Multi-Domain Spoken Language Understanding Using Domain- and Task-Aware Parameterization

Libo Qin[†], **Minheng Ni**[†], **Yue Zhang**[‡§]*, **Wanxiang Che**[†], **Yangming Li**[†], **Ting Liu**[†]
[†]Research Center for Social Computing and Information Retrieval (SCIR),
Harbin Institute of Technology, Harbin, China
[‡]School of Engineering, Westlake University, China
[§]Institute of Advanced Technology, Westlake Institute for Advanced Study
[†]{lbqin,mhni,car,yangmingli,tliu}@ir.hit.edu.cn
[‡§]{yue.zhang}@wias.org.cn

Abstract

Spoken language understanding has been addressed as a supervised learning problem, where a set of training data is available for each domain. However, annotating data for each domain is both financially costly and non-scalable so we should fully utilize information across all domains. One existing approach solves the problem by conducting multi-domain learning, using shared parameters for joint training across domains. We propose to improve the parameterization of this method by using domain-specific and task-specific model parameters to improve knowledge learning and transfer. Experiments on 5 domains show that our model is more effective for multi-domain SLU and obtain the best results. In addition, we show its transferability by outperforming the prior best model by 12.4% when adapting to a new domain with little data.

1 Introduction

Spoken language understanding (SLU) (Young et al., 2013) plays an important role in task-oriented dialog systems. It consists of two typical subtasks, including intent detection and slot filling (Tur and De Mori, 2011). One example is shown in Figure 1, where the input is the utterance "*watch action movie* ?". The outputs consist of an overall intent class label (i.e., `WatchMovie`) and a slot label sequence (i.e., `O`, `B-movie-type`, `I-movie-type`, `O`). Since slots highly depend on the intent information, dominant SLU systems in the literature (Goo et al., 2018; Li et al., 2018; Xia et al., 2018; Qin et al., 2019; E et al., 2019; Liu et al., 2019) adopt joint models for the two tasks.

Though achieving promising performance, existing work mainly focuses on the single-domain

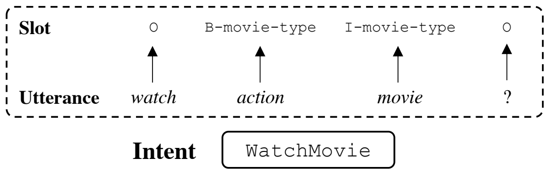
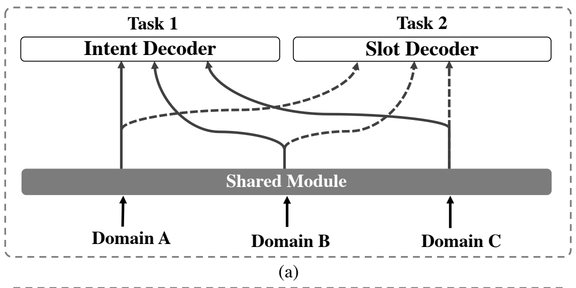


Figure 1: An example with intent and slot annotation (BIO format), which indicates the slot of movie name from an utterance with an intent `WatchMovie`.

scenario, relying on a set of domain-specific training data, which is both financially costly and non-scalable. Ideally, when an SLU system meets a new domain with little labeled data, the model should be able to transfer the existing domain knowledge to a new domain effectively. To this end, some existing work has endeavored towards using resources from all domains to train a model (Hakkani-Tür et al., 2016; Kim et al., 2017). In particular, as shown in Figure 2(a), Kim et al. (2017) build a joint model by combining labeled data from each domain for jointly training intent detection and slot filling. Unfortunately, the method is *domain-agnostic* and *task-agnostic* in the sense that the same set of model parameters are used for jointly representing cross-domain and cross-task information. While this can be useful for feature integration, it does not offer fine-grained channels for learning domain and task-specific knowledge, which can be useful for improving model performances. Take the sentence "*watch action movie*" in Figure 1 for example, the shared and domain-specific knowledge on the words "*watch*" and "*action movie*" is subtle, because "*action movie*" is domain-specific while "*watch*" can be shared with other domains. Similarly, different tasks require different features.

To address this issue, we propose a domain-aware and task-aware model for multi-domain joint intent detection and slot filling. The main idea is to make use of domain-aware and task-aware param-

*Corresponding Author. The work was done when the first author visits Westlake University.



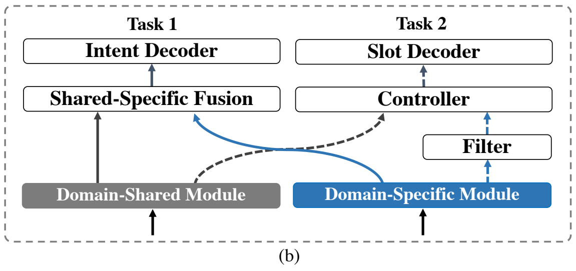


Figure 2: Methods for multi-domain spoken language understanding. Prior work trains a single model on mixed dataset (a). Our proposed domain-aware task-aware model (b). Dash line denotes information flow to slot filling and solid line denotes information flow to intent detection. Blue color represents Domain-Specific Module and gray color denotes Domain-Shared Module. Better viewed in color.

eterization, structuring the model so that domain-specific and task-specific components interact for maximizing the strength. As shown in Figure 2(b), the model consists of a domain-shared module for common knowledge across domains, and a domain-specific module for explicitly extracting specific feature for each domain. In addition, to make our model aware of the characteristics of tasks, we design separate model parameters to explicitly transfer features for intent detection and slot filling tasks across domains. For sentence-level intent detection transfer, the learned shared and domain-specific feature representations are directly combined. For token-level slot filling transfer, we propose a two-stage decoder which includes a filter decoder to mask out tokens which do not need domain-specific information, and a controller decoder to automatically calculate weights at the token-level for selecting domain-shared and domain-specific features, so that fine-grained combination of domain knowledge can be achieved. In addition, we further utilize graph convolutional networks (Kipf and Welling, 2016) to encode syntax information, which can help to capture domain-specific and domain-shared features.

We conduct experiments on two benchmarks MTOD (Schuster et al., 2019) and ASMixed (Goo et al., 2018; Coucke et al., 2018), which include five different domains totally. Experiments show that our method achieves state-of-the-art results of 91.27% sentence accuracy on the MTOD dataset, outperforming prior best by 1.91%. On the ASMixed dataset, we achieve 84.81% sentence accuracy, outperforming prior best by 5.63%. In particular, given a new domain with little labeled data, our framework can effectively transfer knowledge by our proposed domain-aware task-aware module and can outperform the existing state-of-the-art model by 12.4% on average. All datasets and code are publicly available at: `https://XXX`.

2 Task Definition

The input to our task is a sentence $X = (x_1, \dots, x_n)$ and two tasks are performed jointly. Slot filling can be treated as a sequence labeling task that predicts the corresponding slots sequence $\mathbf{o}^S = (o_1^S, \dots, o_n^S)$. Intent detection can be seen as a classification problem to decide the intent label o^I.

Multi-Domain Learning Suppose that there is a domain set $D = \{d_1, d_2, ..., d_{|D|}\}$ and a dataset with m data $T = \{t_1, t_2, ..., t_m\}$. For each t in T, $t = (X, \mathbf{o}^S, o^I, d)$, where X represents utterance, $\mathbf{o}^S$ represent target slots, o^I represents target intent and d represents domain of this data, respectively. The task attempts to learn a joint model function F trained on multiple source domain which will be used for each domain. Similar to OneNet (Kim et al., 2017), we assume that the input is still one utterance, without the need to know which domain it comes from. However, since our model requires explicit knowledge of the input domain for calculating a corresponding representation, we first perform a domain classification task. It has been shown that the accuracy of a standard sequence classifier can be rather reliable, reaching the level of 99.7%. More details are given in Section 4.3.

3 Approach

The overall structure of our model is shown in Figure 3. First, a *domain-aware syntactic encoder* is proposed for incorporating domain-shared and domain-specific features, which are combined for intent detection. Second, a two-stage decoder includes *Slot Filter* and *Slot Controller* module, which is used to make fine-grained knowledge transfer for the token-level slot filling. Both intent detection and slot filling parameters are optimized simultaneously via a joint learning scheme.

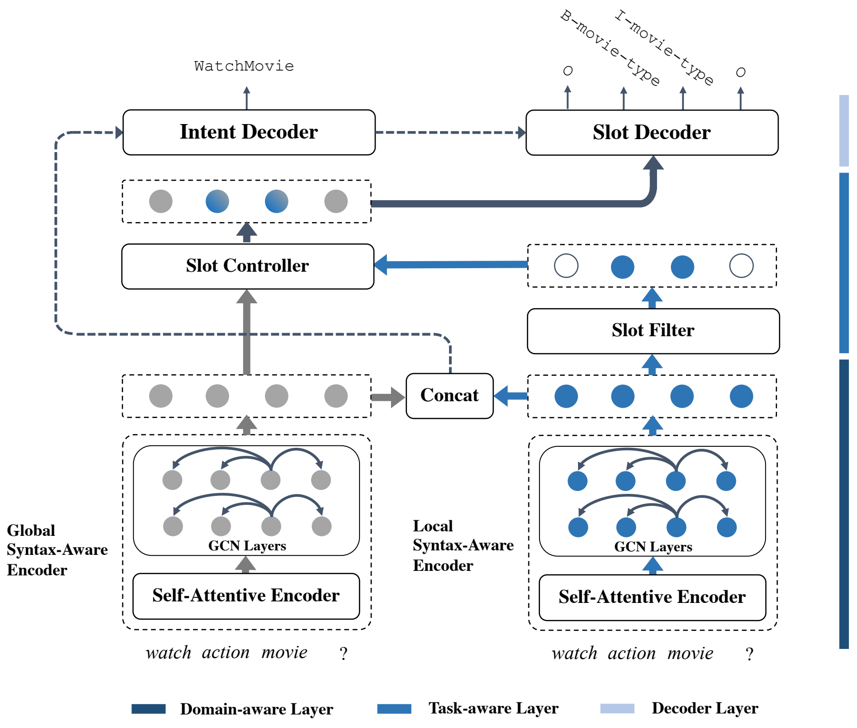


Figure 3: Overview of our proposed framework. It consists of a global-locally syntax-aware encoder, a slot filter, slot controller and two decoders. The gray color represents general features across all domains and the blue color denotes domain-specific features. Better viewed in color.

3.1 Domain-Aware Syntactic Encoder

In our framework, we propose a global-local syntax-aware to make our framework explicitly aware of domain. More specifically, a global syntax-aware encoder is proposed to capture the domain-shared features where instances from all domains will pass the global encoder. Meanwhile, we utilize D local syntax-aware encoders to capture features of differnent domains. The architecture of all local syntax-aware encoders is same with the global syntax-aware encoder. However, for each domain d, parameters of its local syntax-aware encoder are differnet from others. Each instance from domain d only pass into its corresponding local encoder to capture the specific d domain features.

In the following section, we describe the syntax-aware encoder which consists of an self-attention encoder to capture the contextual information and an graph network to encoder syntax information.

Self-Attentive Sentence Representation Following Qin et al. (2019), we first utilize a basic self-attentive encoder to obtain self-attentive representation. Given an n-word sentence $X = (x_1, x_2, .., x_n)$, a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) is used to produce hidden state vectors $\boldsymbol{H} = (\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_n)$. Besides, we also apply self-attention over word embedding to capture context-aware features. We adopt Transform encoder (Vaswani et al., 2017), which maps the matrix of input vectors $\boldsymbol{X} = \{\phi^{emb}(x_1), \ldots, \phi^{emb}(x_n)\} \in \mathbb{R}^{n\times d}$ (ϕ^{emb} represents embedding mapping matrix) to queries ($\boldsymbol{Q}$), keys ($\boldsymbol{K}$) and values ($\boldsymbol{V}$) matrices by using different linear projections and output $\boldsymbol{C} \in \mathbb{R}^{T\times d}$ is a weighted sum of values: $\boldsymbol{C} = \text{softmax}\left(\frac{\boldsymbol{QK}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}$, where d_k denotes the dimension of keys. We concatenate these two representations as the self-attentive encoding representation: $\boldsymbol{E} = \boldsymbol{H} \oplus \boldsymbol{C}$, where $\boldsymbol{E} = (\boldsymbol{e}_1, \boldsymbol{e}_2, .., \boldsymbol{e}_n) \in \mathbb{R}^{n\times 2d}$ and $\oplus$ is concatenation operation.

Graph Convolution over Dependency Trees Syntax information is an important source of features across domains. In our framework, we propose to build a GCN (Kipf and Welling, 2016) over the dependency tree of a sentence to exploit syntactical information. For a given graph with k nodes, an adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{k\times k}$ is used to represent the graph, where $A_{ij} = 1$ if there is an edge going from node i to node j. We denote the output of the l-th layer for node i as $\boldsymbol{g}_i^{(l)}$, where $\boldsymbol{g}_i^{(0)}$ represents the initial state of node i.

In our paper, we set $\boldsymbol{G}^{(0)} = \boldsymbol{E}$ to make nodes aware of context (Zhang et al., 2018, 2019a). For an L-layer GCN, $\boldsymbol{g}_i^{(L)}$ is the final state of node i. Given the dependency tree of the input sentence,[1] the graph convolution operated on the node representation can be written as:

$$\boldsymbol{g}_i^{(l)} = \sigma\big(\sum_{j=1}^{n} \tilde{A}_{ij} \boldsymbol{W}^{(l)} \boldsymbol{g}_j^{(l-1)} + \boldsymbol{b}^{(l)}\big), \tag{1}$$

where $l \in [1, 2, \cdots, L]$, $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ and $\boldsymbol{I}$ is a $n \times n$ identity matrix to consider information itself, $\boldsymbol{W}^{(l)}$ is a linear transformation, $\boldsymbol{b}^{(l)}$ is a bias term and σ is a nonlinear function.

To sufficiently capture syntactical information in a sentence, we stack this operation over L layers, where we use $\boldsymbol{G} = (\boldsymbol{g}_1^{(L)}, \ldots, \boldsymbol{g}_n^{(L)})$ as output word representations.

Hence, we can utilize a global syntax-aware encoder to obtain a global encoding $\boldsymbol{G}^{\mathrm{g}} = (\boldsymbol{g}_1^{\mathrm{g}}, \ldots, \boldsymbol{g}_n^{\mathrm{g}})$, which captures domain-shared features. Similarly, for each domain d, we produce a local encoding $\boldsymbol{G}^{\mathrm{l}}$, using a local d syntax-aware encoder to capture local encoding representation $\boldsymbol{G}^{\mathrm{l}} = (\boldsymbol{g}_1^{\mathrm{l}}, \ldots, \boldsymbol{g}_n^{\mathrm{l}})$.

3.2 Task-Specific Intent Decoder

Domain Shared-Private Feature Fusion After obtaining the domain-shared and domain-specific encoding representation $\boldsymbol{G}^{\mathrm{g}}$, $\boldsymbol{G}^{\mathrm{l}}$, we use a self-attention module (Zhong et al., 2018; Goo et al., 2018) to compute relevant attention context for intent detection. The global attention context can be calculated as:

$$\boldsymbol{a}^{\mathrm{g}} = \boldsymbol{W}^{\mathrm{g}} \boldsymbol{G}^{\mathrm{g}} + \boldsymbol{b}^{\mathrm{g}}, \tag{2}$$

$$\boldsymbol{p}^{\mathrm{g}} = \mathrm{softmax}\left(\boldsymbol{a}^{\mathrm{g}}\right). \tag{3}$$

$\boldsymbol{c}^{\mathrm{g}}$ is then the sum of each element $\boldsymbol{g}_i^{\mathrm{g}}$, weighted by the corresponding normalized global self-attention score p_i^{g}:

$$\boldsymbol{c}^{\mathrm{g}} = \sum_i p_i^{\mathrm{g}} \boldsymbol{g}_i^{\mathrm{g}}. \tag{4}$$

We similarly compute the local self-attention context $\boldsymbol{c}^{\mathrm{l}}$. After obtaining the global encoding representation $\boldsymbol{c}^{\mathrm{g}}$ and local encoding representation $\boldsymbol{c}^{\mathrm{l}}$, we combine them through a simple concatenation operation which can be written as:

$$\boldsymbol{c}^{m} = [\boldsymbol{c}^{\mathrm{g}}, \boldsymbol{c}^{\mathrm{l}}], \tag{5}$$

where $\boldsymbol{c}^{m}$ is the mixed syntax-aware encoding feature, which can be seen as combining the domain-shared and domain-specific features.

Intent Prediction The combined context encoding $\boldsymbol{c}^{m}$ is used for intent detection:

$$\boldsymbol{y}^{I} = \mathrm{softmax}\left(\boldsymbol{W}_h^{I} \boldsymbol{c}^{m}\right), \tag{6}$$

$$o^{I} = \mathrm{argmax}(\boldsymbol{y}^{I}), \tag{7}$$

where $\boldsymbol{y}^{I}$ is the intent output distribution of the utterance; o^{I} represents the intent label and $\boldsymbol{W}_h^{I}$ are trainable parameters of the model.

3.3 Task-Specific Slot Filling Decoder

We propose a two-stage decoder to consider task characteristic for slot filling. The first stage uses a filter to mask out those domain-general tokens which does not need domain-specific features. Such tokens include those annotated with a domain-general slot-label.[2] The second stage makes use of a controller module to achieve fine-grained knowledge transfer by automatically obtaining the weight to select how to combine the domain-shared and domain-specific features at the token-level.

Slot Filter We adopt a simple feedforward network as our filter, which aims to mask the domain-general tokens. In this way, we can allow our model to focus on knowledge transfer for more inferable tokens. Given the global-local encoder output: $\boldsymbol{G}^{\mathrm{g}}$ and $\boldsymbol{G}^{\mathrm{l}}$, we concat them and pass them to the Filter module, which can be written as:

$$\boldsymbol{F} = \mathrm{Sigmoid}\left(\boldsymbol{W}_f [\boldsymbol{G}^{\mathrm{g}}, \boldsymbol{G}^{\mathrm{l}}] + \boldsymbol{b}_f\right), \tag{8}$$

where $\boldsymbol{W}_f$ are trainable parameters of the model, and we define the label $\boldsymbol{F} = (f_1, ..., f_n)$ as the probability of domain general tokens and $(1 - \boldsymbol{F}) = (1 - f_1), ..., (1 - f_n))$ as the probability of domain specific tokens.

After obtaining the probability of domain specific tokens and domain-specific features $\boldsymbol{G}^{\mathrm{l}} = (\boldsymbol{g}_1^{\mathrm{l}}, \ldots, \boldsymbol{g}_n^{\mathrm{l}})$, we can utilize the output of Filter module to mask out domain-general tokens, each filtered i^{th} domain-specific information representation can be written as:

$$\boldsymbol{u}_i^{\mathrm{l}} = (1 - f_i) * \boldsymbol{g}_i^{\mathrm{l}}. \tag{9}$$

where $\boldsymbol{U} = \{u_1^{l}, \ldots, u_n^{l}\}$ represents the remained domain-specific presentation.

[1] We use Stanford CoreNLP (Manning et al., 2014) to generate the dependency tree.

[2] We define slot-label is domain-general when a slot-label exists in all domains (e.g. O).

Fine-grained Fusion for Slot Filling Here, we describe the process of fine-grained fusion for slot filling. We first propose a controller, which generates weights deciding how to combine domain-shared and domain-specific features at the token-level. Then, we make a fine-grained fusion for slot filling at the token-level.

Slot Controller Given the utterance X, after obtaining the global-local encoder output: $\boldsymbol{G}^{\mathrm{g}}$ and $\boldsymbol{G}^{\mathrm{l}}$, we concat them and use a simple feedforward network to calculate weights at each token which can be written as follows:

$$\boldsymbol{P} = \mathrm{Sigmoid}\left(\boldsymbol{W}_c[\boldsymbol{G}^{\mathrm{g}}, \boldsymbol{G}^{\mathrm{l}}] + \boldsymbol{b}_c\right). \quad (10)$$

Finally, we adopt $\boldsymbol{P} = (p_1, ..., p_n)$ as the token-level weights to combine domain-shared and domain-specific features.

Fine-grained Fusion For incorporating domain-shared features at the token-level, we use the weight produced by the controller module to fuse domain-shared and domain-specific features.

$$\boldsymbol{u}_i^f = p_i * \boldsymbol{u}_i^{\mathrm{l}} + (1 - p_i) * \boldsymbol{g}_i^{\mathrm{g}}, \quad (11)$$

where $\boldsymbol{u}_i^f$ is the fused representation at i^{th} token.

Slot Prediction We use a unidirectional LSTM as the slot-filling decoder. Following Li et al. (2018) and Qin et al. (2019), we adopt intent information to guide the slot prediction. At i^{th} decoding step, the decoder state $\boldsymbol{h}_i^S$ can be formalized as:

$$\boldsymbol{h}_i^S = \mathrm{LSTM}\left(\boldsymbol{h}_{i-1}^S, \boldsymbol{y}_{i-1}^S, \boldsymbol{y}^I \oplus \boldsymbol{u}_i^f\right), \quad (12)$$

where $\boldsymbol{h}_{i-1}^S$ is the previous decoder state; $\boldsymbol{y}_{i-1}^S$ is the previous emitted slot label distribution and $\boldsymbol{y}^I$ is embedding of intent in the utterance.

Finally, $\boldsymbol{h}_i^S$ is utilized for slot prediction:

$$\boldsymbol{y}_i^S = \mathrm{softmax}\left(\boldsymbol{W}_h^S \boldsymbol{h}_i^S\right), \quad (13)$$

$$o_i^S = \mathrm{argmax}(\boldsymbol{y}_i^S), \quad (14)$$

where o_i^S is the slot label of the i^{th} word in the utterance.

3.4 Joint Training

We adopt a joint model to consider the two tasks and update parameters in a joint optimization. The intent detection objection can be formulated as:

$$\mathcal{L}_1 \triangleq -\sum_{j=1}^{m} \hat{\boldsymbol{y}}^{j,I} \log\left(\boldsymbol{y}^{j,I}\right). \quad (15)$$

Dataset	Domains	Train	Dev	Test
MTOD	Reminder, Weather, Alarm	30,521	4,181	8,621
ASMixed	ATIS, SNIPS	17,562	1,200	1,593

Table 1: Statistics of datasets.

Similarly, the slot filling task objection is defined as:

$$\mathcal{L}_2 \triangleq -\sum_{j=1}^{m}\sum_{i=1}^{n_j} \hat{\boldsymbol{y}}_i^{j,S} \log\left(\boldsymbol{y}_i^{j,S}\right), \quad (16)$$

where $\hat{\boldsymbol{y}}_j^I$ and $\hat{\boldsymbol{y}}_i^{j,S}$ are the gold intent label and gold slot label, respectively; m is the number of training data and n_j is the number of tokens in j^{th} data.

In addition, to further strengthen the Filter ability, we add an auxiliary loss to train Filter as a classification task, the loss function can be denotes as:

$$\mathcal{L}_3 \triangleq -\sum_{j=1}^{m}\sum_{i=1}^{n_j} \hat{y}_i^{j,F} \log\left(f_i^j\right) + (1 - \hat{y}_i^{j,F}) \log\left(1 - f_i^j\right), \quad (17)$$

where $\hat{y}_i^{j,F}$ is the gold representation of Filter. The final joint objective is formulated as:

$$\mathcal{L}_\theta = \alpha_1\mathcal{L}_1 + \alpha_2\mathcal{L}_2 + \alpha_3\mathcal{L}_3, \quad (18)$$

where α_1, α_2 and α_3 are hyper-parameters.

4 Experiments

4.1 Datasets

We conduct experiments on the benchmark MTOD (Schuster et al., 2019). The dataset contains three domains including Alarm, Reminder, Weather domain. We follow the same format and partition as in Schuster et al. (2019). To justify the generalization of the proposed model, we construct another multi-domain SLU dataset (ASMixed) by mixing the ATIS (Hemphill et al., 1990) and SNIPS (Coucke et al., 2018) dataset and keep the train/dev/test partition unchanged. In addition, the constructed multi-domain datasets will be available for further research. The detailed statistics of two datasets are presented in Table 1.

4.2 Experimental Settings

The dimensionalities of the embedding we adopt is 64 and LSTM hidden units is 256. The dropout ratio we use in our framework is 0.4 and the batch size is 16. The layer of GCN is 3 for MTOD dataset

	MTOD						ASMixed				
Model	Overall Exact	Slot	Intent	Reminder Exact	Alarm Exact	Weather Exact	Overall Exact	Slot	Intent	ATIS Exact	SNIPS Exact
Shared-LSTM (Hakkani-Tür et al., 2016)	88.71	94.87	98.70	82.06	90.19	90.99	76.71	92.55	94.41	81.69	70.41
Separated-LSTM (Hakkani-Tür et al., 2016)	89.73	94.89	99.01	84.59	89.81	92.18	79.53	92.94	94.79	80.96	77.71
Multi-Domain adv (Liu and Lane, 2017)	88.82	94.41	98.87	82.09	88.86	92.05	79.47	91.80	96.48	82.75	75.29
One-Net (Kim et al., 2017)	89.36	95.25	98.56	83.27	90.15	91.83	78.28	93.38	93.72	81.85	73.80
Locale-agnostic-Universal (Lee et al., 2019)	88.54	94.16	99.12	81.63	89.58	91.21	79.35	92.10	96.48	82.19	75.71
Ours framework	**91.27***	**95.69***	**99.20***	**85.62***	**92.37***	**93.29***	**84.81***	**94.30***	**97.30***	**86.53***	**82.62***
Ours framework (oracle)	95.45	97.80	99.18	95.56	92.81	97.38	87.26	94.93	96.92	86.76	87.89

Table 2: Main Results (Overall Exact, Slot and Intent denote the corresponding metrics on whole datasets and domain exact represents the exact accuracy on each domain separately). The numbers with * indicate that the improvement of our framework over all baselines is statistically significant with $p < 0.01$ under t-test.

and 2 for ASMixed dataset. In the framework, we use Adam (Kingma and Ba, 2014) to optimize the parameters in our model and adopt the suggested hyper-parameters. For all the experiments, we select the model which works the best on dev set, and then evaluate it on test set.

4.3 Baselines

We compare our model with several existing state-of-the-art multi-domain SLU baselines including: 1) **Shared-LSTM** Hakkani-Tür et al. (2016) trained a single model for intent detection and slot filling using data from all the domains. 2) **Separated-LSTM**. Hakkani-Tür et al. (2016) proposed a multi-task modeling approach for joint modeling of slot-filling and intent determination for each domain separately. 3) **Multi-Domain Adv**. Liu and Lane (2017) applied an adversarial training method for learning common features and representations that can be shared across multiple domains. For fair comparison, we add an intent detection module and train the two tasks jointly. 4) **One-Net**. Kim et al. (2017) adopted a principled architecture for multi-task learning to fold in the state-of-the-art models for intent detection and slot filling. 5) **Locale-agnostic-Universal**. Lee et al. (2019) proposed a locale-agnostic universal domain classification model based on multi-task learning that learns a joint representation of an utterance over locales with different sets of domains and allows locales to share knowledge selectively depending on the domains. To verify the ability of our proposed framework, we implement a joint model for slot filling and intent detection with the method.

Domain Classification: We adopt a syntax-aware encoder which is the same as the global syntax-aware encoder shown in Section 3 to directly predict domain of given utterance, and the result is 99.9% in MTOD dataset and 99.7% in ASMixed dataset. After obtaining the predicted domain, we put the utterance to the corresponding local encoder during the test period.

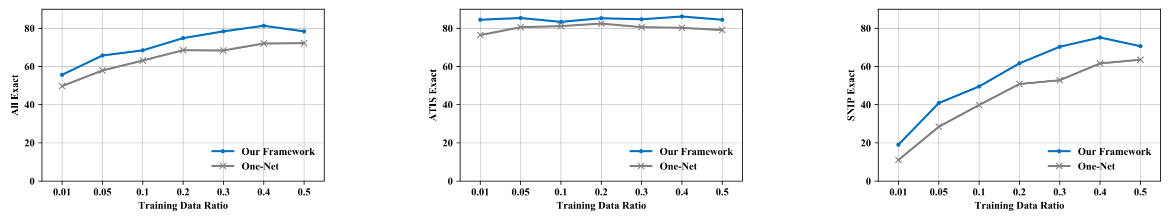
4.4 Overall Results

Following prior works (Goo et al., 2018; Qin et al., 2019), we evaluate the SLU performance of slot filling using F1 score, the performance of intent prediction using accuracy and sentence-level semantic frame parsing using exact accuracy. Table 2 shows the experiment results of the proposed models on two datasets. We can observe that: 1) *Locale-agnostic-Universal* has achieved the best performances on intent detection among all baselines, which indicates that explicitly model domain-shared and domain-specific is more effective than implicitly capturing shared knowledge with sharing parameters. 2) Our model significantly outperforms all the baselines by a large margin and achieves the state-of-the-art performance. In the MTOD dataset, compared with the best prior joint work *One-Net*, we achieve 2.35% improvement on Reminder domain, 2.22% improvement on Alarm domain and 1.46% improvement on Weather domain. In the ASMixed dataset, the same trend has been witnessed. This indicates the effectiveness of our domain-aware and task-aware framework which can effectively improve performance for each domain. 3) To see the role of our two-stage decoder intuitively, we also present the result when using the oracle Filter information.[3] The result is shown in the oracle row of Table 2. We can see that given the oracle Filter label where we correctly mask the token that does not need domain-specific feature will lead to better slot filling performance, which further verifies the effectiveness of our two-stage decoder.

4.5 Analysis

We conduct experiment analysis on ASMixed to answer the following questions: 1) Does the

[3]We provide the gold Filter one-hot vector during the training and test process.



(a) Overall Exact (Acc). (b) ATIS Exact (Acc). (c) SNIPS Exact (Acc)

Figure 4: Performance (Exact Acc) of domain adaption on different subsets of original training data on ASMixed dataset.

Model	Overall Exact	Slot	Intent	ATIS Exact	SNIPS Exact
Full Model	**84.81**	**94.30**	**97.30**	**86.53**	**82.62**
w/o Local Syntax-Aware Encoder	82.93	93.52	97.24	84.96	80.34
w/o Filter & Controller	82.23	93.28	97.05	83.61	80.48
w/o GCN	83.82	93.82	96.99	85.52	81.62

Table 3: Ablation Experiments on ASMixed datasets.

domain-aware global-local module benefits for multi-domain SLU? 2) Does the task-aware two-stage decoder successfully transfer fine-grained knowledge feature for token-level slot filling across all domains? 3) Can syntactic information better generalize across domains in SLU tasks? 4) Can our framework effectively transfer knowledge for a new domain with little labeled data? 5) Does our framework successfully captures the domain-shared and domain-specific features?

4.5.1 Effectiveness of Domain-Aware Parameterization

In order to verify our claim that domain-aware parameterization can effectively capture domain-specific features. We remove the local syntax-aware encoder and conduct the experiment. Result is shown in *w/o Local Syntax-Aware Encoder* row in Table 3. We observe that the ATIS Exact acc drops 1.57% and SNIPS Exact acc drops 2.28%. In addition, the whole exact acc drops 1.98%, which shows that domain-specific feature extracted by local module is important for multi-domain SLU tasks, and domain-aware parameterization is better than only using the shared parameters for capturing domain-specific features.

4.5.2 Effectiveness of Task-Aware Parameterization

To verify the effectiveness of the task-aware parameterization, we conduct ablation experiment where we remove the two-stage decoder for slot filling. In this setting, we incorporate shared and domain-specific feature by summation other than with our two-stage slot filling decoder. This model is an effective domain-aware version of OneNet (Kim et al., 2017), with multi-tasking between intent classification and slot filling only through parameter sharing. The results are shown in *w/o Filter & Controller* row in Table 3. We can see a 1.58% and 1.02% drop in exact and slot filling metric, respectively, which verifies the effectiveness of our proposed two-stage decoder. We attribute this to the fact that our filter successfully filters the domain-general token and the model automatically learn weights on how to combine shared and domain-specific feature for each token slot prediction.

4.5.3 Effectiveness of Syntactic Information

We remove the GCN layers in our framework and only adopt the self-attentive encoder to verify the effectiveness of syntax information. The result is shown in *w/o GCN* row in Table 3. We can see that the performance drops significantly in all metrics, which demonstrates the effectiveness of syntax information in multi-domain SLU tasks. We think utterances in different domains have different syntactic patterns, which helps the model better capture the shared and domain-specific features.

4.5.4 Domain Adaption

We conduct domain adaption experiments on ATSmixed dataset by simulating given a new domain with little labeled data to explore the transferability of our framework. We keep ATIS dataset unchanged, and the ratio of the except domain SNIPS from original data varies from [1%, 5%, 10%, 20%, 30%, 40%, 50%]. The results are shown in Figure 4. We can find that our framework outperforms *One-Net* on all ratios of the original dataset. In particular, our framework trained with 20% training dataset can achieve comparable and even better performance compared to *One-Net* with 50% training dataset on some domains. Especially, with 5% training data, our model outperforms *One-Net*

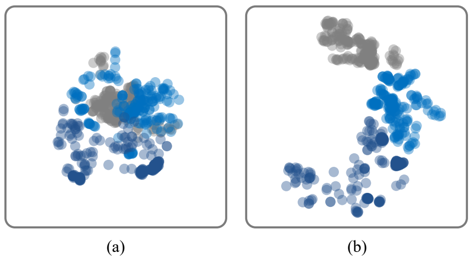


Figure 5: t-SNE visualization of sentences vector space from the global encoder (a) and with each domain specific encoder (b). Each color represents 200 sentence representations from different domains.

	from	*Indianapolis*	*to*	*Orlando*	*around*	*December*	*twenty*	*fifth*
One-Net	O	From City	O	To City	O	Time Month	Time Day	
Our Model	O	From City	O	To City	**Time Relative**	Time Month	Time Day	

Figure 6: A case of slot filling task between our model and *One-Net*. The blue slot is correct while the red one is wrong. Better viewed in color.

by 12.4% on SNIPS exact. This implies that our framework effectively transfers knowledge from other domains to achieve better performance for the low-resources new domain.

4.5.5 Visualization

In order to see whether our framework successfully captures the domain-shared and domain-specific features. In practice, we put 600 sentences which includes 200 sentences from Alarm domain, 200 sentences from Reminder domain and other sentences from Weather domain into the global syntax-aware encoder and get those sentence representations. Sentences from each domain are fed into its local syntax-aware encoder to get their sentence representations. Then, we use t-SNE to visualize those sentence representations obtained by the global and local encoder, as shown in Figure 5. We can observe that those representations from the global encoder become closer and overlap with each other, which verifies that our framework successfully captures domain-shared features. Meanwhile, each domain sentence representations appeal in a cluster and there is nearly no overlap between different domains, which demonstrates our local syntax-aware encoder capture the domain-specific features.

4.5.6 Case Study

To better understand how our proposed task-aware two-stage decoder affects and contributes to the final result, we conduct a case study of slot filling task between our model and the baseline model *One-Net*. This case is shown in Figure 6. For the word "*around*", *One-Net* predicts its slot label as "O" wrongly. Token"*around*" is more likely treated as a domain-general token because it usually does not have real meaning in SLU system. The result indicates that *One-Net* can not capture enough domain information to predict it correctly. In contrast, our model predicts the slot label correctly. We attribute this to the fact that the proposed two-stage decoder successfully learns to capture more domain-specific knowledge for this token.

5 Related Work

Joint Model for SLU Recently, dominate models (Zhang and Wang, 2016; Goo et al., 2018; Li et al., 2018; Wang et al., 2018; Qin et al., 2019; Zhang et al., 2019b) adopted a joint model for training slot filling and intent detection simultaneously. Zhang and Wang (2016) proposed the joint work using LSTM for learning the correlation between intent and slots. Goo et al. (2018) proposed a slot-gated to model the relationship and interaction between two tasks. Li et al. (2018); Qin et al. (2019) proposed to use intent information to explicitly model the semantic correlation between slots and intent. Our work in line is solving the same joint task rather than one of the sub tasks. Besides, we first propose to use intent information to explicitly guide slot filling in a multi-domain learning scenario.

Multi-Domain SLU Hakkani-Tür et al. (2016) proposed a single LSTM model over mixed multi-domain dataset implicitly learning the domain-shared features. Kim et al. (2017) adopted one network to jointly modeling slot filling, intent detection and domain classification. Compared with their work, we propose a domain-aware model which uses different parameters to extract shared feature while adopt a local module to capture domains-specific feature. Liu and Lane (2017) also used a shared LSTM to capture domain-shared knowledge and a private LSTM to extract domain-specific feature, but simply combine them with equal weight for multi-domain slot filling. In contrast, we use a two-stage decoder to automatically calculate weights on how to combine shared and domain-specific knowledge. In addition, our framework explicitly incorporated intent information for guiding slot filling in a joint model while they only perform slot filling task. Lee et al. (2019) proposed a shared-private framework to capture shared and domain-specific feature for domain classification task. In our framework, we propose a task-aware model, not only using a shared-private framework for sentence-level intent detection, but also adopting a two-stage decoder for token-level slot filling. To our best of knowledge, this is the first work to consider fine-grained domain knowledge transfer in the multi-domain SLU task in a joint model.

6 Conclusion

We investigated domain-aware and task-aware parameterization for multi-domain SLU by building a model with separate domain- and task-specific parameters. Unlike existing methods, which use the same parameters for multi-task learning, our model can better capture fine-grained knowledge. Experiments on two publicly available datasets including five domains show the effectiveness of the proposed models and achieve the state-of-the-art performance. In addition, the model can quickly adapt to a new domain given little labeled data, which makes it more robust and scalable.

References

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

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proc. of ACL*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proc. of NAACL*.

Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Interspeech*.

Charles T Hemphill, John J Godfrey, and George R Doddington. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8).

Young-Bum Kim, Sungjin Lee, and Karl Stratos. 2017. Onenet: Joint domain, intent, slot prediction for spoken language understanding. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 547–553. IEEE.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Jihwan Lee, Ruhi Sarikaya, and Young-Bum Kim. 2019. Locale-agnostic universal domain classification model in spoken language understanding. In *Proc. of NAACL*.

Changliang Li, Liang Li, and Ji Qi. 2018. A self-attentive model with gate mechanism for spoken language understanding. In *Proc. of EMNLP*.

Bing Liu and Ian Lane. 2017. Multi-domain adversarial learning for slot filling in spoken language understanding. *arXiv preprint arXiv:1711.11310*.

Yijin Liu, Fandong Meng, Jinchao Zhang, Jie Zhou, Yufeng Chen, and Jinan Xu. 2019. Cm-net: A novel collaborative memory network for spoken language understanding. *arXiv preprint arXiv:1909.06937*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proc. of ACL*.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proc. of EMNLP*.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proc. of NAACL*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł *ukasz* Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A bi-model based rnn semantic frame parsing model for intent detection and slot filling. In *Proc. of ACL*.

Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip Yu. 2018. Zero-shot user intent detection via capsule neural networks. In *Proc. of EMNLP*.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019a. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proc. of EMNLP*.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. 2019b. Joint slot filling and intent detection via capsule neural networks. In *Proc. of ACL*.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *Proc. of IJCAI*.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proc. of EMNLP*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proc. of ACL*.