A Framework to Generate High-Quality Datapoints for Multiple Novel Intent Detection

[1]Ankan Mullick* [2]Sukannya Purkayastha* † [1]Pawan Goyal [1,3]Niloy Ganguly
[1]Department of Computer Science and Engineering, IIT Kharagpur
[2]TCS Research, India
[3]Leibniz University of Hannover, Germany
ankanm@kgpian.iitkgp.ac.in, sukannya.purkayastha@tcs.com
{pawang, niloy}@cse.iitkgp.ac.in

Abstract

Systems like Voice-command based conversational agents are characterized by a pre-defined set of skills or intents to perform user specified tasks. In the course of time, newer intents may emerge requiring retraining. However, the newer intents may not be explicitly announced and need to be inferred dynamically. Thus, there are two important tasks at hand (a). identifying emerging new intents, (b). annotating data of the new intents so that the underlying classifier can be retrained efficiently. The tasks become specially challenging when a large number of new intents emerge simultaneously and there is a limited budget of manual annotation. In this paper, we propose **MNID** (**M**ultiple **N**ovel **I**ntent **D**etection) which is a cluster based framework to detect multiple novel intents with budgeted human annotation cost. Empirical results on various benchmark datasets (of different sizes) demonstrate that MNID, by intelligently using the budget for annotation, outperforms the baseline methods in terms of accuracy and F1-score.

1 Introduction

The conversational agents such as Amazon Alexa, Apple Siri are characterised by the skill of understanding *intents* which help them to efficiently handle a user's query. For example, the query **'Will it be colder in Ohio'** requires getting the weather updates for the city 'Ohio' and would be associated to the intent *GetWeather*. The agents are trained with a pre-defined set of intents such as {*GetWeather*, *RateBook*, *BookRestaurant*} so as to perform the goal-oriented user tasks. But with the passage of time, a user may be interested in performing newer tasks adding hitherto unknown intents. For example, **'Play some music from 1954'** would be associated to the intent *PlayMusic* that may not be a part of the set of pre-defined intents.

Emergence of novel intent detection has been periodically checked by different models in the last decade. There are works on incremental learning in dynamic environment for evolving new classes (Zhou and Chen, 2002; Kuzborskij et al., 2013; Scheirer et al., 2012). There are also several approaches (Sun et al., 2016; Masud et al., 2010; Haque et al., 2016; Wang et al., 2020; Mu et al., 2017b,a) to detect new classes in the form of outlier detection but they do not distinguish among multiple new class labels so are not effective in novel multi-class detection. Xia et al. (2018); Siddique et al. (2021) detect user intents using zero-shot generalized intent detection framework. However, they assume that the unseen intent class LABELS are already known, while in our case neither the number of unseen intent classes, nor the corresponding class labels are known. The other line of works (Xia et al., 2021; Halder et al., 2020) supply the system with new intents, albeit with a limited amount of tagged data per class and then have an efficient algorithm to incrementally learn new classes. These models work on the assumption that some instances of these new classes would be provided for model building. However, in a realistic setting, the system may not have any knowledge of the number and types of new intents appearing, it may at most understand that some new out-of-domain samples are generated. So, the problem statement is to probe the incoming data wisely and **use minimum human intervention to identify all types of novel intents emerging and intelligently tag a limited set of data covering all discovered intents**, which can be be fed into a model for retraining.

More concretely the system is at first trained with an initial set of known intents; side-by-side an out-of-distribution (OOD) detector classifier is also trained to identify datapoints which do not fit the known intents. When substantial amount of

* Equal contribution

† Work done while the author was a student at IIT Kharagpur

Findings of the Association for Computational Linguistics: NAACL 2022, pages 282 - 292
July 10-15, 2022

such points are detected, the task is to (a) identify whether the points are originating from introduction of a single novel intent or multiple and (b) choose (a limited number of) samples to annotate so that the classifier can be retrained efficiently.

In order to determine the number of novel intents present in the OOD data, we undertake a clustering based approach with the idea that each cluster would represent a novel intent. By increasing the cluster number progressively, we can make a highly accurate estimate of the number of novel intents. If sample points of an intent mainly correspond to a well formed cluster, the implication is that without much probing we can shortlist enough training samples (through silver tagging) for that class. On the other hand, if the sample points of an intent tend to intertwine with other intent points in the feature space, these can be considered as uncertain points and require human intervention for tagging (gold tagging). With this intuition in place, we design a mix of silver and gold tagging to produce high-quality training samples which can be used to retrain the classifier.

Our proposed framework of Multiple Novel Intent Detection (**MNID**) is compared with competitive baselines and evaluated across several standard public datasets in NLU domain where it performs substantially better. We use datasets with different number of intent classes. SNIPS (Coucke et al., 2018) and ATIS (Tur et al., 2010) are smaller datasets, consisting of less number intent classes - 7 and 21, respectively. HWU (Liu et al., 2019a), BANKING (Casanueva et al., 2020) and CLINC (Larson et al., 2019) consist of large number of intent classes - 64, 77 and 150, respectively.

The paper is organized as follows. We discuss the Problem Setting and solution overview in Section 2. Our algorithmic framework is described in Section 3. We present the datasets with experimental statistics and data pre-processing in Section 4. In Section 5, we discuss the experimental design and baselines. Detail evaluation results with different algorithmic variations are in Section 6. We conclude with a summary in Section 7[1].

2 Problem Setting and Solution Overview

Problem Setting: To formally describe the problem setting, let there be a dataset W containing overall N classes. However, the value of N is not known apriori. Let $T \in W$ be the test set and $W - T = D$ be the rest of the dataset, out of which $|D_{init}|$ ($<<$ $|D|$) labelled data of N_{init} ($<$ N)

[1] Codes are in - https://github.com/sukannyapurkayastha/MNID

Algorithm 1 Multiple Novel Intent Detection (**MNID**)

Input
  D_{init} Initial Labelled Data
  $\mathcal{T}$ Blind Test Data For Evaluation
  B Total Annotation Budget
Parameters
  D Total Data points
  $\mathcal{L} \leftarrow D_{init}$
  $\mathcal{OS} \leftarrow$ OODD(D, D_{init})
procedure MULTIPLE NOVEL INTENT DETECTION
  $\mathcal{L}$, N_{new}, $\mathcal{CL} \leftarrow$ NCD($\mathcal{OS}$, $\mathcal{L}$)
  if $\mathcal{L} < B$ **then**
    $\mathcal{L}$, $\mathcal{G_{CL}}$, $\mathcal{B_{CL}} \leftarrow$ CBQA($\mathcal{L}$, $\mathcal{CL}$)
  end if
  Train Model $\mathcal{M}$ on $\mathcal{L}$, predict on the remaining points in the clusters to get the confidence score (CS) of each data point and store in All_{CS}.
  if $\mathcal{L} < B$ **then**
    $\mathcal{L} \leftarrow$ PPAS($\mathcal{L}$, All_{CS}, $\mathcal{G_{CL}}$, $\mathcal{B_{CL}}$, B)
  end if
  Train model $\mathcal{M}$ on $\mathcal{L}$ and test on $\mathcal{T}$ to find out Accuracy, F1 for all classes.
end procedure

Algorithm 2 OOD Detection Algorithm **OODD**($|D|$, $|D_{init}|$)

Train OOD-SDA on D_{init} and predict on $(D - D_{init})$ to get OOD samples, $\mathcal{OS}$
Return $\mathcal{OS}$

Algorithm 3 Novel Class Detection **NCD**($\mathcal{OS}$, $\mathcal{L}$)

Initial number of clusters, $K = 1$.
Number of new classes, $N_{new} = 0$.
while $N_{new} \geq \lfloor K/2 \rfloor$ **do**
  Perform K-Means Clustering on $\mathcal{OS}$
  Annotate x (≥ 2) points from each cluster. That results in discovering of n' new classes
  Add $x * K$ point labels to $\mathcal{L}$
  $N_{new} \leftarrow N_{new} + n'$
  $K \leftarrow 2 * K$
end while
$\mathcal{CL}$ = Store All K Clusters
Return ($\mathcal{L}$,N_{new}, $\mathcal{CL}$)

Algorithm 4 Cluster Quality Based Annotation **CQBA**($\mathcal{L}$, $\mathcal{CL}$)

Take p points from each of the clusters ($\mathcal{CL}$) and annotate to find Good Cluster ($\mathcal{G}_{\mathcal{CL}}$) and Bad Cluster ($\mathcal{B}_{\mathcal{CL}}$).
Add annotated $p * |\mathcal{CL}|$ point labels to $\mathcal{L}$.
for each Bad Cluster **do**
    Take q more points from Bad cluster and annotate.
    Add $q * |\mathcal{B}_{\mathcal{CL}}|$ point labels to $\mathcal{L}$.
end for
Return($\mathcal{L}$, $\mathcal{G}_{\mathcal{CL}}$, $\mathcal{B}_{\mathcal{CL}}$)

Algorithm 5 Post-Processing Annotation Strategy **PPAS**($\mathcal{L}$, All_{CS}, $\mathcal{G}_{\mathcal{CL}}$, $\mathcal{B}_{\mathcal{CL}}$, B)

for Each point with CS in All_{CS} **do**
    if $CS \geq \mathcal{TH}$ and point in $\mathcal{G}_{\mathcal{CL}}$ and average cosine similarity with already annotated points of that class $\geq \tau$ **then**
        $L_s \leftarrow$ Silver Annotation Strategy
    end if
end for
while $|\mathcal{L}| < B$ **do**
    Select datapoint with least CS
    if $\mathcal{B}_{\mathcal{CL}}$ exists **then**
        From $\mathcal{B}_{\mathcal{CL}}$ in Round-Robin way
    else
        From $\mathcal{G}_{\mathcal{CL}}$ in Round-Robin way
    end if
    $L_g \leftarrow$ Gold Annotation Strategy
    $\mathcal{L} \leftarrow \mathcal{L} + L_g$
end while
$\mathcal{L} \leftarrow \mathcal{L} \cup L_s \cup L_g$
Return ($\mathcal{L}$)

classes is initially provided, while the rest of the data is unlabelled. The task is to design an algorithm to (a). detect all the remaining $N - N_{init}$ classes and (b). spent a limited budget (B - $|D_{init}|$) to annotate high fidelitous new datapoints, so that the classifier can achieve high accuracy when retraining.

Solution Overview: The solution steps are as follows: (a) Identify the OOD (out of distribution) datapoints which do not belong to the initial N_{init} classes. This can be considered as a preprocessing step. (b) Use a part of the allotted budget to annotate a portion of these OOD datapoints. These points (for annotation) are selected by repeatedly running a clustering algorithm with increasing number of clusters as input, and choosing cluster centre points to identify the unknown classes. **Rationale:** The intuition/expectation is that each cluster hosts a separate intent, hence annotating the cluster centres would lead to discovery of maximum number of novel intents. (c) Further identify the classes which are well clustered in feature space and which are not. Use another portion of the budget to increase the annotations of not-so well formed clusters and then build up a classifier with all the classes. **Rationale:** If a cluster is well-formed, most likely it is hosting a single class, hence there is no need to annotate further points there, rather annotate more points in not-so-well-formed clusters. (d) Use the classifier to classify points from the clusters. Identify low-confidence points from the bad clusters and annotate them. High-confidence points from good clusters are silver annotated. **Rationale:** The low-confidence points in the bad clusters are the most uncertain points, hence annotating them helps in increasing classifier accuracy. Similarly high-confident points in the good clusters almost surely will belong to that particular cluster, hence silver annotation is pursued. (e). Retrain the classifier.

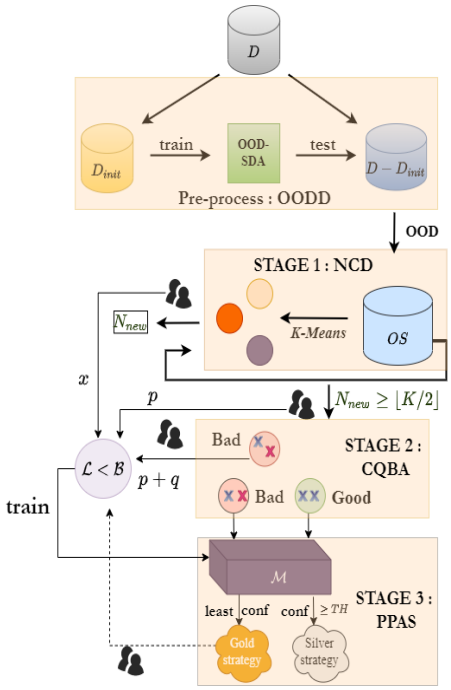


Figure 1: End-to-end architecture of MNID: Multiple Novel Intent Detection

The overall MNID framework with different algorithmic modules is shown in Fig. 1.

3 MNID: Solution Detail

The proposed framework for Multiple Novel Intent Detection (MNID) is explained through Algorithm 1. As highlighted in the overview, the algorithm consists of data pre-processing step, followed by three stages, each of them are discussed below. The total budget of (gold) annotation is B. Besides the system can undertake unlimited silver annotation. The advantage of silver strategy is that it is free as no human probing is required. However, it is also likely to bring in noise if used indiscriminately.

Pre-process: OOD Detection (OODD): For the dataset (D), this module (Algorithm 2) takes the initial labelled data (D_{init}) as input and predicts the Out-of-Domain (OOD) samples on the remaining data, ($D - D_{init}$). We call the set of OOD samples predicted as $\mathcal{OS}$. This is a part of the data pre-processing.

Stage 1. Novel Class detection (NCD): In this sub-module (Algorithm 3), we aim at finding all the new classes, N_{new}. On the OOD samples ($\mathcal{OS}$), obtained in the previous sub-module (Algorithm 2), we do clustering using K-Means. We start the algorithm with $K = 1$ and number of new classes, N_{new} = 0. We perform - (i) K-Means clustering. (ii) Annotate x points from each cluster, add those points to $\mathcal{L}$ and identify n' new classes. (iii) Increase new class count (N_{new}+ n'). (iv) Double the number of cluster count (we compare N_{new} with $K/2$). We execute the above steps until cluster count exceeds the new intent count. The algorithm returns current annotations ($\mathcal{L}$), newly discovered class count (N_{new}) and newly formed clusters ($\mathcal{CL}$). The budget spent in this step is B_1.

Stage 2. Cluster Quality Based Annotation (CQBA): In this step (Algorithm 4), we evaluate the quality of each of the clusters obtained by the previous algorithm. We annotate p points from each of these clusters and if all the p points belong to the same class, we term it as a *good cluster* or else a *bad cluster*. An example of a *bad cluster* in BANKING would be the one containing data points from multiple classes, which may have high similarity, such as: *declined_cash_withdrawal* and *pending_cash_withdrawal*. For the *bad clusters*, we annotate q more points. All these annotated points are then added to the labelled data, $\mathcal{L}$. The budget spent in this step is B_2. Hence the remaining budget B - (B_1 + B_2) is used in the next step.

Stage 3. Post Processing Annotation Strategy (PPAS): In this step (Algorithm 5), we add more data to the labelled set, $\mathcal{L}$, through gold annotation (*gold strategy*), as well as silver-annotated data (*silver strategy*). To select these data points, we first train a classifier ($\mathcal{M}$) with the labelled set, $\mathcal{L}$ as obtained in the last step (CBQA), and consider the clusters $\mathcal{CL}$. We predict on the remaining points of the clusters to get the confidence of the datapoints. We perform silver strategy based on confidence score (CS) and gold strategy in round-robin way to operate on each cluster one after another.

Gold Strategy: Least confident data-points are annotated from the bad clusters (if present) or else from the good clusters. Gold strategy is performed in a round-robin way to retrieve data points with the least score for each cluster until our budget exhausts.

Silver strategy: If the confidence score (CS) of a datapoint from a cluster is greater than a predefined threshold ($\mathcal{TH}$), we measure the average cosine similarity of points annotated within that cluster with this point. If similarity is above a predefined threshold (τ), we label this point with class label of that cluster. The predefined threshold (τ) is required to choose good samples selectively instead of choosing all the points. Silver strategy does not require human intervention therefore there is no extra addition to the annotation cost, but the multiple conditions are checked to prevent noise in the training set.

Final Step: We again train the neural model $\mathcal{M}$ on $\mathcal{L}$ and test on $\mathcal{T}$ to find out Accuracy and F1.

4 Dataset and Pre-Processing

We perform our experiments on a variety of datasets, which are widely used as benchmarks for Natural Language Understanding tasks. The datasets are SNIPS (Coucke et al., 2018), ATIS (Tur et al., 2010), HWU (Liu et al., 2019a), BANKING (Casanueva et al., 2020) and CLINC (Larson et al., 2019). SNIPS (7) and ATIS (21) are smaller datasets consisting of less number of intents (in bracket) where HWU (64), BANKING (77) and CLINC (150) are larger datasets with many intents. ATIS is the most imbalanced, skewed dataset. In BANKING data - several intents are highly similar among themselves. The detailed statistics of these datasets including our experimental framework are shown in Table 1. Since the datasets are already fully labelled, annotation essentially means utilizing the already available labels. Hence, we

Dataset (W)	# Intent Class ($\|N\|$)	Dataset Size ($\|W\|$)	#Labelled $\|D_{init}\|$	#Unlab ($\|D\|$ - $\|D_{init}\|$)			#Test ($\|T\|$))
				#IND	#OOD	#Total	
SNIPS	7 (5+2)	14484	50	8601	3449	12050	2384
ATIS	21 (13+8)	5871	130	3155	1586	4741	1000
HWU*	64 (10+54)	11036	100	1408	8452	9860	1076
BANKING*	77 (10+67)	13083	100	1026	8877	9903	3080
CLINC*	150 (10+140)	22500	100	1100	16800	17900	4500

Table 1: Statistics based on our split for five Datasets. * represents pre-defined train-test splits. In # Intent Class, (- + -) represents (known + unknown) intents

do not have to deal with usual issues of annotation accuracy, inter-annotator agreement, etc.

Data Pre-Processing

Dataset	DOC		MSP		LMCL		FS-OOD	
	A	F1	A	F1	A	F1	A	F1
SNIPS	**77.3**	72.1	78.2	71.7	74.7	69.3	76.8	**72.9**
ATIS	55.8	47.2	56.1	44.7	54.5	40.6	**74.9**	**68.6**
HWU	61.4	57.2	59.9	29.9	53.1	44.3	**68.2**	**64.1**
BANKING	56.3	20.4	52.5	20.2	52.9	51.3	**73.7**	**64.1**
CLINC	54.8	18.7	53.4	20.5	54.1	59.9	**77.7**	**65.7**

Table 2: Accuracy (A) and F1-Score in (%) of various OODD algorithms to detect OOD points from different datasets. **Bold** denotes the best for each dataset.

In the pre-processing step, we filter the out-of-domain samples. We consider four algorithms for detecting out-of-domain samples. i) Softmax Prediction Probability (MSP) (Hendrycks and Gimpel, 2018) predicts out-of-domain samples based on a threshold on the softmax prediction scores. ii) Deep Open Classification (DOC) (Shu et al., 2017) method builds a multi-class classifier with one vs rest layer of sigmoids. iii) Large Margin Cosine Loss (LMCL) (Lin and Xu, 2019) trains a network with margin loss and predictions are then fed into an algorithm called Local Outlier Factor (LOF) for outlier detection. iv) Few-shot OOD (FS-OOD) (Tan et al., 2019) uses a Proto-Typical Network to detect OOD examples and classifying in-domain examples with few-shot examples from the in-domain class. We fine-tune BERT embeddings using all these out of domain sample detection algorithms. We use bert-base-uncased for these methods. We set the threshold for MSP as 0.5 as in Lin and Xu (2019), Xu et al. (2020). The results of all these algorithms are shown in Table 2. FS-OOD (Tan et al., 2019) provides us the best accuracy and F1 for detecting OOD samples ($\mathcal{OS}$). Only DOC performs better in case of SNIPS but overall FS-OOD outperforms other approaches so we use FS-OOD produced out-of-sample data.[2]

[2] FS-OOD: https://github.com/SLAD-ml/few-shot-ood and other OOD models: https://huggingface.co

5 Experimental Setup

The efficacy of the algorithm needs to be tested on two aspects. (a). The number of unknown intents identified. (b). The accuracy achieved when the data is annotated by our algorithm, MNID. To test the accuracy, we use state-of-the-art several classification algorithms used for intent detection.

Different Neural Models: We explore different neural models to evaluate **MNID** as discussed next:

1. IFSTC (Xia et al., 2021): This finetunes a trained model on few shot data of new classes using an entailment and hybrid strategy. We use the hybrid strategy (best performing in their case).

2. PolyAI (Casanueva et al., 2020): It performs intent classification task based on dual sentence encoders - Universal Sentence Encoders (USE) (Cer et al., 2018) and ConveRT. Since authors have taken down the ConveRT model, we apply USE only. [3]

Along with the above two, we also consider other standard models, **3. BERT ('bert-base-uncased')** (Devlin et al., 2019) and **4. RoBERTa ('roberta-base-uncased')** (Liu et al., 2019b) for evaluation on these datasets. We finetune these pre-trained language models for 15 epochs for the smaller datasets (SNIPS, ATIS) 50 epochs for the larger datasets (HWU, BANKING, CLINC) and with a learning rate of 2e-05 and Adam optimizer[4]. Early stopping was employed to stop training. For all methods, we provide the same number of gold annotated data obtained using our pipeline and report its performance.

Baselines: We compare the performance of our method using two annotation techniques for choosing $B - \|D_{init}\|$ data points: 1) Gl_F: This is the ideal scenario where we are given F (=10) data points for each of the new classes - $Gold_{Few}$, abbreviated as 'Gl_F'. 2) Rn_F : Here, we randomly choose $B - \|D_{init}\|$ data points from the unlabelled data - $Random_{Few}$, abbreviated as 'Rn_F'.

Clustering Algorithms: One of the building blocks of **MNID** is to cluster datapoints, so the efficacy of **MNID** depends on employing an efficient clustering algorithm. We do a detailed study by employing several unsupervised and semi-supervised clustering algorithms and choose the best.

The unsupervised algorithms are: (i) **K-Means (KM)** (MacQueen et al., 1967) (ii) **Agglomera-**

[3] We use author's implementation of IFSTC (PyTorch) and re-implement PolyAI (Tensorflow)

[4] We use https://huggingface.co/

Method		IFSTC (Gl_F, Rn_F, MNID)	PolyAI (Gl_F, Rn_F, MNID)	BERT (Gl_F, Rn_F, MNID)	RoBERTa (Gl_F, Rn_F, MNID)
SNIPS	A	**85.4**, 78.1, 84.7	93.2, 85.7, **95.1**	92.7, 91.6, **93.3**	94.9, 92.3, **95.3**
	F1	**84.2**, 79.4, **84.2**	93.2, 84.3, **94.9**	92.6, 91.9, **93.9**	**94.8**, 91.9, **94.8**
ATIS	A	88.4, 70.1, **88.8**	87.8, 71.8, **88.6**	88.1, 70.2, **88.2**	87.9, 70.8, **88.6**
	F1	87.3, 65.8, **87.8**	84.3, 74.5, **87.0**	86.3, 73.9, **86.9**	84.6, 74.1, **85.1**
HWU	A	78.2, 72.4, **79.7**	83.8, 75.2, **83.8**	82.6, 73.6, **82.7**	82.5, 75.3, **83.7**
	F1	76.4, 71.4, **78.4**	83.7, 77.3, **84.2**	81.7, 74.3, **82.4**	81.3, 77.2, **82.4**
BANKING	A	78.3, 72.8, **79.0**	84.2, 79.0, **84.7**	80.1, 75.5, **82.8**	83.4, 77.0, **84.5**
	F1	77.7, 74.1, **80.0**	83.1, 79.0, **84.4**	80.0, 76.4, **83.7**	**83.9**, 78.7, 83.8
CLINC	A	88.7, 77.1, **88.9**	92.1*, 83.2, **94.9**	90.8, 77.6, **91.4**	91.3, 84.5, **92.3**
	F1	85.7, 76.4, **88.3**	93.5, 83.7, **95.2**	90.7, 78.8, **91.0**	91.7, 85.3, **92.8**

Table 3: Overall Accuracy (A) and Macro F1 in (%) across all datasets for different scenarios - ideal (Gl_F), random (Rn_F) and MNID (The best outcomes among three scenarios in **Bold**). *Casanueva et al. (2020) report accuracy of 90.15 with OOS and 92.14 without OOS.

tive Clustering (AG) (Gowda and Krishna, 1978) (iii) **Deep Clustering Network (DCN)** (Yang et al., 2017) and (iv) **Deep Embedded Clustering (DEC)** (Xie et al., 2016) which uses the stacked auto-encoder based reconstruction loss. The semi-supervised algorithms are: (i) **DeepAligned (DAL)** (Zhang et al., 2021) which uses limited data for pre-training and cluster assignments as pseudo labels for cluster refinement. (ii) **DTC** (Han et al., 2019) develops on the DCN algorithm by scaling it to the transfer learning setting and can estimate the number of known classes in unlabelled data. It is however highly dependent on availability of labelled data (iii) **KCL** (Hsu et al., 2017) which transfers the knowledge to target dataset considering KL-divergence based distance loss (iv) **MCL** (Hsu et al., 2019) which uses meta-classification based likelihood criterion for pairwise similarity evaluation (v) **CDAC+** (Lin et al., 2020) which uses prior data to refine the clustering process and KL-divergence based loss [5].

Other than KM and AG, all the other unsupervised methods along with some of the semi-supervised methods such as DTC and CDAC need the information of the ground truth number of clusters for training and we provide them so (it is an extra advantage for them). For semi-supervised methods such as KCL, MCL, DTC and DAL, we start with double the number of ground truth clusters and let the method determine the number of clusters.

Hyper-parameters and Settings: For Post-Processing annotation strategy of MNID, we set the cosine similarity threshold, τ as 0.8 and the confidence threshold, $\mathcal{TH}$ as 0.5 [6]. For all datasets, we use a setting similar to κ-shot with κ = 10. For N intents, we define our total budget $B = \kappa \times N$. We use same budget for all our experiments. We experiment on NVIDIA Tesla K40m GPU with 12 GB RAM, 6 Gbps clock cycle and GDDR5 memory. All the methods took less than 8 GPU hours for training.

6 Experimental Results

In this section, we discuss the experimental outcomes for MNID and competing baselines. We also show results of different clustering algorithms and variations of distinct components of MNID.

(A) Class Discovery: MNID is very effective in identifying almost all new intents. For HWU, BANKING and CLINC, 54 out of 54 (100%), 66 out of 67 (98.5%) and 139 out of 140 (99.3%) new intents from the unknown intent set were discovered, respectively. For SNIPS and ATIS, we could discover 2 out of 2 (100%) and 7 out of 8 (87.5%) new intent classes, respectively. Due to data skewness (ATIS) and high intent similarity (BANKING, CLINC) MNID misses one intent.

(B) Performance of MNID: Table 3 shows the performance of different models - IFSTC, PolyAI, BERT and RoBERTa when trained with datasets provided by MNID. In order to maintain the fairness, MNID, Rn_F and Gl_F use the (overall) same number of gold-annotated data points. Besides MNID uses silver-annotated data points, while the others do not have any way of creating high quality silver annotated data. Each cell in the table consists of values from Gl_F, Rn_F and MNID. As expected, Rn_F performs the worst across all settings. However, except two scenarios, we observe that MNID consistently performs better than the Gl_F dataset. For all these four different settings across five datasets, MNID improvements over Gl_F predictions are statistically significant ($p < 0.05$) as

[5] Code: https://github.com/thuiar/TEXTOIR

[6] This combination of τ and $\mathcal{TH}$ provides the best results among different experimented results.

per McNemar's Test. It is observed that our approach also works well on the highly imbalanced ATIS dataset in which some of the classes have less than 10 data points and highly similar BANKING dataset in which the intents are closely related eg., 'top-up-reverted' and 'top-up-failed'. This is because although Gl_F chooses uniformly across all classes, MNID selectively labels datapoints having high uncertainty thus providing the classifier with the right ingredient to perform better. In IFSTC on SNIPS dataset, MNID underperforms as compared to Gl_F but with a very small margin. This happens because in the case of SNIPS dataset, the number of new classes is very less, hence Gl_F can choose ideal candidates. The best performance of MNID as well as the two baselines is in the PolyAI setting when it is used with Universal Sentence Encoders (USE). Since PolyAI performs the best, all our subsequent results are provided on PolyAI (USE).

(C) Distribution of gold annotated points: Fig 2 shows the count of the gold annotated points ($Y - axis$) for new classes (class indices on $X - axis$). The dotted line is at the frequency of 10, corresponding to the average annotations per class. For 76.2% (HWU), 81.5% (BANKING) and 67.3% (CLINC) classes in good clusters require '$\leq$ 10' annotations. More than 10 annotations are needed for 65.4% (HWU), 68.5% (BANKING) and 54.5% (CLINC) classes in bad clusters.

(D) Budgets: For novel intent class discovery, a minimum number of human annotation is necessary. For NCD to work, at least 4 shot, 6-shot and 7-shot annotations are required for HWU, BANKING and CLINC datasets respectively.

Different Variations of MNID

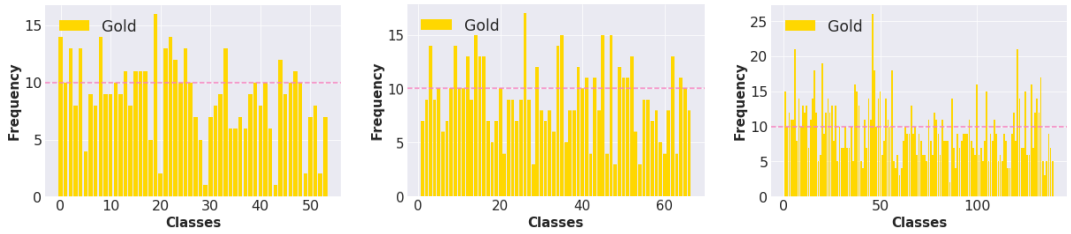
MNID consists of three steps (a). novel class detection (NCD), (b). cluster quality based annotation (CQBA) and (c). post-processing annotation strategy (PPAS). In each of these steps, certain parameters can be varied. We systematically discuss the impact of these parameters on MNID performance.

Variations at NCD

(a) Performance of Clustering Algorithms: We explore different unsupervised and semi-supervised clustering algorithms in our **MNID** framework. Overall accuracy and F1-Score for open intent discovery by different approaches are shown in Table 4. From Table 4, it is seen that unsupervised approaches perform better than semi-supervised models. The semi-supervised techniques get biased by the initial seed and fail to discover diverse clusters needed to detect all the new intent classes. K-Means (KM) performs the best across all datasets in terms of accuracy and F1 score except for HWU dataset where DEC and DTC (F1 only) outperforms it. This is most probably due to its robustness and absence of any outlier in the dataset. So we use K-Means as the clustering algorithm for MNID.

(b) Class Discovery with number of clusters: From Fig 3a, we observe an increasing trend in the number of classes discovered with increasing number of clusters which show that classes get evenly distributed across clusters as the number of clusters increases. The rate at which new classes are discovered is linear with the new clusters until significant classes are detected. The horizontal lines represent the gold number of new intents.

(c) Effect of number of points (x) used in clustering: Fig 3b shows that the accuracy on all datasets drops as we increase the number of points used for new class discovery in clustering beyond $x = 2$. This is because most of the budget gets exhausted while clustering and we have a very small budget to annotate low-confidence points in the next steps. Note that at least two points from a cluster need to be annotated for new class discovery.



(a) HWU (b) BANKING (c) CLINC

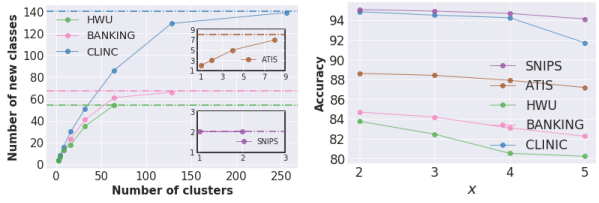
Figure 2: Count of gold annotated points for newly discovered classes

Dataset	Unsupervised Clustering Algorithms								Semi-Supervised Clustering Algorithms									
	KM		AG		DCN		DEC		DAL		DTC		KCL		MCL		CDAC+	
	A	F1	A	F1	A	F1	A	F1	A	F1	A	F1	A	F1	A	F1	A	F1
SNIPS	**95.1**	**94.9**	92.7	92.9	89.2	88.7	89.6	88.2	92.2	92.2	87.6	87.2	73.3	70.4	78.2	74.1	80.4	79.2
ATIS	**88.6**	**87.0**	85.8	86.4	77.7	79.78	83.1	85.42	86.9	87.0	84.3	85.9	76.7	80.8	80.4	83.2	77.9	81.6
HWU	83.8	84.2	83.2	83.3	84.1	83.6	**84.7**	84.4	83.7	82.6	83.6	**85.2**	73.3	74.1	78.1	74.8	83.1	81.1
BANKING	**84.7**	**84.4**	84.2	84.1	80.1	83.2	80.1	80.5	80.5	81.1	79.9	78.2	71.8	72.4	74.2	73.1	83.4	82.6
CLINC	**94.9**	**95.2**	93.9	94.8	93.4	94.2	93.4	94.9	93.9	92.6	93.9	93.2	83.4	84.5	81.0	82.3	92.1	92.5

Table 4: Accuracy (A) and F1-Score (F1) in (%) for various Open Intent Discovery Based Clustering Algorithms across all datasets. The best results for each dataset in **Bold**.

p, q	2, 1	2, 2	2, 3	3, 0	3, 1	**3, 2**	4, 1
HWU	82.1	81.2	82.5	81.8	83.7	**83.8**	83.2
BANKING	84.1	83.0	84.2	82.9	84.0	**84.7**	84.1
CLINC	94.8	93.9	94.2	92.7	93.0	**94.9**	94.2

Table 6: Accuracy (%) based on point selections from Good and Bad clusters



(a) Class discovery with number of clusters

(b) Accuracy vs points annotated (x) for clustering

Figure 3: Variations of NCD

Variations at CQBA

(a) Effect of number of points selected from Good and Bad clusters: We experiment with different values of point selection (p, q) for the module CQBA (Algo 4) and observe how accuracy changes for three larger datasets - HWU, BANKING and CLINC. We get the best accuracy for (p, q) = (3, 2) i.e 3 (p) points from good cluster and 5 ($p + q$) points from bad cluster as shown in Table 6. Since, we perform gold annotation strategy on the bad clusters, a higher number of point selection is required to identify classes.

(b) Distribution of good and bad clusters: For CLINC, BANKING and HWU we obtain 256, 128 and 64 clusters respectively by NCD. The percentage of good clusters obtained for CLINC, BANKING and HWU are 70.70% (181 out of 256), 46.88% (60 out of 128) 56.25 % (36 out of 64), respectively. For BANKING, since the entire dataset is from a single domain with multiple intents being similar among themselves, we obtain more bad clusters than the good clusters. For SNIPS and ATIS, however, all the clusters are good clusters.

Variations at PPAS

(a) Different Variations of Gold and Silver Strategies: The results for different variations of MNID methods (based on Silver and Gold Strategy applications) for all the datasets are provided in Table 5. We observe that the best result is obtained on MNID-9, i.e., choosing high confidence points from the good clusters for silver strategy and low confidence points from the bad clusters (if detected or else from the good clusters) only for gold strategy. This strategy ensures that during silver annotation we choose points with high fidelity and side by side for gold annotation choose points with high uncertainty, both of which help in developing a highly accurate classifier. Silver strategy on

Method	Silver Strategy	Gold Strategy	SNIPS		ATIS		HWU		BANKING		CLINC	
			A	F1	A	F1	A	F1	A	F1	A	F1
MNID-1	Good Clusters†	✗	94.4	93.2	87.2	85.4	78.5	78.8	77.5	78.4	89.2	89.8
MNID-2	Good Clusters†	Any Point from Bad Clusters	94.4	93.2	87.2	85.4	80.9	80.9	79.3	80.0	90.8	90.7
MNID-3	✗	Low-Conf from Any Cluster	94.7	94.0	87.9	86.1	81.2	81.1	81.7	81.1	91.3	91.0
MNID-4	High-Conf from Good Clusters	✗	94.8	93.9	87.7	86.1	81.5	81.4	82.2	81.8	91.8	91.9
MNID-5	High-Conf from Good Clusters	Low-Conf from Any Cluster	**95.1**	**94.9**	**88.6**	**87.0**	82.9	82.2	82.7	81.8	92.1	93.5
MNID-6	Good Clusters†	Low-Conf from Bad Clusters	94.4	93.2	87.2	85.4	83.1	82.8	83.9	83.1	93.9	93.7
MNID-7	✗	Low-Conf from Bad Clusters*	94.7	94.0	87.9	86.1	81.9	81.6	83.0	82.4	92.8	92.7
MNID-8	Good Clusters†	Low-Conf from Bad Clusters*	94.9	94.4	88.2	86.4	83.1	82.8	83.9	83.1	93.9	93.7
MNID-9	High-Conf from Good Clusters	Low-Conf from Bad Clusters*	**95.1**	**94.9**	**88.6**	**87.0**	**83.8**	**83.2**	**84.7**	**84.4**	**94.9**	**95.2**

Table 5: Accuracy (A) and F1-score (F1) in (%) across all datasets for different variations of silver and gold strategy of MNID. [* - If no bad cluster exists then the strategy is applied on good clusters (SNIPS, ATIS have no bad cluster). Detailed in line 6-11 of Algorithm 5. † We use $\mathcal{TH} = 0$ in Algorithm 5. ✗: denotes we are not using this. **Bold** notifies the best for each dataset.]

Silver Strategy on	SNIPS	ATIS	HWU	BANKING	CLINC
All Clusters (%)	96.2	95.9	82.5	84.3	85.2
Good Clusters (%)	96.2	95.9	93.6	95.4	97.2
High-Conf from Good Clusters (%)	**97.8**	**96.2**	**95.6**	**97.2**	**98.1**
Average Points per Class	10.1	8.2	17.0	22.8	15.9

Table 7: Accuracy (in %) and usage of average number of datapoints per class in silver strategy

(a) Cosine similarity threshold (τ)

(b) Confidence score threshold ($\mathcal{TH}$)

Figure 4: Variations of PPAS

high confidence points from good cluster (7 vs 9) and gold strategy on low confidence points from bad cluster (4 vs 9) alone enhances ∼1- 3% accuracy and F1 for the three large datasets.MNID-9 corresponds to our proposed approach, MNID.

(b) Silver Strategy Analysis: We inspect silver strategy based on cosine similarity, confidence score and strategy accuracy.

(i) Effect of Cosine Similarity and Confidence Score (CS): We study the effect of cosine similarity of silver strategy for MNID. From Fig. 4a, we observe that the best results are always obtained using a higher threshold of 0.8 cosine similarity. In case of BANKING, HWU and ATIS accuracy drops at 0.9 whereas for other datasets it remains almost identical. Fig. 4b shows how accuracy varies for different confidence scores. We observe that for all the datasets the best results are obtained at a threshold of 0.5. This is because a lower threshold allows more diverse datapoints to be selected using cosine similarity and this in turn improves the model performance. In both the cases if the cosine similarity or the threshold is increased beyond the optimal point, that results in selection of too less datapoints which is not enough for the classifier to do a meaningful learning. Hence accuracy drops. So we choose the parameters - cosine similarity = 0.8 and τ = 0.5 - while choosing *high confidence* point to be annotated by silver strategy.

(ii) Strategy Accuracy: The accuracy of data point selection by silver strategy for different MNID variations is shown in Table 7. We see the strategy of choosing high-confidence points from good clusters produce points with high fidelity. Table 7 also shows the average number of points per class as selected by this strategy for various datasets. Here we see that enough number of silver points are annotated even after considering a very strict criterion. Note, average points per intent class count is the highest for BANKING because multiple intents are very similar to each other and hence more points qualify the cosine similarity threshold, τ.

.

7 Conclusion

We have developed MNID (**M**ultiple **N**ovel **I**ntent **D**etection), an end-to-end framework to identify multiple novel intents within a fixed annotation cost. The algorithm intelligently uses the concept of clusters to first discover the classes and then estimate the nature in which datapoints of a class is distributed, that is, whether the datapoints of a class congregate strongly within themselves and separate from other classes or are entangled with datapoints of other classes. In the two types of situations, we propose two different strategies, silver strategy to take advantage of the clusters so that we can annotate many points without any extra human cost and gold strategy to annotate highly uncertain points. This two-pronged approach helps us to annotate highly precise points automatically while annotating the most uncertain (with respect to the class it belongs) points using human assistance. We have done a very rigorous analysis/experimentation to establish the core idea of our algorithm. We observe that the accuracy of classifiers when fed with the dataset created by MNID can beat the standard best few-shot setting where it is assumed that 'κ' instances of each class are provided and annotated by human whereas in our case we have to first discover the classes and then have to find the instances of each class.

One limitation of MNID is that it is not able to detect intents where classes are very similar to each other. For example, the query "Can you explain why my payment is still pending?" in BANKING dataset is from the "pending transfer" category but our system detects as "pending card payment" intent as both intents are quite similar. We shall try to address this issue in future. We have presently worked on a setting where novel intents appear in one step, we would strive to extend this framework to explore the dynamics of periodically evolving intents.

References

Inigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. *arXiv preprint arXiv:2003.04807*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

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

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. 2020. Task-aware representation of sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213.

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*, pages 8401–8409.

Ahsanul Haque, Latifur Khan, and Michael Baron. 2016. Sand: Semi-supervised adaptive novel class detection and classification over data stream. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Dan Hendrycks and Kevin Gimpel. 2018. A baseline for detecting misclassified and out-of-distribution examples in neural networks.

Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. 2017. Learning to cluster in order to transfer across domains and tasks. *arXiv preprint arXiv:1711.10125*.

Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. 2019. Multi-class classification without multi-class labels. *arXiv preprint arXiv:1901.00544*.

Ilja Kuzborskij, Francesco Orabona, and Barbara Caputo. 2013. From n to n+ 1: Multiclass transfer incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3358–3365.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. 2019. An evaluation dataset for intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027*.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. *arXiv preprint arXiv:1906.00434*.

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

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019a. Benchmarking natural language understanding services for building conversational agents.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Mohammad Masud, Jing Gao, Latifur Khan, Jiawei Han, and Bhavani M Thuraisingham. 2010. Classification and novel class detection in concept-drifting data streams under time constraints. *IEEE Transactions on Knowledge and Data Engineering*, 23(6):859–874.

Xin Mu, Kai Ming Ting, and Zhi-Hua Zhou. 2017a. Classification under streaming emerging new classes: A solution using completely-random trees. *IEEE Transactions on Knowledge and Data Engineering*, 29(8):1605–1618.

Xin Mu, Feida Zhu, Juan Du, Ee-Peng Lim, and Zhi-Hua Zhou. 2017b. Streaming classification with emerging new class by class matrix sketching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. 2012. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772.

Lei Shu, Hu Xu, and Bing Liu. 2017. Doc: Deep open classification of text documents.

AB Siddique, Fuad Jamour, Luxun Xu, and Vagelis Hristidis. 2021. Generalized zero-shot intent detection via commonsense knowledge. In *Proceedings of the 44th International ACM SIGIR Conference on*

Research and Development in Information Retrieval, pages 1925–1929.

Yu Sun, Ke Tang, Leandro L Minku, Shuo Wang, and Xin Yao. 2016. Online ensemble learning of data streams with gradually evolved classes. *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1532–1545.

Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. 2019. Out-of-domain detection for low-resource text classification tasks. *arXiv preprint arXiv:1909.05357*.

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

Min Wang, Ke Fu, Fan Min, and Xiuyi Jia. 2020. Active learning through label error statistical methods. *Knowledge-Based Systems*, 189:105140.

Congying Xia, Wenpeng Yin, Yihao Feng, and Philip Yu. 2021. Incremental few-shot text classification with multi-round new classes: Formulation, dataset and system. *arXiv preprint arXiv:2104.11882*.

Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip S Yu. 2018. Zero-shot user intent detection via capsule neural networks. *arXiv preprint arXiv:1809.00385*.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR.

Hong Xu, Keqing He, Yuanmeng Yan, Sihong Liu, Zijun Liu, and Weiran Xu. 2020. A deep generative distance-based classifier for out-of-domain detection with mahalanobis space. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1452–1460.

Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*, pages 3861–3870. PMLR.

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

Zhi-Hua Zhou and Zhao-Qian Chen. 2002. Hybrid decision tree. *Knowledge-based systems*, 15(8):515–528.