Few-shot Slot Tagging with Collapsed Dependency Transfer and Label-enhanced Task-adaptive Projection Network

Yutai Hou[1], **Wanxiang Che**[1]*, **Yongkui Lai**[1], **Zhihan Zhou**[3],
Yijia Liu[2], **Han Liu**[3], **Ting Liu**[1]
[1]Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology
[2]Alibaba Group [3]Department of Computer Science, Northwestern University
{ythou, car, yklai, tliu}@ir.hit.edu.cn, oneplus.lau@gmail.com
zhihanzhou2020@u.northwestern.edu, hanliu@northwestern.edu

Abstract

In this paper, we explore the slot tagging with only a few labeled support sentences (a.k.a. few-shot). Few-shot slot tagging faces a unique challenge compared to the other few-shot classification problems as it calls for modeling the dependencies between labels. But it is hard to apply previously learned label dependencies to an unseen domain, due to the discrepancy of label sets. To tackle this, we introduce a *collapsed dependency transfer* mechanism into the conditional random field (CRF) to transfer abstract label dependency patterns as transition scores. In the few-shot setting, the emission score of CRF can be calculated as a word's similarity to the representation of each label. To calculate such similarity, we propose a *Label-enhanced Task-Adaptive Projection Network (L-TapNet)* based on the state-of-the-art few-shot classification model – TapNet, by leveraging label name semantics in representing labels. Experimental results show that our model significantly outperforms the strongest few-shot learning baseline by 14.64 F1 scores in the one-shot setting.[1]

1 Introduction

Slot tagging (Tur and De Mori, 2011), a key module in the task-oriented dialogue system (Young et al., 2013), is usually formulated as a sequence labeling problem (Sarikaya et al., 2016). Slot tagging faces the rapid changing of domains, and the labeled data is usually scarce for new domains with only a few samples. Few-shot learning technique (Miller et al., 2000; Fei-Fei et al., 2006; Lake et al., 2015; Vinyals et al., 2016) is appealing in this scenario since it learns the model that borrows the prior experience from old domains and adapts to new domains quickly with only very few examples (usually one or two examples for each class).

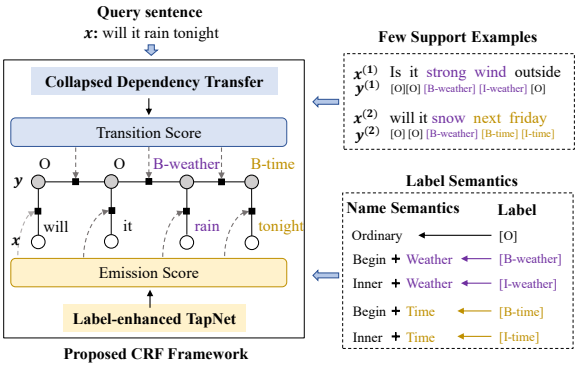

Figure 1: Our few-shot CRF framework for slot tagging.

Previous few-shot learning studies mainly focused on classification problems, which have been widely explored with similarity-based methods (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018; Yan et al., 2018; Yu et al., 2018). The basic idea of these methods is classifying an (query) item in a new domain according to its similarity with the representation of each class. The similarity function is usually learned in prior rich-resource domains and per class representation is obtained from few labeled samples (support set). It is straightforward to decompose the few-shot sequence labeling into a series of independent few-shot classifications and apply the similarity-based methods. However, sequence labeling benefits from taking the dependencies between labels into account (Huang et al., 2015; Ma and Hovy, 2016). To consider both the item similarity and label dependency, we propose to leverage the conditional random fields (Lafferty et al., 2001, CRFs) in few-shot sequence labeling (see Figure 1). In this paper, we translate the emission score of CRF into the output of the similarity-based method and calculate the transition score with a specially designed transfer mechanism.

The few-shot scenario poses unique challenges in learning the emission and transition scores of CRF. It is infeasible to learn the transition on the

*Corresponding author.
[1]Code is available at: https://github.com/AtmaHou/FewShotTagging

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1381–1393
July 5 - 10, 2020.

few labeled data, and prior label dependency in source domain cannot be directly transferred due to discrepancy in label set. To tackle the label discrepancy problem, we introduce the *collapsed dependency transfer* mechanism. It transfers label dependency information from source domains to target domains by abstracting domain-specific labels into abstract domain-independent labels and modeling the label dependencies between these abstract labels.

It is also challenging to compute the emission scores (word-label similarity in our case). Popular few-shot models, such as Prototypical Network (Snell et al., 2017), average the embeddings of each label's support examples as label representations, which often distribute closely in the embedding space and thus cause misclassification. To remedy this, Yoon et al. (2019) propose TapNet that learns to project embedding to a space where words of different labels are well-separated. We introduce this idea to slot tagging and further propose to improve label representation by leveraging the semantics of label names. We argue that label names are often semantically related to slot words and can help word-label similarity modeling. For example in Figure 1, word *rain* and label name *weather* are highly related. To use label name semantic and achieve good-separating in label representation, we propose *Label-enhanced TapNet* (L-TapNet) that constructs an embedding projection space using label name semantics, where label representations are well-separated and aligned with embeddings of both label name and slot words. Then we calculate similarities in the projected embedding space. Also, we introduce a *pair-wise embedding* mechanism to representation words with domain-specific context.

One-shot and five-shot experiments on slot tagging and named entity recognition show that our model achieves significant improvement over the strong few-shot learning baselines. Ablation tests demonstrate improvements coming from both L-TapNet and collapsed dependency transfer. Further analysis for label dependencies shows it captures non-trivial information and outperforms transition based on rules.

Our contributions are summarized as follows: (1) We propose a few-shot CRF framework for slot tagging that computes emission score as word-label similarity and estimate transition score by transferring previously learned label dependencies. (2) We introduce the collapsed dependency transfer mechanism to transfer label dependencies across domains with different label sets. (3) We propose the L-TapNet to leverage semantics of label names to enhance label representations, which help to model the word-label similarity.

2 Problem Definition

We define sentence $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ as a sequence of words and define label sequence of the sentence as $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$. A domain $\mathcal{D} = \left\{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\right\}_{i=1}^{N_D}$ is a set of $(\boldsymbol{x}, \boldsymbol{y})$ pairs. For each domain, there is a corresponding domain-specific label set $\mathcal{L}_\mathcal{D} = \{\ell_i\}_{i=1}^{N}$. To simplify the description, we assume that the number of labels N is same for all domains.

As shown in Figure 2, few-shot models are usually first trained on a set of source domains $\{\mathcal{D}_1, \mathcal{D}_2, \ldots\}$, then directly work on another set of unseen target domains $\{\mathcal{D}'_1, \mathcal{D}'_2, \ldots\}$ without fine-tuning. A target domain $\mathcal{D}'_j$ only contains few labeled samples, which is called support set $\mathcal{S} = \left\{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\right\}_{i=1}^{N_\mathcal{S}}$. $\mathcal{S}$ usually includes k examples (K-shot) for each of N labels (N-way).

The K-shot sequence labeling task is defined as follows: given a K-shot support set $\mathcal{S}$ and an input query sequence $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, find $\boldsymbol{x}$'s best label sequence $\boldsymbol{y}^*$:

$$\boldsymbol{y}^* = (y_1, y_2, \ldots, y_n) = \arg\max_{\boldsymbol{y}} \; p(\boldsymbol{y} \mid \boldsymbol{x}, \mathcal{S}).$$

3 Model

In this section, we first show the overview of the proposed CRF framework (§3.1). Then we discuss how to compute label transition score with collapsed dependency transfer (§3.2) and compute emission score with L-TapNet (§3.3).

3.1 Framework Overview

Conditional Random Field (CRF) considers both the transition score and the emission score to find the global optimal label sequence for each input. Following the same idea, we build our few-shot slot tagging framework with two components: Transition Scorer and Emission Scorer.

We apply the linear-CRF to the few-shot setting by modeling the label probability of label $\boldsymbol{y}$ given query sentence $\boldsymbol{x}$ and a K-shot support set $\mathcal{S}$:

$$p(\boldsymbol{y} \mid \boldsymbol{x}, \mathcal{S}) = \tfrac{1}{Z} \exp(\text{TRANS}(\boldsymbol{y}) + \lambda \cdot \text{EMIT}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{S})),$$

where $Z = \sum_{\boldsymbol{y}' \in \boldsymbol{Y}} \exp(\text{TRANS}(\boldsymbol{y}') + \lambda \cdot \text{EMIT}(\boldsymbol{y}', \boldsymbol{x}, \boldsymbol{S}))$,

$\text{TRANS}(\boldsymbol{y}) = \sum_{i=1}^{n} f_T(y_{i-1}, y_i)$ is the Transition

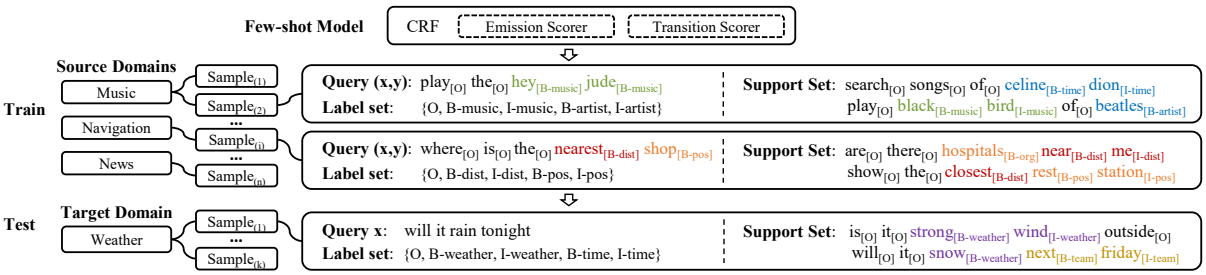

Figure 2: Overviews of training and testing. This figure illustrates the procedure of training the model on a set of source domains, and testing it on an unseen domain with only a support set.

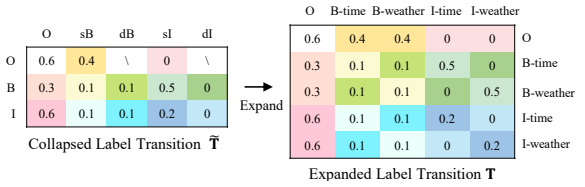

Figure 3: An example of collapsed label dependency transfer. We learn a collapsed label transition $\tilde{\boldsymbol{T}}$ and obtain specific label transition $\boldsymbol{T}$ by filling each position of it with value from $\tilde{\boldsymbol{T}}$ in the same color.

Scorer output and $\text{EMIT}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{S}) = \sum_{i=0}^{n} f_E(y_i, \boldsymbol{x}, \mathcal{S})$ is the Emission Scorer output. λ is a scaling parameter which balances weights of the two scores.

We take $L_{\text{CRF}} = -\log(p(\boldsymbol{y} \mid \boldsymbol{x}, \mathcal{S}))$ as loss function and minimize it on data from source domains. After the model is trained, we employ Viterbi algorithm (Forney, 1973) to find the best label sequence for each input.

3.2 Transition Scorer

The transition scorer component captures the dependencies between labels.[2] We model the label dependency as the transition probability between two labels:

$$f_T(y_{i-1}, y_i) = p(y_i \mid y_{i-1}).$$

Conventionally, such probabilities are learned from training data and stored in a transition matrix $\boldsymbol{T}^{N \times N}$, where N is the number of labels. For example, $\boldsymbol{T}_{\text{B-loc,B-team}}$ corresponds to $p(\text{B-loc} \mid \text{B-team})$. But in the few-shot setting, a model faces different label sets in the source domains (train) and the target domains (test). This mismatch on labels blocks the trained transition scorer directly working on a target domain.

Collapsed Dependency Transfer Mechanism We overcome the above issue by directly modeling the transition probabilities between abstract labels. Intuitively, we collapse specific labels into three abstract labels: O, B and I. To distinguish whether two labels are under the same or different semantics, we model transition from B and I to the same B (sB), a different B (dB), the same I (sI) and a different I (dI). We record such abstract label transition with a Table $\tilde{\boldsymbol{T}}^{3\times5}$ (see Figure 3). For example, $\tilde{\boldsymbol{T}}_{B,sB} = p(B\text{-}\ell_m \mid B\text{-}\ell_m)$ is the transition probability of two same B labels. And $\tilde{\boldsymbol{T}}_{B,dI} = p(I\text{-}\ell_n \mid B\text{-}\ell_m)$ is the transition probability from a B label to an I label with different types, where $\ell_m \neq \ell_n$. $\tilde{\boldsymbol{T}}_{O,sB}$ and $\tilde{\boldsymbol{T}}_{O,sI}$ respectively stands for the probability of transition from O to any B or I label.

To calculate the label transition probability for a new domain, we construct the transition matrix $\boldsymbol{T}$ by filling it with values in $\tilde{\boldsymbol{T}}$. Figure 3 shows the filling process, where positions in the same color are filled by the same values. For example, we fill $\boldsymbol{T}_{\text{B-loc,B-team}}$ with value in $\tilde{\boldsymbol{T}}_{B,dB}$.

3.3 Emission Scorer

As shown in Figure 4, the emission scorer independently assigns each word an emission score with regard to each label:

$$f_E(y_i, \boldsymbol{x}, \mathcal{S}) = p(y_i \mid \boldsymbol{x}, \mathcal{S}).$$

In few-shot setting, a word's emission score is calculated according to its similarity to representations of each label. To compute such emission, we propose the L-TapNet by improving TapNet (Yoon et al., 2019) with label semantics and prototypes.

3.3.1 Task-Adaptive Projection Network

TapNet is the state-of-the-art few-shot image classification model. Previous few-shot models, such as Prototypical Network, average the embeddings of each label's support example as label representations and directly compute word-label similarity in word embedding space. Different from them,

[2] Here, we ignore *Start* and *End* labels for simplicity. In practice, *Start* and *End* are included as two additional abstract labels.

TapNet calculates word-label similarity in a projected embedding space, where the words of different labels are well-separated. That allows TapNet to reduce misclassification. To achieve this, TapNet leverages a set of per-label reference vectors $\mathbf{\Phi} = [\boldsymbol{\phi}_1; \cdots; \boldsymbol{\phi}_N]$ as label representations. and construct a projection space based on these references. Then, a word x's emission score for label ℓ_j is calculated as its similarity to reference $\boldsymbol{\phi}_j$:

$$f_E(y_j, \boldsymbol{x}, \mathcal{S}) = \text{Softmax}\{\text{SIM}(\mathbf{M}(E(x)), \mathbf{M}(\boldsymbol{\phi}_j)\} \ ,$$

where $\mathbf{M}$ is a projecting function, E is an embedder and SIM is a similarity function. TapNet shares the references $\mathbf{\Phi}$ across different domains and constructs $\mathbf{M}$ for each specific domain by randomly associating the references to the specific labels.

Task-Adaptive Projection Space Construction Here, we present a brief introduction for the construction of projection space. Let $\mathbf{c}_j$ be the average of the embedded features for words with label ℓ_j in support set S. Given the $\mathbf{\Phi} = [\boldsymbol{\phi}_1; \cdots; \boldsymbol{\phi}_N]$ and support set S, TapNet constructs the projector $\mathbf{M}$ such that (1) each $\mathbf{c}_j$ and corresponding reference vector $\boldsymbol{\phi}_j$ align closely when projected by $\mathbf{M}$. (2) words of different labels are well-separated when projected by $\mathbf{M}$.

To achieve these, TapNet first computes the alignment bias between $\mathbf{c}_j$ and $\boldsymbol{\phi}_j$ in original embedding space, then it finds a projection $\mathbf{M}$ that eliminates this alignment bias and effectively separates different labels at the same time. Specifically, TapNet takes the matrix solution of a `linear error nulling` process as the embedding projector $\mathbf{M}$. For the detail process, refer to the original paper.

3.3.2 Label-enhanced TapNet

As mentioned in the introduction, we argue that label names often semantically relate to slot words and can help word-label similarity modeling. To enhance TapNet with such information, we use label semantics in both label representation and construction of projection space.

Projection Space with Label Semantics Let prototype $\mathbf{c}_j$ be the average of embeddings of words with label ℓ_j in support set. And $\mathbf{s}_j$ is semantic representation of label ℓ_j and Section 3.3.3 will introduce how to obtain it in detail. Intuitively, slot values ($\mathbf{c}_j$) and corresponding label name ($\boldsymbol{s}_j$) often have related semantics and they should be close

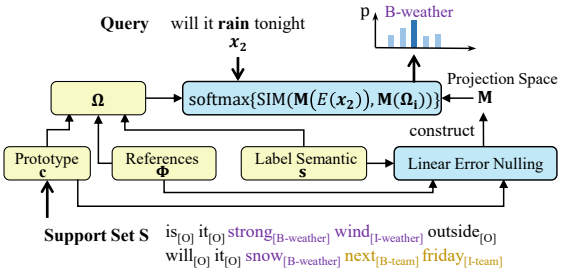

Figure 4: Emission Scorer with L-TapNet. It first constructs a projection space $\boldsymbol{M}$ by *linear error nulling* for given domain, and then predicts a word's emission score with its distance to label representation $\mathbf{\Omega}$ in the projection space.

in embedding space. So, we find a projector $\mathbf{M}$ that aligns $\mathbf{c}_j$ to both $\boldsymbol{\phi}_j$ and $\mathbf{s}_j$. The difference with TapNet is that it only aligns $\mathbf{c}_j$ to references $\boldsymbol{\phi}_j$ but we also require alignments with label representation. The label-enhanced reference is calculated as:

$$\boldsymbol{\psi}_j = (1 - \alpha) \cdot \boldsymbol{\phi}_j + \alpha \mathbf{s}_j,$$

where α is a balance factor. Label semantics $\mathbf{s}_j$ makes $\mathbf{M}$ specific for each domain. And reference $\boldsymbol{\phi}_j$ provides cross domain generalization.

Then we construct an $\mathbf{M}$ by linear error nulling of alignment error between label enhanced reference $\boldsymbol{\psi}_j$ and $\mathbf{c}_j$ following the same steps of TapNet.

Emission Score with Label Semantic For emission score calculation, compared to TapNet that only uses domain-agnostic reference $\boldsymbol{\phi}$ as label representation, we also consider the label semantics and use the label-enhanced reference $\boldsymbol{\psi}_j$ in label representation.

Besides, we further incorporate the idea of Prototypical Network and represent a label using a **prototype reference** $\mathbf{c}_j$ as $\mathbf{\Omega}_j = (1 - \beta) \cdot \mathbf{c}_j + \beta \boldsymbol{\psi}_j$. Finally, the emission score of x is calculated as its similarity to label representation $\mathbf{\Omega}$:

$$f_E(y_j, \boldsymbol{x}, \mathcal{S}) = \text{Softmax}\{\text{SIM}(\mathbf{M}(E(x)), \mathbf{M}(\mathbf{\Omega}_j)\} \ ,$$

where SIM is the dot product similarity function and E is a word embedding function which will be introduced in the next section.

3.3.3 Embeddings for Word and Label Name

For the **word embedding function** E, we proposed a *pair-wise embedding* mechanism. As shown in Figure 5, a word tends to mean differently when concatenated to a different context. To tackle the representation challenges for similarity computation, we consider the special query-support setting in few-shot learning and embed query and

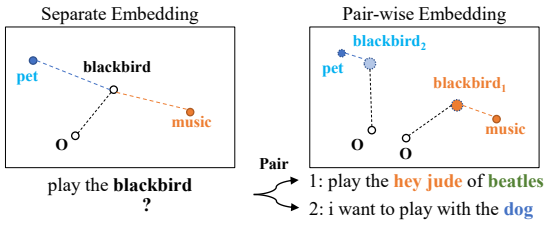

Figure 5: An example of pair-wise embedding. When embedding query and support sentences separately (left), it is hard to tag *blackbird* according to its similarity to labels. But if we embed query by pairing it with different support sentences (right), the domain specific context provide *blackbird* certain meanings close to *pet* and *song* respectively.

Domain	1-shot		5-shot	
	Ave. $\|S\|$	Samples	Ave. $\|S\|$	Samples
We	6.15	2,000	28.91	1,000
Mu	7.66	2,000	34.43	1,000
Pl	2.96	2,000	13.84	1,000
Bo	4.34	2,000	19.83	1,000
Se	4.29	2,000	19.27	1,000
Re	9.41	2,000	41.58	1,000
Cr	1.30	2,000	5.28	1,000

Table 1: Overview of few-shot slot tagging data. Here, "Ave. $|S|$" corresponds to the average support set size of each domain. And "Sample" stands for the number of few-shot samples we build from each domain.

support words pair-wisely. Such pair-wise embedding can make use of domain-related context in support sentences and provide domain adaptive embeddings for the query words. This will further help to model the query words' similarity to domain-specific labels. To achieve this, we represent each word with self-attention over both query and support words. We first copy query sentence $\boldsymbol{x}$ for $N_{\mathcal{S}} = |\mathcal{S}|$ times, and pair them with all support sentences. Then the $N_{\mathcal{S}}$ pairs are passed to a BERT (Devlin et al., 2019) to get $N_{\mathcal{S}}$ embeddings for each query word. We represent each word as the average of $N_{\mathcal{S}}$ embeddings. Now, representations of query words are conditioned on domain-specific context. We use BERT as it can naturally capture the relation between sentence pairs.

To get label representation $\mathbf{s}$, we first concatenate abstract label name (e.g., *begin* and *inner*) and label name (e.g., *weather*). Then, we insert a `[CLS]` token at the first position, and input them into a BERT. Finally, the representation of `[CLS]` is used as the **label semantic embedding**.

4 Experiment

We evaluate the proposed method on *slot tagging* and test its generalization ability on a similar sequence labeling task: *name entity recognition* (NER). Due to space limitation, we only present the detailed results for 1-shot/5-shot slot tagging, which transfers the learned knowledge from source domains (training) to an unseen target domain (testing) containing only a 1-shot/5-shot support set. The results of NER are consistent and we present them in the supplementary Appendix B.

4.1 Settings

Dataset For slot tagging, we exploit the `snips` dataset (Coucke et al., 2018), because it contains 7 domains with different label sets and is easy to simulate the few-shot situation. The domains are Weather (We), Music (Mu), PlayList (Pl), Book (Bo), Search Screen (Se), Restaurant (Re) and Creative Work (Cr). Information about original datasets is shown in Appendix A.

To simulate the few-shot situation, we construct the few-shot datasets from original datasets, where each sample is the combination of a query data $(\boldsymbol{x}^{\boldsymbol{q}}, \boldsymbol{y}^{\boldsymbol{q}})$ and corresponding K-shot support set $\mathcal{S}$. Table 1 shows the overview of the experiment data.

Few-shot Data Construction Different from the simple classification of single words, slot tagging is a structural prediction problem over the entire sentence. So we construct support sets with sentences rather than single words under each tag.

As a result, the normal *N-way K-shot* few-shot definition is inapplicable for few-shot slot tagging. We cannot guarantee that each label appears K times while sampling the support sentences, because different slot labels randomly co-occur in one sentence. For example in Figure 1, in the 1-shot support set, label `[B-weather]` occurs twice to ensure all labels appear at least once. So we approximately construct K-shot support set $\mathcal{S}$ following two criteria: (1) All labels within the domain appear at least K times in $\mathcal{S}$. (2) At least one label will appear less than K times in $\mathcal{S}$ if any $(\boldsymbol{x}, \boldsymbol{y})$ pair is removed from it. Algorithm 1 shows the detail process.[3]

Here, we take the 1-shot slot tagging as an example to illustrate the data construction procedure. For each domain, we sample 100 different 1-shot support sets. Then, for each support set, we sample 20 unincluded utterances as queries (query set). Each support-query-set pair forms one **few-shot episode**.

[3]Due to the removing step, Algorithm 1 has a preference for sentences with more slots. So in practice, we randomly skip removing by the chance of 20%.

Algorithm 1: Minimum-including

Input: # of shot K, domain $\mathcal{D}$, label set $\mathcal{L}_\mathcal{D}$

Initialize support set $\mathcal{S} = \{\}$, $\text{Count}_{\ell_j} = 0$ $(\forall \ell_j \in \mathcal{L}_\mathcal{D})$

for ℓ *in* $\mathcal{L}_\mathcal{D}$ **do**

  while $Count_\ell < k$ **do**

    From $\mathcal{D} \setminus \mathcal{S}$, randomly sample a $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ pair that $\boldsymbol{y}^{(i)}$ includes ℓ

    Add $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ to $\mathcal{S}$

    Update all Count_{ℓ_j} $(\forall \ell_j \in \mathcal{L}_\mathcal{D})$

for *each* $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ *in* $\mathcal{S}$ **do**

  Remove $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ from $\mathcal{S}$

  Update all all Count_{ℓ_j} $(\forall \ell_j \in \mathcal{L}_\mathcal{D})$

  if *any* $Count_{\ell_j} < k$ **then**

    Put $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ back to $\mathcal{S}$

    Update all Count_{ℓ_j} $(\forall \ell_j \in \mathcal{L}_\mathcal{D})$

Return $\mathcal{S}$

Eventually, we get 100 episodes and 100×20 samples (1 query utterance with a support set) for each domain.

Evaluation To test the robustness of our framework, we cross-validate the models on different domains. Each time, we pick one target domain for testing, one domain for development, and use the rest domains as source domains for training. So for slot tagging, all models are trained on 10,000 samples, and validated as well as tested on 2,000 samples respectively.

When testing model on a target domain, we evaluate F1 scores within each few-shot episode.[4] Then we average 100 F1 scores from all 100 episodes as the final result to counter the randomness from support-sets. All models are evaluated on same support-query-set pairs for fairness.

To control the nondeterministic of neural network training (Reimers and Gurevych, 2017), we report the average score of 10 random seeds.

Hyperparameters We use the uncased `BERT-Base` (Devlin et al., 2019) to calculate contextual embeddings for all models. We use ADAM (Kingma and Ba, 2015) to train the models with batch size 4 and a learning rate of 1e-5. For the CRF framework, we learn the scaling parameter λ during training, which is important to get stable results. For L-TapNet, we set α as 0.5 and β as 0.7. We fine-tune BERT with Gradual Unfreezing trick (Howard and Ruder, 2018). For both proposed and baseline models, we take early stop in training and fine-tuning when there is no loss decay withing a fixed number of steps.

[4]For each episode, we calculate the F1 score on query samples with `conlleval` script: https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt

4.2 Baselines

Bi-LSTM is a bidirectional LSTM (Schuster and Paliwal, 1997) with GloVe (Pennington et al., 2014) embedding for slot tagging. It is trained on the support set and tested on the query samples.

SimBERT is a model that predicts labels according to cosine similarity of word embedding of non-fine-tuned BERT. For each word x_j, SimBERT finds its most similar word x'_k in support set, and the label of x_j is predicted to be the label of x'_k.

TransferBERT is a domain transfer model with the NER setting of BERT (Devlin et al., 2019). We pretrain the it on source domains and select the best model on the same dev set of our model. We deal with label mismatch by only transferring bottleneck feature. Before testing, we fine-tune it on target domain support set. Learning rate is set as 1e-5 in training and fine-tuning.

WarmProtoZero (WPZ) (Fritzler et al., 2019) is a few-shot sequence labeling model that regards sequence labeling as classification of every single word. It pre-trains a prototypical network (Snell et al., 2017) on source domains, and utilize it to do word-level classification on target domains without training. Fritzler et al. (2019) use randomly initialized word embeddings. To eliminate the influence of different embedding methods, we further implement WPZ with the pre-trained embedding of GloVe (Pennington et al., 2014) and BERT.

Matching Network (MN) is similar to WPZ. The only difference is that we employ the matching network (Vinyals et al., 2016) with BERT embedding for classification.

4.3 Main Results

Results of 1-shot Setting Table 2 shows the 1-shot slot tagging results. Each column respectively shows the F1 scores of taking a certain domain as target domain (test) and use others as source domain (train & dev). As shown in the tables, our L-TapNet+CDT achieves the best performance. It outperforms the strongest few-shot learning baseline WPZ+BERT by average F1 scores of 14.64.

Our model significantly outperforms Bi-LSTM and TransferBERT, indicating that the number of labeled data under the few-shot setting is too scarce for both conventional machine learning and transfer

Model	1-shot Slot Tagging							
	We	Mu	Pl	Bo	Se	Re	Cr	Ave.
Bi-LSMT	10.36	17.13	17.52	53.84	18.44	22.56	8.64	21.21
SimBERT	36.10	37.08	35.11	68.09	41.61	42.82	23.91	40.67
TransferBERT	55.82	38.01	45.65	31.63	21.96	41.79	38.53	39.06
MN	21.74	10.68	39.71	58.15	24.21	32.88	**69.66**	36.72
WPZ	4.53	7.43	14.43	39.15	11.69	7.78	10.09	13.59
WPZ+GloVe	17.92	22.37	19.90	42.61	22.30	22.79	16.75	23.52
WPZ+BERT	46.72	40.07	50.78	68.73	60.81	55.58	67.67	55.77
TapNet	51.12	40.65	48.41	77.50	49.77	54.79	61.39	54.80
TapNet+CDT	66.30	55.93	57.55	83.32	64.45	65.65	67.91	65.87
L-WPZ+CDT	71.23	47.38	59.57	81.98	69.83	66.52	62.84	65.62
L-TapNet+CDT	**71.53**	**60.56**	**66.27**	**84.54**	**76.27**	**70.79**	62.89	**70.41**

Table 2: F1 scores on 1-shot slot tagging. `+CDT` denotes `collapsed dependency transfer`. Score below mid-line are from our methods, which achieve the best performance. Ave. shows the averaged scores. Results with standard deviations is showed in Appendix D.

Model	5-shots Slot Tagging							
	We	Mu	Pl	Bo	Se	Re	Cr	Ave.
Bi-LSMT	25.17	39.80	46.13	74.60	53.47	40.35	25.10	43.52
SimBERT	53.46	54.13	42.81	75.54	57.10	55.30	32.38	52.96
TransferBERT	59.41	42.00	46.07	20.74	28.20	67.75	58.61	46.11
MN	36.67	33.67	52.60	69.09	38.42	33.28	72.10	47.98
WPZ	9.54	14.23	18.12	44.65	18.98	12.03	14.05	18.80
WPZ+GloVe	26.61	34.25	22.11	50.55	28.53	34.16	23.69	31.41
WPZ+BERT	67.82	55.99	46.02	72.17	73.59	60.18	66.89	63.24
TapNet	53.03	49.80	54.90	83.36	63.07	59.84	67.02	61.57
TapNet+CDT	66.48	66.36	68.23	**85.76**	73.60	64.20	68.47	70.44
L-WPZ+CDT	**74.68**	56.73	52.20	78.79	80.61	69.59	67.46	68.58
L-TapNet+CDT	71.64	**67.16**	**75.88**	84.38	**82.58**	**70.05**	**73.41**	**75.01**

Table 3: F1 score results on 5-shots slot tagging. Our methods achieve the best performance. Results with standard deviations is showed in Appendix D.

learning models. Moreover, the performance of SimBERT demonstrates the superiority of metric-based methods over conventional machine learning models in the few-shot setting.

The original WarmProtoZero (WPZ) model suffers from the weak representation ability of its word embeddings. When we enhance it with GloVe and BERT word embeddings, its performance improves significantly. This shows the importance of embedding in the few-shot setting. Matching Network (MN) performs poorly in both settings. This is largely due to the fact that MN pays attention to all support word equally, which makes it vulnerable to the unbalanced amount of O-labels.

More specifically, those models that are fine-tuned on support set, such as Bi-LSTM and TransferBERT, tend to predict tags randomly. Those systems can only handle the cases that are easy to generalize from support examples, such as tags for proper noun tokens (e.g. city name and time). This shows that fine-tuning on extremely limited examples leads to poor generalization ability and undertrained classifier. And for those metric based methods, such as WPZ and MN, label prediction is much more reasonable. However, these models are easy to be confused by similar labels, such as *current_location* and *geographic_poi*. It indicates the necessity of well-separated label representations. Also illegal label transitions are very common, which can be well tackled by the proposed collapsed dependency transfer.

To eliminate unfair comparisons caused by additional information in label names, we propose the **L-WPZ+CDT** by enhancing the WarmProtoZero (WPZ) model with label name representation same to L-TapNet and incorporating it into the proposed CRF framework. It combines label name embedding and prototype as each label representation. Its improvements over WPZ mainly come from label semantics, collapsed dependency transfer and pair-wise embedding. L-TapNet+CDT outperforms L-WPZ+CDT by 4.79 F1 scores demonstrating the effectiveness of embedding projection. When compared with TapNet+CDT, L-TapNet+CDT achieves

an improvement of 4.54 F-score on average, which shows that considering label semantics and prototype helps improve emission score calculation.

Results of 5-shots Setting Table 3 shows the results of 5-shots experiments, which verify the proposed model's generalization ability in more shots situations. The results are consistent with 1-shot setting in general trending.

4.4 Analysis

Ablation Test To get further an understanding of each component in our method (L-TapNet+CDT), we conduct ablation analysis on both 1-shot and 5-shots setting in Table 4. Each component of our method is removed respectively, including: *collapsed dependency transfer*, *pair-wise embedding*, *label semantic*, and *prototype reference*.

When collapsed dependency transfer is removed, we directly predict labels with emission score and huge F1 score drops are witnessed in all settings. This ablation demonstrates a great necessity for considering label dependency.

For our method without pair-wise embedding, we represent query and support sentences independently. We address the drop to the fact that support sentences can provide domain-related context, and pair-wise embedding can leverage such context and provide domain-adaptive representation for words in query sentences. This helps a lot when computing a word's similarity to domain-specific labels.

When we remove the label-semantic from L-TapNet, the model degenerates into TapNet+CDT enhanced with prototype in emission score. The drops in results show that considering label name can provide better label representation and help to model word-label similarity. Further, we also tried to remove the inner and beginning words in label representation and observe a 0.97 F1-score drop on 1-shot SNIPS. It shows that distinguishing B-I labels in label semantics can help tagging.

And if we calculate emission score without the prototype reference, the model loses more performance in 5-shots setting. This meets the intuition that prototype allows model to benefit more from the increase of support shots, as prototypes are directly derived from the support set.

Analysis of Collapsed Dependency Transfer While collapsed dependency transfer (CDT) brings significant improvements, two natural questions arise: whether CDT just learns simple transition rules and why it works.

Model	1-shot	5-shots
Ours	70.41	75.01
- dependency transfer	-10.01	-8.08
- pair-wise embedding	-8.29	-7.74
- label semantic	-9.57	-4.87
- prototype reference	-1.73	-3.33

Table 4: Ablation test over different components on slot tagging task. Results are averaged F1-score of all domains.

Model	1-shot	5-shots
L-TapNet	60.40	66.93
L-TapNet+Rule	65.30	69.64
L-TapNet+CDT	**70.41**	**75.01**

Table 5: Comparison between transition rules and collapsed dependency transfer (CDT).

To answer the first question, we replace CDT with transition rules in Table 5,[5] which shows CDT can bring more improvements than transition rules.

To have a deeper insight into the effectiveness of CDT, we conduct an accuracy analysis of it. We assess the label predicting accuracy of different types of label bi-grams. The result is shown in Table 6. We further summarize the bi-grams into 2 categories: **Border** includes the bi-grams across the border of a slot span; **Inner** is the bi-grams within a slot span. We argue that improvements of **Inner** show successful reduction of illegal label transition from CDT. Interestingly, we observe that CDT also brings improvements by correctly predict the first and last token of a slot span. The results of **Border** verified our observation that CDT may helps to decide the boundaries of slot spans more accurately, which is hard to achieve by adding transition rules.

5 Related Works

Traditional few-shot learning methods depend highly on hand-crafted features (Fei-Fei, 2006; Fink, 2005). Classical methods primarily focus on metric learning (Snell et al., 2017; Vinyals et al., 2016), which classifies an item according to its similarity to each class's representation. Recent efforts (Lu et al., 2018; Schwartz et al., 2019) propose to leverage the semantics of class name to enhance class representation. However, different from us, these methods focus on image classification where effects of name semantic are implicit and label dependency is not required.

Few-shot learning in natural language process-

[5]Transition Rule: We greedily predict the label for each word and block the result that conflicts with previous label.

Bi-gram Type		Proportion	L-TapNet	+CDT
Border	O-O	28.5%	82.7%	**83.7%**
	O-B	24.5%	78.3%	**81.5%**
	B-O	8.2%	72.4%	**74.8%**
	I-O	5.8%	76.7%	**81.7%**
	I-B/B-B	7.8%	65.0%	**72.5%**
Inner	B-I	13.3%	78.5%	**83.6%**
	I-I	12.1%	77.8%	**82.7%**

Table 6: Accuracy analysis of label prediction on 1-shot slot tagging. The table shows accuracy and proportion of different bi-gram types in dataset.

ing has been explored for classification tasks, including text classification (Sun et al., 2019; Geng et al., 2019; Yan et al., 2018; Yu et al., 2018), entity relation classification (Lv et al., 2019; Gao et al., 2019; Ye and Ling, 2019), and dialog act prediction (Vlasov et al., 2018). However, few-shot learning for slot tagging is less investigated. Luo et al. (2018) investigated few-shot slot tagging using additional regular expressions, which is not comparable to our model due to the usage of regular expressions. Fritzler et al. (2019) explored few-shot named entity recognition with the Prototypical Network, which has a similar setting to us. Compared to it, our model achieves better performance by considering both label dependency transferring and label name semantics. Zero-shot slot tagging methods (Bapna et al., 2017; Lee and Jha, 2019; Shah et al., 2019) share a similar idea to us in using label name semantics, but has a different setting as few-shot methods are additionally supported by a few labeled sentences. Chen et al. (2016) investigate using label name in intent detection. In addition to learning directly from limited example, another research line of solving data scarcity problem in NLP is data augmentation (Fader et al., 2013; Zhang et al., 2015; Liu et al., 2017). For data augmentation of slot tagging, sentence generation based methods are explored to create additional labeled samples (Hou et al., 2018; Shin et al., 2019; Yoo et al., 2019).

6 Conclusion

In this paper, we propose a few-shot CRF model for slot tagging of task-oriented dialogue. To compute transition score under few-shot setting, we propose the collapsed dependency transfer mechanism, which transfers the prior knowledge of the label dependencies across domains with different label sets. And we propose L-TapNet to calculate emission score, which improves label representation with label name semantics. Experiment results validate that both the collapsed dependency transfer and L-TapNet can improve the tagging accuracy.

Acknowledgments

We sincerely thank Ning Wang and Jiafeng Mao for the help on both paper and experiments. We are grateful for the helpful comments and suggestions from the anonymous reviewers. This work was supported by the National Natural Science Foundation of China (NSFC) via grant 61976072, 61632011 and 61772153.

References

Ankur Bapna, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *arXiv preprint arXiv:1707.02363*.

Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. 2016. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models. In *Proc. of the ICASSP*, pages 6045–6049. IEEE.

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.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proc. of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. of the NAACL-HLT, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proc. of the ACL*.

Li Fei-Fei. 2006. Knowledge transfer in learning to recognize visual objects classes. In *International Conference on Development and Learning*, pages 1–8.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611.

Michael Fink. 2005. Object classification from a single example utilizing class relevance metrics. In *NIPS*, pages 449–456.

G David Forney. 1973. The viterbi algorithm. *Proc. of the IEEE*, 61(3):268–278.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proc. of the SAC*, pages 993–1000.

Tianyu Gao, Xu Han, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2019. Neural snowball for few-shot relation learning. *arXiv preprint arXiv:1908.11007*.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *Proc. of the EMNLP-IJCNLP*, pages 3895–3904.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1234–1245.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proc. of the ACL, Volume 1: Long Papers*, pages 328–339.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of the ICLR*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the ICML*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.

Sungjin Lee and Rahul Jha. 2019. Zero-shot adaptive transfer for conversational language understanding. In *Proc. of the AAAI*, volume 33, pages 6642–6649.

Ting Liu, Yiming Cui, Qingyu Yin, Wei-Nan Zhang, Shijin Wang, and Guoping Hu. 2017. Generating and exploiting large-scale pseudo training data for zero pronoun resolution. In *Proc. of the ACL*, pages 102–111.

Zhiwu Lu, Jiechao Guan, Aoxue Li, Tao Xiang, An Zhao, and Ji-Rong Wen. 2018. Zero and few shot learning with semantic feature synthesis and competitive learning. *arXiv preprint arXiv:1810.08332*.

Bingfeng Luo, Yansong Feng, Zheng Wang, Songfang Huang, Rui Yan, and Dongyan Zhao. 2018. Marrying up regular expressions with neural networks: A case study for spoken language understanding. In *Proc. of the ACL, Volume 1: Long Papers*, pages 2083–2093.

Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2019. Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations. *arXiv preprint arXiv:1908.11513*.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proc. of the ACL, Volume 1: Long Papers*.

Erik G Miller, Nicholas E Matsakis, and Paul A Viola. 2000. Learning from one example through shared densities on transforms. In *Proc. of the CVPR*, volume 1, pages 464–471. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proc. of the EMNLP*, pages 1532–1543.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proc. of the CoNLL*, pages 143–152.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging. In *Proc. of the EMNLP*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proc. of the CoNLL-HLT-NAACL*, pages 142–147.

Ruhi Sarikaya, Paul A Crook, Alex Marin, Minwoo Jeong, Jean-Philippe Robichaud, Asli Celikyilmaz, Young-Bum Kim, Alexandre Rochette, Omar Zia Khan, Xiaohu Liu, et al. 2016. An overview of end-to-end language understanding and dialog management for personal digital assistants. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 391–397. IEEE.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45(11):2673–2681.

Eli Schwartz, Leonid Karlinsky, Rogerio Feris, Raja Giryes, and Alex M Bronstein. 2019. Baby steps towards few-shot learning with multiple semantics. *arXiv preprint arXiv:1906.01905*.

Darsh J. Shah, Raghav Gupta, Amir A. Fayazi, and Dilek Hakkani-Tür. 2019. Robust zero-shot cross-domain slot filling with example values. In *Proc. of the ACL, Volume 1: Long Papers*, pages 5484–5490.

Y. Shin, K. M. Yoo, and S. Lee. 2019. Utterance generation with variational auto-encoder for slot filling in spoken language understanding. *IEEE Signal Processing Letters*, 26(3):505–509.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NIPS*, pages 4077–4087.

Shengli Sun, Qingfeng Sun, Kevin Zhou, and Tengchao Lv. 2019. Hierarchical attention prototypical networks for few-shot text classification. In *Proc. of the EMNLP-IJCNLP*, pages 476–485.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proc. of the CVPR*, pages 1199–1208.

Gokhan Tur and Renato De Mori. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *NIPS*, pages 3630–3638.

Vladimir Vlasov, Akela Drissner-Schmid, and Alan Nichol. 2018. Few-shot generalization across dialogue tasks. *arXiv preprint arXiv:1811.11707*.

Leiming Yan, Yuhui Zheng, and Jie Cao. 2018. Few-shot learning for short text classification. *Multimedia Tools and Applications*, pages 1–12.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-level matching and aggregation network for few-shot relation classification. In *Proc. of the ACL, Volume 1: Long Papers*, pages 2872–2881.

Kang Min Yoo, Youhyun Shin, and Sang-goo Lee. 2019. Data augmentation for spoken language understanding via joint variational generation. In *Proc. of the AAAI*, volume 33, pages 7402–7409.

Sung Whan Yoon, Jun Seo, and Jaekyun Moon. 2019. Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. In *Proc. of the ICML*, pages 7115–7123.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proc. of the IEEE*, 101(5):1160–1179.

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 *Proc. of the NAACL-HLT, Volume 1 (Long Papers)*, pages 1206–1215.

Amir Zeldes. 2017. The gum corpus: creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proc. of the NIPS*, pages 649–657.

Appendices

A Detail of Dataset

Table 7 shows the statistics of the original dataset used to construct few-shot experiment data.

Task	Dataset	Domain	# Sent	# Labels
Slot Tagging	Snips	We	2,100	10
		Mu	2,100	10
		Pl	2,042	6
		Bo	2056	8
		Se	2,059	8
		Re	2,073	15
		Cr	2,054	3
NER	CoNLL	News	20679	5
	GUM	WiKi	3,493	12
	WNUT	Social	5,657	7
	OntoNotes	Mixed	159,615	19

Table 7: Statistic of Original Dataset

B Few-shot experiments for Name entity recognition

Name entity recognition (NER) that identify predefined name entities, such as the person names, organizations and locations, can be modeled as a slot tagging task. Also, the data scarcity problem for a new domain exists in the NER task. For the above reasons, we conduct few-shot NER experiments to test our model's generation ability.

Domain	1-shot		5-shots	
	Ave. $\|S\|$	Samples	Ave. $\|S\|$	Samples
News	3.38	4,000	15.58	1,000
Wiki	6.50	4,000	27.81	1,000
Social	5.48	4,000	28.66	1,000
Mixed	14.38	2,000	62.28	1,000

Table 8: Overview of few-shot data for NER experiments. Here, "Ave. $|S|$" corresponds to the average support set size of each domain. And "Sample" stands for the number of few-shot samples we build from each domain.

Experiment Data for Few-shot NER For named entity recognition, we utilize 4 different datasets: `CoNLL-2003` (Sang and Meulder, 2003), `GUM` (Zeldes, 2017), `WNUT-2017` (Derczynski et al., 2017) and `Ontonotes` (Pradhan et al., 2013), each of which contains data from only 1 domain. The 4 domains are News, Wiki, Social and Mixed. Detail of the original data set is showed in Table 7 and statistic of constructed few-shot data is showed in Table 8.

Model	1-shot Named Entity Recognition				
	News	Wiki	Social	Mixed	Ave.
Bi-LSMT	2.57 ±0.14	3.29 ±0.19	0.67 ±0.07	2.11 ±0.15	2.16 ±0.14
SimBERT	19.22 ±0.00	6.91 ±0.00	5.18 ±0.00	13.99 ±0.00	11.32 ±0.00
TransferBERT	4.75 ±1.42	0.57 ±0.32	2.71 ±0.72	3.46 ±0.54	2.87 ±0.75
MN	19.50 ±0.35	4.73 ±0.16	17.23 ±2.75	15.06 ±1.61	14.13 ±1.22
WPZ	3.64 ±0.08	2.00 ±0.02	0.92 ±0.04	0.66 ±0.03	1.80 ±0.04
WPZ+GloVe	9.40 ±0.06	3.23 ±0.01	2.29 ±0.02	2.56 ±0.01	4.37 ±0.03
WPZ+BERT	32.49 ±2.01	3.89 ±0.24	10.68 ±1.40	6.67 ±0.46	13.43 ±1.03
L-TapNet+CDT	**44.30** ±3.15	**12.04** ±0.65	**20.80** ±1.06	**15.17** ±1.25	**23.08** ±1.53

Table 9: F1 scores on 1-shot name entity recognition. `CDT` denotes `collapsed dependency transfer`. Scores below mid-line are from our models, which achieve the best performance. Ave. shows the averaged scores.

Model	5-shots Named Entity Recognition				
	News	Wiki	Social	Mixed	Ave.
Bi-LSMT	6.81 ±0.40	8.40 ±0.16	1.06 ±0.16	13.17 ±0.17	7.36 ±0.22
SimBERT	32.01 ±0.00	10.63 ±0.00	8.20 ±0.00	21.14 ±0.00	18.00 ±0.00
TransferBERT	15.36 ±2.81	3.62 ±0.57	11.08 ±0.57	**35.49** ±7.60	16.39 ±2.89
MN	19.85 ±0.74	5.58 ±0.23	6.61 ±1.75	8.08 ±0.47	10.03 ±0.80
WPZ	4.09 ±0.16	3.19 ±0.13	0.86 ±0.23	0.93 ±0.14	2.27 ±0.17
WPZ+GloVe	16.94 ±0.10	5.33 ±0.07	5.53 ±0.12	3.54 ±0.03	7.83 ±0.08
WPZ+BERT	**50.06** ±1.57	9.54 ±0.44	17.26 ±2.65	13.59 ±1.61	22.61 ±1.57
L-TapNet+CDT	45.35 ±2.67	**11.65** ±2.34	**23.30** ±2.80	20.95 ±2.81	**25.31** ±2.65

Table 10: . F1 score results on 5-shots name entity recognition. Our methods achieve the best performance.

Model	1-shot	5-shots
Ours	22.19	24.12
- dependency transfer	-4.55	-4.83
- label semantic	-6.93	-1.46

Table 11: Ablation test over different components on NER task. Results are averaged F1-score of all domains.

1-shot and 5-shots Results for NER Table 9 and Table 10 respectively show the 1-shot and 5-shots name entity recognition results. Our best model outperforms all baseline in both settings.

The trend of results is consistent with slot-tagging results. But the overall score is much lower than slot-tagging results. this is because NER domains are from different datasets and the domain gap is much larger.

Our improvements on 5-shots is narrowed in margin. This is because NER domains have different genres and vocabulary. So compared to SNIPS, it is harder to transfer knowledge but benefits more to rely on domain-specific support examples. This trend is even more pronounced with more shots. In 5-shots setting, the strongest baseline WPZ benefits more from the increased shots because it only uses support set for prediction. But the benefit of more shots is weaker for our model because it uses more prior knowledge.

Ablation Analysis on NER We investigate effectiveness of collapsed dependency transfer and label semantic on the NER task. We perform ablations on two proposed components and observe performance drops on both 1-shot and 5-shots settings, which demonstrate the generalization ability of proposed two mechanism.

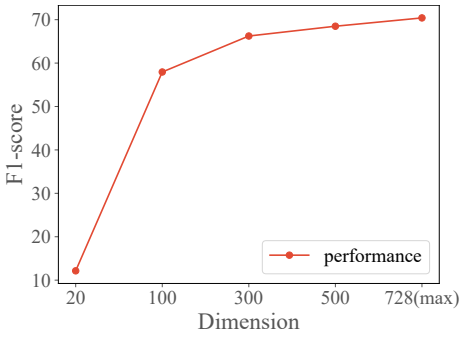

Figure 6: Impacts of projection space dimensionality.

C Analysis of Projection Space Dimensionality

Fig 6 shows the performance on 1-shot Snips when using different projected-space dimensions in L-TapNet. As shown in the trend in the figure, the performance of the model becomes better as the dimension of the mapping space increases and gradually stabilizes. This shows the possibility of reducing the dimension without losing too much performance (Yoon et al., 2019).

D Slot Tagging Result with Standard Deviations

Table 12 and 13 show the complete results with standard deviations for slot tagging task.

Model	1-shot Slot Tagging							
	We	Mu	Pl	Bo	Se	Re	Cr	Ave.
Bi-LSMT	10.36±0.36	17.13±0.61	17.52±0.76	53.84±0.57	18.44±0.44	22.56±0.10	8.64±0.41	21.21±0.46
SimBERT	36.10±0.00	37.08±0.00	35.11±0.00	68.09±0.00	41.61±0.00	42.82±0.00	23.91±0.00	40.67±0.00
TransferBERT	55.82±2.75	38.01±1.74	45.65±2.02	31.63±5.32	21.96±3.98	41.79±3.81	38.53±7.42	39.06±3.86
MN	21.74±4.60	10.68±1.07	39.71±1.81	58.15±0.68	24.21±1.20	32.88±0.64	**69.66**±1.68	36.72±1.67
WPZ	4.53±0.18	7.43±0.31	14.43±0.73	39.15±1.10	11.69±0.16	7.78±0.38	10.09±0.74	13.59±0.51
WPZ+GloVe	17.92±0.05	22.37±0.11	19.90±0.08	42.61±0.08	22.30±0.03	22.79±0.05	16.75±0.08	23.52±0.07
WPZ+BERT	46.72±1.03	40.07±0.48	50.78±2.09	68.73±1.87	60.81±1.70	55.58±3.56	67.67±1.16	55.77±1.70
TapNet	51.12±5.36	40.65±2.83	48.41±2.27	77.50±1.09	49.77±1.36	54.79±2.32	61.39±2.41	54.80±2.52
TapNet+CDT	66.30±3.81	55.93±1.78	57.55±6.57	83.32±0.96	64.45±4.07	65.65±1.74	67.91±3.32	65.87±3.18
L-WPZ+CDT	71.23±6.00	47.38±4.18	59.57±5.55	81.98±2.08	69.83±1.94	66.52±2.72	62.84±0.58	65.62±3.29
L-TapNet+CDT	**71.53**±4.04	**60.56**±0.77	**66.27**±2.71	**84.54**±1.08	**76.27**±1.72	**70.79**±1.60	62.89±1.88	**70.41**±1.97

Table 12: 1-shot slot tagging results with standard deviations.

Model	5-shots Slot Tagging							
	We	Mu	Pl	Bo	Se	Re	Cr	Ave.
Bi-LSMT	25.17±0.42	39.80±0.52	46.13±0.42	74.60±0.21	53.47±0.45	40.35±0.52	25.10±0.94	43.52±0.50
SimBERT	53.46±0.00	54.13±0.00	42.81±0.00	75.54±0.00	57.10±0.00	55.30±0.00	32.38±0.00	52.96±0.00
TransferBERT	59.41±0.30	42.00±2.83	46.07±4.32	20.74±3.36	28.20±0.29	67.75±1.28	58.61±3.67	46.11±2.29
MN	36.67±3.64	33.67±6.12	52.60±2.84	69.09±2.36	38.42±4.06	33.28±2.99	72.10±1.48	47.98±3.36
WPZ	9.54±0.19	14.23±0.19	18.12±1.41	44.65±2.58	18.98±0.58	12.03±0.58	14.05±0.63	18.80±0.88
WPZ+GloVe	26.61±0.54	34.25±0.16	22.11±0.04	50.55±0.15	28.53±0.05	34.16±0.43	23.69±0.07	31.41±0.21
WPZ+BERT	67.82±4.11	55.99±2.24	46.02±3.19	72.17±1.75	73.59±1.60	60.18±6.96	66.89±2.88	63.24±3.25
TapNet	53.03±7.20	49.80±3.02	54.90±2.72	83.36±1.03	63.07±1.96	59.84±1.57	67.02±2.51	61.57±2.86
TapNet+CDT	66.48±4.09	66.36±1.77	68.23±3.99	**85.76**±1.65	73.60±1.09	64.20±4.99	68.47±1.93	70.44±2.79
L-WPZ+CDT	**74.68**±2.43	56.73±3.23	52.20±3.22	78.79±2.11	80.61±2.27	69.59±2.78	67.46±1.91	68.58±2.56
L-TapNet+CDT	71.64±3.62	**67.16**±2.97	**75.88**±1.51	84.38±2.81	**82.58**±2.12	**70.05**±1.61	**73.41**±2.61	**75.01**±2.46

Table 13: 5-shot slot tagging results with standard deviations.