Intent Focused Semantic Parsing and Zero-Shot Learning for Out-of-Domain Detection in Spoken Language Understanding

NIRAJ KUMAR AND BHIMAN KUMAR BAGHEL
Samsung Research Institute, Bengaluru 560036, India
Corresponding author: Niraj Kumar (niraj.kumar@samsung.com)

ABSTRACT In Spoken Language Understanding (SLU), the ability to detect out-of-domain (OOD) input dialog plays a very important role (e.g., voice assistance and chatbot systems). However, most of the existing OOD detection methods rely heavily on manually labeled OOD data. Manual labeling of the OOD data for a dynamically changing and evolving area is time-consuming and not immediately possible. It limits the feasibility of these models in practical applications. So, to solve this problem, we are considering the scenario of having no OOD labeled data (i.e., zero-shot learning). To achieve this goal, we have used the intent focused semantic parsing, extracted with the help of Transformer-based techniques [e.g., BERT (Devlin et al., 2018)]. The two main components of the intent-focused semantic parsing are - (a) the sentence-level intents and (b) token-level intent classes, which show the relation of slot tokens with intent classes. Finally, we combine both information and use a One Class Neural Network (OC-NN) based zero-shot classifier. Our devised system has shown better results compared to the state-of-the-art on four publicly available datasets.

INDEX TERMS Spoken language understanding, out-of-domain (OOD) detection, zero-shot learning, intent focused semantic parsing.

I. INTRODUCTION
Most of the voice assistant systems and chatbot systems provide supports for a fixed number of domains. We consider it as in-domain (ID) texts/utterances. The intent-slot-based systems smoothly run on the ID texts/utterances. However, the presence of the out-of-domain texts badly affects the quality of the system. Thus, the efficiency of voice assistance systems directly depends upon ID and OOD text classifiers. According to [18], correctly identifying out-of-scope cases is crucial to avoid performing the wrong action by SLU systems. The following examples and research directions will be helpful in understanding the research problems in this area.

1) SUPERVISED APPROACH, FEW-SHOT AND ZERO-SHOT STRATEGIES
The traditional supervised state-of-the-art OOD and ID deep learning (DL) classifiers often require a massive amount of ID and OOD labeled data (e.g., [8], [9], and [10]). In reality, many applications contain limited ID labeled data without having OOD labeled data. Collecting the OOD data is highly labor-intensive work, and there are several chances that it may not cover all the areas of OOD data.

However, the main advantages of such strategies are - (a) stability, (b) robustness, and (c) the absence of data-specific thresholds (like we use in unsupervised strategies).

To solve the shortcomings of supervised approaches for OOD detection, [20] suggested the use of ‘few-shot learning’ (to support applications having very limited OOD labeled data) and ‘zero-shot learning’ (to support applications having no OOD labeled data).

2) UNSUPERVISED OOD DETECTION STRATEGIES
Some unsupervised OOD detection algorithms, like- [21], and [22], use a threshold on the ID classifier’s probability estimate. Similarly, [23] and [24] propose a likelihood ratio method to effectively correct the confounding background statistics. Reference [17] employ an unsupervised density-based novelty detection algorithm and local outlier factor (LOF) to detect unseen intents.

Reference [18], presents a study on Softmax-classifier-based OOD detection techniques. According to them, deep...
neural networks with the Softmax classifiers are known to produce highly overconfident posterior distributions even for such abnormal OOD samples. These techniques highly depend upon the effectiveness of the selected thresholds. It finally decreases the performance of the system.

Reference [25] conducts a thorough comparison of out-of-domain intent detection methods. The main findings show that Mahalanobis distance, together with utterance representations, derived from Transformer-based encoders, shows better performance on state-of-the-art datasets. But like other unsupervised strategies, the performance depends upon the threshold. In a similar effort, the authors of [18] have proposed a strong generative distance-based classifier. It estimates the class-conditional distribution on feature spaces of Deep Neural Networks (DNNs) via Gaussian discriminant analysis (GDA) to avoid over-confidence problems. Finally, it uses two distance functions, Euclidean and Mahalanobis distances, to measure the confidence score of whether a test sample belongs to OOD. Reference [18], also proves that, the performance with Mahalanobis distances is better. However, all such unsupervised algorithms depend upon some thresholds.

Based on the above-discussed facts, we believe that an end-to-end supervised ‘zero-shot learning’ system will be a good option for OOD detection. Through this, we will be able to exclude our dependencies on both, i.e., (a) OOD data and (b) threshold to differentiate between OOD and ID.

A. OUR APPROACH - MOTIVATION

We have introduced the intent focused semantic parsing of text (see Fig. 1) for OOD detection. This semantic parsing captures the connection of important terms (e.g., slots) with the corresponding intent classes and the connection of other terms (i.e., non-slot terms) with an additional non-intent class. In addition to this, we also collect the sentence-level intent information. This strategy of capturing combined intent-focused information at the sentence-level and at token-level gives a better picture of the utterance’s intent. This shows better performance in classifying between intent level ID vs. OOD classes. Next, to achieve the target of zero-shot classification, we use OC-NN. This OC-NN architecture uses the intent-focused semantic information from the text in classifying ID Vs. Out-domain utterances. The following contains a detailed explanation, to show, how Intent focused combined semantic parsing works. Finally, in the next subsection, we discuss our contributions.

1) CASE-1: ANALYSING THE USE OF SENTENCE-LEVEL INTENTS

Mostly all the conversational AI systems provide supports for a finite set of tasks (i.e., ID tasks supported by a finite set of intents). In the traditional conversational AI systems, intents play a major in deciding, whether the given utterance will be ID (i.e., supported by the current system) or OOD (i.e., not supported by the current system). However, the complete dependency on sentence-level intents may give false-positive results.

Most of the deep learning system shows bias towards verbs or other terms, which occurs frequently and plays a key role in deciding the intent classes. For example, suppose we train our system for air-line travel ticket booking domain and use the statements like – ‘Please book a Friday Evening ticket for flight from San Francisco to New York (see Table 1, Case-1)’. In this case, the system trained on the ID dataset confuses and gives ‘Flight_Booking’ as an intent class with a little lower confidence score. Here, due to the presence of common
words like – ‘book’, and ‘ticket’, the system confuses with the nearest matching ID intent class ‘Flight_Booking’.

As, we cannot always control the range of threshold values to differentiate between ID and OOD classes, so it affects the quality of the result. Our results (see ‘Additional Baseline-3’) also reflect this point. However, in some other cases, where the utterances have lower similarities, this strategy works well.

2) CASE-2: BENEFITS OF USING TOKEN-LEVEL INTENTS

We took the concept of token-level intents from [44]. In this work, authors have used a fine-grain representation of intents and slots, which works at the token level. This representation helps in getting the relationship between the slots and the corresponding intents in the multi-intent environment. To achieve this, the authors of [44] have assigned the intent class labels to tokens, which belong to the corresponding slot classes. For this, the system uses the semantic and contextual information given in the text. It also assigns the other label (i.e., Non-Intent label) to all non-slot tokens.

Thus, the token-level intent information contains information about a slot also. We find this information very useful in identifying the OOD class texts. Our experimental results (see Additional Baseline-4), also supports this fact. To test the behavior of the token-level intent annotation, we have made the following observation. We train the system by using the ID dataset and tested its predictions on unseen ID and OOD utterances. The following points contain the summary of the entire observation.

1) For utterance related to OOD: This summarizes the behavior of the token-level intent classifier (trained on the ID dataset) with OOD utterances. In most cases, the trained model gives – ‘O’ (other or non-intent class) for all the tokens in the text related to the ‘OOD’ class. However, a few times it misclassifies and gives ID-class labels to a few tokens of OOD class utterance/text (but with a very low Softmax probability). A system can use these probabilities and learn to differentiate between ID and OOD classes.

2) For utterances related to ID. This summarizes the behavior of the token-level intent classifier on unseen ID utterances. We use the model trained on the ID dataset to observe the token-level predictions on unseen ID utterances. The trained model correctly classifies tokens into intent and other classes and gives high Softmax scores for them. Thus, it effectively helps the classifier to differentiate between ID and OOD class utterances.

Thus, these points give sufficient token-level information to differentiate between ID and OOD class texts. The following contains an example to demonstrate the above observations.

a: EXAMPLE

This example shows the prediction analysis of a token-level intent and slot classifier trained on the ID dataset (see TABLE 1). This table presents three cases, i.e., ‘Case-1’, ‘Case-2’, and ‘Case-3’. These cases show the token-level intent and slot class outputs by the token-level intent classifier (trained on the ID dataset with intent class ‘Flight Booking’ from ATIS dataset, by using [44]). In this example, the utterance given in Case-1 belongs to the ID class. Similarly, utterances given in Case-2 and Case-3 belong to the OOD class for the current trained system.

b: ANALYSIS WITH OOD UTTERANCES

From details, given in TABLE 1, it is clear that the token-level intent output for ‘Case-2’ text”, shows only ‘O’ (other/Non-Intent Classes) for all tokens. Similarly, the token-level intent output for ‘Case-3’ text”, shows just two token-level intent ID-Classes (given in different colors) but with low Softmax probability scores. The system has detected the ‘O’ (other/Non-Intent) classes for the rest of the tokens. In this table we only show the maximum Softmax probability outputs with the corresponding label for each of the tokens.

c: ANALYSIS SUMMARY

This example demonstrates how the token-level intent class Softmax probability provides good insight, helpful in differentiating between ID and OOD type texts.

3) CASE-3: BENEFITS OF COMBINING SENTENCE-LEVEL AND TOKEN-LEVEL INTENTS

From the above discussion, it is clear that sentence-level intents give the idea of action, but token-level intents give the idea of the object on which the action is performed. Combining both features and applying an OC-NN type deep learning system, learns to effectively differentiate between ID and OOD utterances.

For this, we concatenate the confidence matrices obtained from both classifiers (i.e., sentence-level intent classifier and token-level intent classifier). Next, OC-NN learns from this matrix and classifies the given text into ID and OOD.

Thus, the system does not require any pre-fixed thresholds. Our experimental results also prove that combining the information from both types of intents helps in differentiating between ID and OOD classes.

B. CONTRIBUTION

Based on the above discussion, we have made the following contributions:

1) We have introduced the use of intent-focused semantic parsing of text/utterances for OOD detection. The intent-focused semantic parsing contains sentence and token-level intent information.

2) We have introduced the use of OC-NN based zero-shot OOD classifier. It uses the semantic parsing score to learn to differentiate between ID and OOD classes. The experimental results also support our way of detecting OOD utterances.
TABLE 1. Use of token-level intents in OOD detection.

| Ideal Intent Slot output, when all intent classes are used in the training |
|---|
| 1 Case-1: sentence-level Intent Class: [Flight_Booking]  |
| Text: Please book a Friday Evening ticket from San Francisco to New York.  |
| Token Level Slot: Please[O:0.99] book[O:0.92] a[O:0.99] Friday[Day_B: 0.98] Evening[Time_B: 0.98] flight[O:0.79] ticket[O:0.98] from[O:0.99] San[From_Location_B: 0.99] Francisco[From Location_J: 0.99] to New[To_Location_B: 0.99] York[To Location_I:0.97].  |
| 2 Case-2: sentence-level Intent Class: [Movie_Ticket_Booking]  |
| Text: Please book a Friday Evening ticket for movie show at Empirical Theater.  |
| Token Level Slot: Please[O:0.99] book[O:0.93] a[O:0.99] Friday[Day_B: 0.98] Evening[Time_B: 0.98] ticket[O:0.97] for[O:0.99] movie[O:0.99] show[O:0.99] at[O:0.99] Empirical[Location_Location_B:0.99] Theater[Location_I:1.00].  |
| 3 Case-3: sentence-level Intent: [Flight_Listing]  |
| Text: Show me all Friday Evening flight from San Francisco to New York.  |
| Token Level Slot: Show[O:0.99] me[O:0.99] all[O:0.99] Friday[Day_B: 0.98] Evening[Time_B: 0.98] flight[O:0.99] from[O:0.99] San[From_Location_B: 0.99] Francisco[From Location_J: 0.99] to[O:0.99] New[To_Location_B: 0.99] York[To Location_I:0.99].  |

II. RELATED WORK

The presence or absence of the labeled OOD data plays a very important role in differentiating between ID and OOD. Generally, authors classify their approaches into supervised (with sufficient labeled OOD data, “Few-shot”-having few OOD data, and “zero-shot” – having no OOD data) and unsupervised OOD detection. Other than, such approaches, one-class classifier based approaches are also used by several researchers. The following contains the summary of related works and key technologies applied by them.

Supervised OOD detection [8]–[12], and [13] represents that there are extensive labeled OOD samples in the training data. However, authors of [20], have proposed few-shot and zero-shot learning based text classification tasks, but authors have used Amazon review data and SLU data (not publicly available).

The unsupervised OOD detection techniques like - [14], [15], [17], [18], [21]–[25] successfully reduced its dependencies on labeled OOD samples, however, dependency on thresholds are one of the key issue.

Some of the existing methods often formulate the OOD task as a one-class classification problem, then use appropriate methods to solve it (e.g., one-class SVM [38] and one-class DL-based classifiers [39]). A group of researchers also proposed an auto-encoder based approach and its variation to tackle OOD tasks [40]. However, the reported accuracy is low.

For unsupervised OOD detection, [21] and [22] use a threshold on the ID classifier’s probability estimate. References [23], and [24] proposes a likelihood ratio method to effectively correct the confounding background statistics. Reference [17] employ an unsupervised density-based novelty detection algorithm, local outlier factor (LOF) to detect unseen intents.

From the above discussion, it is clear that most of the techniques use supervised intent classifiers as base systems and then apply unsupervised OOD detection on top of it. Some of them have used embedding based features also. Different from these proposed approaches, we decided to apply a more sophisticated approach with zero-shot learning strategy.

III. METHODOLOGY

In this section, we describe our system architecture. First of all, we preprocess the data by correcting the case. Next, we use OC-NN based zero-shot OOD classifier to learn to differentiate between ID and OOD utterances. Our zero-shot OOD classifier uses intent-focused semantic information from the given text. The following contains the detailed architecture.

A. PREPROCESSING

We have used “truecase” [1] library to correct the case of the alphabet of the given text. For example, when we pass the given text “what are the flights from pittsburgh to denver?”

1 https://pypi.org/project/truecase/
from the ATIS dataset, ‘truecase’ library restores/corrects its case and returns the correct text like – “What are the flights from Pittsburgh to Denver?”. We restore the case of all texts from all the given datasets. Next, we use the corrected text for training and test operation. The following contains the details.

B. INTENT FOCUSED SEMANTIC PARSING & OC-NN BASED ZERO-SHOT OOD CLASSIFIER

This system has two parts (see Fig. 2). The first part of this system identifies ‘token + sentence’ level intent information from text (used intent focused semantic parsing to represent it). Next, we use the OC-NN based zero-shot learning system. This OC-NN system takes concatenated outputs obtained from sentence-level ‘multi-level’ intent classifier and token-level Softmax probability scores as input. For this, we modify the token-level Intent classifier [44] and also follow the token-level data annotation strategies (as discussed in [44]). We train it on the ID dataset. The following contains the details.

1) THE SENTENCE-LEVEL AND TOKEN-LEVEL INTENT CLASSIFIER

For this, we use the annotated data (token level intent annotation) from [44]. However, to extract sentence-level intents, we maintain the sentence-level intent classes also. Next, to jointly calculate the token-level and sentence-level intents, we modify the algorithm discussed in [44].

Traditionally, we can extract two types of outputs from the pre-trained BERT Model - (a) Sequence output of the shape (batch_size, sequence_length, ‘D’), and (b) The pooled output of the shape (batch_size, ‘D’) [4], [26], [28]. Here, the sequence output is a representation of each token, where pooled output encoded representation of the entire sentence. Hence we use sequence output for token-level intent extraction and pooled output for multi-level intent classification. Here ‘D’ is the dimension of the output. The ‘sequence_length’ is the count of the total number of tokens. Here ‘batch_size’ depends upon the performance tuning (i.e., size at which the model shows the best performance).

Let us represent utterance as input token sequence x = [w₁, w₂, ..., wᵢ, ..., w_N] where N is maximum sequence length (in the terms of count of maximum number of tokens) obtained after ‘trucasing’ (see section III-A). Let Hᵢ = [h₁, h₂, ..., hᵢ, ..., h_N], where hᵢ ∈ Rd is a D-dimensional vector for each of the token wᵢ obtained as BERT sequence output of last layer. Similarly, let H_p represents the pooled output of the last layer. Actually the token-level intent class annotation contains an ‘Other’ class also. Thus, K_I contains a total count of distinct intent classes and one ‘Other’ class.

a: TOKEN LEVEL INTENT CLASSIFICATION

We have used a TimeDistributedDense layer to get the Softmax probability for the token-level intents. This system generates the Softmax intent class output for each of the N words/tokens. The TimeDistributedDense layer used for this task uses the BERT sequence output Hᵢ as input and generates the following output for each token and map it to the output intent classes.

\[ y_i^t = \text{Softmax}(W^t H_i + b^t), \quad i = 1, \ldots, N \]  \hspace{1cm} (1)

where Hᵢ is the hidden state corresponding to the first sub-token of x. Here Wᵢ is the trainable weight matrix of the Dense layer and bᵢ is the associated bias. Each output (i.e., yᵢ^t) of the Softmax classification layer consists of K_I outputs, which indicates the probability of each of the K_I intent classes.

Thus, the trained model returns (N × K_I) intents for each of the utterances.

b: SENTENCE LEVEL INTENT CLASSIFICATION

For this, we apply the multi-level intent classification.

\[ y^j = \sigma(W^j H_p + b^j) \]  \hspace{1cm} (2)

where, y^j shows the ‘sigmoid’ σ output of jth intent, here, where j ∈ [1, (K_I - 1)]. Here, W^j is trainable weight matrix, H_p is the pooled output of the last layer and b^j is the associated bias. After model training, we resize the prediction output size to K_I by appending an element with value zero. The main aim is to resize the matrix to concatenate it with token-level intent output. Thus the trained model returns prediction intent output of size (1 × K_I) for each of the utterances.

c: JOINT CLASSIFIER TRAINING

Our system jointly learns for token-level and sentence-level intent prediction. Let yᵢ^t represents the sentence/utterance level multi-level intent classification task and yᵢ^S represents the sequence classification task for token-level intents. To jointly model both tasks, the learning objectives can be
The learning objective is to maximize the conditional probability \( P(y^f, y^S|x) \) for the given input token sequence \( x \). The model is fine-tuned end to end via minimizing the cross entropy loss.

As, we have a fixed and small training data so, to generate the input data for OC-NN, we use the following strategy.

\[ P(y^f, y^S|x) = P(y^f|x) \prod_{i=1}^{N} P(y_i^S|x) \quad (3) \]

The learning objective is to maximize the conditional probability \( P(y^f, y^S|x) \) for the given input token sequence \( x \). The model is fine-tuned end to end via minimizing the cross entropy loss.

We perform \( k' \)-fold cross-validation on the joint sentence and token-level Intent classifier and collect the cross-validated predicted values. In the current system, we use \( k' = 5 \). (Note: Scalability is the main reason behind the selection of 5-fold cross-validation, instead of using 10-fold or higher, same as discussed in [44]). In this way, each time, one-fold will be used for prediction and the rest of the four-folds will be used for training. Thus, we get sentence and token-level Softmax probability scores for the entire training data. After resizing the sentence-level intent output format from \( (1 \times (K_f - 1)) \) to \( (1 \times K_f) \) format, we concatenate it with token-level intent output. Thus the concatenated output for each of the utterances results in output format \( (N \times K_f) + (1 \times K_f) \). Here '+' is used to represent the concatenation operation.

\[ \text{d: GENERATING TRAINING DATA FOR OC-NN} \]

We perform \( k' \)-fold cross-validation on the joint sentence and token-level Intent classifier and collect the cross-validated predicted values. In the current system, we use \( k' = 5 \). (Note: Scalability is the main reason behind the selection of 5-fold cross-validation, instead of using 10-fold or higher, same as discussed in [44]). In this way, each time, one-fold will be used for prediction and the rest of the four-folds will be used for training. Thus, we get sentence and token-level Softmax probability scores for the entire training data. After resizing the sentence-level intent output format from \( (1 \times (K_f - 1)) \) to \( (1 \times K_f) \) format, we concatenate it with token-level intent output. Thus the concatenated output for each of the utterances results in output format \( (N \times K_f) + (1 \times K_f) \). Here '+' is used to represent the concatenation operation.

\[ \text{2) THE ONE-CLASS NETWORK (OC-NN)} \]

We have used the one-class Neural Network (OC-NN) model [45] for OOD detection. It uses a neural architecture using a One-Class SVM (OC-SVM) ([29], [46]) equivalent loss function. Using OC-NN, we are able to exploit and refine features obtained from 'token-level + sentence-level' intent classification for OOD detection. The OC-NN uses a simple feed-forward network with one hidden layer having linear or sigmoid activation and one output node. Generalizations to deeper architectures are straightforward. The OC-NN objective can be formulated as:

\[ \min_{w, V, r} \frac{1}{2} \|w\|^2 + \frac{1}{2} \|V\|^2 + \frac{1}{v} \cdot \frac{1}{N} \sum_{n=1}^{N} \max (0, r - \langle w, g(VX_n) \rangle) - r \quad (4) \]

where \( w \) is the scalar output obtained from the hidden to output layer. \( V \) is the weight matrix from input to the hidden unit. Thus the key insight of the paper is to replace the dot product \( \langle w, \Phi(X_n) \rangle \) in OC-SVM with the dot product \( \langle w, g(VX_n) \rangle \). This change makes it possible to leverage the joint intent output, obtained using a trained deep learning model and create an additional layer to refine the features for OOD detection.

To understand the use of \( \langle w, \Phi(X_n) \rangle \) in OC-SVM, we will go through, the OC-SVM. Intuitively in OC-SVM all the data points are considered as positively labelled instances and the origin as the only negative labelled instance. More specifically, given a training data \( X \), a set without any class information, and \( \Phi(X) \) a reproducing kernel Hilbert space (RKHS) map function [45] from the input space to the feature space \( F \). A hyper-plane or linear decision function \( f(X_n) \) in the feature space \( F \) is constructed as \( f(X_n) = w^T \Phi(X_n) - r \), to separate as many as possible of the mapped vectors \( \Phi(X_n), n = 1, 2, 3, \ldots, N \); from the origin. Here \( w \) is the norm, perpendicular to the hyperplane and \( r \) is the bias of the hyper-plane. In order to obtain \( w \) and \( r \), we need to solve the following optimization problem,

\[ \min_{w, r} \frac{1}{2} \|w\|^2 + \frac{1}{v} \cdot \frac{1}{N} \sum_{n=1}^{N} \max (0, r - \langle w, \Phi(X_n) \rangle) - r \quad (5) \]

The OC-NN, gives output in the range \([-\infty, +\infty]\).

\[ \text{IV. EXPERIMENTS} \]

We have used publically available datasets in all experiments. The following contains the model description, hyperparameter details, dataset description, and baseline systems. Next, we discuss our evaluation strategy and present the comparative results with other published results.

\[ \text{A. BASELINES FROM PAPERS} \]

Most of the supervised systems use heavy OOD utterances in the training dataset, while our system does not use OOD utterances in any stage of the training (due to the zero-shot framework). While on the other side, most of the unsupervised systems use the features from trained models and use unsupervised strategies on top of it to get the OOD utterances. We have introduced a new joint model to extract Sentence and token-level intents. Next, we applied OC-NN (i.e., deep-learning based one-class classifier, instead of using unsupervised strategies). Thus, we find such types of unsupervised systems as good nearest baselines to compare our results. We compare our methods with the following state-of-the-art baselines using macro f1-scores of OOD intents.

\[ \text{DOC (Softmax)} \] - It is based on the paper ‘Deep Open Classification (DOC)’ [22]. They proposed a method to solve the open-world classification problem. It uses sigmoid activation function at the final layer and computes the confidence threshold for each class to further tighten the decision boundary of the sigmoid function. The current baseline (used by [18]) uses a variant of DOC, which replaces the sigmoid activation function with the softmax at the final layer, and show the better performance compared to the original ‘DOC’.

\[ \text{LOF (LCML)} \] - It is based on the paper [17]. It first uses Large Cosine Margin Loss (LCML) [47] to train a feature extractor, and then uses a density-based novelty detection algorithm Local Outlier Factor (LOF) [48] for out-of-distribution detection.

\[ \text{LOF (Softmax)} \] - A variant of LOF (LCML), which replaces LCML with the normal softmax loss to train the feature extractor.
‘GDA + Euclidean distance’ and ‘GDA + Mahalanobis distance’ - [18] has proposed a Deep generative distance-based classifier, which used Euclidean and Mahalanobis distances respectively.

The conversational dataset used by [20], is not publicly available, so we failed to use it in our experimental evaluation.

B. ADDITIONAL BASELINES

We consider some additional baselines for experiment. The main aim was to check other possibilities in end-to-end supervised and zero-shot framework. Some of the baselines are helpful in understanding the impact of different features.

Additional Baseline-1: This baseline tests the impact of the direct use of BERT based token-level embedding in differentiating between ID and OOD utterances. This baseline uses the BERT sequence output (as used in [44] for token-level intent classification). Instead of classifying intents, we directly pass the embedding output to OC-NN (as discussed in Section III-B). The rest part (i.e., the use of OC-NN and the corresponding parameter setting strategy) is the same as given in Section III-B.

Additional Baseline-2: Actually, BERT generates two different types of output (a) sequence output used to represent token-level sentence embedding and (b) pooled output used to represent sentence-level embedding. In this baseline, we have used the pooled output of BERT to differentiate between ID and OOD utterances. This is the only difference between ‘Additional Baseline-1’, and the current baseline. The rest part (i.e., the use of OC-NN and the corresponding parameter setting strategy) is the same as given in section III-B.

Additional Baseline-3: In this baseline, we use confidence scores of ‘sentence-level intents’ (obtained by using multi-level classification, see section III-B) as a feature and pass it to OC-NN. The main aim is to check the effectiveness of ‘sentence-level intents’ in classifying the OOD.

Additional Baseline-4: In this baseline, we use ‘token level intents’ (i.e., token-level intent prediction score/softmax score) as a feature and pass it to OC-NN. The main aim is to check the effectiveness of ‘token level intents’ in classifying the OOD.

C. MODEL AND HYPERPARAMETER DETAILS

We have used the BERT-Base, cased model, which contains 12-layers (i.e., Transformer blocks), hidden size $H = 768$, $A = 12$ attention heads, and trained on 110M parameters. The max token count is data-dependent and calculated separately for each dataset. The token count is the count of tokens in a single utterance; thus, the max token count is the count of the maximum number of tokens in any of the utterances of the given dataset. We have used the early-stopping [16] based on minimizing the loss with callback and patience $= 2$ for the best model selection.

D. DATASETS

We perform experiments on four public benchmark OOD datasets, including SNIPS [18], ATIS [18], CLINC-Full, and CLINC-Imbal [11]. The details are given in TABLE 2.

E. IMPLEMENTATION DETAILS

To conduct a fair comparison, we follow a similar evaluation setting as discussed in [17] and [18]. In each experiment, we randomly sample a set of intent classes among all classes in the dataset, considering them as ID classes and the rest as out-of-domain classes. We use train samples from only ID classes to train our model and use test samples both from ID and out-of-domain classes for OOD sample detection. We vary the proportion of ID classes at 25%, 50%, and 75% of all classes. For each proportion, we rerun the experiment 10 times (each with a different set of ID classes) and report the average F1-score on OOD sample detection.

F. RESULTS AND DISCUSSION

TABLES 3, 4, 5, and 6 show the experiment results. Based on the experimental results, we can draw the following conclusions.

1) OUR-MODEL consistently outperforms all baselines in all settings.

2) However, when we compare our baselines like “Additional Baseline-1”, “Additional Baseline-2”, “Additional Baseline-3”, and “Additional Baseline-4”, we find that, “Additional Baseline-4” is also consistently performing better than other systems. This shows the effectiveness of token-level intent features.
TABLE 4. Results on ATIS dataset.

| % of known intents | 25%  | 50%  | 75%  |
|--------------------|------|------|------|
| DOC (SoftMax)      | 63.6 | 63.3 | 38.7 |
| LOF (SoftMax)      | 67.3 | 61.8 | 38.9 |
| LOF (LMCL)         | 68.5 | 63.4 | 39.6 |
| GDA + Euclidean distance | 77.9 | 75.4 | 43.7 |
| GDA + Mahalanobis distance | 78.5 | 72.8 | 42.1 |
| Additional Baseline-1 | 72.0 | 72.4 | 42.0 |
| Additional Baseline-2 | 71.1 | 68.3 | 41.0 |
| Additional Baseline-3 | 73.1 | 70.1 | 43.0 |
| Additional Baseline-4 | 81.4 | 76.0 | 64.6 |
| OUR-MODEL          | 81.9 | 76.6 | 66.3 |

TABLE 5. Results on CLINC-full dataset.

| % of known intents | 25%  | 50%  | 75%  |
|--------------------|------|------|------|
| DOC (SoftMax)      | -    | -    | -    |
| LOF (SoftMax)      | 91.1 | 83.1 | 63.5 |
| LOF (LMCL)         | 91.3 | 83.3 | 62.8 |
| GDA + Euclidean distance | 91.4 | 84.2 | 65.1 |
| GDA + Mahalanobis distance | 91.4 | 84.4 | 65.1 |
| Additional Baseline-1 | 91.1 | 83.5 | 64.2 |
| Additional Baseline-2 | 91.0 | 83.4 | 63.4 |
| Additional Baseline-3 | 91.0 | 83.3 | 63.5 |
| Additional Baseline-4 | 93.6 | 85.9 | 70.2 |
| OUR-MODEL          | 94.2 | 86.4 | 70.9 |

TABLE 6. Results on CLINC-Imbal dataset.

| % of known intents | 25%  | 50%  | 75%  |
|--------------------|------|------|------|
| DOC (SoftMax)      | -    | -    | -    |
| LOF (SoftMax)      | 88.4 | 77.6 | 57.5 |
| LOF (LMCL)         | 88.7 | 78.9 | 56.7 |
| GDA + Euclidean distance | 91.1 | 81.2 | 60.8 |
| GDA + Mahalanobis distance | 91.5 | 81.5 | 61.3 |
| Additional Baseline-1 | 90.7 | 80.1 | 58.0 |
| Additional Baseline-2 | 88.9 | 78.7 | 59.4 |
| Additional Baseline-3 | 89.4 | 78.8 | 59.8 |
| Additional Baseline-4 | 93.7 | 83.7 | 67.5 |
| OUR-MODEL          | 94.3 | 84.9 | 68.9 |

3) When we compare the results of “Additional Baseline-1”, and “Additional Baseline-2”, we find that token-level embedding better represents the text. Thus, the “Additional Baseline-1”, shows better performance. Even “Additional Baseline-1”, shows little better performance w.r.t., “Additional Baseline-3”. This means that the token-level embedding better represents the utterances compared to sentence-level intents.

4) When we compare the results between, “Additional Baseline-2” and “Additional Baseline-3”, we find that, “Additional Baseline-3” shows a little better performance. This means the sentence-level intent based feature is more important w.r.t., sentence embedding.

5) When we compare the results of “Additional Baseline-1” and “Additional Baseline-4”, we find that “Additional Baseline-4” shows better performance. This means the token-level intent based feature shows better performance w.r.t. token-level embedding.

6) In the majority of the cases, the performance of sentence-level intent based feature (see “Additional Baseline-3”) is a little poor compared to the other baselines like – “GDA + Euclidean distance” and “GDA + Mahalanobis distance” and other systems. This shows that the direct use of sentence-based embedding is not highly effective (compared to the combined use of token and sentence-level intents).

7) “Additional Baseline-3”, which represents the sentence-level intent based feature, shows relatively lower performance w.r.t., “Additional Baseline-4”. But, combining the sentence-level and token-level intent feature helps OC-NN to learn to effectively differentiate between ID and OOD utterances (See the performance of “OUR-MODEL”).

8) However, we have seen a decrease in the performance with the increase in the % intents. After domain analysis, we have identified that all intent classes of each of the four datasets are related to a fixed domain. For example, in the ATIS dataset, all intent classes are related to the “Airline Travel Information System” domain. When we increase the percentage count of the ID intent class, its similarity with the rest of the classes (related to the same domain) increases. Thus, it results in a decrease in the accuracy of the system. We believe that with the increase in the amount of training data, we can reduce such negative effects.

V. CONCLUSION

This paper shows the effectiveness of intents focused semantic features, like - combined use of fine grain (i.e., token-level intents) and sentence-level intents in the automatic identification of OOD classes. The use of OC-NN on top of intent-focused semantic features helped in developing the fully supervised system. It also helped in removing the threshold based process, like another unsupervised system is doing. As future work, we can try an additional better set of features in enhancing the performance to a next level.

REFERENCES

[1] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, “The Stanford CoreNLP natural language processing toolkit,” in Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics, Syst. Demonstrations, 2014, pp. 55–60.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 5998–6008.

[3] Y. Wu et al., “Google’s neural machine translation system: Bridging the gap between human and machine translation,” 2016, arXiv:1609.08144.

[4] Q. Chen, Z. Zhao, and W. Wang, “BERT for joint intent classification and slot filling,” 2019, arXiv:1902.10909.

[5] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized autoregressive pretraining for language understanding,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 1–18.

[6] B. Dzmitry, C. Kyunghyun, and B. Yoshua, “Neural machine translation by jointly learning to align and translate,” in Proc. ICLR, 2015, pp. 1–15.

[7] B. Shu, F. Ren, and Y. Yao, “Investigating LSTM with k-max pooling for text classification,” in Proc. 11th Int. Conf. Intell. Comput. Technol. Autom. (ICICTA), Sep. 2018, pp. 31–34.
[8] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, “Toward open set recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 7, pp. 1757–1772, Jul. 2013.

[9] G. Fei and B. Liu, “Breaking the closed world assumption in text classification,” in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., 2016, pp. 506–514.

[10] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, “Likelihood ratios for out-of-distribution detection,” 2019, arXiv:1906.02845.

[11] L. Kimin, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” 2018, arXiv:1807.03888.

[12] K. He, Y. Yan, and W. Xu, “Learning to tag OOV tokens by integrating contextual representation and background knowledge,” in Proc. 58th Annul. Meeting Assoc. Comput. Linguistics, 2020, pp. 619–624.

[13] Y. Zheng, G. Chen, and M. Huang, “Out-of-domain detection for natural language understanding in dialog systems,” IEEE/ACM Trans. Audio, Speech, Language Process., vol. 28, pp. 1198–1209, 2020.

[14] J. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. Mars, “An evaluation dataset for intent classification and out-of-scope prediction,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 1311–1316.

[15] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, “Deep one-class classification,” in Proc. 35th ICML, 2018, pp. 4393–4402.

[16] J. K. Kim and Y.-B. Kim, “Joint learning of domain classification and acceptance rates,” 2018, arXiv:1807.00072.

[17] J. Kimin, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” 2018, arXiv:1807.03888.

[18] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, “Likelihood ratios for out-of-distribution detection,” 2019, arXiv:1906.02845.

[19] L. Prechelt and G. B. Orr, “Early stopping–but when,” in Proc. 28th Int. Conf. Comput. Linguistics, 2020, pp. 53–67.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805.

[21] J.-K. Kim and Y.-B. Kim, “Joint learning of domain classification and acceptance rates,” 2018, arXiv:1807.00072.

[22] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, “Likelihood ratios for out-of-distribution detection,” 2019, arXiv:1906.02845.

[23] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, “Deep one-class classification,” in Proc. 35th ICML, 2018, pp. 4393–4402.

[24] J. Kimin, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” 2018, arXiv:1807.03888.

[25] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, “Likelihood ratios for out-of-distribution detection,” 2019, arXiv:1906.02845.

[26] L. Prechelt and G. B. Orr, “Early stopping–but when,” in Neural Networks: Tricks of the Trade (Lecture Notes in Computer Science), G. Montavon and K.-R. Müller, Eds., Berlin, Germany: Springer, Jan. 2012, pp. 53–67.

[27] T.-E. Lin and H. Xu, “Deep unknown intent detection with margin loss,” in Proc. 57th Annul. Meeting Assoc. Comput. Linguistics, 2019, pp. 1–6.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805.

[29] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, “Likelihood ratios for out-of-distribution detection,” 2019, arXiv:1906.02845.

[30] L. Prechelt and G. B. Orr, “Early stopping–but when,” in Neural Networks: Tricks of the Trade (Lecture Notes in Computer Science), G. Montavon and K.-R. Müller, Eds., Berlin, Germany: Springer, Jan. 2012, pp. 53–67.

[31] T.-E. Lin and H. Xu, “Deep unknown intent detection with margin loss,” in Proc. 57th Annul. Meeting Assoc. Comput. Linguistics, 2019, pp. 1–6.

[32] Y. Zheng, G. Chen, and M. Huang, “Out-of-domain detection for natural language understanding in dialog systems,” IEEE/ACM Trans. Audio, Speech, Language Process., vol. 28, pp. 1198–1209, 2020.

[33] J. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. Mars, “An evaluation dataset for intent classification and out-of-scope prediction,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 1311–1316.

[34] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” Neural Comput., vol. 13, no. 7, pp. 1443–1471, Jul. 2004.

[35] J. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. Mars, “An evaluation dataset for intent classification and out-of-scope prediction,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 1311–1316.

[36] J. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. Mars, “An evaluation dataset for intent classification and out-of-scope prediction,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 1311–1316.

[37] J. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, and J. Mars, “An evaluation dataset for intent classification and out-of-scope prediction,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 1311–1316.