Learning Class-Transductive Intent Representations for Zero-shot Intent Detection

Qingyi Si[1,2], **Yuanxin Liu**[1,2], **Peng Fu**[1*], **Zheng Lin**[1*], **Jiangnan Li**[1,2], **Weiping Wang**[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{siqingyi, liuyuanxin, fupeng, linzheng, lijiangnan, wangweiping}@iie.ac.cn

Abstract

Zero-shot intent detection (ZSID) aims to deal with the continuously emerging intents without annotated training data. However, existing ZSID systems suffer from two limitations: 1) They are not good at modeling the relationship between seen and unseen intents. 2) They cannot effectively recognize unseen intents under the generalized intent detection (GZSID) setting. A critical problem behind these limitations is that the representations of unseen intents cannot be learned in the training stage. To address this problem, we propose a novel framework that utilizes unseen class labels to learn **C**lass-**T**ransductive **I**ntent **R**epresentations (CTIR). Specifically, we allow the model to predict unseen intents during training, with the corresponding label names serving as input utterances. On this basis, we introduce a multi-task learning objective, which encourages the model to learn the distinctions among intents, and a similarity scorer, which estimates the connections among intents more accurately. CTIR is easy to implement and can be integrated with existing methods. Experiments on two real-world datasets show that CTIR brings considerable improvement to the baseline systems.[1]

1 Introduction

In recent years, smart devices with built-in personal assistants like Google Assistant and Siri are becoming omnipresent. Behind these systems, a key question is how to identify the underlying intent of a user utterance, which has triggered a large amount of work on intent detection [Ravuri and Stolcke, 2015; Liu and Lane, 2016; Nam *et al.*, 2016]. Most existing intent detection systems are built on models trained on annotated data. However, as user demands and the functions of smart devices continue to grow, collecting supervised data for every new intent becomes labor-intensive.

To address this issue, some studies tackle intent detection in the zero-shot learning (ZSL) manner, attempting to utilize the learned knowledge of seen classes to help detect unseen classes. Recent methods of ZSID can be roughly divided into two categories: The first category [Xia *et al.*, 2018; Liu *et al.*, 2019], referred to as the transformation-based methods, utilizes word embeddings of label names to establish a similarity matrix, which is then used to transfer the prediction space of seen intents to unseen intents. Another line is the compatibility-based methods [Chen *et al.*, 2016; Kumar *et al.*, 2017], which aims to encode the label names and utterances into the same semantic space and then calculate their similarity. However, in both kinds of methods, most existing ZSID methods are *inductive*, which do not consider any information about the unseen classes in the training stage. Consequently, the representations of unseen intents cannot be learned, resulting in two limitations.

First, the ZSID methods are not good at modeling the relationship between seen and unseen intents, especially when the label names are given in the form of raw phrases or sentences. For the transformation-based methods, word embeddings of label names are inadequate to associate the connections between seen and unseen intents. For example, "BookRestaurant" is similar to "RateBook" when measured by word embeddings. However, the meaning of these two intents are not that relevant. For the compatibility-based methods, since the unseen intents are not included in learning the semantic space shared by utterance and label names, it cannot effectively detect unseen intents during the test stage, especially when the expressions of utterances are diverse.

Second, the vanilla ZSL methods are not applicable to generalized zero-shot intent detection (GZSID), where the models (at test time) are presented with not only unseen class utterances but also seen class utterances. In GZSID, existing ZSL models usually suffer from the domain shift [Fu *et al.*, 2015a] problem, in which utterances from unseen intents are almost always mistakenly classified into seen intents.

The two limitations are caused by inadequate learning of unseen intents. Naturally, the label name provides a proper sketch of the intent meaning. Existing models use it during the test stage. In contrast, we introduce the *class-transductive* [Wang *et al.*, 2019; Xian *et al.*, 2019] setting into ZSID, which uses semantic information about the unseen classes (e.g., the label names) for model training. Specifically, we include the unseen intents into the prediction space during training, with the label names serving as the pseudo utter-

*Peng Fu and Zheng Lin are the corresponding authors.

[1]The code, datasets and Appendix are available at https://github.com/PhoebusSi/CTIR

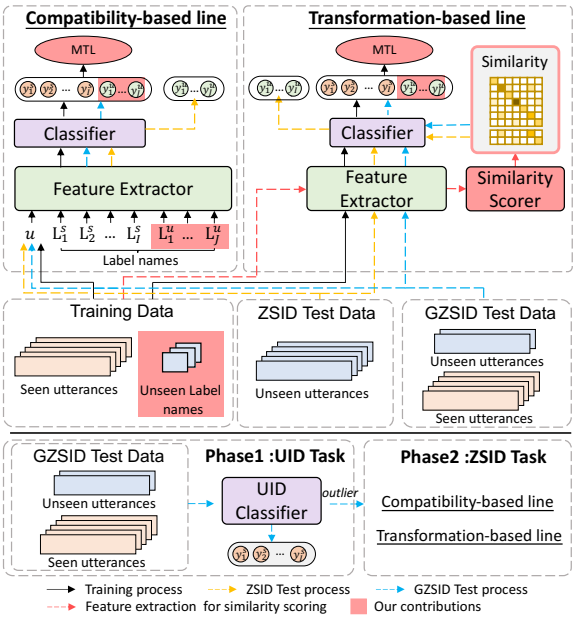

Figure 1: Illustration of how to integrate compatibility-based method (upper left), transformation-based method (upper right) and two-stage GZSID method (lower) with CTIR.

ances. This allows the model to learn a rough boundary of each seen and unseen class in the semantic space. Under this framework, we introduce an assistant task that forces the model to find the distinction between seen and unseen intents, thereby alleviating the domain-shift problem. On this basis, we refine the word embedding based similarity matrix by averaging the representations of all corresponding (seen intent) utterances and (unseen intent) label names. As a result, we can better capture the intent meanings and the similarity matrix reflects more accurate intent connections. In summary, our contribution is three-fold:

- In response to the limitations of existing ZSID systems, we propose a class-transductive framework that makes use of unseen label names in the training stage.
- Under the framework, we present a multi-task learning objective to find the inter-intent distinctions and a similarity scorer to associate the inter-intent connections.
- Empirical results on ZSID and GZSID in two benchmarks show that CTIR can bring improvement to a wide range of ZSID systems with different zero-shot learning strategies and model architectures.

2 Problem Formulation

Zero-shot Intent Detection. In ZSID, the model is trained on the annotated dataset $\{(x,\ y)\}$, where $y \in Y_{seen} = \{y_1^s, y_2^s, \ldots, y_I^s\}$ is the intent label and x is the utterance. At test time, the goal is to identify the intent of an utterance, which belongs to one of the J unseen intents $Y_{unseen} = \{y_1^u, y_2^u, \ldots, y_J^u\}$, where $Y_{seen} \cap Y_{unseen} = \emptyset$.

Generalized Zero-shot Intent Detection. In GZSID, the model is presented with utterances from either seen or unseen intents, and the prediction space is $Y_{seen} \cup Y_{unseen}$.

Unknown Intent Detection (UID). In UID [Lin and Xu, 2019], the training set is the same as ZSID and GZSID. During testing, the model is expected to detect seen intents and decide whether an utterance belongs to the unknown intents. The prediction space is $\{y_1^s, y_2^s, \ldots, y_I^s, y_{unseen}\}$, where all the unknown intents are grouped into a single class y_{unseen}. The GZSID can be solved by breaking it into UID and ZSID tasks(as shown in Figure1 lower).

Simplified Unknown Intent Detection (SUID). The prediction space is reduced to $\{y_{seen}, y_{unseen}\}$ in SUID. In our multi-task learning, SUID serves as an assistant task.

3 CTIR Framework

In this section, we describe how to integrate CTIR into transformation-based, compatibility-based methods and two-stage GZSID method respectively. Figure1 gives an overview. The core idea is to expand the prediction space during training to include unseen classes, with the unseen label names serving as pseudo utterances. During test time, the trained model can be applied to both ZSID and GZSID settings.

3.1 Transformation-based Methods

Feature Extractor. The feature extractor transforms an input text into a sequence of hidden vectors $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_T) \in \mathbb{R}^{T \times D_H}$, where D_H is the hidden dimension and T is the sequence length. Note that the architecture of feature extractor is a free choice in CTIR framework. Specially, on top of the $\mathbf{H}$ produced by a Bi-LSTM, CapsNet [Xia *et al.*, 2018] applies a multi-head attention layer. Each attention head represents a unique semantic feature, which gives rise to $\{\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_R\} \in \mathbb{R}^{R \times D_H}$, where R is the number of attention heads. We follow this operation when combining CapsNet with CTIR.

Intent Classifier. There are two commonly used classifiers in intent detection: 1) the linear classifier and 2) the capsule networks [Sabour *et al.*, 2017]. The linear classifier predicts the intent probabilities as:

$$\mathbf{v}_{tr} = \text{Softmax}((\frac{1}{T}\sum_{t=1:T}\mathbf{h}_t)\mathbf{W}) \tag{1}$$

where $W \in \mathbb{R}^{D_H \times K}$ is the weight matrix, and $K = I + J$ is the total number of seen and unseen classes.

In capsule networks, the Dynamic Routing algorithm [Sabour *et al.*, 2017] is used to aggregate the low-level features $\{\mathbf{m}_1, \mathbf{m}_2, \ldots \mathbf{m}_R\}$ into higher-level representations:

$$\begin{aligned} \hat{\mathbf{u}}_{k|r} &= \mathbf{m}_r \mathbf{W}_{k,r} \\ \mathbf{s}_k &= \textstyle\sum_r c_{kr} \hat{\mathbf{u}}_{k|r} \end{aligned} \tag{2}$$

where c_{kr} is the coefficient that determines how much the r^{th} semantic feature contributes to intent y_k. Following $\mathbf{s}_k$ is the squash function, which gives rise to the activation vectors $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K)$. The 2-norms of these vectors are then used as the probability of different intent classes.

Multi-task Learning Objective. The assistant task SUID has two class labels, namely y_{seen} and y_{unseen}. In order to compute the probabilities for y_{seen} and y_{unseen}, we sum up

the probabilities of all intent classes in the respective categories. For linear classifiers, we compute the sum of the first I dimensions of the vector $\mathbf{v}_{tr}$ for y_{seen} and the last J dimensions for y_{unseen}. Then, the linear classifier is trained with cross-entropy loss:

$$\mathcal{L}_{cross} = \sum_{k=1}^{K} z_k \log(p_k) + \alpha \sum_{n=1}^{2} z'_n \log(P_n) \tag{3}$$

where $z_k \in \{0, 1\}$ indicates whether the k^{th} intent is true, and p_k is the predicted probability of the k^{th} intent. z'_n and P_n are respectively the ground truth and the predicted probability of each class in SUID. α is a down-weighting coefficient.

For capsule networks, we sum up the 2-norms of the activation vectors $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_I)$ and $(\mathbf{v}_{I+1}, \mathbf{v}_{I+2}, \ldots, \mathbf{v}_K)$, respectively. Then, the max-margin loss for capsule networks is:

$$\begin{aligned}\mathcal{L}_{margin} = &\sum_{k=1}^{K} \Big\{ T_k \cdot \max\left(0, m^+ - \|\mathbf{v}_k\|\right)^2 \\ &+\lambda\left(1 - T_k\right) \cdot \max\left(0, \|\mathbf{v}_k\| - m^-\right)^2 \Big\} \\ &+ \lambda' \sum_{n=1}^{2} \Big\{ T'_n \cdot \max\left(0, m'^+ - P_n\right)^2 \\ &+\lambda\left(1 - T'_n\right) \cdot \max\left(0, P_n - m'^-\right)^2 \Big\}\end{aligned} \tag{4}$$

where $T_k = 1$ when the k^{th} intent is ground-truth, and otherwise $T_k = 0$. T'_n is defined in the same way for SUID. λ and λ' are the down-weighting coefficients, m^+, m^- and m'^+, m'^- are the margins. In addition, a regularization term is added to $\mathcal{L}_{margin}$ to encourage the discrepancy among attention heads [Xia *et al.*, 2018].

Similarity Scorer. Similarity Scorer, which measures the connections between intent classes, is a key component for transformation-based methods. Inspired by Chao [2016], we average the representations of all utterances to represent the seen intents. For the unseen intents, we use the representations of label names. The representation of each utterance or label name is obtained by averaging over different time steps or attention heads (for CapsNet). In practice, the representations are computed during the last training epoch, which considers the entire training set (i.e., the parameters of feature extractors are updated in this process).

After the averaging operation, we have the intent representations $\{\mathbf{g}_1, \mathbf{g}_2, \ldots \mathbf{g}_K\} \in \mathbb{R}^{K \times D_H}$, which is used to compute the similarity matrices $\mathbf{L}_{zsl} \in \mathbb{R}^{K \times J}$ for ZSID and $\mathbf{L}_{gzsl} \in \mathbb{R}^{K \times K}$ for GZSID. The similarity between intent k_1 and k_2 is computed as $L_{k_1 k_2} = \text{Cosine}(\mathbf{g}_{k_1}, \mathbf{g}_{k_2})$, where $\mathbf{g}_{k1}$, $\mathbf{g}_{k2}$ are the representations of $k1$ and $k2$, respectively.

Inference Process. During inference, a test utterance is first encoded into a prediction vector of K dimensions, in the same way as the training process. Based on this vector, we further employ the Similarity Scorer to obtain the final prediction. In terms of the linear classifier, we have:

$$\mathbf{v}_{te} = \text{Softmax}((\frac{1}{T} \sum_{t=1:T} \mathbf{h}_t)\mathbf{W}\mathbf{L}) \tag{5}$$

where $\mathbf{L}$ refers to $\mathbf{L}_{zsl}$ or $\mathbf{L}_{gzsl}$. When it comes to the capsule networks:

$$\mathbf{s}_j = \sum_r L_{jk}(c_{kr}\hat{\mathbf{u}}_{j|k}) \tag{6}$$

where L_{jk} is the k^{th} entry in the j^{th} row of $\mathbf{L}$. After Dynamic Routing, we can get J activation vectors for ZSID and K activation vectors for GZSID.

3.2 Compatibility-based Methods

Feature Extractor. Similar to the transformation-based methods, the first step of compatibility-based methods is to encode the utterance or label name into a dense vector. In this paper we study two kinds of compatibility-based methods: Zero-shotDNN [Kumar *et al.*, 2017] and CDSSM [Chen *et al.*, 2016], which extract the text representation with a tanh-activated nonlinear layer and CNN respectively.

Intent Classifier. With the representations of an utterance and all label names, compatibility-based methods compute the cosine similarity between the utterance u and each intent, which results in $\mathbf{S} = \{sim(u, y) | y \in Y_{seen} \cup Y_{unseen}\} \in \mathbb{R}^K$. Then, u can be classified into a particular intent class according to $\mathbf{v}_{tr} = \text{Softmax}(\mathbf{S})$.

Multi-task Learning Objective. To perform SUID with the compatibility-based classifier, we sum up the first I and last J positions in $\mathbf{S}$ as $\sum_{i=1:I} \mathbf{S}_i$ and $\sum_{i=I+1:K} \mathbf{S}_i$, the result of which is passed to a binary softmax function to obtain the final probabilities for y_{seen} and y_{unseen}. The multi-task learning objective is defined in the same way as Equation 3.

Inference Process. The inference process of compatibility-based methods is basically the same as the classification process during training. Given an input utterance, we compute its similarity with each candidate intent in the learned representation space, and classify it to the most similar intent.

3.3 Two-stage GZSID Method

The main idea of two-stage method is to first determine whether an utterance belongs to unseen intents (i.e., Y_{seen}), and then classify it into a specific intent class. This method bypasses the need to classify an input sentence among all the seen and unseen intents, thereby alleviating the domain shift problem. To verify the performance of integrating CTIR into the two-stage method, we design a new two-stage pipeline. In Phase1, a test utterance is classified into one of the classes from $Y_{seen} \cup \{y_{unseen}\}$ using the UID classifier. In practice, we use the density-based algorithm LOF (LMCL) [Lin and Xu, 2019] to perform UID. In Phase2, we perform ZSID for the utterances that have been classified into y_{unseen}, using the methods described in Section 3.1 and Section 3.2.

4 Experiments

4.1 Datasets and Experimental Setup

We conduct experiments on two benchmarks for intent detection. **SNIPS** [Coucke *et al.*, 2018] is a corpus to evaluate the performance of voice assistants, which contains 5 seen intents and 2 unseen intents. **CLINC** [Larson *et al.*, 2019] includes out-of-scope queries and 22,500 in-scope queries covering intent classes from 10 domains. We use the in-scope data to

Model	SNIPS		CLINC	
	Acc	F1	Acc	F1
LSTM (Ours)	79.47	79.18	71.73	68.73
+CTIR	**93.43**	**93.42**	**84.48**	**84.35**
CNN (Ours)	65.15	60.91	73.03	70.94
+CTIR	**94.73**	**94.73**	**85.11**	**85.20**
BERT (Ours)	73.66	73.32	62.73	59.45
+CTIR	**96.13**	**96.12**	**88.72**	**88.45**
CDSSM (Chen et al. 2016)	68.98	66.52	64.80	61.14
+key	83.03	82.86	-	-
+CTIR	**94.14**	**94.14**	**83.07**	**82.52**
ZSDNN [Kumar *et al.*, 2017]	80.96	80.74	82.20	82.18
+key	93.49	93.49	-	-
+CTIR	**95.07**	**95.07**	**93.57**	**93.62**
CapsNet [Xia *et al.*, 2018]	74.21	72.58	64.51	61.87
+key	93.49	93.49	-	-
+CTIR	**94.84**	**94.84**	**87.01**	**86.91**

Table 1: Results of ZSID. (Ours) represents our implementation.

build our dataset with 50 seen intents and 10 unseen intents. For more details of the datasets, please refer to Appendix A[1].

Dataset Processing. For ZSID, we use all utterances from seen intents to construct the training set and those from unseen intents to construct the test set. For the training set of GZSID, we randomly select 70% utterances of each seen intent and replicate the unseen label names to roughly the same number. Although our focus is the zero-shot problem, utterances from seen intents still account for the majority in real-world applications. In light of this phenomenon, we balance the sample number of unseen and seen classes to build the test set: selecting the remaining 30% utterances of each seen intent and 30% random utterances of each unseen intent (30% seen and 100% unseen intent utterance in the setting of existing works). Besides, we use the raw label names without modification while most existing work modified the label names slightly to achieve better performance, e.g. from "SearchScreeningEvent" to "SearchMovie".

Experimental Setup. We use the test set for hyperparameter(Appendix B[1]) tuning, which is the same with most ZSID work. We run the experiments for five times with different random seeds. We consider two evaluation metrics: Accuray (Acc), and F1, both of which are computed with the average value weighted by their support on each class.

4.2 Baselines

We integrate CTIR with representative ZSID systems. For transformation-based methods with linear classifier, we explore the use of CNN, LSTM, and BERT [Devlin *et al.*, 2019] as the feature extractor, which are denoted as CNN, LSTM and BERT, respectively. We find that use BERT to compute the similarity matrix leads to poor results, as the label names are very short. Therefore, we combine BERT feature extractor with CNN-CTIR computed similarity matrix for the BERT baseline. For capsule network classifier, we adopt CapsNet [Xia *et al.*, 2018] as the baseline system. In terms of the compatibility-based methods, we study the combination of CTIR with Zero-shotDNN [Kumar *et al.*, 2017] and CDSSM [Chen *et al.*, 2016].

We also introduce two strong baselines targeting the two limitations related to ZSID and GZSID respectively, as discussed in Section 1. For ZSID, we replace the original label names with manually selected keywords (denoted as + key), which reflect the inter-intent relationships more accurately. For GZSID, we build a two-stage approach (denoted as + LOF) as described in Section 3.3.

ReCapsNet [Liu *et al.*, 2019] and the two-stage method SEG [Yan *et al.*, 2020], which achieve the SOTA performance, are not open-sourced. Different from our two-stage method, SEG ensembles UID and ZSID models in Phase1 and conducts seen intent prediction in Phase2. To compare with them, we follow their way of data processing and run our model.

4.3 Results and Analysis

Performance in ZSID. As can be seen in Table 1, the performance of CDSSM, ZSDNN and CapsNet can be significantly improved with the manually composed keyword labels. This demonstrates the first limitation that existing ZSID methods are not good at associating the relationship between seen and unseen intents when the label names are given in raw phrases or sentences. Our proposed framework, regardless of the backbone network and ZSID strategy, consistently outperforms the baselines with comfortable margin (11.49 $\sim$ 33.82 absolute F1 improvement). More importantly, the CTIR-enhanced models also achieve better results than the manually selected keywords, with an encouraging improvement of 11.28 F1 score for the CDSSM baseline. This shows the effectiveness of the proposed method to alleviate the effect of poor-quality label names, dispensing with the need of manual modification.

Performance in GZSID. From the results in Table 2, we can derive four observations: 1) The baselines work well on the seen intents, while they fail to detect the unseen intents. This phenomenon attests to the second limitation that the ZSID systems cannot effectively work in the GZSID scenario. 2) The two-stage framework (+LOF) brings significant improvement on the unseen intents, successfully alleviating the domain-shift problem. 3) Our CTIR framework improves the performance to a larger extent, which on average outperforms +LOF with 10.04 F1 on unseen intents and 3.37 F1 in terms of overall performance. 4) In terms of Acc, CTIR performs better in seen intents while LOF achieves higher scores in unseen intents. We analyze this trade-off phenomenon in Appendix C[1]. 5) CTIR can further promote the strong two-stage baseline: Enhancing the Phase2 ZSID model with CTIR (+LOF+CTIR) results in a substantial improvement of +LOF on the unseen intents, where the Acc is increased by 19.77 on average and F1 score is increased by 16.52 on average. The overall performance is thereby also improved.

Relieving of two limitations. We qualitatively analyze the effectiveness of CTIR in relieving the two limitations by answering the following questions. Q1: Can CTIR model a better inter-intent relationship in ZSID? Q2: Can CTIR distinguish between the samples of seen and unseen intents?

Q1 A good inter-intent relationship is crucial to transform the model from seen intent to unseen intent. To illustrate the problem of word embedding based inter-intent similarity, we compute the similarity with the sum of original

Model	SNIPS						CLINC					
	Seen		Unseen		Overall		Seen		Unseen		Overall	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
LSTM (Ours)	**97.22**	83.94	0.00	0.00	69.66	60.14	93.19	85.95	0.00	0.00	77.65	71.62
+CTIR	95.42	**86.00**	**31.34**	**45.32**	**77.26**	**74.47**	**93.40**	**87.55**	**20.50**	**30.96**	**81.24**	**78.12**
CNN (Ours)	**97.95**	85.09	0.00	0.00	70.18	60.97	95.35	87.89	0.00	0.00	79.46	73.24
+CTIR	96.59	**88.41**	**46.22**	**62.37**	**82.31**	**81.03**	**96.25**	**90.06**	**15.82**	**23.11**	**82.85**	**78.91**
BERT (Ours)	96.43	84.70	0.00	0.00	69.1	60.69	88.14	82.17	0.23	0.44	73.49	68.55
+CTIR	**96.93**	**89.62**	**49.38**	**61.15**	**83.45**	**81.55**	**97.84**	**91.19**	**12.55**	**19.74**	**83.62**	**79.29**
CDSSM (Chen et al. 2016)	**97.99**	84.54	0.19	0.37	70.32	60.74	93.62	86.66	2.09	3.69	78.35	72.82
+LOF	81.52	**88.22**	60.75	49.14	75.63	77.14	79.96	86.94	42.80	28.96	73.76	77.28
+CTIR	92.55	84.63	42.91	56.46	78.51	76.66	**94.72**	**89.11**	23.64	32.39	**82.87**	79.65
+LOF+CTIR	81.52	**88.22**	**87.00**	**73.01**	**83.08**	**83.90**	79.96	86.94	**61.32**	**44.58**	76.89	**79.97**
ZSDNN [Kumar *et al.*, 2017]	94.79	83.29	10.50	17.40	70.93	64.64	85.45	81.14	26.79	36.07	75.63	73.73
+LOF	81.52	88.22	66.37	55.24	77.23	78.86	79.96	86.94	64.04	47.13	77.30	80.30
+CTIR	**95.82**	**89.11**	58.03	73.00	**85.12**	**84.55**	**91.75**	**88.18**	47.15	**58.69**	**84.30**	**83.18**
+LOF+CTIR	81.52	88.22	**88.52**	**74.30**	83.51	84.27	79.96	86.94	**75.47**	54.80	79.21	81.85
CapsNet [Xia *et al.*, 2018]	**97.99**	84.34	0.00	0.00	70.22	60.43	**97.92**	90.31	0.23	0.42	81.64	75.33
+LOF	81.52	88.22	64.74	53.00	76.76	78.23	79.96	86.94	46.04	33.80	74.30	78.08
+CTIR	97.47	**89.15**	47.17	63.90	**83.21**	81.99	97.71	**92.29**	30.95	43.09	**86.58**	**84.09**
+LOF+CTIR	81.52	88.22	**86.63**	**72.91**	82.97	**83.88**	79.96	86.94	**64.40**	**46.76**	77.36	80.24

Table 2: Results of GZSID. ZSDNN is short for Zero-shotDNN.

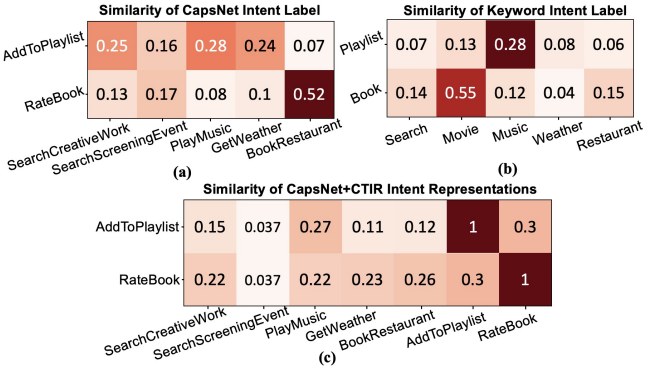

Figure 2: The similarities computed by (a) CapsNet, (b) keyword intent labels and (c) CapsNet+CTIR. Note that the similarity scores for CapsNet+CTIR are not normalized in our experiments.

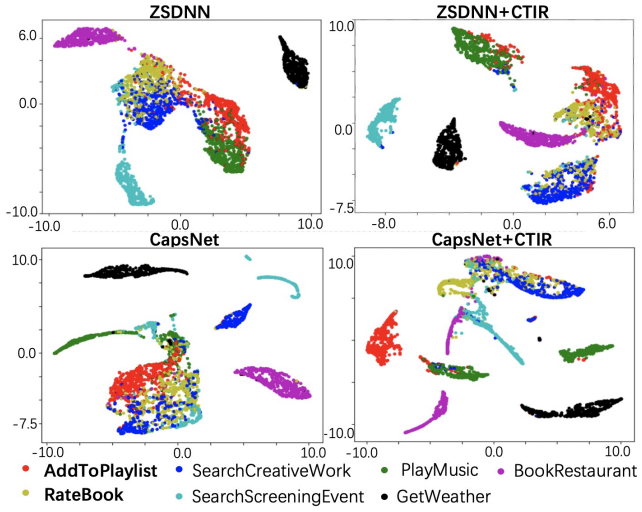

Figure 3: Data visualization on SNIP test set. "AddToPlaylist" and "RateBook" are the unseen intents.

label word embeddings (denoted as CapsNet Intent Labels), the keyword embedding (denoted as Keyword Intent Labels), and our similarity scorer (CapsNet+CTIR), which is shown in Figure2. When using the original label names, "BookRestaurant" is much more similar to "RateBook" than the other labels, as a result of the shared word "Book". In comparison, the keyword-based similarity matrix and our similarity scorer successfully avoid the pitfall of "Book" polysemy, and the similarities among other intents are also reasonable (e.g., "AddToPlaylist" and "PlayMusic"). This demonstrates that the intent representation of CapsNet+CTIR is well-learned, which can better associate the inter-intent relationship and dispenses with manually selecting keywords. We also find that for CapsNet+CTIR the two unseen classes are more similar to each other than to the seen classes. This is partly due to the introduction of the simplified unseen intent detection task, which aims to distinguish between seen classes and unseen classes.

Q2 To examine whether the CTIR framework can actually learn to distinguish between the unseen and seen intent representations, we visualize the utterance representations using the feature extractors of four methods (see Figure 3). We can see that the two unseen intents "AddToPlaylist" (red) and "RateBook" (yellow) are entangled with "SearchCreativeWork" (blue) and "PlayMusic" (green) in the CapsNet representation space. For ZSDNN representations, yellow dots and blue dots appear as a single cluster, so it is with red dots and green dots. These entangled representations make it difficult to perform GZSID. In comparison, the CapsNet+CTIR successfully learns an independent sub-space for "AddToPlaylist" and disentangles a part of "RateBook" from the seen intents. ZSDNN+CTIR pulls the two unseen classes as a whole out from the seen classes, with a vague but identifiable boundry in between.

Comparison with SOTA methods. Table 3 presents the comparison of the proposal with SOTA methods under their experimental settings as described in the Baselines. As

Model	ZSID		GZSID	
	Acc	F1	Acc	F1
ReCapsNet [Liu *et al.*, 2019]	79.96	79.80	47.05	38.26
SEG [Yan *et al.*, 2020]	-	-	76.85	76.74
ZSDNN+CTIR	**95.05**	**95.05**	74.41	75.44
ZSDNN+LOF	-	-	76.93	77.07
CapsNet+LOF+CTIR	-	-	82.90	83.19
ZSDNN+LOF+CTIR	-	-	**84.10**	**84.33**

Table 3: Comparison with SOTA methods in SNIPS.

Model	SNIPS				CLINC			
	ZSID		GZSID		ZSID		GZSID	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
CNN + CTIR	**94.73**	**94.73**	**82.31**	**81.03**	**85.11**	**85.20**	82.91	78.88
w/o MT	94.09	94.08	82.24	80.96	84.96	85.01	82.85	78.73
w/o SS	94.54	94.53	80.55	79.60	83.78	83.82	**84.24**	**80.25**
SS → ES	81.89	80.95	80.51	79.50	82.44	82.54	82.54	78.29
CapsNet + CTIR	**94.84**	**94.84**	**83.21**	**81.99**	**87.01**	**86.91**	**86.58**	**84.09**
w/o MT	93.68	93.67	81.02	79.07	85.17	85.07	86.19	83.64
w/o SS	90.30	90.26	83.20	81.94	85.37	85.16	86.26	83.69
SS → ES	82.54	81.96	83.11	81.90	79.97	79.79	86.57	83.92
ZSDNN + CTIR	**95.07**	**95.07**	**85.12**	**84.55**	**93.57**	**93.62**	**84.30**	**83.18**
w/o MT	93.55	93.52	84.90	84.30	93.40	93.46	84.03	82.81
CDSSM + CTIR	**94.14**	**94.14**	78.51	**76.66**	**83.07**	**82.52**	**82.87**	**79.65**
w/o MT	93.84	93.83	**78.73**	76.01	82.68	82.12	82.79	79.57

Table 4: Results of ablation study. w/o MT and w/o SS indicate removing multi-task learning and the similarity scorer, respectively. SS → ES means replacing our Similarity Scorer with the word embedding based similarity.

we can see, +CTIR methods clearly outperform ReCapsNet in ZSID, and the +LOF+CTIR methods outperform SEG, which is also a two-stage approach, in GZSID. The +LOF method is used as our stronger baseline and its performance is comparable with SEG in GZSID. One may find that ZSDNN+LOF+CTIR is better than ZSDNN+CTIR in Table 3, while lags behind in our GZSID setting. The reason is that their dataset splitting assigns less importance to the seen intents, where the performance of +LOF is relatively poor.

Ablation Study. We can observe from Table 4 that: 1) Our similarity scorer clearly outperforms the word embedding based intent similarity, especially in the ZSID setting. The advantage of SS over ES is less obvious in GZSID because the model trained with CTIR can directly perform GZSID, which makes the effect of the similarity matrix less significant. This can also explain for the phenomenon that w/o SS outperforms CNN+CTIR in CLINC under GZSID setting. 2) In spite of this, removing the similarity scorer has a negative impact on the model performance. 3) Multi-task learning consistently benefits the CTIR framework in all circumstances. For GZSID, the improvement directly comes from the disentangling effect of MT. For ZSID, we attribute the improvement to the well-learned relationship between seen and unseen intents, which is a side-effect of increasing the distance between unseen and seen intent representations. We also investigate the performance variation with the increase of α and λ', please see Appendix D[1] for details.

5 Related Work

Zero-shot Learning. In computer vision, ZSL is a well-developed sub-field, where a common approach is to relate unseen classes with seen classes through visual attributes [Farhadi *et al.*, 2009; Parikh and Grauman, 2011] or word2vec representations of the class names [Mikolov *et al.*, 2013; Frome *et al.*, 2013; Socher *et al.*, 2013]. For ZSID, transformation-based methods [Xia *et al.*, 2018; Liu *et al.*, 2019] calculate the inter-intent similarity based on the word embeddings of intent labels, and use it to transform the predictions from seen intents to unseen intents. Compatibility-based methods [Chen *et al.*, 2016; Kumar *et al.*, 2017] attempt to learn a shared semantic space for label names and utterances from seen data, and then measure the similarity between a test utterance and each unseen label in this space. There are also studies resorting to external knowledge, e.g., label ontologies [Ferreira *et al.*, 2015] or human-defined attributes [Yazdani and Henderson, 2015; Zhang *et al.*, 2019], which, however, are laborious to obtain.

Generalized Zero-shot Learning. Socher [2013] and Zhang [2016] design different model architectures to consider the probability of each sample coming from an unseen class before final classifying. In the task of intent detection, ReCapsNet-ZS [Liu *et al.*, 2019] enhanced CapsNet [Xia *et al.*, 2018] in GZSID by modeling the correlation between the dimensions of word embeddings, which can find a more accurate connection between seen and unseen intents. On the basis of ReCapsNet-ZS, Yan [2020] proposed a two-stage GZSID method that combines unknown intent detection and ZSID, which successfully resolves the domain-shift problem. However, this is at the cost of the performance in seen intents, which is the majority in real-world applications.

Class-transductive Zero-shot Learning. Class- transductive ZSL supports the use of semantic information (typically textual descriptions) about the unseen classes during training. In the CV field, class-transductive methods are used to infer the relationship between seen and unseen classes [Fu *et al.*, 2015b; Fu *et al.*, 2018] or directly predict the parameters of unseen intent classifiers [Elhoseiny *et al.*, 2013; Wang *et al.*, 2018]. In comparison, CTIR uses the unseen label names as training instances to learn unseen intent representations, which takes advantage of the fact that label names and utterances both come from the textual domain.

6 Conclusions

In this paper, we propose a class-transductive framework, CTIR, to overcome the limitations of existing ZSID models. CTIR utilizes the unseen label names as input utterances and includes the unseen classes into the prediction space during training. Under this framework, we present a multi-task learning objective in the training stage to encourage the model to learn the distinctions between unseen and seen intents. In the inference stage, we develop a similarity scorer, which can better associate the inter-intent connections based on the learned representations. Experiments on two benchmarks show that CTIR can bring considerable improvement to ZSID systems with different ZSL strategies and backbone networks.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 61976207, No. 61906187)

References

[Chao *et al.*, 2016] Wei-Lun Chao, Soravit Changpinyo, Boqing Gong, and Fei Sha. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *ECCV*, 2016.

[Chen *et al.*, 2016] Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models. In *ICASSP*, 2016.

[Coucke *et al.*, 2018] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*, 2018.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[Elhoseiny *et al.*, 2013] Mohamed Elhoseiny, Babak Saleh, and Ahmed M. Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013.

[Farhadi *et al.*, 2009] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

[Ferreira *et al.*, 2015] Emmanuel Ferreira, Bassam Jabaian, and Fabrice Lefèvre. Zero-shot semantic parser for spoken language understanding. In *INTERSPEECH*, 2015.

[Frome *et al.*, 2013] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *NeurIPS*, 2013.

[Fu *et al.*, 2015a] Yanwei Fu, Timothy M Hospedales, Tao Xiang, and Shaogang Gong. Transductive multi-view zero-shot learning. *IEEE T PATTERN ANAL*, 2015.

[Fu *et al.*, 2015b] Zhen-Yong Fu, Tao A. Xiang, Elyor Kodirov, and Shaogang Gong. Zero-shot object recognition by semantic manifold distance. In *CVPR*, 2015.

[Fu *et al.*, 2018] Zhenyong Fu, Tao Xiang, Elyor Kodirov, and Shaogang Gong. Zero-shot learning on semantic class prototype graph. *IEEE T PATTERN ANAL*, 2018.

[Kumar *et al.*, 2017] Anjishnu Kumar, Pavankumar Reddy Muddireddy, Markus Dreyer, and Björn Hoffmeister. Zero-shot learning across heterogeneous overlapping domains. In *INTERSPEECH*, 2017.

[Larson *et al.*, 2019] 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. An evaluation dataset for intent classification and out-of-scope prediction. In *EMNLP*, 2019.

[Lin and Xu, 2019] Ting-En Lin and Hua Xu. Deep unknown intent detection with margin loss. In *ACL*, 2019.

[Liu and Lane, 2016] Bing Liu and Ian Lane. Attention-based recurrent neural network models for joint intent detection and slot filling. In *INTERSPEECH*, 2016.

[Liu *et al.*, 2019] Han Liu, Xiaotong Zhang, Lu Fan, Xuandi Fu, Qimai Li, Xiao-Ming Wu, and Albert YS Lam. Reconstructing capsule networks for zero-shot intent classification. In *EMNLP*, 2019.

[Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *ICLR (Workshop)*, 2013.

[Nam *et al.*, 2016] Jinseok Nam, Eneldo Loza Mencía, and Johannes Fürnkranz. All-in text: Learning document, label, and word representations jointly. In *AAAI*, 2016.

[Parikh and Grauman, 2011] Devi Parikh and Kristen Grauman. Relative attributes. In *ICCV*, 2011.

[Ravuri and Stolcke, 2015] Suman Ravuri and Andreas Stolcke. Recurrent neural network and lstm models for lexical utterance classification. In *INTERSPEECH*, 2015.

[Sabour *et al.*, 2017] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *NeurIPS*, 2017.

[Socher *et al.*, 2013] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *NeurIPS*, 2013.

[Wang *et al.*, 2018] Xiaolong Wang, Yufei Ye, and Abhinav Gupta. Zero-shot recognition via semantic embeddings and knowledge graphs. In *CVPR*, 2018.

[Wang *et al.*, 2019] Wei Wang, Vincent W. Zheng, Han Yu, and Chunyan Miao. A survey of zero-shot learning: Settings, methods, and applications. *ACM Trans. Intell. Syst. Technol.*, 2019.

[Xia *et al.*, 2018] Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip S. Yu. Zero-shot user intent detection via capsule neural networks. In *EMNLP*, 2018.

[Xian *et al.*, 2019] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly. *IEEE T PATTERN ANAL*, 2019.

[Yan *et al.*, 2020] Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert Y. S. Lam. Unknown intent detection using gaussian mixture model with an application to zero-shot intent classification. In *ACL*, 2020.

[Yazdani and Henderson, 2015] Majid Yazdani and James Henderson. A model of zero-shot learning of spoken language understanding. In *EMNLP*, 2015.

[Zhang *et al.*, 2016] Hanwang Zhang, Xindi Shang, Wenzhuo Yang, Huan Xu, Huanbo Luan, and Tat-Seng Chua. Online collaborative learning for open-vocabulary visual classifiers. In *CVPR*, 2016.

[Zhang *et al.*, 2019] Jingqing Zhang, Piyawat Lertvittayakumjorn, and Yike Guo. Integrating semantic knowledge to tackle zero-shot text classification. In *NAACL*, 2019.