An Explicit-Joint and Supervised-Contrastive Learning Framework for Few-Shot Intent Classification and Slot Filling

Han Liu[1,2] **Feng Zhang**[1,2,3] **Xiaotong Zhang**[1,2*]
Siyang Zhao[1,2] **Xianchao Zhang**[1,2]
School of Software, Dalian University of Technology[1]
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province[2]
School of Electronics Engineering and Computer Science, Peking University[3]
{hanliu,zhangxt}@dlut.edu.cn, zhangfeng@stu.pku.edu.cn
zhao_siyang@mail.dlut.edu.cn, xczhang@dlut.edu.cn

Abstract

Intent classification (IC) and slot filling (SF) are critical building blocks in task-oriented dialogue systems. These two tasks are closely-related and can flourish each other. Since only a few utterances can be utilized for identifying fast-emerging new intents and slots, data scarcity issue often occurs when implementing IC and SF. However, few IC/SF models perform well when the number of training samples per class is quite small. In this paper, we propose a novel explicit-joint and supervised-contrastive learning framework for few-shot intent classification and slot filling. Its highlights are as follows. (i) The model extracts intent and slot representations via bidirectional interactions, and extends prototypical network to achieve explicit-joint learning, which guarantees that IC and SF tasks can mutually reinforce each other. (ii) The model integrates with supervised contrastive learning, which ensures that samples from same class are pulled together and samples from different classes are pushed apart. In addition, the model follows a not common but practical way to construct the episode, which gets rid of the traditional setting with fixed way and shot, and allows for unbalanced datasets. Extensive experiments on three public datasets show that our model can achieve promising performance.

1 Introduction

With the vigorous development of conversational AI, task-oriented dialogue systems have been widely-used in many applications, e.g., virtual personal assistants like Apple Siri and Google Assistant, and chatbots deployed in various domains (Liu et al., 2019a; Yan et al., 2020). Intent classification (IC) and slot filling (SF) are key components in task-oriented dialogue systems, and their performance will directly affect the downstream dialogue management and natural language generation tasks (Xu and Sarikaya, 2013). Traditional IC/SF models have achieved impressive performance (Gupta et al., 2019), but they often require large amount of labeled instances per class, which is expensive and unachievable in industry especially in the initial phase of a dialogue system.

Few-shot learning aims to solve the data scarcity issue, which can recognize novel categories effectively with only a handful of labeled samples by leveraging the prior knowledge learned from previous categories. Most few-shot learning studies concentrate on computer vision domain (Fei-Fei et al., 2006; Finn et al., 2017; Jung and Lee, 2020). Recently, to handle various new or unacquainted intents popped up quickly from different domains, some few-shot IC/SF models are proposed (Geng et al., 2020; Hou et al., 2020). Nevertheless, these methods usually focus on a single task and do not attempt to address these two tasks simultaneously.

Intuitively, IC and SF are two complementary tasks and the information of one task can be utilized in the other task to improve the performance. Existing joint IC and SF models have achieved impressive performance in supervised learning scenarios (Weld et al., 2021). But only a couple of methods are custom-designed for few-shot joint IC and SF task. Krone et al. (2020) directly apply the popular few-shot learning models MAML and prototypical network to explore the few-shot joint IC and SF. During the same period, Bhathiya and Thayasivam (2020) also attempt to utilize MAML to deal with this problem in a similar way. Though these models outperform the single task model, they just implicitly model the relationship between IC and SF. The mutual interaction between IC and SF in these methods is still unknowable, which seems to be a black box (not using a concrete formula to characterize the interaction), thus difficult to analyze the internal mechanism.

In this paper, we propose to model the relationship between IC and SF precisely and clearly, as

[*] Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 1945–1955
November 7–11, 2021. ©2021 Association for Computational Linguistics

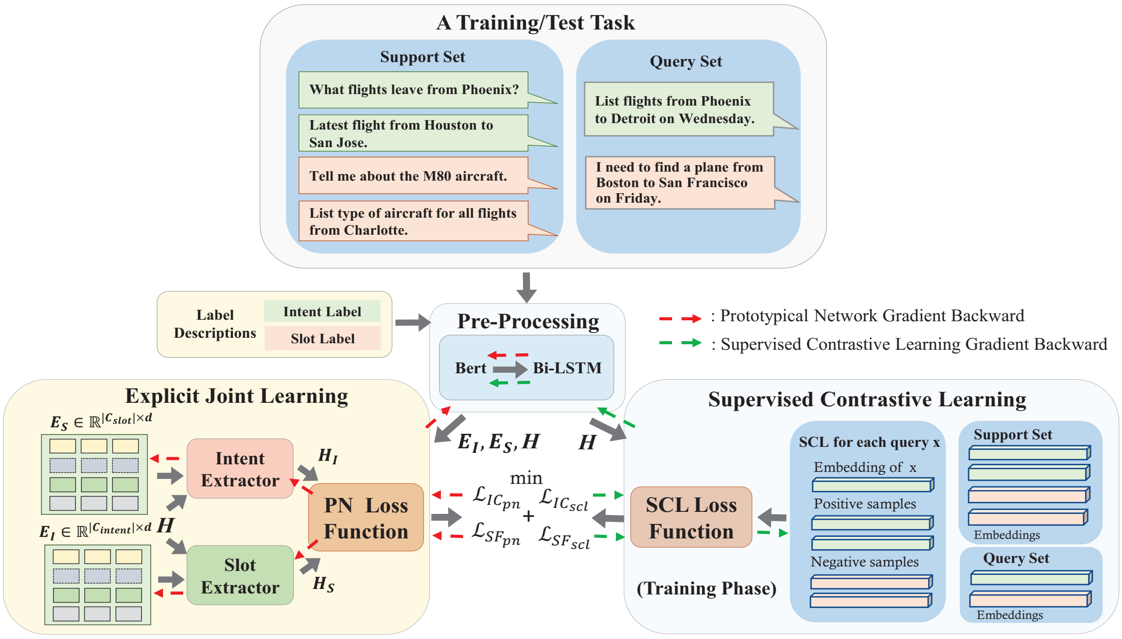

A Training/Test Task
Support Set
What flights leave from Phoenix?
Latest flight from Houston to San Jose.
Tell me about the M80 aircraft.
List type of aircraft for all flights from Charlotte.
Query Set
List flights from Phoenix to Detroit on Wednesday.
I need to find a plane from Boston to San Francisco on Friday.
Label Descriptions
Intent Label
Slot Label
Pre-Processing
Bert
Bi-LSTM
: Prototypical Network Gradient Backward
: Supervised Contrastive Learning Gradient Backward
Explicit Joint Learning
$E_S \in \mathbb{R}^{|C_{slot}| \times d}$
$E_I \in \mathbb{R}^{|C_{intent}| \times d}$
H
Intent Extractor
Slot Extractor
H_I
H_S
PN Loss Function
E_I, E_S, H
H
min
$\mathcal{L}_{IC_{pn}}$
$\mathcal{L}_{IC_{scl}}$
+
$\mathcal{L}_{SF_{pn}}$
$\mathcal{L}_{SF_{scl}}$
Supervised Contrastive Learning
SCL Loss Function
(Training Phase)
SCL for each query x
Embedding of x
Positive samples
Negative samples
Support Set
Embeddings
Query Set
Embeddings

Figure 1: Illustration of our framework. In the training process, labeled utterances from support set and query set are first encoded by pre-processing module. Meanwhile, intent and slot labels' descriptions are fed into pre-processing module to generate intent embedding matrix and slot embedding matrix. Then the two matrices and utterance's embedding are fed into explicit joint learning module, while utterance's embedding is put forward into supervised contrastive learning module. In explicit joint learning module, intent and slot extractors are used to extract intent and slot information, which leverage the attention mechanism. Then, we can obtain slot-attention-based intent representation and intent-attention-based slot representation. Next, prototypical network uses intent labels to guide slot embedding learning and vice versa. In supervised contrastive learning module, we construct contrastive samples for each query instance using support set. And the SCL loss function can push samples from the same class more close and samples from different classes further apart. In the testing process, prototypical network is used to predict intent and slot labels, while supervised contrastive learning module is disabled.

well as integrating with contrastive learning. As illustrated in Figure 1, our framework consists of two main components. First, we present an explicit-joint learning framework for few-shot intent classification and slot filling, which effectively utilizes the bidirectional connection between IC and SF via leveraging slot-attention-based intent representation and intent-attention-based slot representation. In addition, we integrate with supervised contrastive learning to obtain more class-discriminative embeddings, which is a strong complementary part to improve our framework.

To verify the effectiveness of the proposed model, we conduct extensive experiments on three public datasets. Catering to the unbalanced datasets and very limited labeled samples in real application scenarios, we adopt a not common but practical way to construct the episode for few-shot learning, i.e., in each episode, the way and shot are variable. The empirical study validates our proposal and shows promising results of our framework on IC and SF tasks.

2 Related Work

Few-shot learning Few-shot learning aims to use the knowledge learned from seen classes, of which abundant labeled samples are available for training, to recognize unseen classes, of which limited labeled samples are provided (Wang et al., 2020a). It has been widely studied in computer vision such as classification (Fei-Fei et al., 2006; Wang et al., 2020b), segmentation (Wang et al., 2019; Rakelly et al., 2018) and generation (Liu et al., 2019b). Recently it has been expanded to natural language processing such as intent detection (Yu et al., 2021; Kumar et al., 2021).

Few-shot classification is an important and challenging task. Several methods have been proposed to tackle this problem. In particular, several metric-based methods (Vinyals et al., 2016; Snell et al., 2017; Yu et al., 2018; Geng et al., 2019; Bao et al., 2020) have been proposed, which first learn an embedding space and then utilize a metric to classify instances of new categories according to proximities with the labeled examples. In addition to metric-based methods, some optimization-based approaches (Ravi and Larochelle, 2017; Finn et al., 2017; Yoon et al., 2018) have also been explored for few-shot classification.

Contrastive learning Contrastive learning applied to self-supervised representation learning has seen a resurgence of interest in recent years, leading to state-of-the-art performance in unsupervised training of deep image models (Chen et al., 2020). Khosla et al. (2020) extend the self-supervised batch contrastive approach to the fully-supervised setting, allowing us to effectively leverage label information. Recently, Gunel et al. (2020) propose a novel objective function that contains a supervised contrastive learning term for fine-tuning pre-trained language models, which improves the model generalization ability significantly.

Joint intent classification and slot filling Due to the close relationship between IC and SF, Liu and Lane (2016); Zhang and Wang (2016); Goo et al. (2018); Qin et al. (2019, 2021) propose joint models to consider the correlation between these two tasks. These models can be classified into two categories. One type of approaches (Liu and Lane, 2016; Zhang and Wang, 2016) adopt a multi-task framework to solve these two tasks simultaneously. Although these models outperform the single-task model, they just model the relationship implicitly by sharing the encoder parameters. The other type of approaches (Goo et al., 2018; Qin et al., 2019) explicitly adopt the intent information to guide the slot filling task. Qin et al. (2021) further propose a co-interactive transformer which considers the cross-impact between these two tasks. These explicit-joint learning models have achieved very remarkable performance, but they mainly focus on the traditional supervised learning setting.

3 Problem Definition

A labeled utterance with T words (tokens) can be represented as (x, t, y), where $x = (w_1, w_2, ..., w_T)$ is an utterance with T words, $t = (t_1, t_2, ..., t_T)$ is composed of slot labels of each word in x, y is the intent label of x. In this paper, few-shot classification is conducted via episode learning strategy. In the training period, we partition the training set into multiple episodes. Each episode consists of a *support set* $\mathcal{S}$ and a *query set* $\mathcal{Q}$. In particular, we randomly select N classes from the training classes, and obtain a class set $\mathcal{C}$ in each episode. Then the support set is formed by randomly selecting k_c labeled samples (utterances) from each of the N classes, i.e., $\mathcal{S} = \bigcup_{c \in \mathcal{C}} \mathcal{S}_c$, where $\mathcal{S}_c = \{(x^i, t^i, y_c) | i \in (1, ..., k_c)\}$. And a fraction of the remainder of these N classes' samples (k_q examples per class) serve as the query set, i.e., $\mathcal{Q} = \bigcup_{c \in \mathcal{C}} \mathcal{Q}_c$, where $\mathcal{Q}_c = \{(x^j, t^j, y_c) | j \in (1, ..., k_q)\}$. In the test period, we also partition the test set into multiple episodes. Each episode contains a support set $\mathcal{S} = \bigcup_{c \in \mathcal{C}} \mathcal{S}_c$, where $\mathcal{S}_c = \{(x^i, t^i, y_c) | i \in (1, ..., k_c)\}$, and a query set $\mathcal{Q} = \bigcup_{c \in \mathcal{C}} \mathcal{Q}_c$, where $\mathcal{Q}_c = \{x^j | j \in (1, ..., k_q)\}$. There is no overlap between the training classes and test classes. Table 1 summarizes the symbol explanation in details.

Symbol	Explanation
$\mathcal{C}$	set of intent classes in each episode
$\mathcal{S}$	support set of an episode
$\mathcal{Q}$	query set of an episode
$\mathcal{S}_c$	set of support data in the c-th class
$\mathcal{Q}_c$	set of query data in the c-th class
x	an utterance with T words, $x = (w_1, ..., w_T)$
t	slot labels of each word in x, $t = (t_1, ..., t_T)$
y	intent label of utterance x
k_c	number of supports in $\mathcal{S}_c$
k_q	number of queries in $\mathcal{Q}_c$
$\boldsymbol{H}$	pre-processed utterance embedding
$\boldsymbol{E}_I$	intent label embedding
$\boldsymbol{E}_S$	slot label embedding
$\boldsymbol{H}_I$	slot-attention-based intent representation
$\boldsymbol{H}_S$	intent-attention-based slot representation
$\boldsymbol{c}$	sentence embedding of utterance x

Table 1: Symbol explanation.

4 Approach

4.1 Pre-processing

Given an utterance $x = (w_1, w_2, ..., w_T)$ with T words (tokens), each word in the utterance can obtain its word embedding by BERT (Devlin et al., 2019). And each word can be further encoded using a recurrent neural network such as bidirectional LSTM, i.e.,

$$\begin{aligned} \overrightarrow{\boldsymbol{h}}_t &= \mathrm{LSTM}_{fw}(w_t, \overrightarrow{\boldsymbol{h}}_{t-1}), \\ \overleftarrow{\boldsymbol{h}}_t &= \mathrm{LSTM}_{bw}(w_t, \overleftarrow{\boldsymbol{h}}_{t+1}), \end{aligned} \tag{1}$$

where LSTM_{fw} and LSTM_{bw} denote the forward and backward LSTM respectively, and $\overrightarrow{\boldsymbol{h}}_t \in \mathbb{R}^{d_h}$ and $\overleftarrow{\boldsymbol{h}}_t \in \mathbb{R}^{d_h}$ are the hidden states of the t-th word learned from LSTM_{fw} and LSTM_{bw} respectively. The entire hidden state of the t-th word is represented by concatenating $\overrightarrow{\boldsymbol{h}}_t$ and $\overleftarrow{\boldsymbol{h}}_t$, i.e., $\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}}_t, \overleftarrow{\boldsymbol{h}}_t]$, and the hidden state matrix of the utterance is $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_T]^\top \in \mathbb{R}^{T \times 2d_h}$. To express concisely, we use $d = 2d_h$ to represent the dimension of hidden state and obtain $\boldsymbol{H} \in \mathbb{R}^{T \times d}$.

4.2 Extracting Intent and Slot Representations via Bidirectional Interaction

To explicitly establish the interaction between intent classification and slot filling, for each utterance, we first use the attention mechanism over slot and intent label descriptions to get the initial intent and slot representations (Cui and Zhang, 2019; Qin et al., 2021). Then, these initial representations are concatenated with the utterance embedding matrix to produce the final slot-attention-based intent representation and intent-attention-based slot representation.

In particular, we first use the embeddings of intent labels' descriptions to produce intent embedding matrix $\boldsymbol{E}_I \in \mathbb{R}^{|\mathcal{C}_{intent}| \times d}$, and use the embeddings of slot labels' descriptions to produce slot embedding matrix $\boldsymbol{E}_S \in \mathbb{R}^{|\mathcal{C}_{slot}| \times d}$, where $|\mathcal{C}_{intent}|$ is the number of intents in the episode, $|\mathcal{C}_{slot}|$ is the number of slots in the episode, d is the dimension of hidden state. $\boldsymbol{E}_I$ and $\boldsymbol{E}_S$ are initialized by pre-processing intent and slot labels' descriptions, and they are learnable and can be updated during training. Then we calculate slot-attention-based intent representation and intent-attention-based slot representation as follows.

Slot-attention-based Intent Representation

$$\boldsymbol{H}_I = softmax(\boldsymbol{H}(\boldsymbol{E}_S)^T)\boldsymbol{E}_S \,||\, \boldsymbol{H}. \tag{2}$$

Intent-attention-based Slot Representation

$$\boldsymbol{H}_S = softmax(\boldsymbol{H}(\boldsymbol{E}_I)^T)\boldsymbol{E}_I \,||\, \boldsymbol{H}. \tag{3}$$

Here $\boldsymbol{H}_I = (\boldsymbol{h}^I_1, \boldsymbol{h}^I_2, ..., \boldsymbol{h}^I_T) \in \mathbb{R}^{T \times 2d}$, $\boldsymbol{H}_S = (\boldsymbol{h}^S_1, \boldsymbol{h}^S_2, ..., \boldsymbol{h}^S_T) \in \mathbb{R}^{T \times 2d}$, and they carry the corresponding intent and slot information respectively.

4.3 Explicit Joint Learning with Prototypical Networks

Inspired by (Krone et al., 2020), we also extend the prototypical networks to perform joint intent classification and slot filling. Different from (Krone et al., 2020), when calculating the prototype of slot label, instead of only considering the words in the front, we use the window strategy to take the contextual words into account simultaneously, which seems more reasonable.

In general, for each intent class or slot class, its corresponding prototype is the mean vector of the sample embeddings in that class. Given a support set, $\mathcal{S}_c = \{(x^i, t^i, y_c) | i \in (1, ..., k_c)\}$ is the set of support data with intent class c, where $x^i = (w^i_1, w^i_2, ..., w^i_T)$ is the i-th utterance, and $t^i = (t^i_1, t^i_2, ..., t^i_T)$ is the corresponding slot labels. $\mathcal{S}_o = \{(x^i, t^i, y^i) | t^i_j = o\}$ is the set of support data with slot label o. The prototype $\boldsymbol{p}_c$ of intent label c and the prototype $\boldsymbol{p}_o$ of slot label o can be computed as follows:

$$\boldsymbol{p}_c = \frac{1}{|\mathcal{S}_c|} \sum_{x^i \in \mathcal{S}_c} \boldsymbol{c}^i, \tag{4}$$

$$\boldsymbol{p}_o = \frac{1}{|\mathcal{S}_o|} \sum_{x^i \in \mathcal{S}_o} \frac{1}{2l+1} \sum_{k=j-l}^{j+l} (\boldsymbol{h}^S_k)^i, \tag{5}$$

where $\boldsymbol{c}^i = mean(\boldsymbol{H}_I) \in \mathbb{R}^{2d}$ is the embedding of the i-th utterance x^i. $\frac{1}{2l+1} \sum_{k=j-l}^{j+l} (\boldsymbol{h}^S_k)^i$ is the embedding of j-th word with slot label o, which considers the contextual words simultaneously with the window size $2l + 1$.

Given a query data $(x^*, t^*, y^*) \in \mathcal{Q}$, we compute the conditional probability $p\,(y = c |\, x^*,\, \mathcal{S})$ to predict its intent based on negative squared Euclidean distance.

$$p\,(y = c |\, x^*, \mathcal{S}) = \frac{exp(-||\boldsymbol{c}^* - \boldsymbol{p}_c||^2_2)}{\sum_{c'} exp(-||\boldsymbol{c}^* - \boldsymbol{p}_{c'}||^2_2)}. \tag{6}$$

Here $\boldsymbol{c}^*$ is the embedding of x^*. Similarly, we can compute the conditional probability $p\,(t_j = o |\, x^*,\, \mathcal{S})$ to predict the slot.

Finally, we perform the cross-entropy loss on all query instances to construct the IC and SF prototypical loss functions.

$$\mathcal{L}_{IC_{pn}} = \frac{1}{|\mathcal{Q}|} \sum_{x^* \in \mathcal{Q}} -\log p\,(y = y^* |\, x^*, \mathcal{S}). \tag{7}$$

$$\mathcal{L}_{SF_{pn}} = \frac{1}{|\mathcal{Q}|} \sum_{x^* \in \mathcal{Q}} \sum_{t^*_j \in t^*} -logp\left(t_j = t^*_j \middle|\, x^*, \mathcal{S}\right). \tag{8}$$

4.4 Integrating with Supervised Contrastive Learning

Supervised contrastive learning has achieved great success in computer vision, which aims to maximize similarities between instances from the same class and minimize similarities between instances from different classes. Here we integrate with supervised contrastive learning to generate better intent representations and slot representations.

We first construct contrastive samples for each query instance using support set. For a query instance x, we can take the support instances which have the same label with x as the positive samples, and the negative samples are those with different labels. Then for an episode, the SCL loss of IC can be written as:

$$\mathcal{L}_{IC_{scl}} = \frac{1}{|\mathcal{Q}|}\sum_{x^i\in\mathcal{Q}} -\frac{1}{N_{y^i}}\sum_{x^j\in\mathcal{S}} \mathbf{1}_{y^i=y^j} \, log \, \frac{exp(\mathbf{z}^i\cdot\mathbf{z}^j/\tau)}{\sum_{x^k\in\mathcal{S}} exp(\mathbf{z}^i\cdot\mathbf{z}^k/\tau)}, \tag{9}$$

where $\mathbf{z}^i \cdot \mathbf{z}^j$ means the inner product of the two vectors. $(x^i,\ t^i,\ y^i)$ is a query instance in query set $\mathcal{Q}$. N_{y^i} is the total number of utterances in support set which have the same intent label y^i. $\mathbf{z}^i = mean(\boldsymbol{H})$ is the pre-processed embedding of x^i. $\tau > 0$ is an adjustable scalar parameter which can control the separation degree of classes.

To analyze Eq. (9), we can do some simple formula manipulation as below.

$$\mathcal{L}_{IC_{scl}} = \frac{1}{|\mathcal{Q}|}\sum_{x^i\in\mathcal{Q}} -\frac{1}{N_{y^i}}\mathcal{L}_{scl},$$

$$\mathcal{L}_{scl} = \sum_{x^j\in\mathcal{S}} \mathbf{1}_{y^i=y^j} \, log \, \frac{exp(\mathbf{z}^i\cdot\mathbf{z}^j/\tau)}{\sum_{x^k\in\mathcal{S}} exp(\mathbf{z}^i\cdot\mathbf{z}^k/\tau)}$$

$$= \underbrace{\sum_{x^j\in\mathcal{S}} \mathbf{1}_{y^i=y^j}(\frac{\mathbf{z}^i\cdot\mathbf{z}^j}{\tau})}_{positive} - \sum_{x^j\in\mathcal{S}} \mathbf{1}_{y^i=y^j} \, log \underbrace{\sum_{x^k\in\mathcal{S}} exp(\frac{\mathbf{z}^i\cdot\mathbf{z}^k}{\tau})}_{positive+negative}.$$

According to the above formula, if we want to minimize $\mathcal{L}_{IC_{scl}}$, we must maximize $\mathcal{L}_{scl}$, where we need to maximize the *positive* term and minimize the *positive+negative* term, so the *negative* term will be decreased. Intuitively, the supervised contrastive learning term can push samples from the same class close and samples from different classes further apart.

In a similar manner, the SCL loss of SF for an episode can be written as:

$$\mathcal{L}_{SF_{scl}} = \frac{1}{|\mathcal{Q}_s|}\sum_{w_i\in\mathcal{Q}_s} -\frac{1}{N_{t_i}}\sum_{w_j\in\mathcal{S}_s} \mathbf{1}_{t_i=t_j} log \, \frac{exp(\boldsymbol{h}_i\cdot\boldsymbol{h}_j/\tau)}{\sum_{w_k\in\mathcal{S}_s} exp(\boldsymbol{h}_i\cdot\boldsymbol{h}_k/\tau)}, \tag{10}$$

where $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ are the embedding representations of w_i and w_j. $\boldsymbol{h}_i \cdot \boldsymbol{h}_j$ means the inner product of $\boldsymbol{h}^i$ and $\boldsymbol{h}^j$. $\mathcal{Q}_s$ represents the set of words in query set, and $\mathcal{S}_s$ represents the set of words in support set. N_{t_i} is the total number of words in support set which have the same slot label t_i. Here the same word in different utterances are considered repeatedly, and the words with slot label "Other" are ignored. Note that different from the symbol t^i which represents the slot labels of each word in an utterance x^i, t_i represents the slot label of w_i.

Combining Eq. (7), (8), (9) and (10), the overall loss function of the proposed framework is:

$$\mathcal{L} = \mathcal{L}_{IC_{pn}} + \lambda\mathcal{L}_{SF_{pn}} + \gamma\mathcal{L}_{IC_{scl}} + \delta\mathcal{L}_{SF_{scl}}, \tag{11}$$

where λ, γ and δ are trade-off hyperparameters.

5 Experiments

5.1 Episode Construction

In this section, we outline the method of sampling episodes used in (Triantafillou et al., 2020) and (Krone et al., 2020), which allows that the "way" N and the "shot" k_c are variable in each episode, and can cater the unbalanced datasets and very limited labeled instances in real application scenarios. Given a data split which contains $|C_{split}|$ intent classes, there are two steps to construct an episode.

Step 1: Sampling the class set for each episode.
(i) We sample the class number N uniformly from the range [3, $|C_{split}|$].
(ii) We sample N intent classes from the data split at random.

Step 2: Sampling the samples for each episode.
(i) Computing the query set size of each class by:

$$k_q = \min\{10, (\min_{c\in\mathcal{C}} \lfloor 0.5 * |U(c)| \rfloor)\},$$

where $\mathcal{C}$ is the set of selected classes, and $U(c)$ denotes the set of utterances belonging to class c.
(ii) Computing the total support set size $|\mathcal{S}|$ by:

$$\min\left\{U_{max}, \sum_{c\in\mathcal{C}} \lceil \beta \min\{20, |U(c)| - k_q\} \rceil \right\},$$

where β is a scalar sampled uniformly from interval $(0, 1]$, and U_{max} is the maximum support set size.
(iii) Computing the number of shots k_c of each class by:

$$k_c = \min\left\{\lfloor R_c * (|\mathcal{S}| - |\mathcal{C}|) \rfloor + 1, |U(c)| - k_q\right\},$$

where the parameter R_c is computed by:

$$R_c = \frac{exp(\alpha_c)|U(c)|}{\sum_{c' \in \mathcal{C}} exp(\alpha_{c'})|U(c')|},$$

where α_c is sampled uniformly from the interval $[log(0.5), log(2))$.

5.2 Datasets

We conduct experiments on three benchmark datasets ATIS (Hemphill et al., 1990), SNIPS (Coucke et al., 2018), and TOP (Gupta et al., 2018). In pre-processing procedure, we follow (Krone et al., 2020) to modify slot label name by adding the associated intent label name as a prefix to each slot.

We divide the dataset into train set (70%), development set (15%), and test set (15%) respectively. For the SNIPS dataset, we choose not to form a development set. This is because that there are only 7 intents in the SNIPS dataset, and we require a minimum of 3 intents per split. Table 2 provides the detailed dataset statistics.

5.3 Baselines

Following the work of Amazon AI (Krone et al., 2020), we compare our framework with some popular few-shot models: first order approximation of model agnostic meta learning (foMAML) (Finn et al., 2017)), prototypical networks (Proto), and a fine-tuning method (Fine-tune) (Goyal et al., 2018). For each model, its embedding layer could be GloVe word embeddings (GloVe), GloVe word embeddings concatenated with ELMo embeddings (ELMo), or BERT embeddings (BERT).

Furthermore, we can train the above models with two modes. One is to train and test the model on a single dataset, the other is to apply joint training approach to train the model on all the three datasets and test it on a single dataset. For example, SNIPS means we train and test the baseline on SNIPS dataset, and SNIPS (joint) means we train the baseline on all the three datasets but test it on SNIPS dataset.

Split	ATIS		SNIPS		TOP	
	#Utt	#In	#Utt	#In	#Utt	#In
Train	4,373	5	8,230	4	20,345	7
Dev	669	6	-	-	4,333	5
Test	829	7	6,254	3	4,426	6
Total	5,871	18	14,484	7	29,104	18

Table 2: Detailed statistics on utterance (Utt) and intent (In) counts for ATIS, SNIPS and TOP.

The above baselines have been performed by Krone et al. (2020), we directly reuse their reported results. And as the second training mode is time consuming, we train our proposed model with the first mode.

In addition, we compare with the latest method Retriever (Yu et al., 2021), which is a span-level retrieval method that learns similar contextualized representations for spans with the same label via a novel batch-softmax objective.

We also evaluate our framework under three cases: our framework (o, o), our framework (w, o) and our framework (w, w), where our framework (o, o) represents $\mathcal{L} = \mathcal{L}_{IC_{pn}} + \lambda \mathcal{L}_{SF_{pn}}$, our framework (w, o) represents $\mathcal{L} = \mathcal{L}_{IC_{pn}} + \lambda \mathcal{L}_{SF_{pn}} + \gamma \mathcal{L}_{IC_{scl}}$, our framework (w, w) represents $\mathcal{L} = \mathcal{L}_{IC_{pn}} + \lambda \mathcal{L}_{SF_{pn}} + \gamma \mathcal{L}_{IC_{scl}} + \delta \mathcal{L}_{SF_{scl}}$. Our framework (w, w) is the whole model.

5.4 Implementation Details

Parameter Settings In this paper, the dimension of hidden state is set to 1536 ($d = 1536$). We freeze 6 layers of BERT, and train all models using AdamW (Loshchilov and Hutter, 2019) optimizer with the initial learning rate 1×10^{-4} and the dropout ratio 0.1. All the models are trained for 30 epochs. For hyperparameters λ, γ and δ, we use the grid searching method to determine them in the range (0, 1). For the hyperparameter τ, we set $\tau = 0.1$ consistently.

Evaluation Metrics We evaluate the performance of intent classification and slot filling with accuracy (Acc) and F1 score (F1), respectively.

5.5 Result Analysis

IC Performance Table 3 summarizes the average IC accuracy over 100 test episodes when the maximum support set size $U_{max} = 20$, where the top 2 results are highlighted in bold. We could make the following observations. (1) When comparing with the baselines that use the same word embeddings (BERT), our framework (w, w) improves upon the strong baseline BERT+Proto by nearly 4%, 22% and 10% on SNIPS, ATIS and TOP respectively, which shows the superiority of our proposed model. (2) When comparing with all the baselines, our framework (w, w) improves upon the strong baseline ELMo+Proto by nearly 15% and 12% on ATIS and TOP respectively. (3) On SNIPS dataset, our framework (w, w) performs a little worse than ELMo+Fine-tune with joint train-

Embed.	Algorithm	IC Accuracy (mean +/- std)					
		SNIPS	SNIPS (joint)	ATIS	ATIS (joint)	TOP	TOP (joint)
GloVe	Fine-tune	69.52 +/- 2.88	70.25 +/- 1.85	49.50 +/- 0.65	58.26 +/- 1.12	37.58 +/- 0.54	40.93 +/- 2.77
GloVe	foMAML	61.08 +/- 1.50	59.67 +/- 2.12	54.66 +/- 1.82	45.20 +/- 1.47	33.75 +/- 1.30	31.48 +/- 0.50
GloVe	Proto	68.19 +/- 1.76	68.77 +/- 1.60	65.46 +/- 0.81	63.91 +/- 1.27	43.20 +/- 0.85	38.65 +/- 1.35
ELMo	Fine-tune	85.53 +/- 0.35	**87.64 +/- 0.73**	49.25 +/- 0.74	58.69 +/- 1.56	45.49 +/- 0.61	47.63 +/- 2.75
ELMo	foMAML	78.90 +/- 0.77	78.86 +/- 1.31	53.90 +/- 0.96	52.47 +/- 2.86	38.67 +/- 1.02	36.49 +/- 0.99
ELMo	Proto	83.54 +/- 0.40	85.75 +/- 1.57	65.95 +/- 2.29	65.19 +/- 1.29	50.57 +/- 2.81	50.64 +/- 2.72
BERT	Fine-tune	76.04 +/- 8.84	77.53 +/- 5.69	43.76 +/- 4.61	50.73 +/- 3.86	39.21 +/- 3.09	40.86 +/- 3.75
BERT	foMAML	67.36 +/- 1.03	68.37 +/- 0.48	50.27 +/- 0.69	48.80 +/- 2.82	38.50 +/- 0.43	36.20 +/- 1.21
BERT	Proto	81.39 +/- 1.85	81.44 +/- 2.91	58.84 +/- 1.33	58.82 +/- 1.55	52.76 +/- 2.26	52.64 +/- 2.58
Retriever		68.81 +/- 0.32		49.22 +/- 0.79		50.67 +/- 0.44	
our framework (o, o)		84.61 +/- 0.78		76.09 +/- 3.75		59.63 +/- 1.48	
our framework (w, o)		**85.81 +/- 0.45**		**80.37 +/- 0.58**		**62.81 +/- 0.96**	
our framework (w, w)		85.15 +/- 0.67		**80.44 +/- 0.62**		**62.85 +/- 0.33**	

Table 3: Average IC accuracy on 100 test episodes when $U_{max} = 20$.

Embed.	Algorithm	IC Accuracy (mean +/- std)					
		SNIPS	SNIPS (joint)	ATIS	ATIS (joint)	TOP	TOP (joint)
GloVe	Fine-tune	72.24 +/- 2.58	73.00 +/- 1.84	49.91 +/- 1.90	56.07 +/- 2.94	39.66 +/- 1.34	41.10 +/- 0.65
GloVe	foMAML	66.75 +/- 1.28	67.34 +/- 2.62	54.92 +/- 0.87	58.46 +/- 1.91	33.62 +/- 1.53	35.68 +/- 0.62
GloVe	Proto	70.45 +/- 0.49	72.66 +/- 1.96	70.25 +/- 0.39	69.58 +/- 0.41	48.84 +/- 1.59	46.85 +/- 0.86
ELMo	Fine-tune	87.69 +/- 1.05	**88.90 +/- 0.18**	49.42 +/- 0.79	56.99 +/- 2.12	47.44 +/- 1.61	48.87 +/- 0.54
ELMo	foMAML	80.80 +/- 0.47	81.62 +/- 1.07	59.10 +/- 2.52	56.16 +/- 1.34	41.80 +/- 1.49	36.24 +/- 0.79
ELMo	Proto	86.76 +/- 1.62	**87.74 +/- 1.08**	70.10 +/- 1.26	71.89 +/- 1.45	58.60 +/- 1.91	56.87 +/- 0.39
BERT	Fine-tune	76.66 +/- 8.68	79.53 +/- 4.25	44.08 +/- 6.05	49.71 +/- 3.84	40.05 +/- 2.35	40.46 +/- 1.74
BERT	foMAML	70.43 +/- 1.56	72.79 +/- 1.11	51.36 +/- 3.74	50.25 +/- 0.88	36.15 +/- 2.17	35.24 +/- 0.35
BERT	Proto	83.51 +/- 0.88	86.29 +/- 1.09	66.89 +/- 2.31	65.70 +/- 2.31	61.30 +/- 0.32	62.51 +/- 1.79
Retriever		71.98 +/- 0.42		54.79 +/- 0.27		51.78 +/- 0.61	
our framework (o, o)		86.35 +/- 1.32		84.92 +/- 1.75		67.98 +/- 1.21	
our framework (w, o)		86.46 +/- 0.89		**86.85 +/- 0.59**		**68.74 +/- 0.61**	
our framework (w, w)		86.79 +/- 0.37		**86.29 +/- 0.42**		**68.51 +/- 0.77**	

Table 4: Average IC accuracy on 100 test episodes when $U_{max} = 100$.

ing mode. This is because that ELMo+Fine-tune with joint training mode trains the model on all the three datasets, but our framework only trains the model on SNIPS. In addition, the word embeddings of ELMo seem more suitable for SNIPS.

Table 4 shows the average IC accuracy over 100 test episodes when the maximum support set size $U_{max} = 100$, where the top 2 results are highlighted in bold. We could make the similar observations. (1) Our framework (w, w) performs the best when comparing with the baselines that use the same word embeddings. (2) Except for SNIPS on which ELMo+Fine-tune and ELMo+Proto get the best two results, our framework (w, w) always performs better than other baselines.

SF Performance Table 5 and Table 6 summarize the average SF F1 score over 100 test episodes when the maximum support set size $U_{max} = 20$ and $U_{max} = 100$ respectively, where the top 2 results are highlighted in bold. It can be seen that (1) When comparing with the baselines that use the same word embeddings (BERT), our framework (w, w) performs the best on all the datasets. (2) When comparing with all the baselines, our framework (w, w) can also obtain satisfactory performance in most cases.

5.6 Ablation Study

Explicit-Joint Learning To verify the effectiveness of slot-attention-based intent representation and intent-attention-based slot representation, we make the ablation study. The results when $U_{max} = 20$ are shown in Table 7. Our framework (o, o) is the model that only contains explicit-joint learning. Only slot-to-intent represents the model that only uses slot-attention-based intent representation while replacing intent-attention-based slot repre-

Embed.	Algorithm	SF F1 Score (mean +/- std)					
		SNIPS	SNIPS (joint)	ATIS	ATIS (joint)	TOP	TOP (joint)
GloVe	Fine-tune	6.72 +/- 1.24	6.68 +/- 0.40	2.57 +/- 1.21	13.22 +/- 1.07	0.90 +/- 0.51	0.76 +/- 0.21
GloVe	foMAML	14.07 +/- 1.01	12.91 +/- 0.43	18.44 +/- 0.91	16.91 +/- 0.32	5.34 +/- 0.43	9.22 +/- 1.03
GloVe	Proto	29.63 +/- 0.75	27.75 +/- 2.52	31.19 +/- 1.15	38.45 +/- 0.97	10.65 +/- 0.83	18.55 +/- 0.35
ELMo	Fine-tune	22.02 +/- 1.13	16.00 +/- 2.07	7.47 +/- 2.60	7.19 +/- 1.71	1.26 +/- 0.46	1.17 +/- 0.32
ELMo	foMAML	33.81 +/- 0.33	32.82 +/- 0.84	27.58 +/- 1.25	24.45 +/- 1.20	22.35 +/- 1.23	15.53 +/- 0.64
ELMo	Proto	**59.88 +/- 0.53**	**59.73 +/- 1.72**	33.97 +/- 0.38	40.90 +/- 2.21	20.12 +/- 0.25	28.97 +/- 0.82
BERT	Fine-tune	12.47 +/- 0.31	8.75 +/- 0.28	9.24 +/- 1.67	15.93 +/- 3.10	3.15 +/- 0.28	1.08 +/- 0.30
BERT	foMAML	12.72 +/- 0.12	13.28 +/- 0.53	18.91 +/- 1.01	16.05 +/- 0.32	5.93 +/- 0.43	8.23 +/- 0.81
BERT	Proto	42.09 +/- 1.11	43.77 +/- 0.54	37.61 +/- 0.82	39.27 +/- 1.84	20.81 +/- 0.40	28.24 +/- 0.53
Retriever		48.30 +/- 0.05		**64.14 +/- 0.99**		34.77 +/- 0.34	
our framework (o, o)		50.03 +/- 0.59		61.79 +/- 3.06		38.41 +/- 1.02	
our framework (w, o)		50.77 +/- 0.92		62.73 +/- 0.53		**38.82 +/- 0.87**	
our framework (w, w)		52.82 +/- 0.70		**63.65 +/- 0.55**		**39.92 +/- 0.42**	

Table 5: Average SF F1 score on 100 test episodes when $U_{max} = 20$.

Embed.	Algorithm	SF F1 Score (mean +/- std)					
		SNIPS	SNIPS (joint)	ATIS	ATIS (joint)	TOP	TOP (joint)
GloVe	Fine-tune	7.06 +/- 1.87	7.76 +/- 0.91	2.72 +/- 1.65	17.20 +/- 3.03	1.26 +/- 0.44	0.67 +/- 0.33
GloVe	foMAML	16.77 +/- 0.67	16.53 +/- 0.32	17.80 +/- 0.42	23.33 +/- 2.89	4.11 +/- 0.81	9.89 +/- 1.13
GloVe	Proto	31.57 +/- 1.28	31.17 +/- 1.31	31.32 +/- 2.79	41.07 +/- 1.14	9.99 +/- 1.08	18.93 +/- 0.77
ELMo	Fine-tune	22.37 +/- 0.91	17.09 +/- 2.57	8.93 +/- 2.86	11.09 +/- 2.00	2.04 +/- 0.41	1.03 +/- 0.24
ELMo	foMAML	36.10 +/- 1.49	37.33 +/- 0.24	26.91 +/- 2.64	26.37 +/- 0.15	18.32 +/- 0.52	16.55 +/- 0.79
ELMo	Proto	**62.71 +/- 0.40**	**62.14 +/- 0.75**	35.20 +/- 2.46	41.28 +/- 2.73	18.44 +/- 2.41	28.33 +/- 1.33
BERT	Fine-tune	14.71 +/- 0.43	10.50 +/- 0.90	11.53 +/- 1.46	20.41 +/- 1.85	4.98 +/- 0.66	1.48 +/- 0.85
BERT	foMAML	14.99 +/- 1.29	15.83 +/- 0.94	17.68 +/- 2.42	17.11 +/- 1.31	3.37 +/- 0.36	10.58 +/- 0.45
BERT	Proto	46.50 +/- 0.75	48.77 +/- 0.71	40.63 +/- 3.37	43.10 +/- 1.76	20.58 +/- 2.27	28.92 +/- 1.09
Retriever		49.39 +/- 0.78		**68.13 +/- 3.06**		37.12 +/- 0.84	
our framework (o, o)		54.29 +/- 0.99		59.13 +/- 1.69		**38.74 +/- 1.53**	
our framework (w, o)		54.52 +/- 0.31		62.01 +/- 0.50		38.40 +/- 0.21	
our framework (w, w)		55.19 +/- 0.41		**64.95 +/- 1.11**		**40.88 +/- 0.63**	

Table 6: Average SF F1 score on 100 test episodes when $U_{max} = 100$.

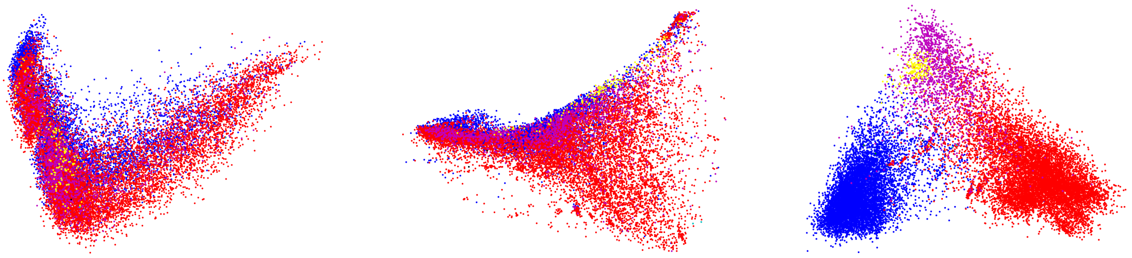
Model	SNIPS		ATIS		TOP	
	IC Acc	SF F1	IC Acc	SF F1	IC Acc	SF F1
only intent-to-slot	81.20	49.57	72.90	59.27	57.11	36.12
only slot-to-intent	82.75	48.95	73.57	54.73	58.39	34.43
our framework (o, o)	84.61	50.03	76.09	61.79	59.63	38.41

Table 7: Ablation study on the ATIS, SNIPS and TOP datasets when $U_{max} = 20$.

sentation with pure slot representation. Similarly, we have the only intent-to-slot model. From the results, it can be seen that our framework (o, o) performs better than the other two baselines, which demonstrates the effectiveness of extracting intent and slot representations via bidirectional interaction.

Supervised Contrastive Learning Our proposed objective function includes a *cross entropy (CE)* term of prototypical network and *supervised contrastive learning (SCL)* term, the latter aims to push samples in the same class close and samples in different classes further apart. By comparing the results of our framework (w, o) with our framework (o, o) in Table 3 and Table 4, we can get that the term $\mathcal{L}_{IC_{scl}}$ brings nearly 0.1% $\sim$ 4.3% improvement for IC accuracy. By comparing the results of our framework (w, w) with our framework (w, o) in Table 5 and Table 6, it can be seen that the term $\mathcal{L}_{SF_{scl}}$ brings nearly 0.6% $\sim$ 2.9% improvement for SF F1 score. The performance improvement demonstrates the effectiveness of the SCL loss for both IC and SF tasks.

Figure 2 visualizes the distribution of sentence embeddings in TOP dataset, we can observe that the original distribution is random in Pic.1. As shown in Pic.2, CE can separate the data in different classes to some extent. In Pic.3, SCL term further encourages more compact clustering of the data points in the same class.



(a) Pic.1: Original (b) Pic.2: CE (c) Pic.3: CE+SCL

Figure 2: Pic.1 shows sentence embeddings obtained from original pre-processing model without any training process. Pic.2 shows sentence embeddings via training the model with cross entropy (CE) loss of prototypical network. Pic.3 shows sentence embeddings via training the model with cross entropy (CE) and supervised contrastive loss (SCL). All the data are from TOP dataset. Data points with the same color come from the same class.

6 Conclusion

In this paper, we propose a new and practicable framework for few-shot intent classification and slot filling. The performance gains of our method come from two aspects: explicit-joint learning and supervised-contrastive learning. By explicit-joint learning, we can effectively utilize the close relationship between IC and SF tasks. By supervised-contrastive learning, we can obtain more class-indicative representations. We thoroughly evaluate our framework on few-shot IC and SF tasks and achieve impressive performance on three public datasets SNIPS, ATIS and TOP. In future work, we plan to explore more explicit-joint learning strategies and extend our framework to deal with multiple-intent classification.

Acknowledgements

The authors are grateful to the anonymous reviewers for their valuable comments and suggestions. This work was supported by National Natural Science Foundation of China (No. 62106035, 61876028), and the Fundamental Research Funds for the Central Universities (No. DUT20RC(3)040, No. DUT20RC(3)066).

References

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *International Conference on Learning Representations (ICLR)*.

Hemanthage S Bhathiya and Uthayasanker Thayasivam. 2020. Meta learning for few-shot joint intent detection and slot-filling. In *International Conference on Machine Learning Technologies (ICMLT)*, pages 86–92.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pages 1597–1607.

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

Leyang Cui and Yue Zhang. 2019. Hierarchically-refined label attention network for sequence labeling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4113–4126.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(4):594–611.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, pages 1126–1135.

Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2020. Dynamic memory induction networks for few-shot text classification. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1087–1094.

Ruiying Geng, Binhua Li, Yongbin Li, Yuxiao Ye, Ping Jian, and Jian Sun. 2019. Few-shot text classification with induction network. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 753–757.

Anuj Kumar Goyal, Angeliki Metallinou, and Spyros Matsoukas. 2018. Fast and scalable expansion of natural language understanding functionality for intelligent agents. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 145–152.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. *CoRR*, abs/2011.01403.

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

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2787–2792.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Proceedings of the Workshop on Speech and Natural Language*.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1381–1393.

Hong-Gyu Jung and Seong-Whan Lee. 2020. Few-shot learning with geometric constraints. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Jason Krone, Yi Zhang, and Mona T. Diab. 2020. Learning to classify intents and slot labels given a handful of examples. *CoRR*, abs/2004.10793.

Manoj Kumar, Varun Kumar, Hadrien Glaude, Cyprien de Lichy, Aman Alok, and Rahul Gupta. 2021. Protoda: Efficient transfer learning for few-shot intent classification. In *IEEE Spoken Language Technology Workshop (SLT)*, pages 966–972.

Bing Liu and Ian R. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 685–689.

Han Liu, Xiaotong Zhang, Lu Fan, Xuandi Fu, Qimai Li, Xiao-Ming Wu, and Albert Y. S. Lam. 2019a. Reconstructing capsule networks for zero-shot intent classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4798–4808.

Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. 2019b. Few-shot unsupervised image-to-image translation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 10550–10559.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2078–2087.

Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021. A co-interactive transformer for joint slot filling and intent detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8193–8197.

Kate Rakelly, Evan Shelhamer, Trevor Darrell, Alexei A. Efros, and Sergey Levine. 2018. Few-shot segmentation propagation with guided networks. *CoRR*, abs/1806.07373.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4077–4087.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. 2020. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations (ICLR)*.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, pages 3630–3638.

Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. 2019. Panet: Few-shot image semantic segmentation with prototype alignment. In *IEEE International Conference on Computer Vision (ICCV)*, pages 9196–9205.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2020a. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):63:1–63:34.

Yikai Wang, Chengming Xu, Chen Liu, Li Zhang, and Yanwei Fu. 2020b. Instance credibility inference for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12833–12842.

Henry Weld, Xiaoqi Huang, Siqi Long, Josiah Poon, and Soyeon Caren Han. 2021. A survey of joint intent detection and slot-filling models in natural language understanding. *CoRR*, abs/2101.08091.

Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular CRF for joint intent detection and slot filling. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU Workshop)*, pages 78–83.

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

Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. 2018. Bayesian model-agnostic meta-learning. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 7343–7353.

Dian Yu, Luheng He, Yuan Zhang, Xinya Du, Panupong Pasupat, and Qi Li. 2021. Few-shot intent classification and slot filling with retrieved examples. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 734–749.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1206–1215.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2993–2999.