *ACC*, *NMI* and *ARI*. In a realistic scenario when ground truth is not available, one can use the Silhouette Score (Rousseeuw, 1987) to decide which inference strategy to use. For the *HIL* approach, we get predictions from *DBSCAN* clustering after model convergence and perform inference after running the cluster merger algorithm. We use existing checkpoints of bert-base-uncased [3] (Devlin et al., 2019) and stsb-roberta-base-v2 [4] (Reimers and Gurevych, 2019) as our PLM. Similar to (Zhang et al., 2021c), we freeze all but the last transformer layer parameters in our PLM for both phases to improve training efficiency and speed up the training process. We use the Adam optimizer (Kingma and Ba, 2014) to update PLM parameters and the learning rate is set to 5e-5 for PLM, 5e-3 for cluster centers in case of *DSSCC-E2E* and 5e-3 for both heads in case of *HIL* Approach . For all of our experiments, the batch size is kept at 400. For *DSSCC-E2E*, the entropy weight(λ) is set as 14.0. Whereas, For *HIL*, the *minimum samples* and *epsilon* for DBSCAN is kept as 3.0 and 0.09 respectively. We run all experiments on an Nvidia Titan A100 GPU. We use classification accuracy on the $\mathcal{D}_{val}$ set for known intents as converge criteria for Phase 1. And for Phase 2 of *DSSCC-E2E*, we calculate the Silhouette Score Rousseeuw (1987) given utterance representations and corresponding predicted cluster-ids on $\mathcal{D}_{train}$. Whereas, for Phase 2 of *HIL*, we converge when the number of predicted clusters by *DBSCAN* clustering is minimum. We use early stopping with a patience value of 20.0 for both phases. For semi-supervised contrastive representation learning, similar to Zhang et al. (2021a), we use contextual augmenter Kobayashi (2018) to generate augmentations corresponding to unlabeled utterances where z% of the words in an utterance are substituted with similar words. We use a suitable value of z% for different datasets based on average utterance length as mentioned in the table 8 following the observations from Zhang et al. (2021a). We do this to preserve the semantics of an utterance while at the same time, substituting words in an utterance to create augmentations. We report the best results averaged over ten different seeds based on the inference details as mentioned in section 4.4. For our codebase, we have adapted existing SupContrast [5] loss in the semi-supervised setting and also utilized data creation steps from Zhang et al. (2021c)[6].

[3] https://huggingface.co/bert-base-uncased

[4] https://huggingface.co/sentence-transformers/stsb-roberta-base-v2

[5] https://github.com/HobbitLong/SupContrast

[6] https://github.com/thuiar/DeepAligned-Clustering

A.5 Results on Intent Detection and Discovery

We have also reported results on five public datasets with our proposed approach on intent detection and intent discovery separately for the case when m is known, are shown in table 11. If we assume that intent detection and discovery are two separate problems, we decouple the results from our joint approach (after training) to see the contribution of *DSSCC* on both tasks. It is clear from the results that, even with very few labeled utterances from known intents, our model maintains the performance on the known intents with at least 83% clustering accuracy on all five datasets. From the results on Intent Discovery, except for BANKING77, *DSSCC (Ours)* gets at least 74% clustering accuracy on 4 datasets. The low performance on intent discovery in BANKING77 is attributed to the complexity of the dataset where all intents are part of one larger domain, i.e., Banking. Whereas, in CLINC150, the intents belong to multiple domains as mentioned in section A.3.

A.6 DSSCC-KM vs DSSCC-CH

We use both inference strategies, i.e., *DSSCC-KM* and *DSSCC-CH* to obtain results for all experiments described in section 6 and report *ACC*, *NMI* and *ARI* as shown in Table 12, 13 and 15. *DSSCC-CH* outperforms *DSSCC-KM* on CLINC150, SNIPS and StackOverflow whereas *DSSCC-KM* gives better results on BANKING77 and DBPedia. This inconsistency between the behaviour of *DSSCC-CH* and *DSSCC-KM* can be attributed to complexity of a given dataset, i.e., *DSSCC-KM* outperforms *DSSCC-CH* on BANKING77 (single domain dataset) and DBpedia whereas *DSSCC-CH* outperforms *DSSCC-KM* on CLINC150 (multi-domain dataset).

In realistic scenario where ground truth is not available, one can use Silhouette Score (SS) to choose between *DSSCC-CH* and *DSSCC-KM*. As shown in Fig. 6, when the difference between *SS* corresponding to *DSSCC-CH* and *DSSCC-KM* is significant, then one should choose inference strategy which gives higher *SS*. And when the difference between *SS* score corresponding to *DSSCC-CH* and *DSSCC-KM* is not significant, then one can choose *DSSCC-CH* for inference, as shown in Fig 5. Above mentioned approach for inference strategy selection correlates with the selection done by majority voting (over *ACC*, *NMI* and *ARI*).

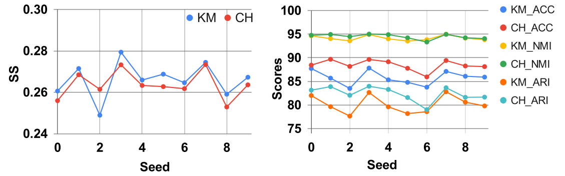


Figure 5: CLINC150 (KIR=75%) *DSSCC-CH* vs *DSSCC-KM* where we use *SBERT* as PLM. The left subfigure shows silhouette scores over different seeds and the right subfigure shows ACC, NMI, and ARI with DSSCC-KM and DSSCC-CH over these seeds.

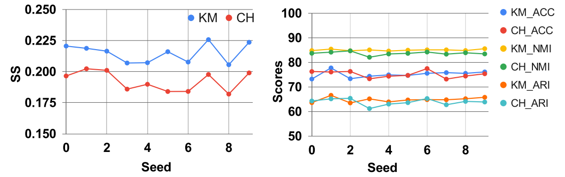


Figure 6: BANKING77 (KIR=75%) *DSSCC-CH* vs *DSSCC-KM* where we use *SBERT* as PLM. The left subfigure shows silhouette scores over different seeds and the right subfigure shows ACC, NMI, and ARI with DSSCC-KM and DSSCC-CH over these seeds.

A.7 Representations from *SBERT* vs *DSSCC*$_{SBERT}$

To showcase the effectiveness of representations learnt by *DSSCC*$_{SBERT}$, we plot the utterance embeddings ($\boldsymbol{h_t}$) with ground truth labels for all five datasets as shown in Fig. 7 and Fig. 8. Initial and final representations correspond to utterance embeddings obtained from *SBERT* and *DSSCC*$_{SBERT}$ respectively. It can be observed that for all five datasets, intents are clearly separable with final representations as compared to initial representations.

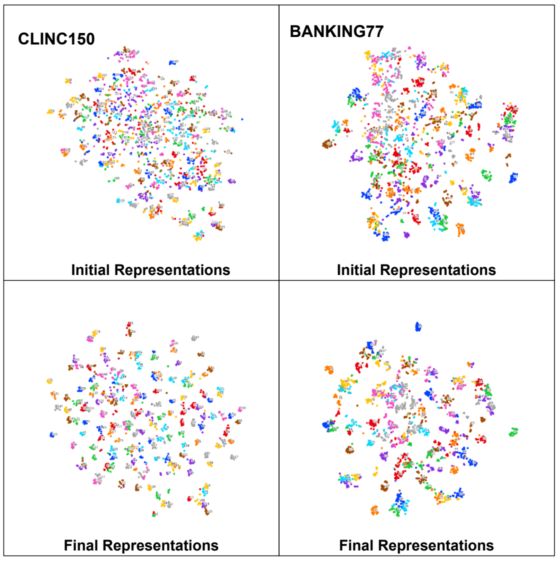


Figure 7: TSNE (van der Maaten and Hinton, 2008) plot for CLINC150 and Banking77 (75% KIR) before and after DSSCC Training.

Dataset	$\mathcal{D}_{train}$	$\mathcal{D}_{val}$	$\mathcal{D}_{test}$	$len(max/mean)$	$\mathcal{T}(n+m)$
CLINC150	18,000	2,250	2,250	25/8.31	150(112+38)
CLINC150$_{OOS}$	19,000	2,250	2,450	25/8.32	150(112+38)
BANKING77	9,003	1,000	3,080	79/11.91	77 (58+19)
SNIPS	13,084	700	700	35/9.03	7 (5+2)
StackOverflow	18,000	1,000	1,000	41/9.18	20 (15+5)
DBPedia	12,600	700	700	54/29.97	14 (11+3)

Table 9: Dataset Details. $\mathcal{D}_{train}$, $\mathcal{D}_{test}$, $\mathcal{D}_{val}$: number of examples in train, validation and test set respectively; $len(max/mean)$: maximum sentence length / mean sentence length; $\mathcal{T}(n+m)$: Total number of classes (Known Intents + New Intents) in case of 75% KIR

		SNIPS		**StackOverflow**		**DBPedia**	
KIR	**Approach**	**ACC**	**NMI**	**ACC**	**NMI**	**ACC**	**NMI**
25%	**DA**$_{BERT}$	86.21	80.42	69.66	70.23	85.89	88.98
	DA$_{SBERT}$	81.16	77.33	72.64	73.43	83.70	85.40
	DSSCC$_{SBERT}$	**94.33**	**89.30**	**81.72**	**76.57**	**89.44**	**89.25**
50%	**DA**$_{BERT}$	85.69	83.03	72.89	74.49	88.63	91.24
	DA$_{SBERT}$	88.83	84.19	73.07	74.08	87.29	88.80
	DSSCC$_{SBERT}$	**95.20**	**91.07**	**82.43**	**77.30**	**92.14**	**92.70**
75%	**DA**$_{BERT}$	90.10	86.94	74.51	76.24	92.17	**93.25**
	DA$_{SBERT}$	92.70	88.22	75.50	75.90	91.17	91.14
	DSSCC$_{SBERT}$	**94.87**	**90.44**	**82.65**	**77.08**	**92.73**	92.58

Table 10: Intent detection and discovery results on three datasets, i.e., SNIPS, StackOverflow and DBPedia where we computed results corresponding to DeepAligned$_{BERT}$ (DA$_{BERT}$), DeepAligned$_{SBERT}$ (DA$_{SBERT}$) using code provided by Zhang et al. (2021c)

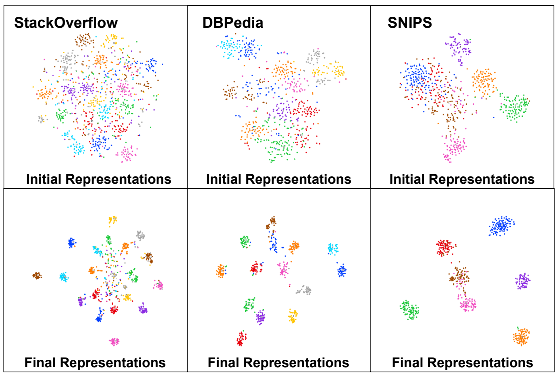


Figure 8: TSNE plot for StackOverflow, DBPedia and SNIPS (75% KIR) before and after DSSCC Training.

		Intent Detection			Intent Discovery		
KIR	Dataset	ACC	NMI	ARI	ACC	NMI	ARI
25%	CLINC150	91.14	96.87	92.84	76.70	90.66	72.14
	BANKING77	78.11	88.53	79.27	60.61	79.09	53.37
	SNIPS	94.36	87.36	88.98	94.32	88.85	89.0
	StackOverflow	86.4	78.82	80.15	80.16	75.26	68.33
	DBPedia	96.0	93.25	94.12	86.82	87.74	82.28
50%	CLINC150	92.03	96.59	91.13	74.97	91.02	74.62
	BANKING77	79.66	88.66	77.19	59.35	80.18	58.0
	SNIPS	95.27	90.78	91.07	95.33	89.90	91.21
	StackOverflow	86.72	81.37	79.50	78.14	72.96	67.99
	DBPedia	95.26	94.96	94.81	88.40	91.07	88.95
75%	CLINC150	92.73	96.44	89.89	75.91	91.52	78.77
	BANKING77	83.60	87.61	74.99	57.72	80.71	62.39
	SNIPS	95.54	91.99	91.84	93.17	81.92	84.77
	StackOverflow	85.37	79.76	75.22	74.48	66.92	64.48
	DBPedia	95.72	94.73	94.09	85.25	86.66	84.02

Table 11: Performance of *DSSCC (Ours)* on Intent Detection and Intent Discovery

		CLINC150			BANKING77		
KIR	Approach	ACC	NMI	ARI	ACC	NMI	ARI
0%	K-means$_{BERT}$	45.06	70.89	26.86	29.55	54.57	12.18
	K-means$_{SBERT}$	61.04	82.22	48.56	55.72	74.68	42.77
	AG	44.03	73.07	27.70	31.58	57.07	13.31
	SAE-KM	46.75	73.13	29.95	38.92	63.79	22.85
	DEC	46.89	74.83	27.46	41.29	67.78	27.21
	DCN	49.29	75.66	31.15	41.99	67.54	26.81
	DAC	55.94	78.40	40.49	27.41	47.35	14.24
	DeepCluster	35.70	65.58	19.11	20.69	41.77	8.95
	SCCL	33.52	66.63	18.89	13.41	34.14	4.02
25%	PCK-means	54.51	68.71	35.38	32.66	48.22	16.24
	BERT-KCL	24.72	65.74	17.97	22.11	52.42	15.75
	BERT-MCL	24.35	65.06	16.82	22.07	51.96	13.94
	BERT-DTC	49.1	74.17	33.05	25.24	48.58	13.32
	CDAC+	64.64	84.25	50.35	48.71	69.78	35.09
	DeepAligned	73.71	88.71	64.27	48.88	70.45	36.81
	DSSCC-KM	74.98	89.19	65.75	55.52	72.73	42.11
	DSSCC-CH	75.72	89.12	66.72	48.38	66.39	33.94
	DSSCC	**75.72**	**89.12**	**66.72**	**55.52**	**72.73**	**42.11**
50%	PCK-means	54.51	68.62	35.23	32.26	48.11	16.02
	BERT-KCL	46.91	78.45	37.94	40.97	65.22	30.03
	BERT-MCL	47.21	78.39	36.72	41.43	65.68	28.87
	BERT-DTC	71.68	86.20	59.62	53.59	71.40	40.65
	CDAC+	69.02	86.18	54.15	53.34	71.53	40.42
	DeepAligned	80.22	**91.63**	72.34	59.23	76.52	47.82
	DSSCC-KM	79.85	91.44	73.48	63.08	77.60	50.64
	DSSCC-CH	81.46	91.39	73.48	59.14	73.11	44.98
	DSSCC	**81.46**	91.39	**73.48**	**63.08**	**77.60**	**50.64**
75%	PCK-means	54.61	68.70	35.40	32.66	48.22	16.24
	BERT-KCL	68.86	86.82	58.79	60.15	75.21	46.72
	BERT-MCL	69.66	87.72	59.92	61.14	75.68	47.43
	BERT-DTC	80.73	90.41	70.92	56.51	76.55	44.70
	CDAC+	69.89	86.65	54.33	53.83	72.25	40.97
	DeepAligned	86.01	**94.03**	79.82	64.90	79.56	53.64
	DSSCC-KM	84.99	93.43	78.35	69.82	81.24	58.09
	DSSCC-CH	87.91	93.87	81.09	68.13	78.15	54.23
	DSSCC	**87.91**	93.87	**81.09**	**69.82**	**81.24**	**58.09**

Table 12: We report ACC, NMI and ARI on CLINC150 and BANKING77 datatsets in the semi-supervised scenario for three different known intent ratios (KIR). Except **K-means**$_{SBERT}$ and **SCCL**, we take all baseline results from Zhang et al. (2021c). For fair comparison we use *BERT* as PLM in DSSCC.

		SNIPS			StackOverflow			DBPedia		
KIR	**Approach**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
25%	**DA**$_{BERT}$	86.21	80.42	74.70	69.66	70.23	53.69	85.89	88.98	79.22
	DA$_{SBERT}$	81.16	77.33	68.38	72.64	73.43	58.30	83.70	85.40	75.67
	DSSCC-KM	93.77	89.05	86.78	79.37	76.93	58.73	89.44	89.25	83.29
	DSSCC-CH	94.33	89.30	87.90	81.72	76.57	68.0	87.70	88.12	81.25
	DSSCC	**94.33**	**89.30**	**87.90**	**81.72**	**76.57**	**68.0**	**89.44**	**89.25**	**83.29**
50%	**DA**$_{BERT}$	85.69	83.03	77.03	72.89	74.49	57.96	88.63	91.24	83.38
	DA$_{SBERT}$	88.83	84.19	79.53	73.07	74.08	59.34	87.29	88.80	81.04
	DSSCC-KM	94.57	90.23	88.47	80.24	77.80	61.08	91.83	92.27	87.85
	DSSCC-CH	95.20	91.07	89.67	82.43	77.30	68.94	92.14	92.70	88.61
	DSSCC$_{SBERT}$	**95.20**	**91.07**	**89.67**	**82.43**	**77.30**	**68.94**	**92.14**	**92.70**	**88.61**
75%	**DA**$_{BERT}$	90.10	86.94	82.42	74.51	76.24	59.45	92.17	**93.25**	88.12
	DA$_{SBERT}$	92.70	88.22	85.40	75.50	75.90	61.21	91.17	91.14	85.94
	DSSCC-KM	94.03	89.20	87.40	80.53	77.25	63.70	92.73	92.58	88.55
	DSSCC-CH	94.87	90.44	89.03	82.65	77.08	68.67	92.13	92.61	88.65
	DSSCC	**94.87**	**90.44**	**89.03**	**82.65**	**77.08**	**68.67**	**92.73**	92.58	**88.55**

Table 13: Intent detection and discovery results on three datasets, i.e., SNIPS, StackOverflow and DBPedia where we computed results corresponding to DeepAligned$_{BERT}$ (DA$_{BERT}$), DeepAligned$_{SBERT}$ (DA$_{SBERT}$) using code provided by Zhang et al. (2021c) and user $SBERT$ as PLM in *DSSCC*.

		CLINC150			BANKING77		
K	**A**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
25%	**DA**$_{BERT}$	73.71	88.71	64.27	48.88	70.45	36.81
	DA$_{SBERT}$	67.78	86.50	57.10	57.0	75.0	45.80
	DSSCC$_{BERT}$	75.72	89.12	66.72	55.52	72.73	42.11
	DSSCC$_{SBERT}$	**80.36**	**91.43**	**72.83**	**64.93**	**80.17**	**53.60**
50%	**DA**$_{BERT}$	80.22	91.63	72.34	59.23	76.52	47.82
	DA$_{SBERT}$	77.69	91.40	70.90	64.14	79.30	52.70
	DSSCC$_{BERT}$	81.46	91.39	73.48	63.08	77.60	50.64
	DSSCC$_{SBERT}$	**83.49**	**92.78**	**76.80**	**69.38**	**82.68**	**58.95**
75%	**DA**$_{BERT}$	86.01	94.03	79.82	64.90	79.56	53.64
	DA$_{SBERT}$	85.89	94.20	79.83	74.08	83.80	63.30
	DSSCC$_{BERT}$	87.91	93.87	81.09	69.82	81.24	58.09
	DSSCC$_{SBERT}$	**88.47**	**94.50**	**82.40**	**75.15**	**85.04**	**64.83**

Table 14: *DA* vs *DSSCC* with *BERT* and *SBERT* as PLM

	CLINC150 ($\mathcal{T}$=150, n=112, $m = 38$)					**BANKING77** ($\mathcal{T}$=77, n=58, m=19)				
Approach	K'	K_{Pred}	**ACC**	**NMI**	**ARI**	K'	K_{Pred}	**ACC**	**NMI**	**ARI**
DA$_{BERT}$	300	130	77.18	92.5	72.26	154	65.1	62.49	78.88	51.71
DA$_{SBERT}$	300	129.6	76.87	92.61	72.05	154	66.9	63.53	80.84	53.26
DSSCC-KM$_{SBERT}$			79.0	92.72	73.58			71.77	84.29	62.13
DSSCC-CH$_{SBERT}$			81.37	92.97	75.49			71.91	82.60	60.69
DSSCC$_{SBERT}$			**81.37**	**92.97**	**75.49**			**71.77**	**84.29**	**62.13**
DA$_{BERT}$	450	189.2	83.81	93.57	79.54	231	99.3	63.98	79.93	53.76
DA$_{SBERT}$	450	190.1	82.57	**93.85**	79.28	231	96.8	66.20	82.11	56.98
DSSCC-KM$_{SBERT}$			82.69	93.59	78.85			72.93	84.97	64.65
DSSCC-CH$_{SBERT}$			84.59	93.64	80.44			72.60	82.78	62.86
DSSCC$_{SBERT}$			**84.59**	93.64	**80.44**			**72.93**	**84.97**	**64.65**
DA$_{BERT}$	600	258.6	72.22	91.8	70.91	308	121.9	61.05	79.95	53.10
DA$_{SBERT}$	600	255.9	72.29	**92.18**	71.38	308	118.1	62.67	82.05	55.75
DSSCC-KM$_{SBERT}$			73.42	92.07	72.19			67.56	84.43	61.87
DSSCC-CH$_{SBERT}$			80.83	92.04	76.28			68.08	81.19	59.37
DSSCC$_{SBERT}$			**80.83**	92.04	**76.28**			**67.56**	**84.43**	**61.87**
DSSCC$_{SBERT}$	150	150	**88.47**	**94.50**	**82.40**	77	77	**75.15**	**85.04**	**64.83**

Table 15: Intent detection and discovery with unknown value of m for KIR=75%. We obtain results corresponding to DeepAligned$_{BERT}$ (DA$_{BERT}$) (Zhang et al., 2021c) and report results for DeepAligned$_{SBERT}$ (DA$_{SBERT}$) using code provided by the authors of Zhang et al. (2021c) with SBERT as the PLM.