Federated Learning for Spoken Language Understanding

Zhiqi Huang[1], Fenglin Liu[1], Yuexian Zou[1,2]*
[1]ADSPLAB, School of ECE, Peking University, Shenzhen, China
[2]Peng Cheng Laboratory, Shenzhen, China
{zhiqihuang, fenglinliu98, zouyx}@pku.edu.cn

Abstract

Recently, spoken language understanding (SLU) has attracted extensive research interests, and various SLU datasets have been proposed to promote the development. However, most of the existing methods focus on a single individual dataset, the efforts to improve the robustness of models and obtain better performance by combining the merits of various datasets are not well studied. In this paper, we argue that if these SLU datasets are considered together, different knowledge from different datasets could be learned jointly, and there are high chances to promote the performance of each dataset. At the same time, we further attempt to prevent data leakage when unifying multiple datasets which, arguably, is more useful in an industry setting. To this end, we propose a federated learning framework, which could unify various types of datasets as well as tasks to learn and fuse various types of knowledge, i.e., text representations, from different datasets and tasks, without the sharing of downstream task data. The fused text representations merge useful features from different SLU datasets and tasks and are thus much more powerful than the original text representations alone in individual tasks. At last, in order to provide multi-granularity text representations for our framework, we propose a novel Multi-view Encoder (MV-Encoder) as the backbone of our federated learning framework. Experiments on two SLU benchmark datasets, including two tasks (intention detection and slot filling) and federated learning settings (horizontal federated learning, vertical federated learning and federated transfer learning), demonstrate the effectiveness and universality of our approach. Specifically, we are able to get 1.53% improvement on the intent detection metric accuracy. And we could also boost the performance of a strong baseline by up to 5.29% on the slot filling metric F1. Furthermore, by leveraging BERT as an additional encoder, we establish new state-of-the-art results on SNIPS and ATIS datasets, where we get 99.33% and 98.28% in terms of accuracy on intent detection task as well as 97.20% and 96.41% in terms of F1 score on slot filling task, respectively.

1 Introduction

In recent years, Spoken Language Understanding (SLU) technology, which typically involves intent detection and slot filling, plays a crucial part in goal-oriented dialogue systems. The intent detection is treated as an utterance classification problem, which can be modeled using various classifiers based on deep neural networks (Liu and Lane, 2016; Zhang et al., 2016a; Zhang et al., 2017; Xia et al., 2018). The slot filling task can be formulated as a sequence labeling problem, and one of the popular approaches with good performance is using CRF and CNN (Xu and Sarikaya, 2013). The slots represent the word-level information while the intent stands for the sentence-level information (E et al., 2019). These two tasks are normally considered as parallel tasks but may have cross-impact on each other (Xu and Sarikaya, 2013). Table 1 gives examples of the two tasks in two utterances, whose intents are *GetWeather* and *SearchCreativeWork* and slots are recorded as BIO format.

Despite the impressive results, most of the existing frameworks focus on a single individual dataset and are directly trained on the specific individual target dataset. The data-specific model may well suffer

*Corresponding Author.

Proceedings of the 28th International Conference on Computational Linguistics, pages 3467–3478
Barcelona, Spain (Online), December 8-13, 2020

Utter.1	it	is	colder	faraway	from	my	current	spot
Slots	O	O	$B-condition\_temperature$	$B-spatial\_relation$	O	O	$B-current\_location$	$I-current\_location$
Intent	$GetWeather$							
Utter.2	find	a	tv	show	called	the	traffic	policeman
Slots	O	O	$B-object\_type$	$I-object\_type$	O	$B-object\_name$	$I-object\_name$	$I-object\_name$
Intent	$SearchCreativeWork$							

Table 1: Utterances with intents and slots annotation (BIO format) sampled from the SNIPS dataset.

from overfitting when the data in the downstream dataset is scarce. In addition, because the design and training of the model are data-specific, it is difficult for the model to transfer the knowledge learned from the target dataset to the other dataset directly, where the two datasets may be completely different. It means that a well-trained model on the target dataset may have poor performance on the other dataset, which indicates the current models suffer from poor robustness. In short, we argue that for the SLU task, studies used to model on a single individual dataset are not friendly to the model robustness. Intuitively, learning from multiple datasets at the same time will improve the performance and robustness of the model, compared to the single dataset training. Thus, it is reasonable to combine multiple datasets for modeling in SLU. A multi-task learning framework, which is used for combining multiple tasks in the previous works (Qin et al., 2019; E et al., 2019), may have the potential to unify multiple datasets, though, it may have data leakage problems, which is unacceptable for collaborative task training between multiple institutions. What's more, the multi-task framework is a special case of one of our federated learning framework, i.e., vertical federated learning, which will be discussed in the following paragraph.

Recently, Konecný et al. (2016a), Konecný et al. (2016b) and McMahan et al. (2017) propose an emerging artificial intelligence technology, i.e., federated learning, whose goal is to train a high quality *centralized model* based on datasets that are distributed across *multiple clients* without sharing the clients' data. Inspired by the success of federated learning, we introduce a federated learning framework to perform the SLU task. In implementations, we treat each sub-task as a *client* and propose an MV-Encoder as the *centralized model*. The motivation for proposing the MV-Encoder into our federated learning framework stems from the fact that a key goal in the SLU task is to effectively utilize the potential linguistic knowledge of the sentence and capture multi-granularity text representations for the input text. Thus, inspired by Zhao et al. (2019), we attempt to obtain multi-granularity text representations for our framework. To this end, our MV-Encoder consists of four sub-encoders, i.e., Position-wise Encoder, Local Encoder, Global Encoder and Time Series Encoder. Finally, we conduct extensive experiments on three federated learning settings, i.e., horizontal federated learning, vertical federated learning and federated transfer learning, and two benchmark datasets, i.e., SNIPS and ATIS, to validate the effectiveness and the universality of our approach. Moreover, BERT (Devlin et al., 2019), a powerful pre-trained language model, is employed to further boost the performance of our framework.

Overall, the main contributions of our work are as follows:

- We introduce a federated learning framework to unify various SLU datasets and perform the SLU task. In the implementation, our framework could unify various types of knowledge, i.e., text representations, from different datasets and tasks, without the sharing of downstream task data.
- We propose the MV-Encoder to implement the centralized model and learn multi-granularity text representations, which provides a solid bias for our proposed federated learning framework.
- We implement our framework on three federated learning settings. The experiments on two SLU datasets verify the effectiveness and the universality of our approach. More encouragingly, we achieve the best performances on the intent detection task of the SNIPS and the ATIS datasets except for the pre-trained BERT model, which further proves the effectiveness of our approach.

2 Related Work

2.1 Intent Detection and Slot Filling

Recently, user intent detection models (Liu and Lane, 2016; Zhang et al., 2016a; Xia et al., 2018) are proposed to classify user intents given their diversely expressed utterances in the natural language. As a

text classification task, the great performance of sentence level intent detection often depends on hidden representations learned from the intermediate layers through a variety of non-linear transformations. While the intent stands for the sentence level information, the slot represents the word level information which means it annotates the utterance with finer granularity (E et al., 2019). The slot filling is usually treated as a sequential labeling task. A recurrent neural network, such as Gated Recurrent Unit (GRU) or Long Short-term Memory Network (LSTM), is used to learn context-aware word representations, and traditional approaches used Conditional Random Fields (CRF) to annotate each word based on its slot type (Raymond and Riccardi, 2007). What's more, models based on the neural network as well as its extensions enhanced the performance on the slot filling task and outperform traditional CRF based models, and some excellent and interesting works with significant improvement such as the self-attention mechanisms for CRF-free sequential labeling are also introduced (Shen et al., 2018; Tan et al., 2018).

2.2 Federated Learning

Federated Learning (FL) is an emerging artificial intelligence technology. Its design goal is to develop efficient machine learning algorithms among multiple participants on the premise of ensuring information security of terminal data and personal data privacy during data exchange (Konecný et al., 2016a; Konecný et al., 2016b; McMahan et al., 2017; Liu et al., 2020). Take the scenario that companies A and B (two data owners) as an example to introduce the architecture of federated learning. Suppose that companies A and B want to jointly train a machine learning model, and they have relevant data for their users independently. Due to data privacy protection and security considerations, A and B cannot directly exchange data and in this situation, they could use the federated learning system to build models. We call each company participating in the jointly modeling as a participant, and according to the different data distribution (datasets) between multiple participants, the federated learning is divided into three categories: horizontal federated learning, vertical federated learning, and federated transfer learning. This paper describes the application of federated learning in spoken language understanding.

2.3 Multi-Task Learning

Multi-task learning (Caruana, 1997) is a sub-field of transfer learning that aims to solve multiple different tasks at the same time, by taking advantage of the similarities between different tasks. It is a machine learning algorithm and makes no requirements for data security and privacy.

Recently, E et al. (2019), Qin et al. (2019) and Zhou et al. (2020) tried to apply multi-task learning framework to the spoken language understanding problem, they design a model for multi-task learning by strengthening the communication of the downstream SLU task in the decoder layer. In their approach, an input is shared by all the tasks, thus they must create a dataset specially designed for multi-task learning, and multiple objectives are given to the identical inputs. While our framework can not only improve the learning efficiency but also it can read the input sentences from different datasets and carry out different downstream tasks which we will discuss in a while.

3 Approach

In this section, we focus on the basic module of our architecture. Firstly, we introduce an MV-Encoder module to extract the multi-granularity sequence information from several views, which is shown in Figure 1. Then, we describe the intent decoder and slot decoder in the decoding stage. Finally, we introduce the implementation of three federated learning settings and the corresponding datasets on our experiments. The architecture of our federated learning framework is shown in Figure 2.

3.1 Multi-view Encoder

In this section, we introduce a feature fusion module named MV-Encoder. In our framework, intent detection task and slot filling task share the same MV-Encoder, in which the Position-wise Encoder, Local Encoder, Self-attention Encoder and Time Series Encoder are jointly used to encode the input sentences. The MV-Encoder and its components are shown in Figure 1.

3.1.1 Position-wise Encoder

Firstly, to guarantee the purity of information at each position, we employ a fully connected feed-forward network (FFN) (Vaswani et al., 2017) as the position-wise encoder. The FFN is applied to each position separately and identically and consists of two linear transformations with a ReLU activation in between. It is parameterized by two matrices $\mathbf{W}_1 \in \mathbb{R}^{d \times d_{ff}}$ and $\mathbf{W}_2 \in \mathbb{R}^{d \times d_{ff}}$ where d is the input hidden state size and d_{ff} is the number of neurons in the intermediate layer of FFN. We use $\mathbf{X} \in \mathbb{R}^{N \times d}$ to denote the input, where N is the sequence length. The output $\mathbf{P} \in \mathbb{R}^{N \times d}$ can be computed as Eq. (1):

$$\mathbf{P} = \mathrm{FFN}(X) = \max\left(XW_1 + b_1, 0\right) W_2 + b_2 \qquad (1)$$

where b_1, b_2 are the bias in the two linear layers.

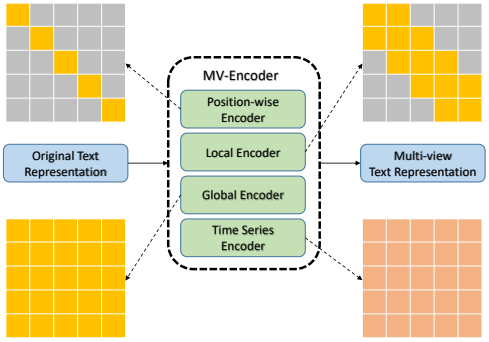


Figure 1: Architecture of the MV-Encoder, which consists of Position-wise Encoder, Local Encoder, Global Encoder and Time Series Encoder. It can provide different text representations for the downstream tasks by making the most of the input information from multiple views.

3.1.2 Local Encoder

Owing to the capability of capturing local correlations of input sentences, Convolutional neural networks (CNN) (Kim, 2014; Kalchbrenner et al., 2014; Lei et al., 2015) have been successfully applied to many sequence labeling tasks (Xu and Sarikaya, 2013; Xu et al., 2018), it extracts high-dimensional features between locally adjacent words by using different sizes of sliding windows for the word vectors of all words of the sentence. Xu and Sarikaya (2013) proposed using Convolutional Neural Network (CNN) based CRF for SLU task, and proved that the CNN model has great performance in the slot filling task. In this paper, we employ CNN in our MV-Encoder. After encoded by CNN, our model output a d dimensional vector in each position. As shown in Figure 1, when we set the kernel size of CNN as 3, the neural network output of each position will contain the information of contiguous three positions of the input sentence. After applying the Local Encoder, the output is represented as $\mathbf{L} \in \mathbb{R}^{N \times d}$.

3.1.3 Global Encoder

Though CNN can extract local features between locally adjacent words, its filter has a limited word capacity and cannot capture long-term dependencies, so it cannot obtain the semantic relationship between non-adjacent words in a sentence. However, due to the pre- and post-dependence of natural language structure, it is important for the sequence labeling task to acquire the global contextual information of the input sentence. Thus, in the global encoder, we adopt the self-attention mechanism (Vaswani et al., 2017), which is excellent in modeling global information (Qin et al., 2019; Tan et al., 2018; Liu et al., 2019; Zhong et al., 2018; Zhao et al., 2019) to capture the context-aware information for each position of the input sentence. In the slot filling task, it helps to determine which position is likely to be a slot. Following Vaswani et al. (2017), suppose the input is $\mathbf{X} \in \mathbb{R}^{N \times d}$ where N is the length of the input sequence and d represents the mapped dimension, we map the matrix of input vectors to queries (Q), keys (K) and values (V) matrices with different linear transformations, where $\mathbf{W}^Q$, $\mathbf{W}^K$, $\mathbf{W}^V$, $\mathbf{W}^O$ $\in \mathbb{R}^{d \times d}$ are projection parameters. The output of the global encoder $\mathbf{G} \in \mathbb{R}^{N \times d}$ is computed as Eq. (2).[1]

$$\begin{aligned} \mathbf{G} &= \mathrm{Attention}(\mathbf{X}) = \mathrm{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}\mathbf{W}^{O\top} \\ &= \mathrm{Softmax}\left(\frac{1}{\sqrt{d_k}}\mathbf{X}\mathbf{W}^Q\mathbf{W}^{K\top}\mathbf{X}^\top\right)\mathbf{X}\mathbf{W}^V\mathbf{W}^{OT} \end{aligned} \qquad (2)$$

3.1.4 Time Series Encoder

Though self-attention mechanism shows great performance in building global information, it is a bit weak in capturing time series information, which means that it cannot get the relative position of two

[1] For detailed explanation of self-attention mechanism, please refer to Vaswani et al. (2017).

tokens. The Bi-LSTM (Hochreiter and Schmidhuber, 1997) has internal mechanisms called gates that can regulate the flow of information. These gates can learn which data in a sequence is important to keep or throw away. By doing that, it can pass relevant information down the long chain of sequences to make predictions and capture long term dependencies. We simplify LSTM recurrence as:

$$\mathbf{h}_i = f\left(\mathbf{h}_{i-1}, \mathbf{x}_i\right) \tag{3}$$

For the input sequence $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$, the Bi-LSTM reads it forwardly and backwardly to produce context-sensitive hidden states. The i-th hidden state $\mathbf{h}_i$ can be denoted as:

$$\mathbf{h}_i \quad = \quad \overrightarrow{\mathbf{h}_i} \oplus \overleftarrow{\mathbf{h}_i} = \overrightarrow{f}\left(\overrightarrow{\mathbf{h}_{i-1}}, \mathbf{x}_i\right) \oplus \overleftarrow{f}\left(\overleftarrow{\mathbf{h}_{i+1}}, \mathbf{x}_i\right) \tag{4}$$

where $\oplus$ is an operation for concatenating two vectors, $\overrightarrow{\mathbf{h}_i}$ and $\overleftarrow{\mathbf{h}_i}$ are the i-th hidden state of the forward and backward LSTM.

Hence, the Time Series Encoder output $\mathbf{H} \in \mathbb{R}^{N \times d}$ can be represented as: $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N)$

After obtaining the output of Position-wise Encoder, Local Encoder, Global Encoder and Time Series Encoder, as shown in Figure 1, we concatenate these four representations as the final MV-Encoder output:

$$\mathbf{E} = \mathbf{P} \oplus \mathbf{L} \oplus \mathbf{G} \oplus \mathbf{H} \tag{5}$$

where $\mathbf{E} \in \mathbb{R}^{N \times 4d}$ and $\oplus$ is concatenation operation.

Through the applying of MV-Encoder, we carry out feature fusion from multiple points of view and generate a useful multi-view text representation. Thus the following decoder layer can utilize such new text representation to effectively solve the downstream tasks. Another benefit of the MV-Encoder is time-friendly because each of the containing encoders can be implemented in parallel.

3.2 Decoder Layer

After extracting feature with the MV-Encoder, we can get the output $\mathbf{E} = (\mathbf{e}_1, ..., \mathbf{e}_N) \in \mathbb{R}^{N \times 4d}$. We then adopt LSTM and linear transformation as a separate intent decoder and slot decoder to perform intent detection and slot filling prediction. For the slot decoder, we use uni-directional LSTM to decode the sequence features. At the decoding step i, the decoder state $\mathbf{h}_i^S$ is calculated by previous decoder state $\mathbf{h}_{i-1}^S$, the previous slot label distribution $\mathbf{y}_{i-1}^S$ and the encoder hidden state $\mathbf{e}_i$, it can be denoted as:

$$\mathbf{h}_i^S = f\left(\mathbf{h}_{i-1}^S, \mathbf{y}_{i-1}^S, \mathbf{e}_i\right) \tag{6}$$

Then the slot label can be computed as:

$$\mathbf{o}_i^S \quad = \quad \operatorname{argmax}\left(\mathbf{y}_i^S\right) = \operatorname{argmax}\left(\operatorname{Softmax}\left(\mathbf{W}^S \mathbf{h}_i^S + \mathbf{b}^S\right)\right) \tag{7}$$

where $\mathbf{y}_i^S$ is the slot output distribution of the i-th token in the utterance, o_i^S represents the slot lable of i-th token and $\mathbf{W}^S$ are trainable parameters of the model.

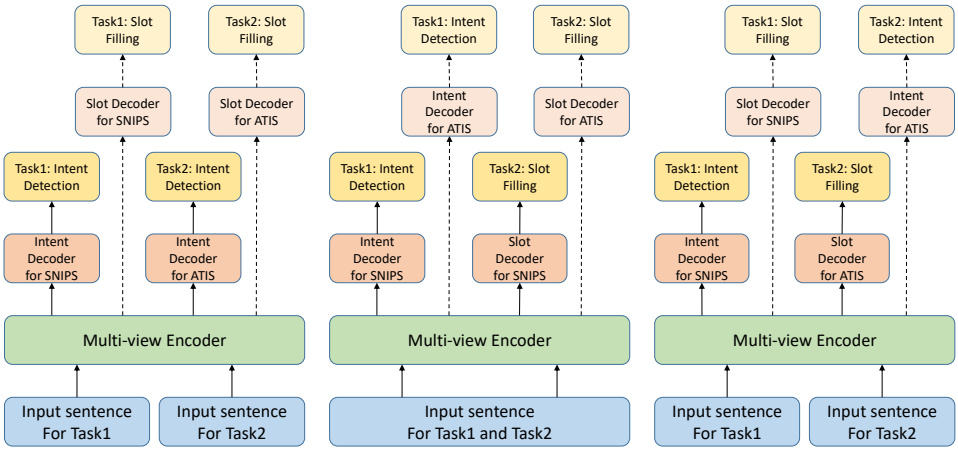
For the intent decoder, we similarly use another LSTM in the intent detection decoder, the LSTM decoder state at decoding step i is denoted as $\mathbf{h}_i^S$:

$$\mathbf{h}_i^I = f\left(\mathbf{h}_{i-1}^I, \mathbf{y}_{i-1}^I, \mathbf{e}_i\right) \tag{8}$$

Then, we sum the LSTM's hidden states at each time step as $\mathbf{h}_{sum}$, and feed it into a linear output layer which projects the component to intent label space. Similar to slot filling, we achieve the normalized distribution over all possible intent labels after softmax:

$$\mathbf{h}_{sum} = \sum_{i \in N} h_i^I; \quad \mathbf{o}^I = \operatorname{argmax}\left(\mathbf{y}^I\right) = \operatorname{argmax}\left(\operatorname{Softmax}\left(\mathbf{W}^I \mathbf{h}_{sum} + \mathbf{b}^I\right)\right) \tag{9}$$

where $\mathbf{y}^I$ is the intent output distribution, o^I represents the intent lable of the utterance and $\mathbf{W}^I$ are trainable parameters of the model.



(a) Horizontal Federated Learning (b) Vertical Federated Learning (c) Federated Transfer Learning

Figure 2: Three settings of federated learning. For (a), we implement the horizontal federated learning setting on SNIPS and ATIS ID datasets (solid black), as well as SNIPS and ATIS SF datasets (dashed black); for (b), we implement the vertical federated learning on SNIPS ID and SF datasets (solid black), as well as ATIS ID and SF datasets (dashed black); for (c), we conduct experiment on ID task of SNIPS dataset and SF task of ATIS datasets (solid black), as well as SF task of SNIPS dataset and ID task of ATIS dataset (dashed black).

3.3 Implementation

In this part, we will introduce the three federated learning settings and our implementation. For each federated setting, we will first introduce its usage scenario. Then we introduce how we apply the federated learning setting to different SLU sub-tasks on SNIPS and ATIS.

3.3.1 Baseline

Due to the lack of decoder interaction in our framework, it doesn't make much sense to directly compare our results with other methods. Thus, in order to better see the improvement brought by our framework, we establish a simple but strong baseline model. Specifically, we take single individual task and dataset as our baseline model, with the cooperation of our MV-Encoder, which means that in this paper, our baseline model performs on SNIPS intent detection dataset, SNIPS slot filling dataset, ATIS intent detection dataset and ATIS slot filling dataset, respectively. The decoder is the same as other models in different federated setting, as described in Section 3.2.

3.3.2 Horizontal Federated Learning

In the case of horizontal federated learning, where the user features of the two data sets overlap more and the user overlaps less, we divide the data set horizontally, i.e., the user dimension, and extract the user features that are the same but the users are not the same to train the model. This method is called horizontal federated learning. For example, there are two banks in different regions, and their user groups are from their respective regions, and the intersection between them is small. However, their businesses are similar, so the recorded user features are the same. At this point, we can use horizontal federated learning to build a joint model.

In implementation, we treat two different intent detection datasets as two banks with respective different "users" (input sentences). And they have the same "user features" (intent detection). As shown in Figure 2a, we implement this setting on SNIPS and ATIS intent detection datasets, as well as SNIPS and ATIS slot filling datasets.

3.3.3 Vertical Federated Learning

In the case of vertical federated learning, where the users of the two data sets overlap more and the user features overlap less, we divide the data set in the vertical direction, i.e., the feature dimension, and take

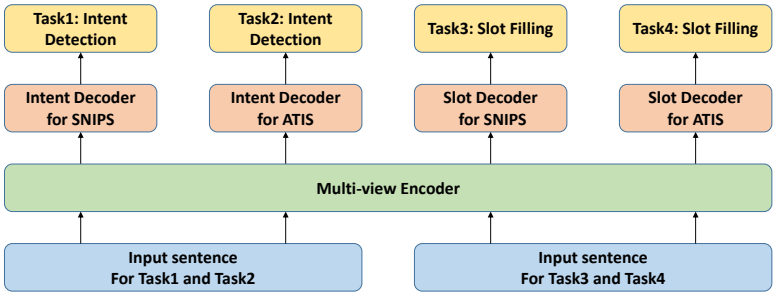


Figure 3: The architecture of full model.

Dataset	SNIPS	ATIS
Vocab Size	11,241	722
Average Sentence Length	9.05	11.28
#Slots	72	120
#Intents	7	21
#Training Samples	13,084	4,478
#Development Samples	700	500
#Test Samples	700	893

Table 2: Statistics of SLU datasets: SNIPS and ATIS

out the same users but different user features to train the model. This method is called vertical federated learning. For example, there are two different institutions. One is a bank in one place and the other is an e-commerce company in the same place. Their user group is likely to include most of the residents of the place and therefore the intersection of users is large. However, because banks record users' income and expenditure behaviors and credit ratings, while e-commerce stores users' browsing and purchasing history, their user features overlap is small. Vertical federation learning is to aggregate these different features in an encrypted state to enhance model capabilities. At present, many machine learning models such as logistic regression models, tree structure models, and neural network models have gradually been proven to be able to build on this federation system.

In our implementation, we treat two different downstream tasks as two different institutions with the same "users" (inputs sentences). And they have different "user features" (intent detection and slot filling). As shown in Figure 2b, we implement this setting on SNIPS intent detection and slot filling datasets, as well as ATIS intent detection and slot filling datasets.

3.3.4 Federated Transfer Learning

In the case where the user and user features of the two datasets have less overlap, we do not segment the data and use the transfer learning to overcome the lack of data or labels. This method is called federated transfer learning. For example, there are two different institutions, one is a bank located in China, and the other is an e-commerce company located in the United States. Due to geographical restrictions, the user groups of the two institutions have very little intersection. At the same time, due to the different types of institutions, only a small part of the data features overlap. In this case, for carrying out federated learning, we use transfer learning to solve the problem of small unilateral data and small label samples, thereby improving the effectiveness of the model.

In our implementation, we treat two different downstream tasks as two different institutions with the two different "users" (inputs sentences). And they have different "user features" (intent detection and slot filling). We implement this setting on intent detection task of SNIPS dataset and slot filling task of ATIS datasets, as well as slot filling task of SNIPS dataset and intent detection task of ATIS dataset, as shown in Figure 2c.

3.3.5 Full Model

After conducting experiments on three federated learning settings, we assume that if all tasks and data could be combined, the performance will be further improved. Thus we further explore the performance of the full model which jointly trains two tasks and two datasets together. The architecture of the full model can be seen in Figure 3.

4 Experiments

In this section, we first describe two benchmark datasets and experimental settings for intent detection and slot filling tasks, and their evaluation metrics. Next, we describe some previous methods and list their performance. Finally, we present our evaluations of the proposed framework on three federated learning

Model	SNIPS		ATIS	
	Intent (*Acc*)	Slot (*F1*)	Intent (*Acc*)	Slot (*F1*)
Joint Seq2Seq (Hakkani-Tür et al., 2016)	96.9	87.3	92.6	94.3
Attention BiRNN (Liu and Lane, 2016)	96.7	87.8	91.1	94.2
Slot-Gated Full Atten (Goo et al., 2018)	97.0	88.8	93.6	94.8
Slot-Gated Intent Atten (Goo et al., 2018)	96.8	88.3	94.1	95.2
Self-Attentive Model (Li et al., 2018)	97.5	90.0	96.8	95.1
Bi-Model (Wang et al., 2018)	97.2	93.5	96.4	95.5
CAPSULE-NLU (Zhang et al., 2019)	97.3	91.8	95.0	95.2
SF-ID Network (E et al., 2019)	97.0	90.5	96.6	95.6
Stack-Propagation (Qin et al., 2019)	98.0	94.2	96.9	95.9
Joint BERT (Chen et al., 2019)	98.6	97.0	97.5	96.1
Joint BERT + CRF (Chen et al., 2019)	98.4	96.7	97.9	96.0
Baseline	97.85	91.91	96.75	95.20
w/ HFL	97.86	92.45	97.09	95.32
w/ VFL	98.43	93.92	97.54	95.55
w/ FTL	97.43	92.82	96.61	95.65
Full Model	98.72	94.89	97.60	96.08
w/ BERT	**99.33**	**97.20**	**98.28**	**96.41**

Table 3: Performance on the SNIPS and ATIS dataset under different federated learning settings.

settings, i.e., horizontal federated learning, vertical federated learning and federated transfer learning, as well as our baseline model and full model.

4.1 Datasets and Settings

To evaluate the efficiency of our proposed model, we conduct experiments on two benchmarks, the widely-used ATIS dataset (Hemphill et al., 1990) and custom-intent-engine dataset called the SNIPS (Coucke et al., 2018), which is collected by snips personal voice assistant. Compared with the ATIS dataset, the SNIPS dataset is more complex due to its large vocabulary and cross-domain intents. The detail of the datasets can be seen in Table 2. Both datasets used in our paper follows the same format and partition as in Qin et al. (2019). Two evaluation metrics are used in the SLU task. The performance of intent detection is measured by accuracy, while slot filling is evaluated with the F1 score.

We adopt the Adam optimizer for optimizing the parameters, with a mini-batch size of 16 and initial learning rate of 0.001. In our experiments, we use the GloVe 300 dimensional word embeddings (Pennington et al., 2014) as our input sentence embeddings. After training 300 epochs, we select the model which works the best on the dev set, and then evaluate it on the test set.

4.2 Results

Following Qin (2019), we use two evaluation metrics in the experiments, i.e., the accuracy for the intent detection task, and the F1 score for the slot filling task. Table 3 shows the experiment results of the proposed framework on SNIPS and ATIS datasets.

Baselines In addition to the previous SLU works, we also propose our own baseline model for easily seeing the promotion of our federated learning framework. As can be seen in Figure 3, even without any decoder interaction and data integration, our baseline model achieves 98.72% and 97.60% accuracy in intent detection task as well as 96.75% and 95.20% F1 score in slot filling task on SNIPS and ATIS datasets, respectively, which outperforms most of the previous methods. From the experimental results, we can observe that: 1) the MV-Encoder performs well in SLU task; 2) though only single task and dataset are used, the result is impressive.

Horizontal Federated Learning As shown in Figure 2, we conduct experiments with HFL on SNIPS and ATIS ID datasets, as well as SNIPS and ATIS SF datasets. Results are shown in Table 3, compared to our baseline model, all the datasets and tasks have improvements on all federated settings, proving the effectiveness of the federated learning framework.

Model	SNIPS		ATIS	
	Intent	Slot	Intent	Slot
Baseline (T)	97.23	91.12	96.06	94.30
w/ G	97.44	91.36	96.37	94.51
w/ G + P	97.61	91.43	96.68	94.72
w/ G + P + L	97.85	91.91	96.75	95.20
w/ G + P + L + VFL	**98.43**	**93.92**	**97.54**	**95.55**
Baseline (full)	**98.72**	**94.89**	**97.60**	**96.08**
w/o T	98.55	94.67	97.38	95.47
w/o G	98.52	94.75	97.37	95.51
w/o P	98.66	94.72	97.46	95.48
w/o L	98.63	94.68	97.47	95.45

Table 4: Ablation analysis of MV-Encoder on the baselines, i.e., the full model and Time Series Encoder. T, G, P, L, VFL stand for Time Series Encoder, Global Encoder, Position-wise Encoder, Local Encoder and vertical federated learning, respectively. Each part of the MV-Encoder is essential for the SLU task.

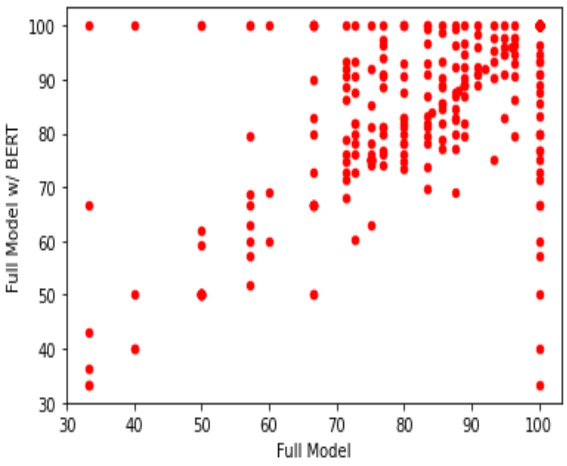


Figure 4: Sentence F1 scores (%) comparison on the ATIS. The x axis presents the F1 scores of the full model and y axis shows the F1 scores of the full model with BERT representation.

Vertical Federated Learning We conduct experiments on two different downstream tasks with the same input sentences in VFL. Table 3 shows that under this setting, we achieve the best performance on most of the metrics (except for ATIS slot) among all the three federated learning settings.

Federated Transfer Learning As for the FTL, we validate the model on SNIPS ID dataset and ATIS SF dataset, as well as SNIPS SF dataset and ATIS ID dataset. The results of FTL can be seen in Tabel 3, all the results on single datasets except ATIS slot are significantly lower than the models under other federated learning settings. We attribute this to the fact that simply training different task from different dataset may have the opposite effect on the model, because it is more difficult for a model to learn knowledge with totally different task and dataset than the model that has the same dataset or the same task such as the above two federated learning settings.

Full Model Base on the above results, we find that the model has obvious improvement on SLU task after applying multiple datasets or multiple tasks, thus, we also conduct an experiment on the full model, aiming to fully integrate all datasets and tasks. From Table 3, We can see that after employ all the datasets and tasks, the model performs best on all the metrics, which further verifies our conclusion.

Full Model w/ BERT Finally, BERT is employed to further boost the performance of the full model by providing abundant input features. Empirically we concatenate the encoding features of BERT with the output of our proposed MV-Encoder and set the decoder same as the full model. As can be seen in Table 3, experimental result shows that, with BERT representation, we get 99.33% and 98.28% in terms of accuracy on ID task as well as 97.20% and 96.41% in terms of F1 score on SF task, respectively.

5 Analysis

Significant improvements among two metrics have been witnessed on both two publicly datasets in Section 4. However, we would like to know the reason for the improvement. In this section, we first study the effect of our proposed MV-Encoder. Next, we explore the effect of the federated learning framework. Finally, we study the effect of BERT representation in our framework.

5.1 Effect of the Multi-view Encoder

Compared with the single task model, our model has obvious improvement in both ID and SF tasks. We attribute this to the fact that, through the sharing of the MV-Encoder, information from different tasks or datasets can be utilized to learn from each other, thus we could fully utilize the potential linguistic

knowledge of the sentence to generate multi-granularity text representations. To verify the effectiveness of the MV-Encoder, we conduct experiments with the two following ablations.

From Table 4, we start with a simple baseline model (i.e., Time Series Encoder), add a sub-encoder each time, and observe that the model performance can be further improved consistently. Since there are so many combinations of this type of addition, we also want to analyze the MV-Encoder from a holistic perspective. From Table 4, we show the results by removing each part of the MV-Encoder, we can clearly see that each part of the MV-Encoder is essential and their combination can provide a strong and robust encoder for our federated learning framework in SLU task. In all, the idea of combining multiple encoder is simple but effective for the SLU task. With the MV-Encoder, our model can generate multi-granularity text representation which is shown to be very helpful to the intent detection and slot filling tasks.

5.2 Effect of Federated Learning Framework

From the last line of Table 4, after employing the overall MV-Encoder with the VFL, results of ID and SF tasks have obvious improvement, which shows that federated learning framework could unify various types of knowledge from different tasks. In addition, compared with the previous model, our model can take data privacy into consideration. It means that our model can avoid data leakage, which is very useful and essential in some specific scenarios. Besides, as shown in Table 3, both of the HFL and VFL can boost the performance of SLU task over the strong baseline model with MV-Encoder, while the FTL setting performs slightly worse than the baseline model on the ID task. We think that it is hard for the model to learn knowledge with totally different task and dataset than the model that can learn from the same dataset on different tasks or the same task on different datasets. Thus, we further propose the full model under our federated learning framework which outperforms most of the existing methods.

5.3 Effect of BERT

As shown by the last two lines in Table 3, compared with the full model without BERT, the improvements brought by our framework over the BERT-enhanced baseline are fairly large on SNIPS SF task, whose absolute F1 score increased by 2.31%. After employing BERT representation, we establish new state-of-the-art results on SNIPS and ATIS datasets. To analyze the difference between the two models (full model and full model w/ BERT), inspired by Zhang et al. (2016b), we conduct an analysis on the sentence performance comparison between them. As shown in Figure 4, we can see that most of the scatter points are distributed near the diagonal line, demonstrating that the full model is similar to the full model with BERT. Though, there are also many points away from the diagonal line, and obviously for these points, the full model with BERT representation performs better, which shows that as for the slot filling task, our MV-Encoder performs better after concatenating with BERT representation. Thus, adding additional BRET representation can further improve the effectiveness of our proposed framework.

6 Conclusion

In this paper, we propose a federated learning framework to perform SLU, where the centralized model is realized by the proposed MV-Encoder to capture the multi-granularity information of the input sentences. Our framework can unify various types of knowledge, i.e., text representations, from different datasets and tasks, without the sharing of downstream task data. Furthermore, we implement our framework on three federated learning settings and conduct extensive experiments on two benchmarks datasets, i.e., SNIPS and ATIS. The experimental results and analysis demonstrate the effectiveness of our framework and the MV-Encoder. Specifically, our approach brings noticeable improvements compared to the baseline. In particular, by leveraging BERT as an additional encoder, we achieve the best performances on SNIPS and ATIS datasets, where we get 99.33% and 98.28% in terms of accuracy on the intent detection task as well as 97.20% and 96.41% in terms of F1 score on the slot filling task, respectively.

Acknowledgments

Special acknowledgments are given to AOTO-PKUSZ Joint Research Center for Artificial Intelligence on Scene Cognition Technology Innovation for its support.

References

Rich Caruana. 1997. Multitask learning. *Machine learning*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. BERT for joint intent classification and slot filling. *CoRR*, abs/1902.10909.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

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

Dilek Hakkani-Tür, Gökhan Tür, Asli **c**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 *HLT*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

Jakub Konecný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. 2016a. Federated optimization: Distributed machine learning for on-device intelligence. *CoRR*, abs/1610.02527.

Jakub Konecný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016b. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492.

Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2015. Molding cnns for text: non-linear, non-consecutive convolutions. In *EMNLP*.

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

Bing Liu and Ian Lane. 2016. Joint online spoken language understanding and language modeling with recurrent neural networks. In *SIGDIAL*.

Fenglin Liu, Xuancheng Ren, Yuanxin Liu, Houfeng Wang, and Xu Sun. 2018. simnet: Stepwise image-topic merging network for generating detailed and comprehensive image captions. In *EMNLP*.

Fenglin Liu, Yuanxin Liu, Xuancheng Ren, Xiaodong He, and Xu Sun. 2019. Aligning visual regions and textual concepts for semantic-grounded image representations. In *NeurIPS*.

Fenglin Liu, Xian Wu, Shen Ge, Wei Fan, and Yuexian Zou. 2020. Federated learning for vision-and-language grounding problems. In *AAAI*.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

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 *EMNLP/IJCNLP*.

Christian Raymond and Giuseppe Riccardi. 2007. Generative and discriminative algorithms for spoken language understanding. In *INTERSPEECH*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI*.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *AAAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz 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 *NAACL-HLT*.

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

Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular CRF for joint intent detection and slot filling. In *ASRU*.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. In *ACL*.

Chenwei Zhang, Wei Fan, Nan Du, and Philip S. Yu. 2016a. Mining user intentions from medical queries: A neural network based heterogeneous jointly modeling approach. In *WWW*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016b. Transition-based neural word segmentation. In *ACL*.

Chenwei Zhang, Nan Du, Wei Fan, Yaliang Li, Chun-Ta Lu, and Philip S. Yu. 2017. Bringing semantic structures to user intent detection in online medical queries. In *BigData*.

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

Guangxiang Zhao, Xu Sun, Jingjing Xu, Zhiyuan Zhang, and Liangchen Luo. 2019. MUSE: parallel multi-scale attention for sequence to sequence learning. *CoRR*, abs/1911.09483.

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

Peilin Zhou, Zhiqi Huang, Fenglin Liu, and Yuexian Zou. 2020. PIN: A novel parallel interactive network for spoken language understanding. In *ICPR*.