Generalized Zero-shot Intent Detection via Commonsense Knowledge

A.B. Siddique, Fuad Jamour, Luxun Xu, Vagelis Hristidis
University of California, Riverside
msidd005,fuadj,lxu051{@ucr.edu}, vagelis@cs.ucr.edu

Abstract

Identifying user intents from natural language utterances is a crucial step in conversational systems that has been extensively studied as a supervised classification problem. However, in practice, new intents emerge after deploying an intent detection model. Thus, these models should seamlessly adapt and classify utterances with both seen and unseen intents – unseen intents emerge after deployment and they do not have training data. The few existing models that target this setting rely heavily on the scarcely available training data and overfit to seen intents data, resulting in a bias to misclassify utterances with unseen intents into seen ones. We propose RIDE: an intent detection model that leverages commonsense knowledge in an unsupervised fashion to overcome the issue of training data scarcity. RIDE computes robust and generalizable *relationship meta-features* that capture deep semantic relationships between utterances and intent labels; these features are computed by considering how the concepts in an utterance are linked to those in an intent label via commonsense knowledge. Our extensive experimental analysis on three widely-used intent detection benchmarks show that relationship meta-features significantly increase the accuracy of detecting both seen and unseen intents and that RIDE outperforms the state-of-the-art model for unseen intents.

1 Introduction

Virtual assistants such as Amazon Alexa and Google Assistant allow users to perform a variety of tasks (referred to as 'skills' in Alexa) through an intuitive natural language interface. For example, a user can set an alarm by simply issuing the utterance *"Wake me up tomorrow at 10 AM"* to a virtual assistant, and the assistant is expected to understand that the user's intent (i.e., "AddAlarm") is to invoke the alarm module, then set the requested alarm accordingly. Detecting the intent implied in a natural language utterance (i.e., *intent detection*) is typically the first step towards performing any task in conversational systems.

Intent detection (or classification) is a challenging task due to the vast diversity in user utterances. The challenge is further exacerbated in the more practically relevant setting where the full list of possible intents (or classes) is not available before deploying the conversational system, or intents are added over time. This setting is an instance of the *generalized zero-shot classification problem* (Felix et al., 2018): labeled training utterances are available for seen intents but are unavailable for unseen ones, and at inference time, models do not have prior knowledge on whether the utterances they receive imply seen or unseen intents; i.e., unseen intents emerge after deploying the model. This setting is the focus of this paper.

Little research has been conducted on building generalized zero-shot (GZS) models for intent detection, with little success. The authors in (Liu et al., 2019) proposed a dimensional attention mechanism into a capsule neural network (Sabour et al., 2017) and computing transformation matrices for unseen intents to accommodate the GZS setting. This model overfits to seen classes and exhibits a strong bias towards classifying utterances into seen intents, resulting in poor performance. Most recently, the authors in (Yan et al., 2020) extended the previous model by utilizing the density-based outlier detection algorithm LOF (Breunig et al., 2000), which allows distinguishing utterances with seen intents from those with unseen ones, which partially mitigates the overfitting issue. Unfortunately, the performance of this model is sensitive to that of LOF, which fails in cases where intent labels are semantically close.

We propose RIDE[1][2], a model for GZS intent

[1] RIDE: Relationship Meta-features Assisted Intent DEtection
[2] Currently under review. GitHub Code Repository will be shared upon acceptance

detection that utilizes commonsense knowledge to compute robust and generalizable unsupervised *relationship meta-features*. These meta-features capture deep semantic associations between an utterance and an intent, resulting in two advantages: (*i*) they significantly decrease the bias towards seen intents as they are similarly computed for both seen and unseen intents and (*ii*) they infuse commonsense knowledge into our model, which significantly reduces its reliance on training data without jeopardizing its ability to distinguish semantically close intent labels. Relationship meta-features are computed by analyzing how the phrases in an utterance are linked to an intent label via commonsense knowledge.

Figure 1 shows how the words (or phrases) in an example utterance are linked to the words in an example intent label through the nodes (i.e., concepts) of a commonsense knowledge graph. In this example, the link ⟨`look for`, $Synonym$, `find`⟩ indicates that `look for` and `find` are synonyms, and the links ⟨`feeling hungry`, $CausesDesire$, `eat`⟩ and ⟨`restaurant`, $UsedFor$, `eat`⟩ can be used to infer the existence of the link ⟨`feeling hungry`, $IsRelated$, `restaurant`⟩, which indicates that `feeling hungry` and `restaurant` are related. These two links, the direct and the inferred ones, carry a significant amount of semantic relatedness, which indicates that the given utterance-intent pair is compatible. Note that this insight holds regardless of whether the intent is seen or not. RIDE utilizes this insight to build relationship meta-feature vectors that quantify the relatedness between an utterance and an intent in an unsupervised fashion.

RIDE combines relationship meta-features with contextual word embeddings (Peters et al., 2018), and feeds the combined feature vectors into a trainable prediction function to finally detect intents in utterances. Thanks to our relationship meta-features, RIDE is able to accurately detect both seen and unseen intents in utterances. Our extensive experimental analysis using the three widely used benchmarks, SNIPS (Coucke et al., 2018), SGD (Rastogi et al., 2019), and MultiWOZ (Zang et al., 2020) show that our model outperforms the state-of-the-art model in detecting unseen intents in the GZS setting by at least 30.36%.

A secondary contribution of this paper is that we managed to further increase the accuracy of GZS intent detection by employing Positive-Unlabeled

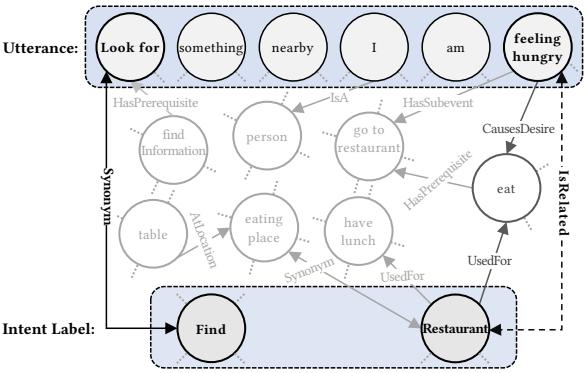


Figure 1: Example utterance, intent, and small commonsense knowledge graph. The presence of direct links such as ⟨`look for`, $Synonym$, `find`⟩ and inferred ones such as ⟨`feeling hungry`, $IsRelated$, `restaurant`⟩ between phrases in the utterance and those in the intent label indicate utterance-intent compatibility.

(PU) learning (Elkan and Noto, 2008) to predict if a new utterance belongs to a seen or unseen intent. PU learning assists intent detection models by mitigating their bias towards classifying most utterances into seen intents. A PU classifier is able to perform binary classification after being trained using only positive and unlabeled examples. We found that the use of a PU classifier also improves the accuracy of existing GZS intent detection works, but our model is again outperforming these works.

2 Preliminaries

2.1 Intent Detection

Let $\mathcal{S} = \{\mathcal{I}_1, \cdots, \mathcal{I}_k\}$ be a set of seen intents and $\mathcal{U} = \{\mathcal{I}_{k+1}, \cdots, \mathcal{I}_n\}$ be a set of unseen intents where $\mathcal{S} \cap \mathcal{U} = \varnothing$. Let $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_m\}$ be a set of labeled training utterances where each training utterance $\mathcal{X}_i \in \mathcal{X}$ is described with a tuple $(\mathcal{X}_i, \mathcal{I}_j)$ such that $\mathcal{I}_j \in \mathcal{S}$. An intent $\mathcal{I}_j$ is comprised of an *Action* and an *Object* and takes the form "ActionObject"[3] (e.g., "FindRestaurant"); an Action describes a user's request or activity and an Object describes the entity pointed to by an Action (Chen et al., 2013; Wang et al., 2015; Vedula et al., 2020). In both the zero-shot (ZS) and the GZS settings, the training examples have intent labels from set $\mathcal{S}$ only, however, the two settings differ as follows.

ZS Intent Detection. Given a test utterance $\mathcal{X}'_i$ whose true label $\mathcal{I}_j$ is known to be in $\mathcal{U}$ a priori, predict a label $\mathcal{I}'_j \in \mathcal{U}$.

[3]If intents are described using a complex textual description, Actions and Objects can be extracted using existing NLP tools such as dependency parsers.

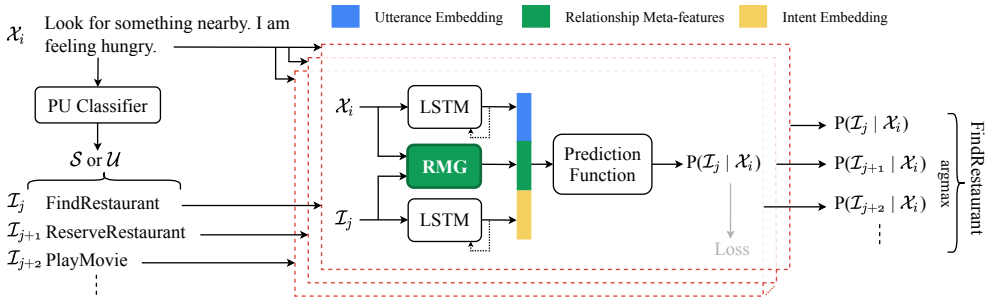


Figure 2: Overview of RIDE.

GZS Intent Detection. Given a test utterance $\mathcal{X}'_i$, predict a label $\mathcal{I}'_j \in \mathcal{S} \cup \mathcal{U}$. Note that unlike in the ZS setting, it is not known whether the true label of $\mathcal{X}'_i$ belongs to $\mathcal{S}$ or $\mathcal{U}$, which exacerbates the challenge in this setting; we focus on this setting in this paper.

2.2 Knowledge Graphs

Knowledge graphs (KG) are structures that capture relationships between entities, and are typically used to capture knowledge in a semi-structured format; i.e., they are used as knowledge bases. Knowledge graphs can be viewed as collections of triples, each representing a fact of the form $\langle$`head`, *relation*, `tail`$\rangle$ where `head` and `tail` describe entities and *relation* describes the relationship between the respective entities. In this work, we use ConceptNet (Speer et al., 2016), which is a rich and widely-used commonsense knowledge graph. Interested readers can check Appendix A.1 for more details on ConceptNet.

2.3 Link Prediction

While large knowledge graphs may capture a large subset of knowledge, they are incomplete: some relationships (or links) are missing. Link prediction (Kazemi and Poole, 2018) augments knowledge graphs by predicting missing relations using existing ones. In the context of this work, we pretrain a state-of-the-art link prediction model (LP) on the ConceptNet KG to score novel facts that are not necessarily present in the knowledge graph. Given a triple (i.e., fact) in the form $\langle$`head`, *relation*, `tail`$\rangle$, a link prediction model scores the triple with a value between 0 and 1, which quantifies the level of validity of the given triple. The details of training our link predictor are available in Appendix A.2.

2.4 Positive-Unlabeled Learning

Positive-Unlabeled (PU) classifiers learn a standard binary classifier in the unconventional setting where labeled negative training examples are unavailable. The state-of-the-art PU classifier (Elkan and Noto, 2008), which we integrate into our model, learns a decision boundary based on the positive and unlabeled examples, and thus can classify novel test examples into positive or negative. The aim of the PU classifier is to learn a probabilistic function $f(\mathcal{X}_i)$ that estimates $P(\mathcal{I}_j \in \mathcal{S} \mid \mathcal{X}_i)$ as closely as possible. In this work, we train a PU classifier using our training set (utterances with only seen intents labeled as positive) and validation set (utterances with both seen and unseen intents as unlabeled). We use 512-dimensions sentence embedding as features when using the PU classifier, generated using a pre-trained universal sentence encoder (Cer et al., 2018).

3 Our Approach

Figure 2 shows an overview of our model: given an input utterance $\mathcal{X}_i$, we first invoke the PU classifier (if it is available) to predict whether $\mathcal{X}_i$ implies a seen or an unseen intent; i.e., whether $\mathcal{X}_i$'s intent belongs to set $\mathcal{S}$ or $\mathcal{U}$. Then, an instance of our core model (the red box in Figure 2) is invoked for each intent in $\mathcal{S}$ or $\mathcal{U}$ based on the PU's prediction. Our core model predicts the level of compatibility between the given utterance $\mathcal{X}_i$ and intent $\mathcal{I}_j$, i.e., the probability that the given utterance implies the given intent $P(\mathcal{I}_j|\mathcal{X}_i) \in [0, 1]$. Finally, our model outputs the intent with the highest compatibility probability, i.e., $\text{argmax}_{\mathcal{I}_j} P(\mathcal{I}_j|\mathcal{X}_i)$.

Our core model concatenates relationship meta-features, utterance embedding, and intent embedding and feeds them into a trainable prediction function. The Relationship Meta-features Generator (RMG) is at the heart of our model, and it is the most influential component. Given an utterance

and an intent, RMG generates meta-features that capture deep semantic associations between the given utterance and intent in the form of a meta-feature vector.

3.1 Relationship Meta-feature Generation

RMG extracts relationship meta-features by utilizing the "ActionObject" structure of intent labels and commonsense knowledge graphs. Relationship meta-features are not only generalizable, but also discriminative: while the example utterance "Look for something nearby. I am feeling hungry." may be related to the intent "ReserveRestaurant", the Action part of this intent is not related to any phrase in the utterance; thus, "ReserveRestaurant" is less related to the utterance than "FindRestaurant".

RMG takes the following inputs: a set of relations in a knowledge graph (35 in the case of ConceptNet) $\mathcal{R} = \{r_1, r_2, .., r_t\}$; the set of n-grams $\mathcal{G}_i = \{g_1, g_2, .., g_q\}$ that correspond to the input utterance $\mathcal{X}_i$, where $|\mathcal{G}| = q$; and an intent label $\mathcal{I}_j = \{\mathcal{A}, \mathcal{O}\}$, where $\mathcal{A}$ and $\mathcal{O}$ are the Action and Object components of the intent, respectively. RMG computes a relationship meta-features vector in four steps, where each step results in a vector of size $|\mathcal{R}|$. The smaller vectors are: $\mathbf{e}^{\overrightarrow{\mathcal{A}}}_{\mathcal{X}_i}$, $\mathbf{e}^{\overrightarrow{\mathcal{O}}}_{\mathcal{X}_i}$, $\mathbf{e}^{\overleftarrow{\mathcal{A}}}_{\mathcal{X}_i}$, and $\mathbf{e}^{\overleftarrow{\mathcal{O}}}_{\mathcal{X}_i}$, where $\mathbf{e}^{\overrightarrow{\mathcal{A}}}_{\mathcal{X}_i}$ captures the Action to utterance semantic relationships and $\mathbf{e}^{\overrightarrow{\mathcal{O}}}_{\mathcal{X}_i}$ captures the Object to utterance relationships. The remaining two vectors capture relationships in the other direction; i.e., utterance to Action/Object, respectively. Capturing bi-directional relationships is important because a relationship in one direction does not necessarily imply one in the other direction – for example, $\langle$`table`, $AtLocation$, `restaurant`$\rangle$ does not imply $\langle$`restaurant`, $AtLocation$, `table`$\rangle$. The final output of RMG is the relationship meta-features vector $\mathbf{e}_{relationship}$, which is the concatenation of the four aforementioned vectors. We explain next how the smaller vectors is computed.

RMG computes $\mathbf{e}^{\overrightarrow{\mathcal{A}}}_{\mathcal{X}_i}$ by considering the strength of each relation in $\mathcal{R}$ between $\mathcal{A}$ and each n-gram in $\mathcal{G}_i$. That is, $\mathbf{e}^{\overrightarrow{\mathcal{A}}}_{\mathcal{X}_i}$ has $|\mathcal{R}|$ cells, where each cell corresponds to a relation $r \in \mathcal{R}$. Each cell is computed by taking $max(LP(\mathcal{A}, r, g))$ over all $g \in \mathcal{G}_i$. LP(`head`, $relation$, `tail`) outputs the probability that the fact represented by the triple $\langle$`head`, $relation$, `tail`$\rangle$ exists. The vector $\mathbf{e}^{\overrightarrow{\mathcal{O}}}_{\mathcal{X}_i}$ is computed similarly, but with passing $\mathcal{O}$ instead of $\mathcal{A}$ when invoking the link predictor; i.e., taking $max(LP(\mathcal{O}, r, g))$ over all $g \in \mathcal{G}_i$ to compute each cell. The vectors $\mathbf{e}^{\overleftarrow{\mathcal{A}}}_{\mathcal{X}_i}$ and $\mathbf{e}^{\overleftarrow{\mathcal{O}}}_{\mathcal{X}_i}$ are computed similarly, but with swapping the `head` and `tail` when invoking the link predictor; i.e., utterance phrases are passed as `head` and Action/Object parts are passed as `tail`. Algorithm 1 outlines the previous process. Finally, the generated meta-features are passed through a linear layer with sigmoid activation before concatenation with the utterance and intent embeddings.

Algorithm 1: RMG

Input: $\mathcal{R} = \{r_1, \cdots, r_t\}$: relations in KG
$\mathcal{G}_i = \{g_1, \cdots, g_q\}$: utterance n-grams
$\mathcal{I}_j = \{\mathcal{A}, \mathcal{O}\}$: intent's Action and Object

Output: $\mathbf{e}_{relationship}$: $\mathcal{X}_i$-$\mathcal{I}_j$ relationship meta-features

Let $\mathbf{e}^{\overrightarrow{\mathcal{A}}}_{\mathcal{X}_i}$ = RM ($\mathcal{A}$, $\mathcal{G}_i$, $\rightarrow$) // Action to utterance
Let $\mathbf{e}^{\overrightarrow{\mathcal{O}}}_{\mathcal{X}_i}$ = RM ($\mathcal{O}$, $\mathcal{G}_i$, $\rightarrow$) // Object to utterance
Let $\mathbf{e}^{\overleftarrow{\mathcal{A}}}_{\mathcal{X}_i}$ = RM ($\mathcal{A}$, $\mathcal{G}_i$, $\leftarrow$) // utterance to Action
Let $\mathbf{e}^{\overleftarrow{\mathcal{O}}}_{\mathcal{X}_i}$ = RM ($\mathcal{O}$, $\mathcal{G}_i$, $\leftarrow$) // utterance to Object
Let $\mathbf{e}_{relationship}$ = [$\mathbf{e}^{\overrightarrow{\mathcal{A}}}_{\mathcal{X}_i}$, $\mathbf{e}^{\overrightarrow{\mathcal{O}}}_{\mathcal{X}_i}$, $\mathbf{e}^{\overleftarrow{\mathcal{A}}}_{\mathcal{X}_i}$, $\mathbf{e}^{\overleftarrow{\mathcal{O}}}_{\mathcal{X}_i}$]
return $\mathbf{e}_{relationship}$

Function RM ($concept$, $phrases$, $direction$) :
  Let **e** = []
  foreach $r \in \mathcal{R}$ **do**
    if $direction = \rightarrow$ **then**
      Let p = Max ($LP(concept, r, g)$) for $g \in phrases$
    if $direction = \leftarrow$ **then**
      Let p = Max ($LP(g, r, concept)$) for $g \in phrases$
    e.append(p)
  return e

3.2 Utterance and Intent Encoders

Given an utterance $\mathcal{X}_i = \{w_1, w_2, \cdots, w_u\}$ with u words, first we compute an embedding $emb(w_j) \in \mathbb{R}^{dim}$ for each word w_j in the utterance, where $emb(w_j)$ is the concatenation of a contextual embedding obtained from a pre-trained ELMo model and parts of speech (POS) tag embedding. Then, we use bi-directional LSTM to produce a d-dimensional representation as follows:

$$\overrightarrow{\mathbf{h}}_j = \text{LSTM}_{fw}(\overrightarrow{\mathbf{h}}_{j-1}, emb(w_j)).$$

$$\overleftarrow{\mathbf{h}}_j = \text{LSTM}_{bw}(\overleftarrow{\mathbf{h}}_{j-1}, emb(w_j)).$$

Finally, we concatenate the output of the last hidden states as utterance embedding $\mathbf{e}_{utterance} = [\overrightarrow{\mathbf{h}}_u; \overleftarrow{\mathbf{h}}_u] \in \mathbb{R}^d$. We encode intent labels similarly to produce an intent embedding $\mathbf{e}_{intent} \in \mathbb{R}^d$.

3.3 Training the Model

Our model has two trainable components: the LSTM units in the utterance and intent encoders and the compatibility probability prediction function. We jointly train these components using training data prepared as follows. The training examples are of the form $((\mathcal{X}_i, \mathcal{I}_j), \mathcal{Y})$, where $\mathcal{Y}$ is a binary label representing whether the utterance-intent pair $(\mathcal{X}_i, \mathcal{I}_j)$ are compatible: 1 means they are compatible, and 0 means they are not. For example, the utterance-intent pair ("I want to play this song", "PlaySong") gets a label of 1, and the same utterance paired with another intent such as "BookHotel" gets a label of 0. We prepare our training data by assigning a label of 1 to the available utterance-intent pairs (where intents are seen ones); these constitute positive training examples. We create a negative training example for each positive one by corrupting the example's intent. We corrupt intents by modifying their Action, Object, or both; for example, the utterance-intent pair ("Look for something nearby. I am hungry.", "FindRestaurant") may result in the negative examples (..., "ReserveRestaurant"), (..., "FindHotel"), or (...,"RentMovies"). We train our core model by minimizing the cross-entropy loss over all the training examples.

4 Experimental Setup

In this section, we describe the datasets, evaluation settings and metrics, competing methods, and implementation details of our proposed method.

4.1 Datasets

Table 1 presents important statistics on the datasets we used in our experiments.

SNIPS (Coucke et al., 2018). A crowd-sourced single-turn NLU benchmark with 7 intents across different domains.

SGD (Rastogi et al., 2019). A recently published dataset for the eighth Dialog System Technology Challenge, Schema Guided Dialogue (SGD) track. It contains dialogues from 16 domains with a total of 46 intents. It is one of the most comprehensive and challenging publicly available datasets. The dialogues contain user intents as part of dialogue states. We only kept utterances where users express an intent by comparing two consecutive dialogue states to check for the expression of a new intent.

MultiWOZ (Zang et al., 2020). Multi-Domain Wizard-of-Oz (MultiWOZ) is a well-known and publicly available dataset. We used the most recent version 2.2 of MultiWOZ in our experiments, which contains utterances spanning 11 intents. Similarly to the pre-processing of SGD dataset, we only kept utterances that express an intent to maintain consistency with the previous work.

Dataset	SNIPS	SGD	MultiWOZ
Dataset Size	14.2K	57.2K	30.0K
Vocab. Size	10.8K	8.8K	9.7K
Avg. Length	9.05	10.62	11.07
# of Intents	7	46	11

Table 1: Dataset statistics.

4.2 Evaluation Methodology

We use standard classification evaluation measures: accuracy and F1 score. The values for all the metrics are per class averages weighted by their respective support. We present evaluation results for the following intent detection settings:

ZS intent Detection. In this setting, a model is trained on all the utterances with seen intents – i.e., all samples $(\mathcal{X}_i, \mathcal{I}_j)$ where $\mathcal{I}_j \in \mathcal{S}$. Whereas at inference time, the utterances are only drawn from those with unseen intents; the model has to classify a given utterance into one of the unseen intents. Note that this setting is less challenging than the GZS setting because models know that utterances received at inference time imply intents that belong to the set of unseen intents only, thus naturally reducing their bias towards classifying utterances into seen intents. For each dataset, we randomly place $\approx 25\%$, $\approx 50\%$, and $\approx 75\%$ of the intents in the seen set for training and the rest into the unseen set for testing, and report the average results over 10 runs. It is important to highlight that selecting seen/unseen sets in this fashion is more challenging to models because all the intents get an equal chance to appear in the unseen set, which exposes models that are capable of detecting certain unseen intents only.

GZS intent Detection. In this setting, models are trained on a subset of utterances implying seen intents. At inference time, test utterances are drawn from a set that contains utterances implying a mix of seen and unseen intents (disjoint set from training set) and the model is expected to select the correct intent from all seen and unseen intents for a given test utterance. This is the most realistic and challenging problem setting, and it is the main focus of this work. For the GZS setting, we decided the train/test splits for each dataset as follows: For

SNIPS, we first randomly selected 5 out of 7 intents and designated them as seen intents – the remaining 2 intents were designated as unseen intents. We then selected 70% of the utterances that imply any of the 5 seen intents for training. The test set consists of the remaining 30% utterances in addition to all utterances that imply one of the 2 unseen intents. Previous work (Liu et al., 2019) used the same number of seen/unseen intents, but selected the seen/unseen intents manually. Whereas we picked unseen intents randomly, and we report results over 10 runs resulting in a more challenging and thorough evaluation. That is, each intent gets an equal chance to appear as an unseen intent in our experiments, which allows testing each model more comprehensively. For SGD, we used the standard splits proposed by the dataset authors. Specifically, the test set includes utterances that imply 8 unseen intents and 26 seen intents; we report average results over 10 runs. For MultiWOZ, we used 70% of the utterance that imply 8 (out of 11) randomly selected intents for training and the rest of the utterances (i.e., the remaining 30% of seen intents' utterances and all utterances implying unseen intents) for testing.

4.3 Competing Methods

We compare our model RIDE against the following state-of-the-art (SOTA) models and several strong baselines:

SEG (Yan et al., 2020). A semantic-enhanced Gaussian mixture model that uses large margin loss to learn class-concentrated embeddings coupled with a density-based outlier detection algorithm LOF to detect unseen intents.

ReCapsNet-ZS (Liu et al., 2019). A model that employs capsule neural network and a dimensional attention module to learn generalizable transformational metrices from seen intents.

IntentCapsNet (Xia et al., 2018). A model that utilizes capsule neural networks to learn low-level features and routing-by-agreement to adapt to unseen intents. This model was originally proposed for detecting intents in the standard ZS setting. We extended it to support the GZS setting with the help of its authors.

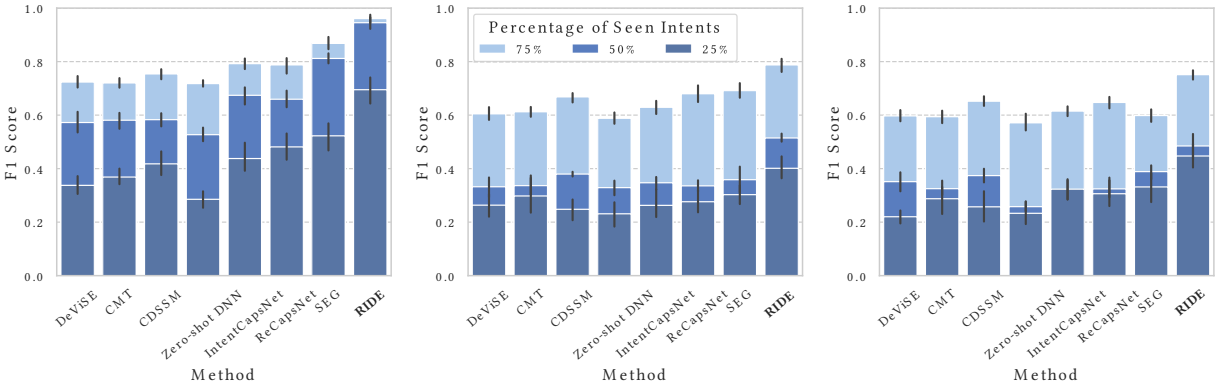
Other Baseline Models. (*i*) Zero-shot DDN (Kumar et al., 2017): A model for ZS intent detection that achieves zero-shot capabilities by projecting utterances and intent labels into the same high dimensional embedding space. (*ii*) CDSSM (Chen et al., 2016): A model for ZS intent detection that utilizes a convolutional deep structured semantic model to generate embeddings for unseen intents. (*iii*) CMT (Socher et al., 2013): A model for ZS intent detection that employs non-linearity in the compatibility function between utterances and intents to find the most compatible unseen intents. (*iv*) DeViSE (Frome et al., 2013): A model that was originally proposed for zero-shot image classification that learns a linear compatibility function. Note that baseline ZS models have been extended to support GZS setting.

4.4 Implementation Details

We lemmatize ConeptNet KG, that has 1 million nodes (English only after lemmatization), 2.7 million edges, and 35 relation types. The link predictor is trained on the lemmatized version of ConceptNet KG. The link predictor has two 200-dimensional embedding layers and a negative sampling ratio of 10; it is trained for $1,000$ epochs using Adam optimizer with a learning rate of 0.05, L2 regularization value of 0.1, and batch size of 4800. Our relationship meta-features generator takes in an utterance's n-grams with n ≤ 4 and an intent label, and uses the pre-trained link predictor to produce relationship meta-features with 140 dimensions. Our utterance and intent encoders use pre-trained ELMo contextual word embeddings with 1024 dimension and POS tags embeddings with 300 dimension, and a two-layer RNN with 300-dimensional bidirectional LSTM as recurrent units. Our prediction function has two dense layers with relu and softmax activation. Our core model is trained for up to 200 epochs with early stopping using Adam optimizer and a cross entropy loss with initial learning rate of 0.001 and ReduceLROnPlateau scheduler (PyTorch, 2020) with 20 patience epochs. It uses dropout rate of 0.3 and batch size of 32. A negative sampling ratio of up to 6 is used. We use the same embeddings generation and training mechanism for all competing models.

5 Results

Standard ZS Intent Detection. Figure 3 presents the F1 scores averaged over 10 runs for all competing models with varying percentages of seen intents in the ZS setting. The performance of all models improves as the percentage of seen intents increases, which is expected because increasing the percentage of seen intent gives models ac-



(a) SNIPS dataset (b) SGD dataset (c) MultiWOZ dataset

Figure 3: F1 scores for competing models in the ZS setting with varying percentages of seen intents. In the ZS setting, utterances with only unseen intents are encountered at inference time, and models are aware of this. Our model RIDE consistently outperforms all other models for any given percentage of seen intents.

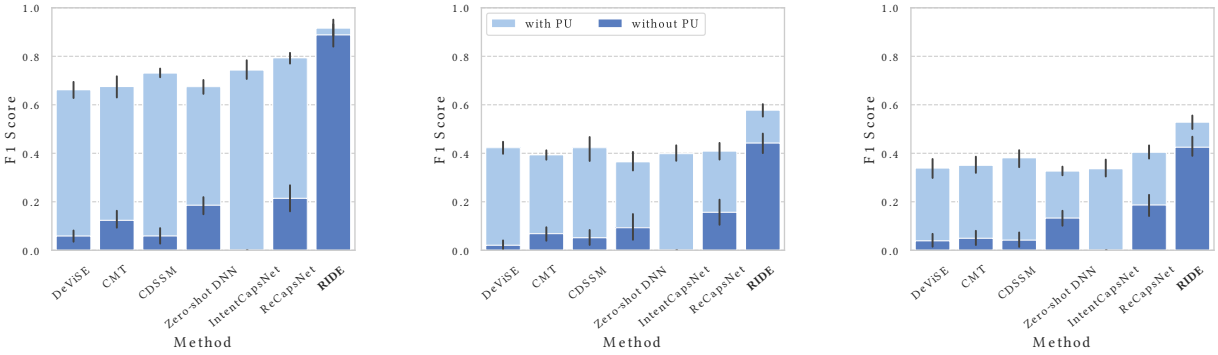
Method	SNIPS				SGD				MultiWOZ			
	Unseen		Seen		Unseen		Seen		Unseen		Seen	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
DeViSE	0.0311	0.0439	0.9487	0.6521	0.0197	0.0177	0.8390	0.5451	0.0119	0.0270	0.8980	0.5770
CMT	0.0427	0.0910	**0.9751**	0.6639	0.0254	0.0621	**0.9014**	0.5803	0.0253	0.0679	0.9025	0.6216
CDSSM	0.0521	0.0484	0.9542	0.7028	0.0367	0.0284	0.8890	0.6379	0.0373	0.0244	0.8861	0.6515
Zero-shot DNN	0.0912	0.1273	0.9437	0.6687	0.0662	0.1168	0.8825	0.6098	0.0802	0.1149	0.8940	0.6012
IntentCapsNet	0.0000	0.0000	0.9749	0.6532	0.0000	0.0000	0.8982	0.5508	0.0000	0.0000	**0.9249**	0.6038
ReCapsNet	0.1249	0.1601	0.9513	0.6783	0.1062	0.1331	0.8762	0.5751	0.1081	0.1467	0.8715	0.6170
SEG	0.6943	0.6991	0.8643	0.8651	0.3723	0.4032	0.6134	0.6356	0.3712	0.4143	0.6523	0.6456
RIDE w/o PU	0.8728	0.9103	0.8906	0.8799	0.3865	0.4634	0.8126	0.8295	0.3704	0.4645	0.8558	0.8816
RIDE $/w$ PU	**0.9051**	**0.9254**	0.9179	**0.9080**	**0.5901**	**0.5734**	0.8315	**0.8298**	**0.5686**	**0.5206**	0.8844	**0.8847**

Table 2: Main results: accuracy and F1 scores for competing models in the GZS setting (i.e., models receive both seen and unseen intents at inference time, which makes the setting more challenging than the ZS setting). We present results for two variants of our model: RIDE w/o PU which does not use a PU classifier, and RIDE $/w$ PU which uses one. Our model consistently achieves the best F1 score for both seen and unseen intents across all datasets, regardless of whether the PU classifier is integrated or not.

cess to more training data and intents. Our model RIDE consistently outperforms the SOTA model SEG (Yan et al., 2020) and all other models in the ZS setting with a large margin across all the datasets. Specifically, it is at least 12.65% more accurate on F1 score than the second best model for any percentage of seen intents on all the datasets. Note that all models perform worse on SGD and MultiWOZ compared to SNIPS because these two datasets are more challenging: they contain closely related intent labels such as "FindRestaurant" and "FindHotel".

GZS Intent Detection. Table 2 shows accuracy and F1 scores averaged over 10 runs for all competing models in the GZS setting. For unseen intents, our model RIDE outperforms all other competing models on accuracy with a large margin. Specifically, RIDE is 30.36%, 58.50%, and 53.18% more accurate than the SOTA model SEG on SNIPS, SGD, and MultiWOZ for unseen intents, respectively. Moreover, our model consistently achieves the highest F1 score on seen as well as unseen intents, which confirms its generalizability. CMT and IntentCapsNet achieve the highest accuracy for utterances with seen intents on all datasets, but their F1 score is among the worst due to their biasedness towards misclassifying utterances with unseen intents into seen ones. RIDE outperforms the SOTA model SEG regardless of whether a PU classifier is incorporated or not. For SNIPS, the role of the PU classifier is negligible as it causes a slight improvement in accuracy and F1 score. For SGD and MultiWOZ, which are more challenging datasets, the PU classifier is responsible for significant improvements in accuracy. Specifically, it provides 20.36 and 19.82 percentage points improvement for SGD and MultiWOZ, respectively, on unseen intents.

Effect of PU Classifier on Other Models. We observed that one of the main sources of error for most models in the GZS setting is their tendency to misclassify utterances with unseen intents into seen ones due to overfitting to seen intents. We investigated whether existing models can be adapted to



(a) SNIPS dataset (b) SGD dataset (c) MultiWOZ dataset

Figure 4: F1 scores for unseen intents for the competing models in the GZS setting after integrating a PU classifier.

Configuration	SNIPS	SGD	MultiWOZ
UI-Embed w/o PU	0.2367	0.1578	0.1723
Rel-M w/o PU	0.7103	0.3593	0.3321
RIDE w/o PU	0.9103	0.4634	0.4645
UI-Embed $/w$ PU	0.7245	0.4202	0.4124
Rel-M $/w$ PU	0.8463	0.5167	0.4781
RIDE $/w$ PU	**0.9254**	**0.5734**	**0.5206**

Table 3: Ablation study: F1 scores for unseen intents in GZS setting; the key reason behind our model's astonishing accuracy is our relationship meta-features.

accurately classify utterances with unseen intents by partially eliminating their bias towards seen intents. Figure 4 presents F1 scores of all the models with and without PU classifier. A PU classifier significantly improves the results of all the competing models. For instance, the IntentCapsNet model with a PU classifier achieves an F1 score of 74% for unseen intents on SNIPS dataset in the GZS setting compared to an F1 score of less than 0.01% without the PU classifier. Note that the PU classifier has an accuracy (i.e., correctly predicting whether the utterance implies a seen or an unseen intent) of 93.69 and an F1 score of 93.75 for SNIPS dataset; 86.13 accuracy and 83.54 F1 score for SGD dataset; and 87.32 accuracy and 88.51 F1 score for MultiWOZ dataset. Interestingly, our model RIDE (without PU classifier) outperforms all the competing models even when a PU classifier is incorporated into them, which highlights that the PU classifier is not the main source of the performance of our model. We did not incorporate the PU classifier into SEG model because it already incorporates an equivalent mechanism to distinguish seen intents from unseen ones (i.e., outlier detection).

Ablation Study. To quantify the effectiveness of each component in our model, we present the results of our ablation study in Table 3 (in the GZS setting). Utilizing utterance and intent embeddings only (i.e., UI-Embed) results in very low F1 score, i.e., 23.67% on SNIPS dataset. Employing relationship meta-features only (i.e., Rel-M) results in significantly better results: an F1 score of 71.03% on SNIPS dataset. When utterance and intent embeddings are used in conjunction with relationship meta-features (i.e., RIDE w/o PU), it achieves better F1 score compared to the Rel-M or UI-Embed configurations. A similar trend can be observed for the other datasets as well. Finally, when our entire model is deployed (i.e., including utterance and intent embeddings, relationship meta-features, and the PU classifier, i.e., RIDE $/w$ PU), it achieves the best results on all the datasets.

6 Related Work

The deep neural networks have proved highly effective for many critical NLP tasks (Siddique et al., 2020; Farooq et al., 2020; Zhang et al., 2018; Williams, 2019; Ma et al., 2019; Siddique et al., 2021; Liu and Lane, 2016; Gupta et al., 2019). We organize the related work on intent detection into three categories: (*i*) supervised intent detection, (*ii*) standard zero-shot intent detection, and (*iii*) generalized zero-shot intent detection.

Supervised Intent Detection. Recurrent neural networks (Ravuri and Stolcke, 2015) and semantic lexicon-enriched word embeddings (Kim et al., 2016) have been employed for supervised intent detection. Recently, researchers have proposed solving the related problems of intent detection and slot-filling jointly (Liu and Lane, 2016; Zhang et al., 2018; Xu and Sarikaya, 2013). Supervised intent classification works assume the availability of a large amount of labeled training data for all intents to learn discriminative features, whereas

we focus on the more challenging and more practically relevant setting where intents are evolving and training data is not available for all intents.

Standard Zero-shot Intent Detection. The authors in (Yazdani and Henderson, 2015) proposed using label ontologies (Ferreira et al., 2015) (i.e., manually annotated intent attributes) to facilitate generalizing a model to support unseen intents. The authors in (Dauphin et al., 2013; Kumar et al., 2017; Williams, 2019) map utterances and intents to the same semantic vector space, and then classify utterances based on their proximity to intent labels in that space. Similarly, the authors in (Gangal et al., 2019) employ the outlier detection algorithm LOF (Breunig et al., 2000) and likelihood ratios for identifying out-of-domain test examples. While these works showed promising results for intent detection when training data is unavailable for some intents, they assume that all utterances faced at inference time imply unseen intents only. Extending such works to remove the aforementioned assumption is nontrivial. Our model does not assume knowledge of whether an utterance implies a seen or an unseen intent at inference time.

Generalized Zero-shot Intent Detection. To the best of our knowledge, the authors in (Liu et al., 2019) proposed the first work that specifically targets the GZS intent detection setting. They attempt to make their model generalizable to unseen intents by adding a dimensional attention module to a capsule network and learning generalizable transformation matrices from seen intents. Recently, the authors in (Yan et al., 2020) proposed using a density-based outlier detection algorithm LOF (Breunig et al., 2000) and semantic-enhanced Gaussian mixture model with large margin loss to learn class-concentrated embeddings to detect unseen intents. In contrast, we leverage rich commonsense knowledge graph to capture deep semantic and discriminative relationships between utterances and intents, which significantly reduces the bias towards classifying unseen intents into seen ones. In a related, but orthogonal, line of research, the authors in (Ma et al., 2019; Li et al., 2020; Gulyaev et al., 2020) addressed the problem of intent detection in the context of dialog state tracking where dialog state and conversation history are available in addition to an input utterance. In contrast, this work and the SOTA models we compare against in our experiments only consider an utterance without having access to any dialog state elements.

7 Conclusion

We have presented an accurate generalized zero-shot intent detection model. Our extensive experimental analysis on three intent detection benchmarks shows that our model is 30.36% to 58.50% more accurate than the SOTA model for unseen intents. The main novelty of our model is its utilization of relationship meta-features to accurately identify matching utterance-intent pairs with very limited reliance on training data, and without making any assumption on whether utterances imply seen or unseen intents at inference time. Furthermore, our idea of integrating Positive-Unlabeled learning in GZS intent detection models further improves our models' performance, and significantly improves the accuracy of existing models as well.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104.

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

Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. 2016. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6045–6049. IEEE.

Zhiyuan Chen, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Identifying intention posts in discussion forums. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1041–1050.

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

Yann N Dauphin, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2013. Zero-shot learning for semantic utterance classification. *arXiv preprint arXiv:1401.0509*.

Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220.

Umar Farooq, A. B. Siddique, Fuad Jamour, Zhijia Zhao, and Vagelis Hristidis. 2020. App-aware response synthesis for user reviews. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 699–708.

Rafael Felix, Vijay BG Kumar, Ian Reid, and Gustavo Carneiro. 2018. Multi-modal cycle-consistent generalized zero-shot learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37.

Emmanuel Ferreira, Bassam Jabaian, and Fabrice Lefevre. 2015. Online adaptative zero-shot learning spoken language understanding using word-embedding. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5321–5325. IEEE.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129.

Varun Gangal, Abhinav Arora, Arash Einolghozati, and Sonal Gupta. 2019. Likelihood ratios and generative classifiers for unsupervised out-of-domain detection in task oriented dialog. *arXiv preprint arXiv:1912.12800*.

Pavel Gulyaev, Eugenia Elistratova, Vasily Konovalov, Yuri Kuratov, Leonid Pugachev, and Mikhail Burtsev. 2020. Goal-oriented multi-task bert-based dialogue state tracker. *arXiv preprint arXiv:2002.02450*.

Arshit Gupta, John Hewitt, and Katrin Kirchhoff. 2019. Simple, fast, accurate intent classification and slot labeling for goal-oriented dialogue systems. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 46–55.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in neural information processing systems*, pages 4284–4295.

Joo-Kyung Kim, Gokhan Tur, Asli Celikyilmaz, Bin Cao, and Ye-Yi Wang. 2016. Intent detection using semantically enriched word embeddings. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 414–419. IEEE.

Anjishnu Kumar, Pavankumar Reddy Muddireddy, Markus Dreyer, and Björn Hoffmeister. 2017. Zero-shot learning across heterogeneous overlapping domains. In *INTERSPEECH*, pages 2914–2918.

Miao Li, Haoqi Xiong, and Yunbo Cao. 2020. The sppd system for schema guided dialogue state tracking challenge. *arXiv preprint arXiv:2006.09035*.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *arXiv preprint arXiv:1609.01454*.

Han Liu, Xiaotong Zhang, Lu Fan, Xuandi Fu, Qimai Li, Xiao-Ming Wu, and Albert YS Lam. 2019. Reconstructing capsule networks for zero-shot intent classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4801–4811.

Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiying Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. 2019. An end-to-end dialogue state tracking system with machine reading comprehension and wide & deep classification. *arXiv preprint arXiv:1912.09297*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

PyTorch. 2020. torch.optim — pytorch 1.3.0 documentation. `https://pytorch.org/docs/stable/optim.html#torch.optim.lr_scheduler.ReduceLROnPlateau`. (Accessed on 11/12/2020).

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*.

Suman Ravuri and Andreas Stolcke. 2015. Recurrent neural network and lstm models for lexical utterance classification. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866.

A. B. Siddique, Samet Oymak, and Vagelis Hristidis. 2020. Unsupervised paraphrasing via deep reinforcement learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 1800–1809, New York, NY, USA. Association for Computing Machinery.

A.B. Siddique, Fuad Jamour, and Vagelis Hristidis. 2021. Linguistically-enriched and context-aware

zero-shot slot filling. In *Proceedings of the Web Conference 2021*, New York, NY, USA. Association for Computing Machinery.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*, pages 935–943.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2016. Conceptnet 5.5: An open multilingual graph of general knowledge. *arXiv preprint arXiv:1612.03975*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. International Conference on Machine Learning (ICML).

Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open intent extraction from natural language interactions. In *Proceedings of The Web Conference 2020*, pages 2009–2020.

Jinpeng Wang, Gao Cong, Wayne Xin Zhao, and Xiaoming Li. 2015. Mining user intents in twitter: A semi-supervised approach to inferring intent categories for tweets. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 318–324. AAAI Press.

Kyle Williams. 2019. Zero shot intent classification using long-short term memory networks. *Proc. Interspeech 2019*, pages 844–848.

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*.

Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 ieee workshop on automatic speech recognition and understanding*, pages 78–83. IEEE.

Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert YS Lam. 2020. Unknown intent detection using gaussian mixture model with an application to zero-shot intent classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1050–1060.

Majid Yazdani and James Henderson. 2015. A model of zero-shot learning of spoken language understanding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 244–249.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip S Yu. 2018. Joint slot filling and intent detection via capsule neural networks. *arXiv preprint arXiv:1812.09471*.

A Appendices

This section provides supplementary details on various aspects of this paper. First, we provide more details on the commonsense knowledge graph we have used as a source of knowledge on semantic relatedness of concepts. Then, we describe the specifics of the datasets we used in our evaluation and our preprocessing procedures. Finally, we provide the details on handling a special case when utterances do not imply intents.

A.1 Knowledge Graph Details

Although creating and maintaining knowledge graphs is laborious and time consuming, the immense utility of such graphs has led many researchers and institutions to make the effort of building and maintaining knowledge graphs in many domains, which lifts the burden off of other researchers and developers who utilize these graphs. For tasks that involve commonsense reasoning such as *generalized zero-shot* intent detection, the ConceptNet (Speer et al., 2016) commonsense knowledge graph stands out as one of the most popular and freely available resources. ConceptNet originated from the crowdsourcing project Open Mind Common Sense, and includes knowledge not only from crowdsourced resources but also expert-curated resources. It is available in 10 core languages, and 68 more common languages. It was employed to show state-of-the-art results at SemEval 2017 (Speer et al., 2017). In this work, we considered 35 relation types to generate our relationship meta-features. The relation types are: $RelatedTo$, $FormOf$, IsA, $PartOf$, $HasA$, $UsedFor$, $CapableOf$, $AtLocation$, $Causes$, $HasSubevent$, $HasFirstSubevent$, $HasLastSubevent$, $HasPrerequisite$, $HasProperty$, $MotivatedByGoal$, $ObstructedBy$, $Desires$, $CreatedBy$, $Synonym$, $Antonym$, $DistinctFrom$,

$DerivedFrom$, $SymbolOf$, $DefinedAs$, $MannerOf$, $LocatedNear$, $HasContext$, $SimilarTo$, $EtymologicallyRelatedTo$, $EtymologicallyDerivedFrom$, $CausesDesire$, $MadeOf$, $ReceivesAction$, $ExternalURL$, and $Self$.

The relationship meta-feature generator produces $35 \times 4 = 140$ dimension vector for each utterance-intent pair. Specifically, we generate relationships: (*i*) from utterance to Object (i.e., Object part in intent label); (*ii*) utterance to Action (i.e., Action part in intent label); and (*iii*) Object to utterance; (*iv*) Action to utterance.

A knowledge graph may not have redundant, but necessary information. For example, a knowledge graph may have the entry $\langle$`movie`, IsA, `film`$\rangle$ but not $\langle$`film`, IsA, `movie`$\rangle$ or vice-versa, because one triple can be inferred from the other based on background knowledge (i.e., symmetric nature of the IsA relation). Similarly, the triple $\langle$`movie`, $HasA$, `subtitles`$\rangle$ can be used to infer the triple $\langle$`subtitles`, $PartOf$, `movie`$\rangle$ based on the background knowledge (i.e., inverse relation between $HasA$ and $PartOf$). So, if this kind of redundant information (i.e., complementing entries for all such triples) is not available in the knowledge graph itself, there is no way for the model to learn these relationships automatically. To overcome this issue, we incorporate the background knowledge that each of the relation types IsA, $RelatedTo$, $Synonym$, $Antonym$, $DistinctFrom$, $LocatedNear$, $SimilarTo$, and $EtymologicallyRelatedTo$ is symmetric; and that the relation types $PartOf$ and $HasA$ are inversely related in our link prediction model as described in (Kazemi and Poole, 2018).

A.2 Training the Link Predictor

The training data for a link prediction model is prepared as follows. First, the triples in the input knowledge graph are assigned a label of 1. Then, negative examples are generated by corrupting true triples (i.e., modifying the `head` or `tail` of existing triples) and assigning them a label of -1 (Bordes et al., 2013). Finally, we train our LP using the generated training data by minimizing the L2 regularized negative log-likelihood loss of training triples (Trouillon et al., 2016).

A.3 Datasets Preprocessing

SNIPS Natural Language Understanding benchmark (SNIPS) (Coucke et al., 2018) is a commonly used dataset for intent detection, whereas Dialogue System Technology Challenge 8, Schema Guided Dialogue dataset (SGD) (Rastogi et al., 2019) and Multi-Domain Wizard-of-Oz (MultiWOZ) (Zang et al., 2020) were originally proposed for the task of dialogue state tracking. For SGD and MultiWOZ, we perform a few trivial preprocessing steps to extract utterances that contain intents, along with their labels, and use them for the task of generalized zero-shot intent detection. First, we provide the details on the preprocessing steps specific to the SGD and MultiWOZ dataset and then describe the preprocessing steps that are common for all datasets.

Steps for SGD and MultiWOZ. To maintain consistency with the previous work on intent detection, we extract only the utterances where user/system expresses an intent, and discard the rest from the original SGD and MultiWOZ datasets. The dialogue state contains a property "active_intent" that keeps track of the user's current intent. After each user utterance, we compare dialogue states to check for the expression of a new user intent, i.e., whether the value of the "active_intent" is modified. Whenever the user expresses a new intent, the value of the "active_intent" is updated. Moreover, sometimes, the bot (i.e., system) also offers new intents to the user (e.g., offering reserving a table to the user, who has successfully searched for a restaurant), which is tracked in the system actions property "act = OFFER_INTENT", and "values = <new_intent>". We also keep such system utterances.

Common Steps. We perform some standard preprocessing steps on all the datasets. We use spaCy to tokenize the sentences. Since intent labels are given in the "ActionObject" format, we tokenize them into "Action Object" phrases before feeding them into our model. For example, the intent labels "FindHotel" and "RateBook", are transformed into "Find Hotel" and "Rate Book", respectively. Note that some Objects parts of intent labels are compound. Consider the intent label "SearchOneWayFlight" whose Action is "Search" and Object is "OneWayFlight". In such cases, our relationship meta-features generator computes meta-features for each part of the compound object then averages them to produce the Object meta-features vector. In the previous example, "OneWayFlight" meta-features vector is computed as the average of the meta-features of "OneWay" and "Flight".

Method	SGD	MultiWOZ
CNN	0.9497	0.9512
GRU	**0.9528**	0.9619
LSTM	0.9512	0.9607
Bi-LSTM	0.9525	**0.9621**

Table 4: F1 score for intent existence binary classifiers.

A.4 Intent Existence Prediction

In real human-to-human or human-to-machine conversations, utterances do not necessarily imply intents. Most existing intent detection models formulate the problem as a classification problem where utterances are assumed to imply an intent, which limits utility of such models in practice. In what follows, we describe a simple method for extending intent detection models (including ours) to accommodate the case when utterances do not necessarily imply intents. We propose to do binary classification as a first step in intent detection, where a binary classifier is used to identify utterances that do not imply an intent. To validate the viability of this proposal, we experimented with several binary classifiers. To train the classifiers, we created datasets of positive and negative examples from seen intents data; positive examples are utterances that imply intents, and negative examples are utterances that do not have intents (See Section A.3 for details on identifying utterances that imply intents). For the SGD dataset, we used the standard train/test splits, and for the MultiWOZ dataset, we used the same splits described in the GZS setting. We report in Table 4 the average F1 score over 5 runs of several binary classifiers for the SGD and the MultiWOZ datasets. All classifiers use ELMo (Peters et al., 2018) and POS tag embeddings. These results show that intent existence classification can be done accurately using the available training data; consequently, intent detection models can be easily and reliably extended to support the case when some input utterances do not imply intents.