FewJoint: few-shot learning for joint dialogue understanding

Yutai Hou · Xinghao Wang · Cheng Chen · Bohan Li · Wanxiang Che · Zhigang Chen

Received: 27 March 2022 / Accepted: 17 June 2022 / Published online: 18 July 2022 © The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract
Few-shot learning (FSL) is one of the key future steps in machine learning and raises a lot of attention. In this paper, we focus on the FSL problem of dialogue understanding, which contains two closely related tasks: intent detection and slot filling. Dialogue understanding has been proven to benefit a lot from jointly learning the two sub-tasks. However, such joint learning becomes challenging in the few-shot scenarios: on the one hand, the sparsity of samples greatly magnifies the difficulty of modeling the connection between the two tasks; on the other hand, how to jointly learn multiple tasks in the few-shot setting is still less investigated. In response to this, we introduce FewJoint, the first FSL benchmark for joint dialogue understanding. FewJoint provides a new corpus with 59 different dialogue domains from real industrial API and a code platform to ease FSL experiment set-up, which are expected to advance the research of this field. Further, we find that insufficient performance of the few-shot setting often leads to noisy sharing between two sub-task and disturbs joint learning. To tackle this, we guide slot with explicit intent information and propose a novel trust gating mechanism that blocks low-confidence intent information to ensure high quality sharing. Besides, we introduce a Reptile-based meta-learning strategy to achieve better generalization in unseen few-shot domains. In the experiments, the proposed method brings significant improvements on two datasets and achieve new state-of-the-art performance.

Keywords Few-shot learning · Joint learning · Dialogue understanding

1 Introduction
Dialogue Understanding (DU) is a crucial module for the task-oriented dialogue system [69]. In recent years, deep learning techniques have brought great progress for DU, but these successes heavily rely on massive annotated data. In fact, DU often suffers from data scarcity in real-world applications, because manual data annotation is costly and task-oriented dialogue system often confronts to new domains that lack data [24, 26, 46]. Few-Shot Learning (FSL) techniques are promising to liberate DU from over-dependence on massive data, which is committed to learning new tasks with only a few examples (usually only one or two per category) [18, 33, 40, 58].

Previous few-shot learning mainly focuses on single task scenarios, where the model learns a single task at a time. However, dialogue understanding usually contains two different but closely related tasks: slot filling and intent detection, and is known to benefit a lot from jointly learning these two tasks [9, 22, 47, 64]. For the example shown in Fig. 1, intent information can guide the slot filling task: “Forrest Gump” is a “FilmName” in “PlayVideo” intent and is a “BookName” in “ReadBook” intent. Unfortunately, joint learning of two tasks becomes challenging in few-shot scenarios: on the one hand, the sparsity of FSL data greatly magnifies the difficulty of modeling connections between the slots and intents; and on the other hand, how to jointly learn multiple tasks in few-shot setting is still less investigated.

Wanxiang Che
car@ir.hit.edu.cn
Yutai Hou
ythou@ir.hit.edu.cn
Xinghao Wang
xhwang@ir.hit.edu.cn
Cheng Chen
cchen@ir.hit.edu.cn
Bohan Li
bhli@ir.hit.edu.cn
Zhigang Chen
zgchen@iflytek.com

1 Research Center for Social Computing and Information Retrieval, Harbin Institute of Technology, Harbin, China
2 State Key Laboratory of Cognitive Intelligence, Hefei, China
Besides the technical challenges above, there are also primary constraints from the lack of joint learning benchmarks. Consequently, new joint methods cannot be easily developed and iteratively improved.

In this paper, we present FewJoint to tackle the above issues, which is the first FSL benchmark for joint dialogue understanding. One of the main obstacles in constructing the FSL benchmarks comes from the special evaluation paradigm of FSL: few-shot models are usually first pre-trained on data-rich domains and then tested on unseen domains. Such a paradigm often requires lots of different train/test domains to learn general prior experience and conquer result randomness, which are often hard to obtain for NLP tasks. Previous NLP FSL works [4, 21] often construct fake domains from a single dataset. They divide the label set into several subsets. Then for each label subset, they construct a synthetic domain with the corresponding samples. Such simulations often disturb the natural distribution of samples, and thus may not reflect well the real-world complexities of few-shot DUs.

Fig. 1 Training and testing examples of the few-shot joint dialogue understanding. We train the model on a set of source domains and test it on an unseen domain containing only a few supporting example.

Further, different from data-rich settings, we observe that joint learning tricks may not necessarily bring improvements in few-shot DU and even hinder the joint performance. A key factor leading to this problem may lie in the low performance of few-shot models on a single task. At this point, directly sharing information between slot and intent tasks may introduce too much noise and harms each other’s performance. For example, suppose the intent accuracy is only 70%, the slot task is likely to be misled on 30% samples. To tackle such negative sharing problems, we guide slot filling with explicit intent information and introduce a novel trust gating mechanism to ensure high-quality information sharing. Specifically, the trust gating mechanism utilizes the confidence of intent predictions to selectively share intent information to the slot task for each sample. Besides, to make the model generalize better and train faster, we introduce a Reptile [44] based meta-learning strategy to the few-shot joint DU tasks. Experiments demonstrate our assumption on the negative sharing problem, and the proposed method can significantly improve model performance on two different datasets.

In summary, our contribution is three-fold: (1) We introduce the first FSL benchmark for joint dialogue understanding, which contains 59 real-world domains that allow evaluating few-shot models without constructing fake domains and better reflect real-world complexities of dialogue understanding, and provide a code platform to ease comparison and implementation of few-shot methods. (2) We propose a
novel trust gating mechanism that introduces a dynamic joint learning process and helps to ensure the quality of information sharing in the few-shot setting. (3) We introduce a Reptile-based meta-learning strategy to achieve better cross-domains generalizability, which is the first exploration of Reptile in this field.

In the following, we first discuss the related work in Sect. 2 and present a problem definition for the few-shot dialogue understanding task in Sect. 3. After that, we describe the proposed model for joint dialogue understanding in Sect. 4 and introduce the proposed FewJoint benchmark in Sect. 5. In Sect. 6, we conduct extensive experiments of few-shot joint dialogue understanding to evaluate proposed methods. At the end of the paper, we summarize the conclusions in Sect. 7.

2 Related work

For the field of machine learning, few-shot learning (FSL) is a challenging and important task that aims to recognize novel classes with only a few labeled samples [7, 17, 19, 59]. There are two main different lines of research dedicated to FSL. (1) representation learning-based approaches, i.e. metric-based or similarity-based methods [58], focus on designing a good feature extractor or training strategy to get better transferable representations from the data-abundant classes, so that the novel classes can be recognized by many distance function or classifier, such as cosine distance, Euclidean distance, etc. [10, 38, 52, 55, 66, 68, 71] (2) optimization-based approaches, i.e. meta-learning based methods [20], aim to get a good model initialization. With a large number of FSL tasks, a task-agnostic meta-learner is learned to accelerate the optimization in the meta-testing stage. [3, 34, 53].

FSL has been studied in many NLP tasks, such as text classification [1, 41, 61], named entity recognition [12, 32, 56, 62], entity relation extraction [67], question answering [8, 39], etc.

As an important NLP application, task-oriented dialogue systems often suffer from a lack of data due to the high cost of annotation and the frequent need to confront new domains that lack data. Owing to the need to handle myriad user expressions, the dialogue understanding module in task-oriented dialogue systems faces particularly challenges from data deficiencies [25]. To remedy this, the two main tasks of DU: intent detection [28, 37, 42, 72] and slot tagging [26, 45, 60], have been widely studied in the few-shot setting. However, previous works mainly address only individual tasks or the two tasks separately [70].

One popular solution for dialogue understanding is joint learning of the two sub-tasks, with many works dedicated to joint DU for data-rich scenarios [48, 50, 63, 73]. Meanwhile, few-shot joint DU is less investigated. ConProm [27] first to explore metric-learning based method for joint DU, which bridges the intent and slot metric spaces with cross-attention. To capture task relation and reduce misclassification, they align related cross-task labels in the bridged metric space and force unrelated labels to separate. Different from metric-learning based methods that have to keep support examples for inference, optimization based methods learn a model to directly predict labels for new queries and can better generalize to cases of a larger support set. To this end, previous works adopt MAML to joint DU task [5, 31], which can learn better initialization parameters for the joint model of intent detection and slot tagging, and can quickly transfer knowledge to the tasks from other domains. But these works achieve joint dialogue understanding by simply sharing the encoding layer between intent and slot models, which fail to capture explicit intent-slot relation. Meanwhile, these methods tend to suffer from the high cost and unstable of second order derivative based optimization in MAML [2, 44]. By contrast, we model explicit intent-slot relation with a novel trust gating mechanism and introduce Reptile-base meta-learning strategy to achieve better domain generalizability.

With the development of few-shot learning, more and more new datasets [14], generalization methods [57] and benchmarks [43, 65, 74] are contributed to promoting research in related communities. Unlike previous datasets, our dataset focuses on a few-shot task that lacks attention, that is, learning multiple tasks jointly, which is also the first dataset for a few-shot joint dialogue understanding task.

3 Problem definition

Before introducing the method and dataset, we present a formal definition of the few-shot language understanding problem here.

Starting from the notions, we define an utterance \( x = (x_1, x_2, \ldots, x_n) \) as a sequence of words and define the corresponding semantic frame as \( y = (c, s) \). \( c \) is the intent label of the utterance; \( s \) is the slot label sequence of the utterance and is defined as \( s = (s_1, s_2, \ldots, s_n) \). A domain \( D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N_D} \) is a set of \((x, y)\) pairs. For each domain, there is a corresponding domain-specific label set \( \mathcal{L}_D \). To simplify the description and ease understanding, we combine the label set definition of intent and slot, and assume that the number of labels \( N \) is the same for all domains.

In few-shot learning scenarios, models are usually first trained on a set of source domains \( \{D_1, D_2, \ldots\} \), then evaluated on another set of unseen target domains \( \{D'_1, D'_2, \ldots\} \). A target domain \( D'_I \) only contains few labeled examples, which is called support set \( S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{NS} \). \( S \) usually includes \( K \) examples (K-shot) for each of \( N \) labels (N-way). Figure 1 shows an example of the training and testing process of 1-shot dialogue understanding.
The K-shot dialogue understanding task is then defined as follows: given an input query utterance \( x = (x_1, x_2, \ldots, x_n) \) and a K-shot support set \( S \) as references, find the most appropriate semantic frame \( y^* \) of \( x \):

\[
y^* = \arg \max_y p(y | x, S).
\]

### 4 Proposed method

In this section, we introduce a novel dialogue understanding model that jointly learns the intent detection and slot filling tasks, and employs the trust gating mechanism to reduce the noise sharing problem in the few-shot setting (Sect. 4.1). To help the model generalize better to unseen few-shot domains, we introduce the Reptile-based learning strategy (Sect. 4.2). The overview of the model is shown in Fig. 2.

#### 4.1 Joint dialogue model

We start with a simple but effective joint dialogue understanding paradigm following previous works [9], where a shared BERT encoder is connected to two separate task-specific linear decoders. Specifically, given the encoder outputs of input tokens \( h = (h_1, \ldots, h_n) \), we feed the averaged token embedding into a linear layer with a Softmax function for intent classification, while each token embedding is fed into another linear classification layer for slot filling.\(^2\) This process can be formulated as:

\[
\begin{align*}
    y_{\text{intent}} & = \text{Softmax}\left( W_{\text{intent}}^{1} \sum_{i=1}^{n} h_i + b_{\text{intent}} \right), \\
    y_{\text{slot}}^i & = \text{Softmax}(W_{\text{slot}}^{i} h_i + b_{\text{slot}}), \quad i \in \{1, \ldots, n\},
\end{align*}
\]

where \( W_{\text{intent}}^{1}, b_{\text{intent}} \) and \( W_{\text{slot}}^{i}, b_{\text{slot}} \) are weights and bias for intent and slot classifier respectively.

This approach achieves implicit joint learning by learning a common representation space with two supervised signals simultaneously, which often faces difficulties in balancing the learning of two tasks and fails to capture the explicit task relations. To remedy this, previous works have demonstrated the effectiveness of explicitly using the intent information to guide the slot filling task [22, 35, 47, 49]. Based on a similar idea, we propose a novel trust gating mechanism to improve the model with explicit joint learning.

#### 4.1.1 Trust gating mechanism

Following [47], we adopt a straightforward but effective manner of using intent information, i.e., directly sharing the intent information of predicted intent distribution. Specifically, unlike the vanilla joint learning described above, we...
concatenate the logits of intent and Encoder outputs $h$ to get the intent-guided features $h'$ for slots. The calculation procedure is as follows:

$$h_{1:n} = \text{Encoder}(x_{1:n}),$$

$$l^{\text{intent}} = W^{\text{intent}} \frac{1}{n} \sum_{i=1}^{n} h_i + b^{\text{intent}},$$

$$h'_{1:n} = (h_i; l^{\text{intent}})_{i=1}^{n},$$

$$y^{\text{intent}} = \text{Softmax}(l^{\text{intent}}),$$

$$y^{\text{slot}}_i = \text{Softmax}(W^{\text{slot}} h'_i + b^{\text{slot}}), \quad i \in \{1, \ldots, n\},$$

where $l$ is the predicted logits of the intent, $(; )$ denotes the concatenation of vectors, and $h'$ is the concatenated new feature for slot prediction, which allows capturing explicit relations between slots and intents.

However, in few-shot settings, sharing intention information directly to the slot filling task may pose some problems. For instance, poor intent classification performance in few-shot scenarios may lead to negative guidance and hurt both performances. Therefore, it is necessary to determine whether the intent information is trustworthy to guide the slot filling task.

To achieve this, we introduce a confidence-based trust gating mechanism that blocks the potential negative sharing of intent information. We introduce the confidence score $C^{\text{intent}}$ of intent, which is calculated as the maximum value of $y^{\text{intent}}$:

$$C^{\text{intent}} = \max(y^{\text{intent}}),$$

where $C^{\text{intent}}$ represents the probability of the most likely intent predicted by the model, which can reflect the confidence that the model predicts correctly. For the case that the confidence score is low, such as $C^{\text{intent}} = 0.2$, the intent information is prone to be noisy and should not be shared with the slot task. Therefore, we set a confidence threshold $t$ to filter out trustworthy information and forms the trust-gate $G$:

$$G = \begin{cases} 1, & C^{\text{intent}} > t, \\ 0, & C^{\text{intent}} \leq t. \end{cases}$$

With trust gating mechanism, the slot filling is selectively guided by intent information as:

$$h'_{1:n} = (h_i; l^{\text{intent}} \ast G)_i^{n},$$

$$y^{\text{slot}}_i = \text{Softmax}(W^{\text{slot}} h'_i + b^{\text{slot}}), \quad i \in \{1, \ldots, n\}.$$ 

The proposed trust gating mechanism is expected to control the intent information sharing, so that the slots are guided by the intent only when the model has a high certainty of the intent correctness, avoiding the problem of negative guidance in few-shot scenarios. The experiments in Table 7 demonstrate that the trust gating mechanism significantly improves the quality of shared intent information.

### 4.2 Model learning

Meta-learning is one of the most popular solutions for few-shot learning problems, which often learns generalization capabilities over a large number of tasks and quickly adapts to a new domain with a few examples. A classical type of these algorithms, such as MAML [20] and Reptile [44], enables fast adaptation of few-shot domains by learning better initialization.

To make the model more generalized and train faster, we introduce a refined approach of Reptile to joint dialogue understanding for the first time.

Reptile is a first-order meta-learning approach, which achieves good performance on many well-established FSL benchmarks while avoiding the high cost of second-order derivations of classic meta algorithm as MAML. Reptile learns a well-generalized initialization by learning a meta-model on multiple source domains. Specifically, it repeatedly samples a single domain, trains a copied model on it, and moves the meta model towards the weights of the trained copied model on the domain. After repeat training, the parameters of the trained meta model are used as the initialization for unseen domains. For each sampled domain $D$, the learning process is as follows:

$$\tilde{\phi} = T_D(\phi),$$

$$\phi \leftarrow \phi - \epsilon \ast (\tilde{\phi} - \phi),$$

where $\phi$ is the parameter of meta-model, i.e., the initialization parameters to be learned, $T_D(\phi)$ denotes training on task $D$ with $\phi$ as the initialization, and $\tilde{\phi}$ is the learned parameter. $\epsilon$ is a learning rate.

Common Reptile algorithm may be unstable because it updates meta-model once for each domain, and domains can be quite different. Switching between domains with big gap can disrupt the learning direction. Besides, with a consistent learning rate, the meta-model may learn too much task-related knowledge on some particular tasks, which may hurt the generalization capability of the meta-model. Moreover, the original reptile uses a fixed learning rate when updating the initialization parameters, which can also hinder the converging.

#### 4.2.1 Joint dialogue understanding training with refined reptile

Unlike common Reptile that learns on one sampled domain at a time, we find that updating the model on multiple tasks at the same time can learn a better-generalized model. We
first split source domains into multiple batches, where the $k$th batch is $B_k = \{D_1^k, D_2^k, \ldots, D_m^k\}$ and $m$ is the batch size. After that, we fine-tune $m$ copied meta-model $\phi$ on each domain with the Cross-Entropy loss function $CE$:

$$loss_{\text{intent}} = CE(y_{\text{intent}}^k, \hat{y}_{\text{intent}}^k),$$

$$loss_{\text{slot}} = \frac{1}{n} \sum_{i=1}^{n} CE(y_{\text{slot}}^i, \hat{y}_{\text{slot}}^i),$$

$$loss = loss_{\text{intent}} + loss_{\text{slot}}.$$

$\hat{y}$ is the real label, then we get a batch of fine-tuned models:

$$\phi_i^k = T_{\text{meta}}(\phi), \quad i \in \{1, \ldots, m\},$$

$$\Phi_{B_k} = \{\phi_1^k, \phi_2^k, \ldots, \phi_m^k\}.$$

We calculate the difference between each fine-tuned model and the original meta-model, and then average the differences for a batch of tasks as the gradient for the original model to update.

$$\text{gradient}^k = \frac{1}{n} \sum_{i=1}^{n} (\phi_i^k - \phi),$$

$$\phi \leftarrow \phi - c \ast (\text{gradient}^k).$$

By averaging the optimization direction of a batch of tasks, we encourage the model to learn more task-agnostic knowledge, avoiding the model going too far on specific tasks, and speeding up the training process.

In Reptile the authors use a constant learning rate to optimize the meta-model with a simple Stochastic Gradient Descent (SGD). However, learning rate warmup and decay [23] have been proved crucial in learning a more generalized model. Besides, adaptive gradient methods like AdaGrad [15], Adam [30] and AdamW [36] that help model escaping local minima, have been a default choice for training deep neural networks. Therefore, we propose to apply linear learning rate decay scheduling and AdamW optimizer with warmup to the optimization of the meta-model.

With the refined Reptile algorithm above, we eliminated the spikes of the optimization direction and smoothed the training procedure to train a more generalized meta-model that can adapt quickly with a few samples.

5 Dataset construction

In this section, we introduce the construction process of the FewJoint dataset, which generally consists of two steps. Firstly, we collect and annotate a complete dialogue understanding corpus with 59 domains (Sects. 5.1 and 5.2). Then, we split the corpus into training and unseen few-shot domains. We also sample support and query sets to simulate few-shot scenarios (Sect. 5.3).

5.1 Dialogue collection

We collect dialogue utterances of real dialogue domains from the AIUI open dialogue platform of iFlytek. Before utterance collection, we select popular domains based on the frequency of API calls, such as “Search information of corona-virus”. We ignore the domains that have no Intent or Slot Schema to ensure joint learning. For schema definition, we leverage the semantic frames and domains defined by AIUI, and also refine parts of domains to remove ambiguous labels. Finally, we gather 59 different dialogue domains together with semantic frame definitions.

Since not all domains have enough real user data, we collect user utterances generally from two sources: (1) Real user utterances. (2) Utterances written by the data annotators. The second method of constructing data is similar to a classical method of dialog data collection, namely the Wizard-of-Oz [6, 16, 29, 75], where the machine learns from annotator-typed conversation logs.

For source (1), we sample existing user utterances from the AIUI platform and remove the sensitive information. For source (2), four data annotators were asked to impersonate users of dialogue agents and write query utterances for the specific domains, such as querying weather. The average ratio of utterances between source (1) and (2) is about 3 : 7.

5.2 Data annotation

After collecting raw user utterances, we label each utterance with both intent (sentence level) and slot labels (token level). The support sets in Fig. 1 show examples of utterances with intent and slots annotations.

The annotating process consists of two steps: Firstly, we obtain rough annotation for each utterance by predicting semantic-frame with the testing tools of the AIUI platform. Then, the human workers validate each roughly annotated utterance and re-annotated the inappropriate ones. The data was divided equally into four parts and then annotated by four workers respectively. The four annotators who participated in utterance writing are also responsible for this part.

After annotating, we perform data re-checking. Another three annotators independently checked all the data, during which incorrect data are re-annotated.

3 http://aiui.xfyun.cn/index-aiui.
5.3 Simulation of few-shot scenarios

Till now, we have collected the annotated dialogue corpus. To test the learning performance of few-shot learning models, we need to reconstruct the data into Few-Shot Learning (FSL) setting. In the FSL setting, models are first pre-trained on data-rich domains and then tested on unseen domains with only a few-shot support set. Figure 1 shows an example of the few-shot learning setting.

5.3.1 Few-shot data construction

To achieve the few-shot problem setting described in Sect. 3, we reserve some domains as few-shot testing domains, which are unseen during training. Specifically, we first split the 59 domains into three parts with no intersection, i.e., train, dev, and test set. Then on each dev or test domain, we construct a K-shot support set and use the other data as the query set. Thus, FewJoint can simulate few-shot scenarios on the unseen testing domains: the models are required to predict the labels of the query samples with only a few support examples. Table 1 shows the details of domain division.

5.3.2 Reconstructing testing domains

We manually reconstruct each test/dev domain into two parts: a few-shot support set and a query set. Here, a K-shot support set is manually constructed by the following principles:

- Ensure each class (intent and slot) appeared at least k times, while keeping the support set as small as possible.
- Avoid duplication between the support set and query set.
- Encourage diversity of both expressions and slot values of support set.

5.3.3 Reconstructing training domains

The training set consists of 45 training domains, which provide prior experience to help quick learning on unseen domains. For few-shot learning, there are two popular strategies for learning such prior experience and their data format is very different. In our benchmark, we provide two kinds of training set format to support these two learning strategies:

(1) Learn the feature encoding layer on all training data, which simply needs to combine all train domain utterances into a single pre-training set.
(2) Learn the ability to learn quickly when given only a few examples, i.e., meta-learning. This requires reconstructing the training set into a series of few-shot episodes (i.e. support set and query set pair).

Strategy (1) does not require special data processing. To support Strategy (2), we need to sample query and support sets to construct few-shot learning episodes within training domains. Here, we adopt the Minimum-including Algorithm [26] to achieve automatic sampling of plentiful few-shot episodes.

Minimum-including Algorithm helps to sample the support set for sequence labeling problems and multi-label problems, where a single instance may be associated with multiple labels. In these problem settings, the normal N-way K-shot support set definition is inapplicable. Since different labels randomly co-occur in one sentence, we cannot guarantee that each label appears K times. Take the dialogue understanding problem as an example, each utterance instance is often associated with multiple labels, including one intent and multiple slots. For example in Fig. 1, in the 1-shot support set, the “City” slot appears twice to ensure all intents appear at least once. Specifically, Minimum-including Algorithm approximately builds K-shot support set S following two criteria:

(1) All labels within the domain appear at least K times in support set S.
(2) At least one label will appear less than K times in S if any (x, y) pair is removed from it.

Algorithm 1 shows the detailed process.

---

Table 1 The domains of FewJoint benchmark

| Item          | Domain name                                                                 |
|---------------|-------------------------------------------------------------------------------|
| Train Domains (45) | query, Capital, app, epg, petrolPrice, dream, animalCries, historyToday, translation, sentence-Making, carNumber, poetry, familyNames, match, clock, weightScaler, cityOfPro, airControl, website, stock, riddle, map, cookbook, music, calendar, crossTalk, wordsMeaning, new, health, home, video, telephone, weather, tchannel, lottery, stroke, radio, contacts, bus, message, train, novel, email, cinemas, flight, childClassics |
| Dev Domains (5)    | wordFinding, garbageClassify, holiday, joke, temperature                      |
| Test Domains (9)   | idiomsDict, timesTable, virusSearch, captialInfo, constellation, drama, length, story, chineseZodiac |

---

4 Benchmark users are free to re-construct training set into any format.
Table 2 Statistic of raw data

| Item                | Count  |
|---------------------|--------|
| Total number of utterances | 6,694  |
| Average length of utterance | 9.9    |
| Total number of intents | 143    |
| Average intents per domain | 2.42   |
| Total number of slots | 205    |
| Average slots per domain | 3.47   |

Algorithm 1 Minimum-including

Input: # of shot $K$, domain $D$, label set $L_D$
1: Initialize support set $S = \emptyset$, $\text{Count}_j = 0$ ($\forall \ell_j \in L_D$)
2: for $\ell$ in $L_D$ do
3: while $\text{Count}_\ell < K$ do
4: From $D \setminus S$, randomly sample a $(x^{(i)}, y^{(i)})$ pair that $y^{(i)}$ includes $\ell$
5: Add $(x^{(i)}, y^{(i)})$ to $S$
6: Update all $\text{Count}_j$ ($\forall \ell_j \in L_D$)
7: end while
8: end for
9: for each $(x^{(i)}, y^{(i)})$ in $S$ do
10: Remove $(x^{(i)}, y^{(i)})$ from $S$
11: Update all $\text{Count}_j$ ($\forall \ell_j \in L_D$)
12: if any $\text{Count}_j < K$ then
13: Put $(x^{(i)}, y^{(i)})$ back to $S$
14: Update all $\text{Count}_j$ ($\forall \ell_j \in L_D$)
15: end if
16: end for
17: Return $S$

5.4 Statistics

This section presents detailed statistics for the constructed dataset. The statistic info of annotated raw paper is shown in Table 2. There are 6,694 utterances included in the corpus and the average length of the utterance is 9.9 (number of Chinese characters). As mentioned before, we collect data for 59 real dialogue domains. Among them, we reserve 14 domains as unseen few-shot domains for evaluation and use all the other 45 domains as training domains. For evaluation, we select 9 domains as the test set and use 5 for development. Overall, our data set contains 143 different intents and 205 different slots.

Table 3 shows the statistics of constructed few-shot data. The main setting (Used in the SMP2020 contest) is 3-shot setting, and we also provide 1, 5, 10 shots setting for extensive evaluation. The support set size and the query set size of different shot settings are included in Table 3. Besides, we also present the number of occurrences for each intent and slot in the support set, which satisfies our construction requirements for shots.

6 Experiments

We conduct 1-shot and 5-shot experiments on two different datasets, where few-shot learning models are asked to transfer universal knowledge from seen source domains (train set) to unseen target domains (test set) that only contain a 1-shot/5-shot support set.

6.1 Settings

Dataset

We evaluate the performance of the proposed method on the 1-shot and 5-shot version of FewJoint dataset, which are described in Section 5.

---

5 The Evaluation of Chinese Human-Computer Dialogue Technology, SMP2020-ECDT task-1. Link: https://smp2020.aconf.cn/smp.html.

6 We choose 1 and 5 shots because they have been a common experiment setting in a few-shot learning study.
In addition to FewJoint, we also conduct experiments on a wide-used public dataset: Snips [11] to verify the generalization. Snips is a dataset for dialogue understanding, containing seven domains with 53 slots. There is only one intent in every domain. Since Snips is intended for data-rich settings, we follow Hou et al. [27], Krone et al. [31] to reconstruct Snips into a few-shot version and use the same data split. Specifically, the Snips dataset is split into three parts: 3 intents in source training domains, 2 intents in developing domains, and 2 intents in testing domains.

We follow previous few-shot learning study ([20, 31, 58]) and construct all datasets into the few-shot episode style. Each episode contains a support set and a query set. The construction of the support set for each episode is described in Sect. 5 and Algorithm 1. For Snips, 200 few-shot episodes are constructed for training, 50 few-shot episodes for developing, and 50 for testing. The query set size is 16 for training/developing, and 100 for testing.

### 6.1.1 Evaluation

Three metrics for evaluation are what our experiment adopt: Intent Accuracy, Slot F1-score, and Joint Accuracy. Among these metrics, Joint Accuracy, which evaluates the sentence level accuracy, is the most important metric for joint dialogue understanding [27]. Only when all slots and intent of one sample are predicted correctly, the sample is considered positive for Joint Accuracy. We calculate Slot F1-score with the conlleval script by following Hou et al. [27]. We report the average score of five random seeds for all results to control script by following Hou et al. [27]. We report the average score of five random seeds for all results to control script by following Hou et al. [27].

### 6.1.2 Implements

We use uncased BERT-base in the Snips dataset and Chinese BERT-base in the FewJoint dataset as the encoder. For each sample, we add special token [CLS] and [SEP] before and after it, and encode the decorated sample with BERT to get token embeddings. After encoding, we only keep the embedding of tokens from original samples and average all token embeddings as sentence embedding. In the inner loop of meta learning, we use Adam [30] to train the models. In the outer loop of our method, we use AdamW [36] to train the model. All initial learning-rates are set as 1e−5. We set the batch size as 4, i.e., learning from 4 tasks for each meta-learning step. We set the confidence threshold value to 0.5. We implement both our and baseline models with the few-shot platform MetaDialog7. In addition, we add Transition Rules to achieve better performance of these methods, i.e. +TR, to ban illegal slot prediction such as an ‘O’ tag before an ‘I’ tag, following Hou et al. [26].

### 6.2 Baselines

We compare our method with various baselines, including similarity-based few-shot learning methods (i.e. non-fine-tune based methods) and fine-tune based transfer learning methods.

SepProto is a similarity-based FSL dialogue understanding model with the prototypical network [54]. Intent detection and slot filling are learned separately with BERT embedding. This model is pretrained on source domains and then directly applies to target domains without any fine-tuning.

JointProto is also a similarity-based model with the prototypical network and it is all the same as SepProto [31]. The difference from SepProto is that intent detection and slot filling are learned jointly with shared BERT embedding. This model is also pretrained on source domains and then directly applies to target domains without any fine-tuning similar to SepProto.

LD-Proto is a similarity-based model with the prototypical network similar to JointProto. The difference from JointProto is that LD-Proto adopts the logits-dependency tricks [22] to achieve joint learning of intent detection and slot filling by depending on the intent and slot prediction on the logits of these accompanying tasks.

JointTransfer is a fine-tune-based domain transfer model. It learns intent detection and slot filling jointly based on the JointBERT [9], which consists of a shared BERT embedder with intent detection and slot filling layers on the top. This model is pretrained on source domains and then directly applies to target domains support sets with fine-tuning.

Meta-JOSFIN is a MAML [20] based meta-learning model (?), which is a joint dialogue understanding model similar to JointTransfer, and is trained with MAML. This method learns initial parameters that can fast adapt to the target domain.

ConProm [27] is a similarity-based few-shot joint dialogue understanding framework and is the current state-of-the-art method. It learns the relations between intents and slots by aligning the related intent-slot prototypes in a shared metric space with contrastive learning.

### 6.3 Main results

This section shows the comparison of our methods and strong baselines on two datasets.8

8 Note that baseline results on FewJoint are slightly higher than those reported in ConProm paper[27]. This is because we conduct experiments on a refined version of FewJoint, which fixes errors in the original version.

---

7 [https://github.com/AtmaHou/MetaDialog](https://github.com/AtmaHou/MetaDialog)

8 Note that baseline results on FewJoint are slightly higher than those reported in ConProm paper[27]. This is because we conduct experiments on a refined version of FewJoint, which fixes errors in the original version.
6.3.1 Result of 1-shot setting

As shown in Table 4, on the FewJoint dataset, our proposed method achieves the best performance on Joint Acc., which is the most important metric that reflects the joint learning capability. Meanwhile, the results of our proposed method are comparable to the strongest baseline (ConProm+FT+TR) on the Snips dataset. These results illustrate the superiority of our method for joint language understanding tasks. The reason for not having the same boost on the SNIPS data as FewJoint is that SNIPS is not a few-shot learning dataset: there is only one test domain for snips, where the contained two intents are always artificially unbalanced because of the simulation of few-shot scenarios.\(^9\) Such an unbalanced domain is detrimental for fine-tuning methods, but relatively advantageous for metric learning methods as the prototypes are naturally normalized against sample numbers. For example, all fine-tuning based methods in Table 4 tend to perform badly on intent tasks. FewJoint overcomes such unfair evaluations by introducing much more domains with various real intent-slot distributions, which can never be achieved by previous synthetic few-shot benchmarks.

On the FewJoint, our proposed method achieves the best results on Slot F1 compared to other baselines. This reflects the effectiveness of the mechanism we provide to use intent information to help slot prediction tasks. Following previous work [27], we reinforce the model with Transition Rule (+TR) tricks that force the model to predict legal slot tags, and both the slot and joint scores are significantly improved. Our model still achieves the best performance compared to those models that also use Transitions rules.

It is worth noting that although our approach does not always achieve the best on a single task, it achieves

\(^9\) During the simulation of 1-shot scenarios, each slot tag is sampled to appear at least 1 time, which lead to over-sampling of intents with much more co-occurring slots.
significant improvements in joint scores. This shows that our approach allows better joint learning of the two tasks and achieves better results overall.

For the other methods, we can see from Table 4 that JointProto, which simply shares parameters to achieve joint learning, does not perform as well as SepProto, which indicates that joint learning in few-shot scenarios is not just simple multi-task learning. Meanwhile, the significant improvement brought by LD-Proto indicates that joint learning can be effective through elaborate strategy design. Comparing JointTransfer and Meta-JOSFIN, it can be seen that the meta-learning approach can be fully effective and outperforms simple fine-tuning when a diverse and sufficient original domain is provided (on FewJoint).

### 6.3.2 Result of 5-shot setting

As shown in Table 5, our method achieves the best Joint Acc. performance on Snips and FewJoint datasets, which verify that the proposed trust gating mechanism and meta-learning strategy can effectively help dialogue understanding in more shot scenarios.

Although the increase in the number of learning shots lead to improved results for all baselines, our method still achieved the optimal results on all datasets and the advantages of our method have expanded. This shows that our model can better exploit the richer intent-slot relations hidden in 5-shot support sets, which makes our method better for joint learning. Also, we can see that fine-tuning methods achieve more improvements with more shots due to less overfitting. This is one of the reasons for the expansion of our advantages. For the same reason, other methods based on fine-tuning, such as Meta-JOSFIN and JointTransfer, gain an advantage over pure metric-learning based methods, such as JointProto and LD-Proto, in the 5-shot setting.

The slot score of our model significantly exceeds that of all baselines: 5.04 F1 improvements on Snips and 9.17 F1 improvements on FewJoint comparing with the strongest baseline (ConProm+FT). Even with the enhancements of the transfer rule, the improvements of our method are still significant. We attribute this to the fact that the intent task introduces much less noise in high-shot settings, and can better assist the slot filling task.

### 6.4 Analysis

#### 6.4.1 Ablation study

In this section, we perform an ablation study to understand the contribution of each component of our proposed method. As shown in Table 6, we conduct the analysis independently for the three components: trust gating mechanism, multi-task learning, and meta-learning.

When we remove the trust gating mechanism of intent information, intent information is directly fed to the slot decoder, and the performance drops in all settings. This demonstrates the importance of filtering out low-quality guidance of intent information and reliable intent information provided by the trust gating mechanism is helpful for the slot prediction task. We conduct a detailed analysis in the following sections.

For the method without multi-task learning, we train the model with a common Reptile that optimizes one task at a time. The losses the score demonstrate that updating the model by averaging the directions obtained from different tasks can better balance the knowledge learned from different tasks in various domains, and the refined Reptile used by our model allows learning better generalization from different tasks, which is crucial for transfer learning across domains.

Finally, we test the performance of the method without meta-learning, and the results show that the use of meta-learning has a very large gain in all settings. When the meta-learning is removed, the model is simply pre-trained with the source domain data, and then fine-tuned with the target domain data. The performance drops show that simple fine-tuning may be sensitive to the differences in knowledge

---

**Table 7** Analysis for the effectiveness of trust gating mechanism on the FewJoint dataset

| Setting | 1-shot | 5-shot | 1-shot | 5-shot |
|---------|--------|--------|--------|--------|
| Ours w/o gate | 64.99 | 85.47 | 86.27 |
| Ours | 46.06 | 63.39 | 66.16 |
| Intent-Guided Sentence Acc. | 30.65 | 57.09 |
| Rate of Correct Guiding | 44.07 | 55.49 |

The best results are highlighted in bold.

---

**Table 6** Ablation study over three main components of our proposed method: trust gating mechanism, smoothed multi-task learning and meta-learning. The score is Joint Accuracy

| Setting | Snips | FewJoint |
|---------|-------|----------|
| 1-shot | 5-shot | 1-shot | 5-shot |
| Ours | 22.05 | 50.73 | 30.65 | 57.09 |
| w/o trust gating mechanism | 2.40 | 3.70 | 1.45 | 1.60 |
| w/o multi-task learning | 2.57 | 1.60 | 3.67 | 1.13 |
| w/o meta learning | 11.97 | 18.03 | 15.78 | 18.68 |
between domains, which can be tackled by learning more generalized initialization with meta-learning.

### 6.4.2 Effectiveness of trust gating mechanism

To gain insight into the effectiveness of the trust gating mechanism and understand how it improves the slot performance, we focus on the samples that explicit intent information are passed to the slot task (through the gate), and perform analysis from three aspects:

1. **Rate of Correct Guiding**: the rate of correct intents among all intents used to guide slot prediction.

2. **Acc. of Intent-guided Slots**: accuracy of slots prediction for those samples that slot prediction is guided by intent information.

3. **Intent-Guided Sentence Acc.**: joint accuracy for those samples that slot prediction is guided by intent information.

Table 7 shows all the experimental results. We can see that trust gating mechanism helps to improve all three metrics in both 1 and 5 shots settings. Our method obtained higher results under the **Rate of Correct Guiding** metric indicating that the gate filtered intent information does have a higher quality and thus can better help slot. The improvements of trust gating mechanism in the **Acc. of Intent-guided Slots** metric shows that filtered intent information plays a good supporting role in the slot prediction task, and the results of the **Intent-Guided Sentence Acc.** metric confirm this conclusion. In general, from these analyses, we know that trust gating mechanism does improve the performance for both the slot task and...
joint accuracy, and this is achieved by refining the quality of intent guidance.

6.4.3 Effect of different confidence threshold values

Table 8 shows the variation of threshold value in trust gating mechanism and corresponding results. Overall, threshold $\theta = 0.5$ provides reliable performance. We have also used this setting in all our experiments. When comparing threshold $\theta = 0$ (guiding slots with all intents) and threshold $\theta = 1$ (predict slots without intent guidance), the performance gaps demonstrate the effectiveness of guiding slot filling with explicit intent information. While applying trust gating mechanism improves the performance, we observe that it is not very sensitive to the threshold value. This implies that the key is to filter out the most untrustworthy part of the intents, and the filtering of the rest of the intents is not so important.

6.4.4 Model performance of more shots setting

We examined the performance of the model under settings with more shots and compare our model with the strongest baseline method, i.e., ConProm. As shown in the Fig. 3, both our model and baseline model perform better as the support shots increases, but in three metrics of all settings, our model significantly outperforms baseline. The advantage of our model expands as the shots increase, which is mainly because our method is optimization-based meta learning method and can make better use of a larger support set than the metric-based method. When reinforce DU model with Transition Rule (+TR) and Fine Tuning (+FT), the prediction errors of both our and baseline model are reduced, but our model still works better, even when adding fine tuning to baseline, which proves the superiority of our method.

7 Conclusion

In this paper, we present a novel few-shot learning benchmark for joint dialogue understanding, which is also the first few-shot NLP benchmark for joint multi-task learning. Compared to the existing few-shot learning dataset that often has to construct fake domains, our benchmark consists of 59 real dialogue domains, which allows to better reflect real-world complexities of dialogue understanding. Further, we introduce a trust gating mechanism based model to guide slot filling with high-quality intent information, and introduce a Reptile-based training process to improve model generalizability on the unseen few-shot domain. In experiments on two datasets, the proposed methods significantly improve performance and achieve new state-of-the-art performance.

Acknowledgments We are grateful for the helpful comments and suggestions from the anonymous reviewers. This work was supported by the National Key R&D Program of China via grant 2020AA0106501 and the National Natural Science Foundation of China (NSFC) via grants 61976072 and 62176078.

References

1. Alex N, Lifland E, Tunstall L, et al (2021) Raft: A real-world few-shot text classification benchmark. In: NeurIPS Datasets and Benchmarks Track (Round 2)
2. Antoniou A, Edwards H, Storkey A (2019) How to train your maml. In: Proc. of ICLR
3. Baik S, Choi M, Choi J, et al (2020) Meta-learning with adaptive hyperparameters. In: NeurIPS
4. Bao Y, Wu M, Chang S, et al (2019) Few-shot text classification with distributional signatures. In: Proc. of ICLR
5. Bhathiya HS, Thayasivam U (2020) Meta learning for few-shot joint intent detection and slot-filling. In: ICMLT, pp 86–92
6. Budzianowski P, Wen TH, Tseng BH, et al (2018) Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In: Proc. of EMNLP, pp 5016–5026
7. Cao K, Brbic M, Leskovec J (2021) Concept learners for few-shot learning. In: Proc. of ICLR
8. Chada R, Natarajan P (2021) Fewshotqa: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. In: Proc. of EMNLP, pp 6081–6090
9. Chen Q, Zhuo Z, Wang W (2019) Bert for joint intent classification and slot filling. arXiv preprint arXiv:1902.10909
10. Chen Z, Ge J, Zhan H, et al (2021) Pareto self-supervised training for few-shot learning. In: Proc. of CVPR, pp 13,663–13,672
11. Coucke A, Saade A, Ball A, et al (2018) Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. CoRR abs/1805.10190
12. Das SSS, Katiyar A, Passonneau RJ, et al (2022) Container: Few-shot named entity recognition via contrastive learning. In: Proc. of ACL
13. Devlin J, Chang M, Lee K, et al (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Proc. of NAACL-HLT, pp 4171–4186
14. Ding N, Xu G, Chen Y, et al (2021) Few-nerd: A few-shot named entity recognition dataset. In: Proc. of ACL-IJCNLP, pp 3198–3213
15. Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research 12(7)
16. Eric M, Goel R, Paul S, et al (2020) Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In: Proceedings of the 12th Language Resources and Evaluation Conference, pp 422–428
17. Fei-Fei L (2006) Knowledge transfer in learning to recognize visual objects classes. In: ICDL, pp 1–8
18. Fei-Fei L, Fergus R, Perona P (2006) One-shot learning of object categories. IEEE transactions on pattern analysis and machine intelligence 28(4):594–611
19. Fink M (2005) Object classification from a single example utilizing class relevance metrics. In: NeurIPS, pp 449–456
20. Finn C, Abbeel P, Levine S (2017) Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML, pp 1126–1135
21. Gao T, Han X, Zhu H, et al (2019) Fewrel 2.0: Towards more challenging few-shot relation classification. In: Proc. of EMNLP-IJCNLP

22. Goo CW, Gao G, Hsu YK, et al (2018) Slot-gated modeling for joint slot filling and intent prediction. In: Proc. of NAACL-HLT, pp 753–757

23. Goyal P, Dollár P, Girshick R, et al (2017) Accurate, large mini-batch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677

24. Henderson M, Vulić I (2021) Convex: Data-efficient and few-shot slot labeling. In: Proc. of NAACL-HLT, pp 3375–3389

25. Hou Y, Liu Y, Che W, et al (2018) Sequence-to-sequence data augmentation for dialogue language understanding. In: Proc. of COLING, pp 1234–1245

26. Hou Y, Che W, Lai Y, et al (2020) Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In: Proc. of ACL

27. Hou Y, Lai Y, Chen C, et al (2021a) Learning to bridge metric spaces: Few-shot joint learning of intent detection and slot filling. In: Findings of ACL-IJCNLP, pp 3190–3200

28. Hou Y, Lai Y, Wu Y, et al (2021b) Few-shot learning for multi-label intent detection. In: Proc. of AAAI, pp 13,036–13,044

29. Kelley JF (1984) An iterative design methodology for user-friendly natural language office information applications. ACM Transactions on Information Systems (TOIS) 2(1):26–41

30. Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: Proc. of ICLR

31. Kroge J, Zhang Y, Diab M (2020) Learning to classify intents and slot labels given a handful of examples. In: Proc. of the 2nd Workshop on Natural Language Processing for Conversational AI

32. La Gatta V, Moscato V, Postiglione M, et al (2021) Few-shot named entity recognition with close questions. arXiv preprint arXiv:2111.12421

33. Lake BM, Salakhutdinov R, Tenenbaum JB (2015) Human-level concept learning through probabilistic program induction. Science 350(6266):1332–1338

34. Lee K, Maji S, Ravichandran A, et al (2019) Meta-learning with differentiable convex optimization. In: Proc. of CVPR, pp 10,657–10,665

35. Li C, Li L, Qi J (2018) A self-attentive model with gate mechanism for spoken language understanding. In: Proc. of EMNLP, pp 3824–3833

36. Loshchilov I, Hutter F (2018) Decoupled weight decay regularization. In: Proc. of ICLR

37. Malik V, Kumar A, Veppa J (2021) Exploring the limits of natural language inference based setup for few-shot intent detection. arXiv preprint arXiv:2112.07434

38. Mangla P, Kumari N, Sinha A, et al (2020) Charting the right manifold: Manifold mixup for few-shot learning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp 2218–2227

39. Meihao F (2021) Few-shot multi-hop question answering over knowledge base. arXiv preprint arXiv:2112.11909

40. Miller EG, Matsakis NE, Viola PA (2000) Learning from one example through shared densities on transforms. In: Proc. of CVPR, pp 464–471

41. Min S, Lewis M, Hajishirzi H, et al (2022) Noisy channel language model prompting for few-shot text classification. In: Proc. of ACL

42. Mittal A, Bharadwaj S, Khare S, et al (2021) Representation based meta-learning for few-shot spoken intent recognition. arXiv preprint arXiv:2106.15238

43. Mukherjee S, Liu X, Zheng G, et al (2021) Clues: Few-shot learning evaluation in natural language understanding. arXiv preprint arXiv:2111.02570

44. Nichol A, Achiam J, Schulman J (2018) On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02990

45. Onguz C, Vu NT (2021) Few-shot learning for slot tagging with attentive relational network. In: Proc. of EACL: Main Volume, pp 1566–1572

46. Peng B, Zhu C, Li C, et al (2020) Few-shot natural language generation for task-oriented dialog. In: Findings of EMNLP, pp 172–182

47. Qin L, Che W, Li Y, et al (2019) A stack-propagation framework with token-level intent detection for spoken language understanding. In: Proc. of EMNLP-IJCNLP, pp 2078–2087

48. Qin L, Xu X, Che W, et al (2020) Towards fine-grained transfer: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In: Findings of EMNLP, pp 1807–1816

49. Qin L, Li Z, Che W, et al (2021a) Co-gat: A co-interaction graph attention network for joint dialog act recognition and sentiment classification. In: Proc. of AAAI, pp 13,709–13,717

50. Qin L, Liu T, Che W, et al (2021b) A co-interaction transformer for joint slot filling and intent detection. In: Proc. of EMNLP

51. Rinze MN, Khan S, Khan FS, et al (2021) Exploring complementary strengths of invariant and equivariant representations for few-shot learning. In: Proc. of CVPR, pp 10,836–10,846

52. Rizvi AA, Rao D, Sygnowski J, et al (2018) Meta-learning with latent embedding optimization. In: Proc. of ICLR

53. Snell J, Swersky K, Zemel R (2017) Prototypical networks for few-shot learning. In: NeurIPS, pp 4077–4087

54. Tian Y, Wang Y, Krishnan D, et al (2020) Rethinking few-shot image classification: a good embedding is all you need? In: ECCV, pp 266–282

55. Tong M, Wang S, Xu B, et al (2021) Learning from miscellaneous other-class words for few-shot named entity recognition. In: Proc. of ACL-IJCNLP

56. Triantafillou E, Larochelle H, Zemel R, et al (2021) Learning a universal template for few-shot dataset generalization. In: ICML, pp 10,424–10,433

57. Vinyals O, Blundell C, Lillicrap T, et al (2016) Matching networks for one shot learning. In: NeurIPS

58. Wang B, Li L, Verma M, et al (2021a) Mtunet: Few-shot image classification with visual explanations. In: Proc. of CVPR, pp 2294–2298

59. Wang H, Wang Z, Fung GPC, et al (2021b) Mcmn: A novel memory-based contrastive meta-learning method for few shot slot tagging. arXiv preprint arXiv:2108.11635

60. Wang J, Wang KC, Rudzicz F, et al (2021c) Grad2task: Improved few-shot text classification using gradients for task representation. In: NeurIPS

61. Wang J, Chu H, Zhang C, et al (2021d) Learning from language description: Low-shot named entity recognition via decomposed framework. In: Findings of EMNLP, pp 1618–1630

62. Wei P, Zeng B, Liao W (2022) Joint intent detection and slot filling with wheel-graph attention networks. Journal of Intelligent & Fuzzy Systems pp 2409–2420

63. Worsham J, Kalita J (2020) Multi-task learning for natural language processing in the 2020s: where are we going? Pattern Recognition Letters

64. Xu L, Lu X, Yuan C, et al (2021) Fewclue: A chinese few-shot learning evaluation benchmark. arXiv preprint arXiv:2107.07498

65. Xu W, Wang H, Tu Z, et al (2020) Attentional constellation nets for few-shot learning. In: Proc. of ICLR

66. Yang S, Zhang Y, Niu G, et al (2021) Entity concept-enhanced few-shot relation extraction. In: Proc. of ACL (Volume 2: Short Papers), pp 987–991
68. Ye HJ, Hu H, Zhan DC, et al (2020) Few-shot learning via embedding adaptation with set-to-set functions. In: Proc. of CVPR, pp 8808–8817
69. Young S, Gašić M, Thomson B, et al (2013) Pomdp-based statistical spoken dialog systems: A review. In: IEEE, pp 1160–1179
70. Yu D, He L, Zhang Y, et al (2021) Few-shot intent classification and slot filling with retrieved examples. In: Proc. of NAACL-HLT
71. Zhang C, Cai Y, Lin G, et al (2020) Deepemd: Few-shot image classification with differentiable earth mover’s distance and structured classifiers. In: Proc. of CVPR, pp 12,203–12,213
72. Zhang J, Bui T, Yoon S, et al (2021a) Few-shot intent detection via contrastive pre-training and fine-tuning. In: Proc. of EMNLP, pp 1906–1912
73. Zhang L, Shi Y, Shou L, et al (2021b) A joint and domain-adaptive approach to spoken language understanding. arXiv preprint arXiv:2107.11768
74. Zheng Y, Zhou J, Qian Y, et al (2022) Fewnlu: Benchmarking state-of-the-art methods for few-shot natural language understanding. In: Proc. of ACL
75. Zhu Q, Huang K, Zhang Z et al (2020) Crosswoz: A large-scale chinese cross-domain task-oriented dialogue dataset. Transactions of the Association for Computational Linguistics 8:281–295

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.