A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding

Xiaodong Zhang and **Houfeng Wang***
Institute of Computational Linguistics, Peking University
{zxdcs, wanghf}@pku.edu.cn

Abstract

Two major tasks in spoken language understanding (SLU) are intent determination (ID) and slot filling (SF). Recurrent neural networks (RNNs) have been proved effective in SF, while there is no prior work using RNNs in ID. Based on the idea that the intent and semantic slots of a sentence are correlative, we propose a joint model for both tasks. Gated recurrent unit (GRU) is used to learn the representation of each time step, by which the label of each slot is predicted. Meanwhile, a max-pooling layer is employed to capture global features of a sentence for intent classification. The representations are shared by two tasks and the model is trained by a united loss function. We conduct experiments on two datasets, and the experimental results demonstrate that our model outperforms the state-of-the-art approaches on both tasks.

1 Introduction

Spoken language understanding (SLU) in human/machine spoken dialog systems aims to automatically identify the intent of the user as expressed in natural language and extract associated arguments or slots towards achieving a goal [Tur *et al.*, 2011]. In recent years, with its widespread application in many areas, e.g. automatic customer service, automatic question answering, voice assistants, etc., SLU has become a hot point in research communities. Typically, a SLU system first transcribes users' voice into text by an automatic speech recognizer (ASR), or the input is directly text typed by users. Then the intent of the user and associated arguments is identified. The system can take the next proper action according to the extracted information to help the users achieve their demands.

An example sentence, "Show flights from Boston to New York today", is demonstrated in Table 1 with In/Out/Begin (IOB) representation. This sentence is derived from airline travel information system (ATIS) corpus [Price, 1990], the most widely used dataset in SLU area. The domain of the sentence is airline travel and the intent is to find a flight. The word "Boston" is labelled as the departure city and "New York" as arrival city. There are also named entity labels. The "Boston" and "New York" are labelled as cities, with which the slot label are easier to predict. The domain and intent determination are usually treated as a semantic utterance classification (SUC) problem and the slot filling as a sequence labelling problem. Since categories of intents are more fine-grained than domains, we focus on intent determination in this work.

Domain	Airline Travel	
Intent	Find Flight	
Sentence	**Slot Label**	**Named Entity**
show	O	O
flights	O	O
from	O	O
Boston	B-dept	B-city
to	O	O
New	B-arr	B-city
York	I-arr	I-city
today	O	O

Table 1: An example utterance from the ATIS dataset

In recent years, RNNs have demonstrated their effectiveness in language modelling [Mikolov *et al.*, 2011]. The state-of-the-art method for SF task is also based on RNNs [Yao *et al.*, 2014]. Nevertheless, RNNs have not been explored in the ID task, let alone a joint model for the two tasks. A joint model is worthwhile for two reasons. Firstly, the two tasks are usually both necessary in a SLU system. Secondly, the information of one task can be utilized in the other task to promote each other and a joint prediction can be made. For example, if the intent of a sentence is to find a flight, it is likely to contain the departure and arrival cities, and vice versa. Based on this idea, we propose a joint model for the two tasks. The input text can be viewed as a sequence. GRU is used to learn the representation of each time step in the sequence. On one hand, these representations are used for predicting slot labels. On the other hand, a global representation of the sequence for intent classification is learned by a max-pooling of these representations. Therefore, the representations learned by GRU are shared by two tasks, and with a joint loss function, the two tasks can interact with each other through the shared representations. Experimental results demonstrate that the joint

*Corresponding author

model outperforms separate models for each task. Furthermore, our model outperforms the state-of-the-art methods for ID and SF on two datasets. Another possible way of combining the two tasks is a pipeline scheme. It first classifies the intent of an utterance and then uses the extra intent information to help slot filling. This scheme is inferior to our joint model because the direction of information sharing is one-way and it suffers from error propagation problem.

2 Related Work

In the 1990s, the SLU research emerged from some call classification systems [Gorin *et al.*, 1997] and the ATIS project. Due to the informality of spoken language and recognition errors, the data-driven statistical approach has become the mainstream. For ID, word n-grams are typically used as features with generic entities, such as dates, locations. Because of the very large dimension of the input space, large margin classifiers such as SVM [Haffner *et al.*, 2003] and Adaboost [Schapire and Singer, 2000] were found to be effective for ID. For SF, a lot of work was based on CRF because of its strong ability on sequence labelling [Raymond and Riccardi, 2007]. Apart from lexical and named entity features, some researchers tried to use syntactic information. Hakkani-Tür *et al.* [2005] presented an approach populating heterogeneous features from syntactic and semantic graphs of utterance for call classification. Moschitti *et al.* [2007] employed syntactic features for slot filling via syntactic tree kernels with SVM. To solve the difficulty in processing complex sentences, Tur *et al.* [2011] proposed a dependency parsing based sentence simplification approach that extracts a set of keywords and uses those in addition to entire utterances for completing SLU tasks. Recently, there has been some work on joint ID and SF. Jeong and Geunbae Lee [2008] proposed triangular CRF, which coupled an additional random variable for intent on top of a standard CRF. Mairesse *et al.* [2009] presented an efficient technique that learned discriminative semantic concept classifiers whose output was used to recursively construct a semantic tree, resulting in both slot and intent labels. Although great successes have been made, these methods require good feature engineering and even additional semantic resources. A promising direction is deep learning, which integrates both feature design and classification into the learning procedure.

Recently, various deep learning models have been explored in SLU. The initial try is deep belief networks (DBNs), which have been used in call routing classification [Sarikaya *et al.*, 2011] and slot filling [Deoras and Sarikaya, 2013]. Tur *et al.* [2012] used deep convex networks (DCNs) for domain classification and produced higher accuracy than a boosting-based classifier. RNNs have shown excellent performance on the SF task [Mesnil *et al.*, 2013; 2015] and outperform traditional models, such as CRF. Yao *et al.* [2014] improved RNNs by using transition features and the sequence-level optimization criterion of CRF to explicitly model dependencies of output labels. As for joint work on ID and SF, Xu and Sarikaya [2013] improved the triangular CRF model by using convolutional neural networks (CNNs) to extract features automatically. Guo *et al.* [2014] adapted recursive neural networks (RecNNs) for joint training of ID and SF. To use RecNNs for sequence labelling, tri-path vector was proposed to capture contextual information. In this paper, we propose a RNNs based joint model for SLU. To the best of our knowledge, this is the first work using RNNs for joint ID and SF.

3 Model

The structure of our model is shown in Figure 1. The input of the network is text S of an utterance, which is a sequence of words $w_1, ..., w_T$, and T is the length of the utterance. The network consists of two kinds of output, i.e. the predicted slot label sequence $\hat{l^s}$ and predicted intent label $\hat{l^u}$. Next we give a detailed description of our model.

3.1 Embeddings

As an alternative to traditional representations, such as one-hot representation, word embeddings are suitable for serving as the input of neural networks. These embeddings are usually trained in an unsupervised way on a large corpus and then fine-tuned during supervised training process.

Mesnil *et al.* [2015] found that a context word window could improve the performance of RNNs on SF. Following their work, we also use a context window as the input of the recurrent layer. With each word mapped to an embedding vector, the d-context word window x_t^d, which considers the d previous words and d next words of the current word w_t, is defined as the ordered concatenation of the $2d+1$ word embedding vectors. Formally,

$$x_t^d = [e(w_{t-d}), ..., e(w_t), ..., e(w_{t+d})] \tag{1}$$

where $e(w_t)$ is the embedding of w_t, and the size of the word window is $2d+1$.

Named entity is an important kind of feature for SLU. To utilize these features, we associate each named entity (including a special label representing non-entity) with an embedding, which is initialized randomly according to a uniform distribution on the interval $[0, 1]$ and is fine-tuned during the training process. The context named entity window is defined like the word window. With named entity embeddings, the input of the recurrent layer at time step t is represented as:

$$\begin{aligned} x_t^d = [&e(w_{t-d}), ..., e(w_t), ..., e(w_{t+d}), \\ &e'(n_{t-c}), ..., e'(n_t), ..., e'(n_{t+c})] \end{aligned} \tag{2}$$

where $e'(n_t)$ is the embedding of the named entity n_t, and the size of named entity window is $2c+1$.

3.2 Recurrent Hidden Layers

As an extension of conventional feed-forward neural networks, RNNs can handle the variable-length sequence by using a recurrent hidden state to take into account the influence of past states.

Traditionally, given a sequence $x = (x_1, x_2, ..., x_T)$, the recurrent hidden state is calculated by

$$h_t = \tanh(Wx_t + Uh_{t-1}) \tag{3}$$

where h_t is the hidden state at time t, W and U are respectively transformation matrices of the input and the previous hidden state.

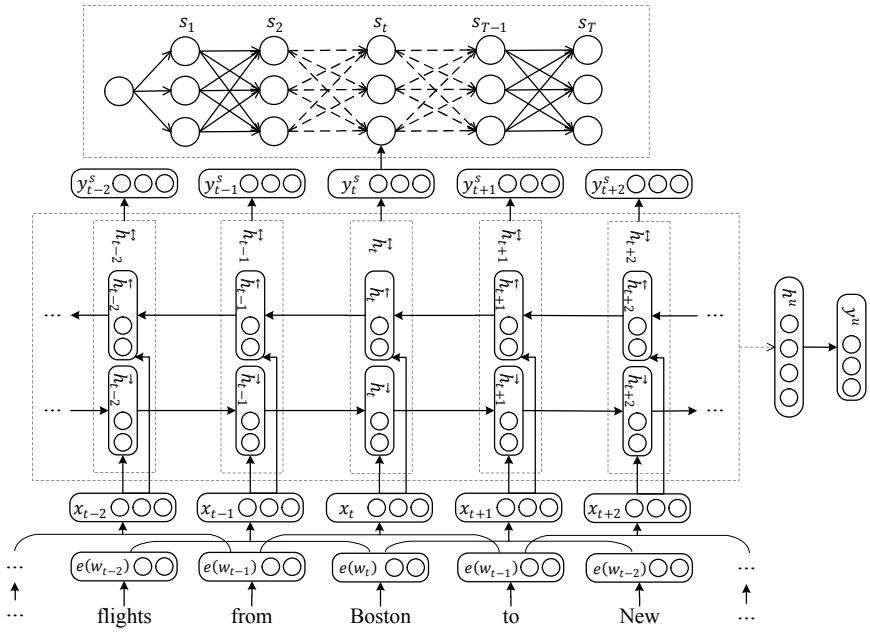

Figure 1: The structure of our model

It was difficult to train RNNs to capture long-term dependencies because the gradients tend to either vanish or explode. Therefore, some more sophisticated activation functions with gating units were designed. Two representative improvements are long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997] and recently proposed gated recurrent unit (GRU) [Cho *et al.*, 2014]. We use GRU in this work, because it performs comparably to LSTM in our experiments, but has less parameters.

GRU Definition: The hidden state h_t of GRU at time t is calculated by

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{4}$$

$$\widetilde{h}_t = \tanh(W x_t + r_t \odot (U h_{t-1})) \tag{5}$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{6}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h}_t \tag{7}$$

where x_t is the input at time t, r and z are reset gate and update gate respectively, σ is sigmoid function, W and U are transformation matrices, and $\odot$ denotes element-wise product of two vectors. For simplification, the above equations are abbreviated with $h_t = GRU(x_t, h_{t-1})$.

For sequence labelling, it is beneficial to consider both past and future information at the same time. Consequently, we use a bidirectional variant of GRU to learn the representations at each time step.

First, we define the forward $\overrightarrow{h}_t$ and the backward $\overleftarrow{h}_t$ hidden layers:

$$\overrightarrow{h}_t = \overrightarrow{GRU}(x_t, \overrightarrow{h}_{t-1}) \tag{8}$$

$$\overleftarrow{h}_t = \overleftarrow{GRU}(x_t, \overleftarrow{h}_{t+1}) \tag{9}$$

where $\rightarrow$ and $\leftarrow$ represent the forward and backward pass respectively.

The bidirectional hidden state $\overleftrightarrow{h}_t$ at time t is defined as the concatenation of the forward and backward hidden states.

$$\overleftrightarrow{h}_t = [\overrightarrow{h}_t, \overleftarrow{h}_t] \tag{10}$$

3.3 Task Specific Layers

The bidirectional hidden states are shared by two tasks. On one hand, the hidden states capture the features at each time step, so they are directly used for predicting slot labels. On the other hand, we use a max-pooling layer to acquire the representation of the whole sequence h^u. The global representation is defined as:

$$h^u = \max_{i=1}^{T} \overleftrightarrow{h}_t \tag{11}$$

where the max function is an element-wise function, and T is the number of words in the utterance. The pooling layer converts texts with variable length into a fixed-length vector, with which the information throughout the entire text can be captured.

The last part is the output layer. The softmax function is applied to the representations with linear transformation to give the probability distribution y_t^s over the t-th slot labels and the distribution y^u over the intent labels. Formally,

$$y_t^s = softmax(W^s \overleftrightarrow{h}_t + b^s) \tag{12}$$

$$y^u = softmax(W^u h^u + b^u) \tag{13}$$

where W^s and W^u are transformation matrices for SF and ID respectively, b^s and b^u are bias vectors.

The labels of slots are inferred at sentence level as in [Chen *et al.*, 2015]. A transition score A_{ij} is introduced to measure

the probability of jumping from label i to label j. For a label sequence $l_{1:T}$, a sentence-level score is given by the sum of label transition scores and predicted scores at each time step:

$$s(l_{1:T}, \theta) = \sum_{t=1}^{T} (A_{l_{t-1} l_t} + y_t^s(l_t)) \quad (14)$$

where $y_t^s(l_t)$ is the predicted probability of label l_t at time step t, which is calculated by Equation 12. A special circumstance is that when $t = 1$, l_0 is involved but not exists. Therefore we add one more label "BOS" to represent l_0. Finally, the predicted slot label sequence $\hat{l^s}$ is the one with the highest score in all possible label sequences L:

$$\hat{l^s} = \underset{l^s \in L}{\operatorname{argmax}}\, s(l^s, \theta) \quad (15)$$

3.4 Training

The parameters θ in our network include:

$$\theta = \{E, E', \overrightarrow{GRU_\theta}, \overleftarrow{GRU_\theta}, W^s, b^s, W^u, b^u, A\} \quad (16)$$

Specifically, $E \in \mathbb{R}^{|e| \times |V|}$, $E' \in \mathbb{R}^{|e'| \times |V'|}$ are respectively word and entity embeddings. $\overrightarrow{GRU_\theta}$ are parameters of the forward GRU, which include the transformation matrices $\overrightarrow{W}, \overrightarrow{W_r}, \overrightarrow{W_z} \in \mathbb{R}^{|x| \times |h|}$, $\overrightarrow{U}, \overrightarrow{U_r}, \overrightarrow{U_z} \in \mathbb{R}^{|h| \times |h|}$ and initial hidden states $\overrightarrow{h}_0 \in \mathbb{R}^{|h|}$. $\overleftarrow{GRU_\theta}$ are parameters of the backward GRU, which is similar to the forward GRU, and we use separate parameters for the forward and backward passes. $W^s \in \mathbb{R}^{2|h| \times L^s}$,$W^u \in \mathbb{R}^{2|h| \times L^u}$ are transformation matrices for two tasks and $b^s \in \mathbb{R}^{L^s}$,$b^u \in \mathbb{R}^{L^u}$ are the bias vectors. $A \in \mathbb{R}^{(L^s+1) \times (L^s+1)}$ is transition score between labels. $|V|, |V'|$ are respectively the number of words and named entities in the vocabulary, and $|e|$,$|e'|$ are dimensions of embeddings of word and named entity. $|h|$ is the dimension of the recurrent hidden units. L^s, L^u are respectively the number of labels of slot and intent. $|x|$ is the dimension of the input of recurrent layer, i.e. $|x| = (2d+1)|e|$ with only lexical features, and $|x| = (2d+1)|e| + (2c+1)|e'|$ with both lexical and named entity features.

Next we define loss function for our networks. We use S to denote the text of an utterance, l^s and l^u to denote the ground truth label of slot and intent.

The loss function for intent is a cross-entropy cost function.

$$\mathcal{L}^u(\theta) = -\log y^u(l^u) \quad (17)$$

The loss function for slot is calculated by a Max-Margin criterion. The structured margin loss $\Delta(l^s, \hat{l^s})$ for ground truth slot label sequence l^s and predicted sequence $\hat{l^s}$ is defined as:

$$\Delta(l^s, \hat{l^s}) = \sum_{t=1}^{T} 1\{l_t^s \neq \hat{l_t^s}\} \quad (18)$$

The loss function of a slot label sequence is defined as:

$$\mathcal{L}^s(\theta) = \max(0, s(\hat{l^s}, \theta) + \Delta(l^s, \hat{l^s}) - s(l^s, \theta)) \quad (19)$$

The training target of the network is minimizing a united loss function:

$$\mathcal{L}(\theta) = \Sigma_{(l^s, l^u, S) \in \mathcal{D}} \left(\alpha \mathcal{L}^s(\theta) + \mathcal{L}^u(\theta)\right) \quad (20)$$

where $\mathcal{D}$ is the dataset, α is a weight factor to adjust the attention paid to two tasks.

Through the united loss function, the shared representations learned by GRUs can consider two tasks jointly. Furthermore, the correlations of the two tasks can be learned and promote each other.

4 Experiments

4.1 Dataset

In order to evaluate our proposed model, we conducted experiments on two datasets. The first set is widely used ATIS dataset, and the second dataset consists of question collected from Baidu Knows[1], the most famous Chinese QA community. We refer to the second dataset as Chinese Question Understanding Dataset (CQUD).

ATIS dataset: The ATIS corpus [Price, 1990] is the most commonly used dataset for SLU research. There are some variants of the dataset. In this paper, we use the ATIS corpus used in [Tur *et al.*, 2010]. The dataset consists of sentences of people making flight reservations. There are 4978 sentences for training and 893 sentences for testing. The numbers of distinct intents and slots are 18 and 63 respectively.

As shown in Table 1, the ATIS dataset also has extra named entity features marked via table lookup, including entities such as city, airline, dates, etc., which nearly determine the slot label. Researchers have different opinions on whether to use these features or not. Some researchers think these features are hand-crafted and not generally available in open domain so that they only use lexical features [Guo *et al.*, 2014; Mesnil *et al.*, 2013]. To make a comprehensive comparison, we give our results with and without named entity features.

CQUD dataset: Although there are some other datasets for SLU, e.g. Cortana Data [Guo *et al.*, 2014] and Bing Query Understanding Dataset [Yao *et al.*, 2014], they are non-public and all English. We intend to conduct experiments on a dataset having some differences with the ATIS. Therefore, we collected questions from Baidu Knows and manually label them. Finally, 3286 questions were filtered and labelled. They are from four domains: Flights, Weather, Express, and Other. The "Other" domain consists of questions that do not belong to the previous three domains. There are totally 43 kinds of intents and 20 kinds of slots. Although questions in Baidu Knows are typed by users, they are far different from formal written language. The style of the questions is informal and filled with colloquialism. They are similar to spoken language except that they are not spoken but typed.

The CQUD dataset distinguishes from the ATIS dataset at three aspects. Firstly, the languages are different so that we can conduct experiments for different languages. Secondly, the domains of CQUD are more diverse than ATIS, and specifically CQUD includes sentences labelled with "Other", which is necessary for real-world applications. Thirdly, the intents and slots in CQUD are more diverse and informal than ATIS, which increases the difficulties for the two tasks. For example, slots of places in ATIS are basically limited to cities in America, while in CQUD the slots are more informal, like "the summit of Changbai Mountain".

[1] http://zhidao.baidu.com/

4.2 Evaluation Metrics

We use accuracy as evaluation metric for ID task. Some utterances in ATIS have more than one intent label. As in previous work [Tur *et al.*, 2010], an utterance is counted as a correct classification if any ground truth label is predicted.

F1-score is used as evaluation metric for SF task. A slot is considered to be correct if its range and type are correct. The F1-score is calculated using CoNLL evaluation script[2].

4.3 Baselines

We compare our model against the following baselines:

SVM: Raymond and Riccardi [2007] used heuristic combinations of forward-moving and backward-moving sequential SVMs classifiers for slot filling.

CRF: A CRF baseline was demonstrated in [Mesnil *et al.*, 2015]. The input is the n-grams in a context window.

RNN: Mesnil *et al.* [2013] employed RNNs for slot filling and the experimental results were updated in [Mesnil *et al.*, 2015].

R-CRF: Yao *et al.* [2014] proposed R-CRF as an improvement to RNNs. It is the state-of-the-art method for SF. Here we use the score reported in [Mesnil *et al.*, 2015] as we use the same dataset.

Boosting: Tur *et al.* [2010] employed AdaBoost.MH algorithm for intent determination. Word n-grams are used as features.

Sentence simplification: Tur *et al.* [2011] parsed the sentence to extract keywords, which served as additional information for SLU task. AdaBoost.MH is the classifier for ID, and CRF for SF. It is the state-of-the-art method for ID.

RecNN: Guo *et al.* [2014] adapted recursive neural networks for joint training of ID and SF. To improve the result on SF, the Viterbi algorithm was applied to optimize the sentence level tag sequence.

4.4 Training Details

We preprocess the datasets as follows. For ATIS, as in [Yao *et al.*, 2013], to deal with unseen words in the test set, we marked all words with only one occurrence in the training data as $\langle \mathrm{UNK} \rangle$ and used this label to represent those unseen words in the test set. We also converted sequences of numbers to the string DIGIT, e.g. "1990" is converted to "DIGIT*4". For CQUD, we use Chinese character rather than word as a basic unit for slot filling. It is because there is no separator between Chinese words and some wrong word segmentations bring inevitable errors for slot filling. Syntactical features are not employed for SF task on CQUD dataset because a parser needs word segmentation first, while for baselines that are set for ID and not joint, such as sentence simplification, we perform word segmentation and syntactical features can be used. Besides, named entity features are not employed on CQUD because we did not label named entities in the dataset.

For our joint model, the average score of the two tasks on the held-out data is used as target to select hyper-parameters. For ATIS, we took out 15% of the training data as hold-out data to determine the hyper-parameters. After optimizing these hyper-parameters, the model is trained on all of the training data and tested on the test data. For CQUD, we report the scores of 5-fold cross validation. GloVe [Pennington *et al.*, 2014] is used to train word embeddings as initializations. For ATIS, the training corpora are Wikipedia and Gigaword. For CQUD, the training corpus is 3 millions unlabelled question collected from Baidu Knows. The dimensions of word embeddings are both 200, and for ATIS the dimension of named entity embeddings is 40. For context window, we set $d = 1$ and $c = 1$. The dimensions of the forward and backward recurrent hidden states are both 300. Model parameters are updated using stochastic gradient descent (SGD). The training set is shuffled at each epoch. The learning rate is initialized to be 0.01 and is updated dynamically by AdaDelta method.

[2]http://www.cnts.ua.ac.be/conll2000/chunking/output.html

4.5 Results and Analysis

The results are demonstrated in Table 2. The second column lists the features used by each method. W, N and S denote lexical, named entity and syntactical features respectively except that W means Chinese character features on CQUD.

We can see that CRF outperforms SVM on SF, showing that with the sequence-level optimization, CRF is suitable for the sequence labelling task. Furthermore, RNN beats CRF because of the ability of capturing long-term dependencies. To model label transfer and acquire the global optimum of the whole sequence, R-CRF combined RNN and CRF, and achieved the state-of-the-art performance on SF task. In fact, the inference of slot labels at sentence level used in our model is similar to R-CRF. For ID task, the state-of-the-art approach is sentence simplification, which uses a dependency parser to extract keywords of a sentence. This method is not a joint work, two individual classifiers are used for the two tasks. RecNN uses the syntactical information in a deep learning scheme. However, the results are worse than the sentence simplification. We think this may be because the scale of the dataset is small such that human-written syntactical features perform better. Nevertheless, Guo *et al.* [2014] compared the joint model and separate models by RecNN and the results demonstrated the effectiveness of the joint model.

On ATIS dataset, our joint model outperforms the state-of-the-art of ID, yielding an absolute improvement of the F1-score of 1.34%, corresponding to a relative error reduction of 44%. As to SF, our model outperforms the state-of-the-art method by 0.43% (relative 12%). The absolute improvement may be not very high, because this dataset have been studied for more than ten years and the score of the state-of-the-art method is very high. Nevertheless, we still achieve obvious relative error reduction. We also observed that the named entity features help the ID only a little, but contribute to the SF a lot, because of the high relevancy of named entities and slot labels. Our method outperforms the previous joint work RecNN significantly, thanks to the powerful ability of recurrent neural networks for modelling sequence and the united loss function that can adjust the weights of two tasks.

The scores on CQUD dataset are lower than ATIS. It is likely because of the higher difficulty of CQUD. As mentioned in Section 4.1, the intents in CQUD are more diverse than ATIS, and the expression of slots are more informal. Besides, the input is a sequence of Chinese characters rather

Model	Features	ATIS		CQUD	
		Intent	Slot	Intent	Slot
SVM [Raymond and Riccardi, 2007]	W	—	89.76	—	81.32
CRF [Mesnil *et al.*, 2015]	W	—	92.94	—	83.40
CRF [Mesnil *et al.*, 2015]	W+N	—	95.16	—	—
RNN [Mesnil *et al.*, 2015]	W	—	95.06	—	85.63
RNN [Mesnil *et al.*, 2015]	W+N	—	96.24	—	—
R-CRF [Yao *et al.*, 2014]	W	—	—	—	**85.88**
R-CRF [Yao *et al.*, 2014]	W+N	—	**96.46**	—	—
Boosting [Tur *et al.*, 2010]	W	95.50	—	93.54	—
Sentence simplification [Tur *et al.*, 2011]	W+S	**96.98**	95.00	**94.46**	—
RecNN [Guo *et al.*, 2014]	W+S	95.40	93.22	—	—
RecNN+Viterbi [Guo *et al.*, 2014]	W+S	95.40	93.96	—	—
Our model	W	98.10	95.49	**96.05**	**87.12**
Our model	W+N	**98.32**	**96.89**	—	—

Table 2: Comparison with previous approaches

than words. Nevertheless, the improvements are consistent in CQUD. Our model outperforms the state-of-the-art methods by 1.59% for ID and 1.24% for SF.

4.6 Joint Model vs Separate Models

First, we give the definitions of joint model and separate models. The joint model is our proposed model in Figure 1. The separate model is similar to the joint model except that there is only one task. For ID, there is only the shared layers and ID specific layers without SF specific layers. It is in the same way for SF. We also implement a pipeline method. First a RNN is trained for ID, and then the predicted intent is used as addition feature to train another RNN for SF.

The speed advantage of the joint model is self-evident, because only one model is needed to train and test. The shared part of the model is only calculated one time for the two tasks. For quantitative analysis, we ran programs of the joint model and separate models using same parameter settings and hardware. On ATIS dataset, the time for training one epoch using joint model is 124 seconds, while the sum of time using separate models is 212 seconds.

Next we compare the performance of the joint model and separate models. In these experiments, only lexical features are used. Here a new concept, joint and with one task oriented, is introduced. In the joint model, we can pay different attentions to the two tasks. This is achieved by adjusting the weight factor α in Equation 20 and using score of one task as target to select hyper-parameters. Larger α means that more attention is paid to SF. The results are listed in Table 3.

The joint model outperforms separate models for two tasks, showing that the joint training is effective. The correlations of two tasks are learned by our joint model and contribute to the two tasks. Because of the two-way information sharing and supervision, our joint model outperforms the one-way pipeline method. Note that if we set one task as oriented in the joint model, higher performance can be acquired for it comparing to treating two tasks equally. This brings flexibility to have tendency to one task if high score is required for it or even only one task is needed in a real application.

In ATIS, α is set to 1.6 for equal model, 1.6 for ID oriented and 1.8 for SF oriented. In CQUD, α is set to 1.5, 2.0 and 1.8 respectively for three models. Intuitively, the performance of one task gets better with higher weight on it. It is not always true in our experiments, which may be because too large weight for one task leads to too quick convergence such that parameters are not well tuned for that task.

Model	ATIS		CQUD	
	Intent	Slot	Intent	Slot
ID only	97.53	—	95.34	—
SF only	—	95.14	—	85.78
Pipeline	97.53	95.41	95.34	86.96
Joint (equal)	98.10	95.49	96.05	87.12
Joint (ID oriented)	98.10	95.49	**96.35**	86.63
Joint (SF oriented)	97.87	**95.61**	95.93	**87.23**

Table 3: Comparison of joint model and separate model

5 Conclusion and Future Work

In this paper we have introduced recurrent neural networks for joint intent determination and slot filling, which are two major tasks in spoken language understanding. Bidirectional GRUs are used to learn the sequence representations shared by two tasks. A global representation is acquired by a max-pooling of the shared representations to predict the label of intent. The labels of slots are predicted by the shared representations and are further inferred at sequence level. Through a united loss function and shared representations, the correlations of the two tasks are learned so as to promote each other. We conducted experiments on two datasets. The joint model demonstrates advantages over separate models and outperforms the state-of-the-art approaches on both tasks.

In future works, we plan to improve our model by using syntactic information. Furthermore, our CQUD dataset is still small-scale for the application of deep learning methods. We would like increase the scale of our dataset, which can be useful for SLU and QA research.

Acknowledgements

Our work is supported by National High Technology Research and Development Program of China (863 Program) (No.2015AA015402), National Natural Science Foundation of China (No.61370117 & No.61433015). We thank Yang Liu for helping revising the paper.

References

[Chen *et al.*, 2015] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*, 2015.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. *Syntax, Semantics and Structure in Statistical Translation*, page 103, 2014.

[Deoras and Sarikaya, 2013] Anoop Deoras and Ruhi Sarikaya. Deep belief network based semantic taggers for spoken language understanding. In *INTERSPEECH*, pages 2713–2717, 2013.

[Gorin *et al.*, 1997] Allen L Gorin, Giuseppe Riccardi, and Jeremy H Wright. How may i help you? *Speech communication*, 23(1):113–127, 1997.

[Guo *et al.*, 2014] Daniel Guo, Gokhan Tur, Wen-tau Yih, and Geoffrey Zweig. Joint semantic utterance classification and slot filling with recursive neural networks. In *IEEE Spoken Language Technology Workshop (SLT)*, pages 554–559. IEEE, 2014.

[Haffner *et al.*, 2003] Patrick Haffner, Gokhan Tur, and Jerry H Wright. Optimizing svms for complex call classification. In *ICASSP*, volume 1, pages I–632. IEEE, 2003.

[Hakkani-Tür *et al.*, 2005] D Hakkani-Tür, G Tur, and A Chotimongkol. Using syntactic and semantic graphs for call classification. In *Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing*, 2005.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Jeong and Geunbae Lee, 2008] Minwoo Jeong and G Geunbae Lee. Triangular-chain conditional random fields. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(7):1287–1302, 2008.

[Mairesse *et al.*, 2009] François Mairesse, Milica Gašić, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. Spoken language understanding from unaligned data using discriminative classification models. In *ICASSP*, pages 4749–4752. IEEE, 2009.

[Mesnil *et al.*, 2013] Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *INTERSPEECH*, pages 3771–3775, 2013.

[Mesnil *et al.*, 2015] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539, 2015.

[Mikolov *et al.*, 2011] Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Honza Černockỳ, and Sanjeev Khudanpur. Extensions of recurrent neural network language model. In *ICASSP*, pages 5528–5531. IEEE, 2011.

[Moschitti *et al.*, 2007] Alessandro Moschitti, Giuseppe Riccardi, and Christian Raymond. Spoken language understanding with kernels for syntactic/semantic structures. In *IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 183–188. IEEE, 2007.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *EMNLP*, 12:1532–1543, 2014.

[Price, 1990] Patti Price. Evaluation of spoken language systems: The atis domain. In *Proceedings of the Third DARPA Speech and Natural Language Workshop*, pages 91–95. Morgan Kaufmann, 1990.

[Raymond and Riccardi, 2007] Christian Raymond and Giuseppe Riccardi. Generative and discriminative algorithms for spoken language understanding. In *INTERSPEECH*, pages 1605–1608, 2007.

[Sarikaya *et al.*, 2011] Ruhi Sarikaya, Geoffrey E Hinton, and Bhuvana Ramabhadran. Deep belief nets for natural language call-routing. In *ICASSP*, pages 5680–5683. IEEE, 2011.

[Schapire and Singer, 2000] Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2):135–168, 2000.

[Tur *et al.*, 2010] Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. What is left to be understood in atis? In *IEEE Spoken Language Technology Workshop (SLT)*, pages 19–24. IEEE, 2010.

[Tur *et al.*, 2011] Gokhan Tur, Dilek Hakkani-Tür, Larry Heck, and Sarangarajan Parthasarathy. Sentence simplification for spoken language understanding. In *ICASSP*, pages 5628–5631. IEEE, 2011.

[Tur *et al.*, 2012] Gokhan Tur, Li Deng, Dilek Hakkani-Tür, and Xiaodong He. Towards deeper understanding: deep convex networks for semantic utterance classification. In *ICASSP*, pages 5045–5048. IEEE, 2012.

[Xu and Sarikaya, 2013] Puyang Xu and Ruhi Sarikaya. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *ASRU*, pages 78–83. IEEE, 2013.

[Yao *et al.*, 2013] Kaisheng Yao, Geoffrey Zweig, Mei-Yuh Hwang, Yangyang Shi, and Dong Yu. Recurrent neural networks for language understanding. In *INTERSPEECH*, pages 2524–2528, 2013.

[Yao *et al.*, 2014] Kaisheng Yao, Baolin Peng, Geoffrey Zweig, Dong Yu, Xiaolong Li, and Feng Gao. Recurrent conditional random field for language understanding. In *ICASSP*, pages 4077–4081. IEEE, 2014.