SEMI-SUPERVISED SLOT TAGGING IN SPOKEN LANGUAGE UNDERSTANDING USING RECURRENT TRANSDUCTIVE SUPPORT VECTOR MACHINES

Yangyang Shi, Kaisheng Yao, Hu Chen, Yi-Cheng Pan, Mei-Yuh Hwang

Microsoft

ABSTRACT

In this paper, we propose a recurrent transductive support vector machine (RTSVM) for semi-supervised slot tagging. Taking advantage of the superior sequence representation capability of recurrent neural networks (RNNs) and the semi-supervised learning capability of transductive support vector machines (TSVMs), the RTSVM is stacking a TSVM on top of a RNN. The performance of the traditional TSVM is sensitive to the regularization weight for unlabeled data in semi-supervised learning. In practice, a suitable unlabeled data regularization weight is difficult to determine. To make the RTSVM semi-supervised learning robust, we propose a *confident subset* regularization method enforcing that the new decision boundaries learned from unlabeled data would not separate the confident clusters learned from labeled data. The experiments based on two datasets show that without using unlabeled data, the supervised version of RTSVM achieves significant F1 score improvement over previous best methods. By taking the unlabeled data into account, the semi-supervised RTSVM gets significant improvement over its supervised opponent.

Index Terms— Recurrent neural networks, transductive support vector machines, semi-supervised learning

1. INTRODUCTION

Slot tagging is a key component in spoken language understanding systems [1, 2, 3, 4, 5], which labels a user query with semantic meaning. The goal of slot tagging is to find a map $\mathcal{F} : \mathcal{X} \rightarrow \mathcal{Y}$ from a sequence of words to a sequence of slot labels, where $\mathcal{X}$ is the word set with vocabulary size N, and $\mathcal{Y}$ the slot set with vocabulary size M. For example, the query "flights from dallas to san francisco" is tagged in the following way. The word "dallas" is labeled as "B-from-city". "san" and "francisco" are labelled as "B-to-city" and "I-to-city", respectively. $\{B, I\}$ is used to indicate the position of a slot label.

flights	from	dallas	to	san	francisco
O	O	B-from-city	O	B-to-city	I-to-city

Recurrent neural network (RNN) and its variants [6, 7, 8, 9, 10, 11] are the cutting-edge techniques for slot tagging. The outstanding performance of RNN based slot tagging is due to its effective sequence feature representation capability. In RNN based slot tagging, a query is represented by a continuous valued vector that implicitly carries the semantic and syntactic information [12, 13]. The special loop structure in RNN acts as a memory to accumulate the history information. Comparing with classic sequential labeling models, for example, conditional random fields (CRFs) [14] and structured support vector machines (SVMs) [15], RNN is able to build slot tagging models from scratch–no feature engineering work is required. However, most of RNN based slot tagging models are still trained via supervised learning.

Supervised learning requires labeled data that needs human annotation. In practice, data manual labeling, especially sequential data manual labeling is time consuming and expensive. In real system, large amount of unlabeled data can be obtained easily from live user logs. It is impractical to manually label all these live queries to improve the real system. One option is semi-supervised learning methods.

The goal of semi-supervised learning methods is to learn a model from a combination of a small size of labeled data set with a large size of unlabeled data set for better performance than to learn from a labeled data set alone. The intuition behind many semi-supervised sequential learning methods is that the intrinsic similar queries share the same label sequence [16]. The intrinsic structure revealed by the unlabeled data can be leveraged to improve the slot tagging models. One widely used discriminative semi-supervised method is transductive support vector machine (TSVM) [17, 18] that is based on the same assumption as most semi-supervised methods but from the other perspective. The assumption of TSVM is that similar data forms the same cluster. In TSVM, the separation of different clusters is achieved by putting decision boundaries in low density area.

In this paper, we propose a recurrent transductive support vector machine (RTSVM) that applies a TSVM on top of a RNN for semi-supervised slot tagging. The stochastic gradient descent algorithm is used to learn the model from the training data. One critical parameter in TSVM is a hyper-parameter for unlabeled data regularization, which is difficult to tune in practice. To make the proposed semi-supervised slot tagging robust to variations of the regularization weight for unlabeled data, an additional constraint is appended to the ob-

jective function. Basically, the additional constraint says that in the training, the confident clusters learned from the labeled samples should not be separated by the decision boundaries learned from the unlabeled data. In this way, we actually dynamically adjust the trade-off weight for labeled data empirical loss and unlabeled data regularization. Experiments on one benchmark dataset namely ATIS (Airline Travel Information Systems) dataset [19, 7], and one internal data set show that the proposed semi-supervised method could achieve significant improvement over its supervised opponent.

In the next section, we discuss about the related work about using RNNs and their variants for slot tagging and other techniques for semi-supervised slot tagging. Section 3 introduces the conventional RNN based slot tagging method. In Section 4, we describe the proposed RTSVMs and the semi-supervised learning method. Section 5 gives the experimental results on ATIS and internal Data. The final section draws conclusions.

2. RELATED WORK

Recently, one interesting direction to improve slot tagging is extracting useful information from large scale web logs and knowledge graphs [20]. In this paper, we improves slot tagging by modifying RNN based slot taggers to make use of unlabeled data extracted form web logs.

Recurrent neural network based slot tagging is first proposed by Yao *et al.*[6]. The first version of RNN-based slot tagger is using local normalization. Each slot is predicted according to current frame lexical information, contextual history information and previous frame slot information. To further exploit the dependency structure of the slot label sequence, Yao *et al.*[8] propose the recurrent conditional random field (RCRF) that stacks a CRF on top of a RNN. RNN and RCRF based slot tagging models proposed by Yao *et al.*are Elman-type of RNN. Slot tagging using the Jordan-type of RNN, bidirectional RNN and the Jordan and Elman hybrid RNN is investigated in [10]. The results in [10] show that Elman-type RNN performs better than Jordan-type RNN and using bidirectional structure helps to improve the performance.

Comparing with classic models, RNN has the advantage of modeling long term dependency. However due to gradient vanish or explosion in error back-propagation, the long term dependency modeling is limited in RNN. To overcome this issue, advanced RNN based slot tagging methods, for example long short term memory (LSTM) networks [21, 22], gated RNN [11] and RNN with external memory (RNNEM) [11] have been proposed. By dedicated designing, these advanced models have better sequence feature representation capability. However, all these methods are based on supervised learning. The proposed RTSVM method uses the TSVM to take the RNN sequence feature representation as input, which can be extended to take the advanced RNN sequence feature representation as input.

There have been extensive literature about semi-supervised learning for structured variables. In general, there are three paradigms for structured varaible semi-supervised learning: generative approach [23, 24], graph-based approach [25, 26, 27, 28] and transductive method [18, 29, 17, 30]. These semi-suprevised methods are based on classic machine learning techniques that need extra feature engineering. In this paper, the proposed RTSVM applies transductive support vector machines (TSVMs) [29] on top of the feature representation of RNNs that have superior sequence representation capability without involving feature engineering work. The performance of TSVM is very sensitive to the hyper parameters for unlabeled data regularization term. Due to the characteristics of stochastic gradient descent algorithm [31], the proposed RTSVM adds another regularization to make the semi-supervised method robust to the hyper-parameter variations for unlabeled data regularization.

3. RECURRENT NEURAL NETWORKS

The conventional RNN based slot taggers [6, 10] illustrated in Fig. 1 have three layers: input layer, hidden layer and output layer. The observation $x(t)$ and the auxiliary feature $f(t)$ is used as input to the network at each frame t. The hidden layer $h(t)$ takes the states from the input layer and previous hidden layer $h(t-1)$ as input. The sigmoid function is used as activation function for the hidden layer. The output layer uses the softmax activation function to compute the multinomial probability distribution $p(y(t)|x(t), f(t))$.

$$p_i(t) = p(y(t) = y_i|x(t), f(t)) = \frac{\exp\left(w_i^T h(t)\right)}{\sum_{j=1}^{m} \exp\left(w_j^T h(t)\right)}, \quad (1)$$

where $p_i(t)$ gives the conditional probability of slot label y_i given input at frame t, and w_i is the weight vector connecting the hidden layer to the output state i. During training cross entropy is applied as the loss function.

In decoding, the slot label $y(t)$ at frame t is selected as follows:

$$y(t) = \underset{y_i(t)\in\{y_1,y_2,\ldots,y_M\}}{\arg\max} p_i(t). \quad (2)$$

To further exploit the dependency structure of slot labels, recurrent conditional random fields (RCRFs) are proposed in [8]. RCRF can be viewed as stacking a CRF on top of a RNN. The RCRF jointly learns the weights for slot label transition and weights for RNN using the maximum likelihood training criteria. In the decoding, Viterbi algorithm is used to find the slot label sequence with the highest probability given the observation sequence.

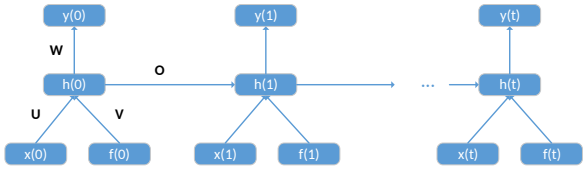

Fig. 1. RNN based slot tagger. U is the weight matrix connecting the word input to the hidden layer, V is the weight matrix connecting the auxiliary feature input to the hidden layer, O is the weight matrix connecting the previous hidden activation state to the current hidden layer state and W is the weight matrix connecting the hidden layer to the output layer.

4. RECURRENT TRANSDUCTIVE SUPPORT VECTOR MACHINES

The proposed RTSVM is illustrated in Fig. 2. The lower part of the figure is a simple RNN that extracts features from user queries. The frame t in the query is represented by a vector $W^t h(t)$ using RNN. On top of the RNN, it is the TSVM that jointly learns the weights matrix A for slot transition features and the weights in RNN using semi-supervised learning.

In semi-supervised learning, the training data is constituted by labeled samples $X^i, i \in [1, n]$, and unlabeled samples $X^j, j \in [n+1, n+m]$, where usually $n << m$. The training objective in RTSVM is the following constrained optimization.

$$\min_{W,A} \frac{1}{2}||W||_2^2 + \frac{1}{2}||A||_2^2 + C\sum_i \zeta_i + \bar{C}\sum_j \bar{\zeta}_j + \hat{C}\sum_k \hat{\zeta}_k$$

$$s.t.\ F(Y^{i^*}) + \zeta_i \geq F(Y^i) + L(Y^i, Y^{i^*}) \quad \forall Y^i \tag{3}$$

$$F(Y^{j^*}) + \bar{\zeta}_j \geq F(Y^j) + L(Y^j, Y^{j^*}) \quad \forall Y^j \tag{4}$$

$$F(Y^{k^*}) + \hat{\zeta}_k \geq F(Y^k) + L(Y^k, Y^{k^*}) \quad \forall Y^k \tag{5}$$

$$\zeta_i, \bar{\zeta}_j, \hat{\zeta}_k \geq 0 \tag{6}$$

$$i \in [1, n], j \in [n+1, n+m], k \in S \subseteq [1, n] \tag{7}$$

where

$$F(Y) = \sum_{t=1}^{T} a_{y(t-1)y(t)} + W_{y(t)}^T h(t) \tag{8}$$

where Y^i and Y^j represent the possible slot sequence for labeled sample X^i and unlabeled sample X^j, respectively. Y^{i^*} is the ground truth slot sequence for labeled sample X^i. For unlabeled sample X^j, Y^{j^*} corresponds the slot label sequence that get the largest discriminative function scores $Y^{j^*} = \arg\max F(Y)$. $a_{y^k(t-1)y^k(t)}$ is the weight for the slot transition features from $y^k(t-1)$ to $y^k(t)$. $L(.)$ defines the loss function of a possible slot label sequence for a training sample. For labeled training sample X^i, $L(Y^i, Y^{i^*})$ is actually used as a margin to separate the score $F(Y^{i^*})$ with score of all other possible slot sequences in Ineq. (3). For unlabeled training sample, $L(Y^j, Y^{j^*})$ is actually used as a margin to separate the slot sequence that get the highest score with all other slot sequences in Ineq. (4). ζ_i, $\bar{\zeta}_j$ and $\hat{\zeta}_k$ are the slack variables that penalize the slot label sequence violating the margin constraints.

Noting that different from traditional TSVM, there is an additional constraint (Ineq. (5)) in the optimization problem. Basically, a subset of *confident* labeled training samples are selected to guarantee that the new decision boundary learned from unlabeled data would not make mistake for these samples. Using stochastic gradient descent algorithm, each iteration of the semi-supervised training consists two phases. In the first phase, the training sweeps over the mixture of labeled training data and unlabeld training data. During the first phase, the *confident* subset S is formed by the labeled training data that satisfying the constraint (Ineq. 3). In the second phase, the model is only trained on the *confident* subset.

Note that Ineq. (3), (4) and (5) are equivalent to the following equations:

$$\zeta = [\max_{Y \neq Y^*} (F(Y) + L(Y, Y^*)) - F(Y^*)]_+, \tag{9}$$

where $[x]_+$ is the Hinge function that maps x to zero when x is smaller than zero, otherwise $[x]_+ = x$. By substituting all the slack variables ζ_i, $\bar{\zeta}_j$ and $\hat{\zeta}_k$ using Eq. (9), the constrained optimization problem is transformed to the following unconstrained optimization problem:

$$\begin{aligned}
&\min_{W,A} \quad F(W, A) = \frac{1}{2C}||W||_2^2 + \frac{1}{2C}||A||_2^2 \\
&+ \sum_{i=1}^{n} [\max_{Y^i \neq Y^{i^*}} \left(F(Y^i) + L(Y^i, Y^{i^*})\right) - F(Y^{i^*})]_+ \\
&+ \bar{\lambda} \sum_{j=n+1}^{n+m} [\max_{Y^j \neq Y^{j^*}} \left(F(Y^j) + L(Y^j, Y^{j^*})\right) - F(Y^{j^*})]_+ \\
&+ \hat{\lambda} \sum_{k \in S \subseteq [1,n]} [\max_{Y^k \neq Y^{k^*}} \left(F(Y^k) + L(Y^k, Y^{k^*})\right) - F(Y^{k^*})]_+.
\end{aligned} \tag{10}$$

where $\bar{\lambda} = \frac{\bar{C}}{C}$ and $\hat{\lambda} = \frac{\hat{C}}{C}$.

In TSVM, the regularization weight $\bar{\lambda}$ for unlabeled samples is difficult to tune. Big $\bar{\lambda}$ usually seriously degrades the performance of model in which labeled training samples have little effect, while small $\bar{\lambda}$ makes model similar to the model trained by labeled data alone. As discussed before, we propose a *confident subset* constraint to make semi-supervised learning robust to the variations of $\bar{\lambda}$. In the experiment, without using the additional *confident subset* regularization term (the last term in Eq. (10)), we rarely get a suitable $\bar{\lambda}$ that makes the semi-supervised learning achieve improvement over its supervised learning opponent. By setting the $\hat{\lambda} = 1$, we find that the models trained using the proposed semi-supervised method achieve similar performance by randomly selecting $\bar{\lambda}$ in the range of $[0.01, 0.09]$. Furthermore,

using the modified TSVM, the model could achieve significant improvement from the unlabeled data over the model trained from labeled data alone.

Another critical factor in TSVM training is the selection of the loss function $L(Y, Y^*)$ that defines the margin between a slot sequence Y with slot sequence Y^*. In our experiment, the following two types of loss functions are investigated.

$$L(Y, Y^*) = \sum_{t=1}^{T} 1\big(y(t) \neq y^*(t)\big) \tag{11}$$

$$L(Y, Y^*) = 1\big(y(1:T) \neq y^*(1:T)\big) \tag{12}$$

Eq. (11) is Hamming loss [32] that is the number of frames at which the slot label in a sequence Y is different from the Y^*. Eq. (12) is sequence level hard loss function that assigns loss one to a slot label sequence that is different from the Y^*. In this paper, we use the margin defined by Eq. (12) as it gives best performance in our experiment.

4.1. Training Procedure For Recurrent Transductive Support Vector Machines

In this paper, models are trained via stochastic gradient descent method. For each training sample $x(1:T)$, the training procedure includes a forward inference and a backward learning.

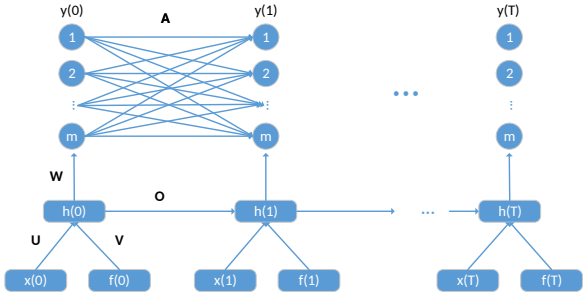

Fig. 2. Recurrent transductive support vector machines for slot tagging. U is the weight matrix connecting the word input to the hidden layer, V is the weight matrix connecting auxiliary feature input to the hidden layer, O is the weight matrix connecting previous hidden state to the current hidden state and W is the weight matrix connecting the hidden layer to the output layer. A represents the weight for slot label transition features.

In the forward inference, a sweep over the RNN part in the Fig. 2 is carried out to calculate the unnormalized score vector $y(t)$ based on the input $x(t)$ and $f(t)$ at each frame t. Taking account of the slot transition feature, a slot lattice is generated for the training sample $x(1:T)$. In the training, the best slot sequence Y^{top} and the runner up slot sequence Y^{second} are derived from the lattice using Viterbi algorithm. In the decoding phase, only the best slot sequence is needed.

Due to the hinge function and maximum function, the objective function of the unconstrained optimization (Eq. 10) is not differentiable. In backward learning, the sub-gradient [33] is used. For labeled training sample X_i, the sub-gradient is

$$\frac{\partial \zeta_i}{\partial \theta} = \frac{\partial F(Y^{i^{\text{top}}})}{\partial \theta} - \frac{\partial F(Y^{i^*})}{\partial \theta}. \tag{13}$$

when the slot sequence $Y^{i^{\text{top}}}$ is not identical to the ground truth slot sequence Y^{i^*}. Otherwise, when the margin between the y^{i^*} and the runner up slot sequence $Y^{i^{\text{second}}}$ is less than the margin $L(Y^i)$, the sub-gradient is

$$\frac{\partial \zeta_i}{\partial \theta} = \frac{\partial F(Y^{i^{\text{second}}})}{\partial \theta} - \frac{\partial F(Y^{i^*})}{\partial \theta}. \tag{14}$$

For the unlabeled data X_j, as we discussed before, the $Y^{j^{\text{top}}}$ is assumed to be the same as Y^{j^*}. So when the margin between the $Y^{j^{\text{top}}}$ and the runner up slot sequence $Y^{j^{\text{second}}}$ is less than the margin $L(Y^j)$, the sub-gradient is

$$\frac{\partial \zeta_j}{\partial \theta} = \frac{\partial F(Y^{j^{\text{second}}})}{\partial \theta} - \frac{\partial F(Y^{j^{\text{top}}})}{\partial \theta}. \tag{15}$$

In Eq. (13), (14) and (15), θ represents the weights in RTSVMs that include W, A, U, V and O. In the backward learning, the weights W, A, U and V are updated using sequence level mini-batch method. The weights O are updated using back-propagation through time (BPTT) [34].

In each training iteration, a *confident subset* of labeled samples is automatically formed and appended to the original training data. Scheduling the *confident subset* labeled data at the end of training data is intend to penalize the new decision boundary learned from unlabeled data that separates the confident clusters learned from labeled training data.

5. EXPERIMENTS

To investigate the effectiveness of the proposed RTSVM method, we need to do two types of experiments. One is to test whether max-margin training criteria on top of RNN feature representation could achieve better performance in slot tagging. This can be tested by setting the regularization weight for unlabeled samples and the *confident subset* to zero. In this way, the proposed RTSVM becomes a supervised method. The other one is to test whether the proposed RTSVM is able to use unlabeled data to improve the model that trained from labeled data alone. To do this test, we partition the original training data into two parts. One part of the training data is treated as unlabeled data by removing all the slot labels.

5.1. Data

In this paper, the experiments are carried out on two datasets. The first one is the benchmark ATIS (Air Traffic Intelligent

Service) dataset [19, 8] that consists of 893 testing queries from ATIS-III, Nov93 and Dec94, and 4978 training utterances from the rest of ATIS-III and ATIS-II. There are 127 unique slot tags.

The second data used in the experiment is internal live log data which contains 8 domains. In total, there are 52506 text queries for training and 5290 text queries for testing. The training data contains 71 unique slot tags.

5.2. Settings

In all the experiments, a predetermined maximum iteration number is set to terminate the training for RTSVMs. AdaGrad [35] is used to dynamically adjust learning rate as follows:

$$\alpha_i(t) = \frac{\alpha}{\sqrt{\sum_{j=1}^{t} g_i(t)^2 + \varepsilon}}, \tag{16}$$

where $\alpha_i(t)$ is the learning rate for weight i at time step t. $\sum_{j=1}^{t} g_i(j)$ is the sum of all the historical gradients of weight i. A small positive ε is used to make the AdaGrad robust. As indicated by Eq. (16), the learning rate gets too small very quickly. In the experiment, the learning rate is reseted every 20 iterations "$\alpha := \alpha/10$" to slow the learning rate reduction.

In all the experiments, we set the hidden layer size to 300 and initial learning rate to 0.1. The regularization weights for unlabeled samples and the *confident subset* are set to 0.01 and 1.0, respectively. The surrounding two words of the current word are used as auxiliary feature represented as bag of words. We set the maximum iteration to 20 for ATIS and 40 for internal live data.

5.3. Results on ATIS

ATIS is a well known benchmark data set in spoken language understanding. The slot tagging F1 scores obtained by different models based on the same data settings are reported in Table. 1. The bottom block gives the results using the supervised version of the proposed RTSVM methods. Using different random seeds, ten different models can be trained from different random weight initialization. "-min", "-max", and "-average" gives the minimum, maximum and average F1 scores obtained by the ten different models.

"CRF [10]" gives the F1 score of the linear chain CRF slot tagger. It is obviously that RNN and its variants based methods achieve superior improvement over CRF. The simple RNN based slot tagger in [6] achieves F1 score of 94.1%. Two different types of RNN, namely Elman RNN and Jordan RNN, are extended and compared in [10]. Their results show that by taking the whole sequence into account, the bidirectional extension improves the Elman and Jordan RNN. The results in [10] show that Elman RNN performs better than Jordan RNN in slot tagging. The best result reported in [10] is achieved by the Elman and Jordan hybrid model that get F1 score 95.1%.

model	F1(%)
CRF [10]	92.9
RNN [6]	94.1
RNN-Jordan[10]	94.3
RNN-embed[36]	94.4
RNN-joint [9]	94.6
RNN-Elman[10]	95.0
RNN-hybrid[10]	95.1
LSTM [7]	94.9
LSTM-ma3[7]	94.9
deep-LSTM [7]	95.0
RNNEM-min[11]	94.7
RNNEM-average[11]	95.0
RNNEM-max[11]	95.2
supervised-RTSVM-min	94.9
supervised-RTSVM-average	95.2
supervised-RTSVM-max	95.5

Table 1. F1 score (in %) for slot tagging on ATIS achieved by different models using only lexicon feature. "-min", "-max", and "-average" each denotes minimum, maximum and average F1 scores for a corresponding method.

By jointly modeling the intent classification and slot tagging [9], "RNN-joint" can obtain F1 score of 94.6%.

In general, the advanced RNN, for example long short term memory network (LSTM) and RNN with external memory (RNNEM) gets better F1 scores than simple RNN by improving the sequence representation capability. Replacing simple RNN with LSTM, the F1 score achieve absolute 0.7% improvement. The RNNEM is recently proposed by Peng et al. [11]. In their experiments, they generated 10 different models using different random seeds. The best F1 score achieved by RNNEM is 95.2%. On average, RNNEM obtains F1 score 95.0%. Using the same method as [11], we also generate 10 different models. As shown in the bottom of Table. 1, the supervised version of RTSVM achieves the state-of-the-art F1 score on ATIS data. The average F1 score of RTSVM is similar to the best score of RNNEM. F1 score distribution of 10 RTSVM models gets significant improvement over the average score of RNNEM (z-test $p - value = 0.0002$).

To test the semi-supervised RTSVM performance on ATIS data, we generate an unlabeled data set from original training data by sampling a subset of labeled data and removing all the slot labels. The rest of labeled data is used as training data for supervised RTSVM. As shown by row "10L+90U" in Table. 2, based on 10% of training data, the supervised RTSVM get average F1 score 87.2%. Taking advantage of additional 90% of original training data (treated as unlabeled data), the proposed semi-supervised method gets absolute 0.7% improvement over the models trained from labeled data alone. On "10L+90U" settings, the paired t-test shows that the semi-supervised method get significant improvement over the su-

setting	ss(%)	s(%)
90L+10U	95.1	94.9
80L+20U	94.9	94.7
70L+30U	94.7	94.6
60L+40U	94.0	94.2
50L+50U	94.0	93.9
40L+60U	93.4	93.2
30L+70U	92.7	92.3
20L+80U	90.9	90.5
10L+90U	87.9	87.2
p-value	0.017	

Table 2. Average F1 score (in %) comparison of semi-supervised RTSVM with supervised RTSVM using different data settings. "XL+YU" means that X percent of data is treated as labeled data, and Y percent of data as unlabeled data. "ss" and "s" columns give the semi-supervised results and supervised results, respectively. Each F1 score in the table is an average F1 score of 10 models trained using different random seeds.

pervised method ($p - value = 0.006$). Based on the average F1 score distributions over different data settings, the paired t-test also show that the semi-supervised RTSVM gets significant improvement over supervised RTSVM with $p - value = 0.017$ shown in the bottom row in Table. 2.

5.4. Results on Internal Live dataset

In a multi-domain language understanding systems [37], a user query is firstly classified into different domains. A domain dependent slot tagger is further applied to extract the slot labels from the query. In this section, we compare different slot models on different domains. In this paper, we focus on slot tagging task. Queries are classified into different domains according to the oracle domain labels. Table. 3 gives the overall performance comparison of different models using the weighted average F1 score over all domains. In this table, we can find that the proposed supervised RTSVM also achieve the best performance. On average supervised RTSVM get 0.6% and 0.7% F1 score improvement over joint-RNN [9] and RCRF [8], respectively.

model	F1(%)
CRF	92.6
RNN	92.8
RCRF	93.9
joint-RNN	94.0
supervised-RTSVM-min	94.0
supervised-RTSVM-average	94.6
supervised-RTSVM-max	95.2

Table 3. The weighted average F1 score of different slot tagging models over all the domains.

Table. 4 gives the average F1 score comparison of semi-supervised RTSVM with supervised RTSVM. Each F1 score in Table. 4 is an average F1 score of 90 different models according to different training data settings and random seeds. For each domain, we consider about 9 different training data settings in which the unlabeled data portion ranges from 10% to 90% as shown in Table. 4. For each domain and each type of data setting, 10 different models are generated using different random seeds. Overall speaking, except on "dom-5" domain, the semi-supervised RTSVM achieves improvement over its supervised opponent. Based on the average F1 score distribution over different domains, the paired t-test shows that semi-supervised RTSVM gets significant improvement from the unlabeled data ($p - value = 0.035$).

domain	ss(%)	s(%)
dom-1	96.54	96.53
dom-2	95.90	95.84
dom-3	93.31	93.15
dom-4	88.00	87.87
dom-5	94.20	94.29
dom-6	87.39	87.11
dom-7	96.08	95.98
dom-8	97.72	97.70
p-value	0.035	

Table 4. The average F1 score comparison of semi-supervised RTSVM with supervised RTSVM. "ss" and "s" columns give the average F1 score of semi-supervised RTSVMs and supervised RTSVMs on different domains.

6. CONCLUSIONS

This paper presented a recurrent transductive support vector machine (RTSVM) for semi-supervised slot tagging. The RTSVM is a combination of a recurrent neural network (RNN) and a transductive support vector machine (TSVM), which uses a RNN to do sequence feature representation and a large margin based TSVM to do semi-supervised learning. In order to make the semi-supervised learning robust to the variations of the regularization weight for unlabeled data, we proposed to add an *confident subset* regularization term to objective function. In the training, the *confident subset* regularization avoids using the new decision boundary learned from unlabeled data to separate the confident clusters learned from labeled data. By removing the unlabeled data regularization and the *confident subset* regularization from the objective function, the proposed RTSVM is transformed to a supervised learning method. Based on the experiments on a benchmark data and an internal live log data, the supervised version of RTSVM achieves the state-of-the-art performance on both datasets. By learning from the unlabeled data, the semi-supervised RTSVM achieved significant improvement over the model learned from labeled data alone.

7. REFERENCES

[1] Charles T. Hemphill, John J. Godfrey, and George R. Doddington, "The atis spoken language systems pilot corpus," in *The Proceedings of the DARPA Speech and Natural Language Workshop*, 1990, pp. 96–101.

[2] Yulan He and S. Young, "A data-driven spoken language understanding system," in *The Proceedings of Automatic Speech Recognition and Understanding*, 2003, pp. 583–588.

[3] Renato De Mori, "Spoken language understanding: a survey," in *The Proceedings of IEEE Workshop on Automatic Speech Recognition Understanding*, 2007, pp. 365–376.

[4] Marco Dinarelli, Alessandro Moschitti, and Giuseppe Riccardi, "Joint generative and discriminative models for spoken language understanding," in *The Proceeding of Spoken Language Technology Workshop*, 2008, pp. 61–64.

[5] Ye-Yi Wang, Li Deng, and Alex Acero, "Spoken language understanding — an introduction to the statistical framework," *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 16–31, 2005.

[6] Kaisheng Yao, Geoffrey Zweig, Mei-Yuh Hwang, Yangyang Shi, and Dong Yu, "Recurrent neural networks for language understanding," in *The Proceedings of Interspeech*, 2013, pp. 2524–2528.

[7] Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi, "Spoken language understanding using long short-term memory neural networks," in *The Proceedings of IEEE workshop on Spoken Language Technology*, 2014, pp. 189–194.

[8] Kaisheng Yao, Baolin Peng, Geoffrey Zweig, Dong Yu, Xiaolong Li, and Feng Gao, "Recurrent conditional random field for language understanding," in *The Proceedings of The International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 4077–4081.

[9] Yangyang Shi, Kaisheng Yao, Hu Chen, Yi-Cheng Pan, Mei-Yuh Hwang, and Baolin Peng, "Contextual spoken language understanding using recurrent neural networks," in *The Proceedings of International Conference on Acoustics, Speech and Signal Processing*, 2015.

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

[11] Baolin Peng and Kaisheng Yao, "Recurrent neural network with external memory for spoken language understanding," in *submitted to Interspeech (http://research.microsoft.com/apps/pubs/?id=246720)*, 2015.

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *The Proceedings of Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[13] Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig, "Linguistic regularities in continuous space word representations," in *The Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 746–751.

[14] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *The Proceedings of the International Conference on Machine Learning*, 2001, pp. 282–289.

[15] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun, "Large margin methods for structured and interdependent output variables," *Journal Of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[16] Xiaojin Zhu, "Semi-supervised learning literature survey," Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[17] Olivier Chapelle and Alexander Zien, "Semi-supervised classification by low density separation," in *The proceedings of International Workshop of Artificial Intelligence*, 1999.

[18] Alexander Zien, Ulf Brefeld, and Tobias Scheffer, "Transductive support vector machines for structured variables," in *Proceedings of the International Conference on Machine Learning*, 2007, pp. 1183–1190.

[19] Charles T. Hemphill, John J. Godfrey, and George R. Doddington, "The atis spoken language systems pilot corpus," in *The Proceedings of the Workshop on Speech and Natural Language*, 1990, pp. 96–101.

[20] Lu Wang, Larry Heck, and Dilek Hakkani-Tur, "Leveraging semantic web search and browse sessions for multi-turn spoken dialog systems," in *The Proceedings of International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 4082–4086.

[21] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton, "Speech recognition with deep recurrent neural networks," in *The Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6645–6649.

[23] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal Of The Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[24] Fernando Pereira and Yves Schabes, "Inside-outside reestimation from partially bracketed corpora," in *Proceedings of the 30th Annual Meeting of the ACL*, 1992, pp. 128–135.

[25] Avrim Blum and Shuchi Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proceedings of the International Conference on Machine Learning*, 2001, pp. 19–26.

[26] Olivier Chapelle, Jason Weston, and Bernhard Schölkopf, "Cluster kernels for semi-supervised learning," in *The Proceedings of Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermayer, Eds., 2003, pp. 601–608.

[27] Martin Szummer and Tommi Jaakkola, "Partially labeled classification with markov random walks," in *The Proceedings of the Advances in Neural Information Processing Systems*, 2002, pp. 945–952.

[28] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *The Proceedings of International Conference of Machine Learning*, 2003, pp. 912–919.

[29] Thorsten Joachims, "Transductive inference for text classification using support vectormachines," in *Proceeding of The International Conference on Machine Learning*, 1999, pp. 200–209.

[30] Vladimir N. Vapnik, *The Nature of Statistical Learning Theory*, 1995.

[31] Constantinos Panagiotakopoulos and Petroula Tsampouka, "The stochastic gradient descent for the primal l1-svm optimization revisited," *CoRR*, 2013.

[32] Nam Nguyen and Yunsong Guo, "Comparisons of sequence labeling algorithms and extensions," *International Conference on Machine Learning*, pp. 681–688, 2007.

[33] Nathan Ratliff, J. Andrew (Drew) Bagnell, and Martin Zinkevich, "(online) subgradient methods for structured prediction," in *Eleventh International Conference on Artificial Intelligence and Statistics (AIStats)*, 2007.

[34] P.J. Werbos, "Backpropagation through time: what it does and how to do it," *The Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[35] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[36] Puyang Xu and Ruhi Sarikaya, "Convolutional neural network based triangular CRF for joint intent detection and slot filling," in *The Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 78–83.

[37] Puyang Xu and Ruhi Sarikaya, "Contextual domain classification in spoken language understanding systems using recurrent nerual network," in *The Proceedings of The International Conference on Accoustics, Speech and Signal Processing*, 2014, pp. 136–140.