BI-DIRECTIONAL RECURRENT NEURAL NETWORK WITH RANKING LOSS FOR SPOKEN LANGUAGE UNDERSTANDING

Ngoc Thang Vu[1,2]*, Pankaj Gupta*[2]*, Heike Adel*[2]*, Hinrich Schütze*[2]

[1]University of Stuttgart, Institute for Natural Language Processing (IMS)
[2]University of Munich, Center for Information and Language Processing (CIS)
thang.vu@ims.uni-stuttgart.de

ABSTRACT

This paper presents our latest investigation of recurrent neural networks for the slot filling task of spoken language understanding. We implement a bi-directional Elman-type recurrent neural network which takes the information not only from the past but also from the future context to predict the semantic label of the target word. Furthermore, we propose to use ranking loss function to train the model. This improves the performance over the cross entropy loss function. On the ATIS benchmark data set, we achieve a new state-of-the-art result of 95.56% F1-score without using any additional knowledge or data sources.

Index Terms— Recurrent neural network, ranking loss, spoken language understanding

1. INTRODUCTION

One of the main tasks of spoken language understanding (SLU) is to assign a semantic concept to each word in a sentence. This is known as slot filling in the speech community. For example, in the sentence 'I want to fly from Munich to Rome', an SLU system should tag 'Munich' as the departure city of a trip and 'Rome' as the arrival city. All the other words, which do not correspond to real slots, are then tagged with an artificial class O. Even after many years of research, the slot filling task along with the intent determination task of SLU are still challenging problems [1, 2]. The main traditional approaches to solving the slot filling task in SLU used generative models, such as hidden markov models (HMM) [3], or discriminative models, such as conditional random fields (CRF) [4, 5]. More recently, neural network models such as recurrent neural networks (RNNs) and convolution neural networks (CNNs) have been applied successfully to this task [17, 18, 19, 20, 21]. This research will focus on the use of RNNs.

RNNs have demonstrated to be successful in many natural language processing tasks, such as language modeling, language understanding and machine translation. A simple RNN consists of an input layer, a recurrent hidden layer, and an output layer. The input layer reads each word and the output layer produces probabilities for the target labels. These can be words for language modeling and machine translation or semantic labels for the slot filling task. The network can be trained with backpropagation through time and therefore can save information from the input words of several time steps to make the prediction for the current word.

In this paper, we apply a bi-directional Elman-type recurrent neural network to the slot filling task. For each input word, the network takes the information not only from the previous words but also from the future words to predict the target slot. Furthermore instead of using cross entropy loss to train the model, we propose to use a ranking loss function. One benefit of this is that it does not force the model to learn a pattern for the artificial class O (which might not exist). Finally, we evaluate the model on the ATIS benchmark data set and show that our model outperforms state-of-the-art results without using any additional knowledge or data sources.

The remainder of the paper is organized as follows: Section 2 gives an overview of related works. We describe the uni- and bi-directional RNN architectures in Section 3. Section 4 presents the ranking loss function which is used to train the network. In Section 5, we show the experimental results with a detailed analysis and comparison with state-of-the-art results. Section 6 summarizes the work and gives possible future work.

2. RELATED WORKS

Based on the success of deep learning in speech recognition [6, 7], this technique has also been applied to intent determination or semantic utterance classification tasks of SLU [8, 9]. Moreover, it has been successfully applied to a number of other human language technology areas including language modeling [10, 11], especially using the recurrent neural networks [12] (with one-hot input encoding).

Another important advance is the invention of word embeddings [13, 14], a projecting of high-dimensional, sparse vectors for word representations into a low-dimensional, dense vector representation for several natural language tasks

[15, 16]. Following this, we use word embeddings as input representation in this work.

Techniques such as recurrent neural networks [17, 18], convolutional neural networks [19] or long short term memory recurrent neural networks [20] have proved to improve the results on the slot filling task of spoken language understanding over traditional approaches. More recently, recurrent neural networks with external memory models [21] were proposed to extend the memory of a simple RNN. In contrast, we revisit the use of past and future context in a bi-directional Elman recurrent neural network [22] for this task. Furthermore, we use a ranking loss function instead of cross entropy to train the model.

3. RECURRENT NEURAL NETWORKS FOR SLOT FILLING

3.1. RNN inputs

As input for the RNN, we use word embeddings which are randomly initialized and jointly trained with the network. Initial experiments showed that concatenating embeddings of trigrams instead of using single words leads to superior results. Hence at time-step t, we do not only give the embedding of word w_t to the model but the concatenated embeddings of the trigram $w_{t-1}w_tw_{t+1}$. In the following figures, however, we depict a standard RNN input (single words) to avoid distractions from the focus of the pictures which is the directionality and not the input of the models.

3.2. Uni-directional RNNs

With uni-directional RNNs, the hidden representation is computed by processing the input word by word and predicting the slot for each word. Thus, history h_t at time-step t is computed conditioned on the information of the preceding words (stored in the preceding history h_{t-1}):

$$h_t = f(U \cdot w_t + V \cdot h_{t-1}) \quad (1)$$

As the non-linear function f, we use sigmoid throughout all our experiments. Figure 1 shows the uni-directional RNN architecture.

3.3. Bi-directional RNNs

Especially for slot filling, the processing of the slot arguments might be easier with knowledge of the succeeding words. Therefore in bi-directional RNNs, not only the previous history of word w_t is regarded but also the future history. Figure 2 shows this in which U_b and U_f are shared and then denoted with U in the equations 2 and 3. Thus, the network can be split into three parts: a forward pass which processes the original sentence word by word; a backward pass which processes the reversed sentence word by word; and a combination of both. All three parts are trained jointly. The

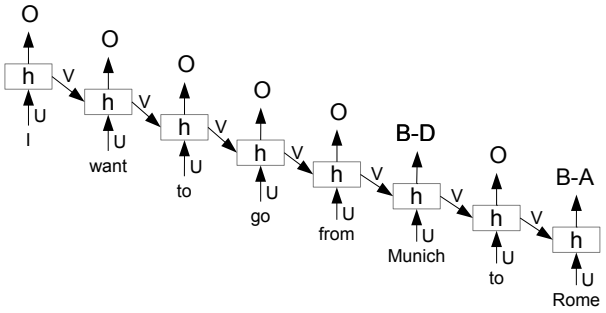

Fig. 1. Uni-directional RNN for slot filling task

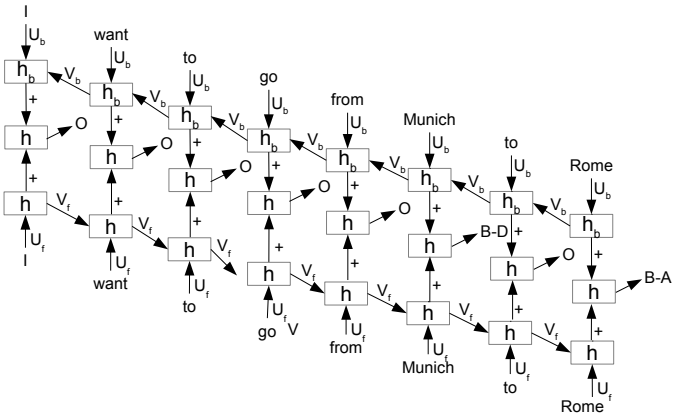

Fig. 2. Bi-directional RNN for relation classification

combination can be done by adding the forward and the backward hidden layer. This leads to the following hidden layer output at time step t:

$$h_t = f(U \cdot w_t + V_b \cdot h_{b_{t+1}} + V_f \cdot h_{f_{t-1}}) \quad (2)$$

Another option for combination is to concatenate the forward and the backward hidden layer.

$$h_t = [f(U \cdot w_t + V_f \cdot h_{f_{t-1}}), f(U \cdot w_t + V_b \cdot h_{b_{t+1}})] \quad (3)$$

The combined hidden layer output is then used to predict the semantic label for the current word.

4. LEARNING OBJECTIVE FUNCTIONS

4.1. Cross entropy

Most approaches use a logistic regression classifier with the softmax activation function in the final layer. The objective function which is mainly used in this case is based on cross entropy:

$$L = -\sum_c y_c \cdot log(s_\theta(x)_c) \quad (4)$$

In this equation, c iterates over all classes, y_c is the correct value for class c and $s_\theta(x)_c$ is the score the network assigned to class c given the current data point x.

4.2. Ranking

Instead of using the sofmax activation function, we train a matrix W^{class} whose columns contain vector representation of the different classes. Therefore, the score for each class c can be computed by using the product

$$s_\theta(x)_c = h_x^t [W^{class}]_c \tag{5}$$

We use a ranking loss function to train the RNN. It learns to maximize the distance between the true label y^+ and the best competitive label c^- given a data point x. The objective function is

$$\begin{aligned} L = &\log(1 + \exp(\gamma(m^+ - s_\theta(x)_{y^+}))) \\ &+ \log(1 + \exp(\gamma(m^- + s_\theta(x)_{c^-}))) \end{aligned} \tag{6}$$

with $s_\theta(x)_{y^+}$ and $s_\theta(x)_{c^-}$ being the scores for the classes y^+ and c^- respectively. This function was proposed by Dos Santos et al. [23] to train convolution neural networks for relation classification. The parameter γ controls the penalization of the prediction errors and m^+ and m^- are margins for the correct and incorrect classes. γ, m^+ and m^- are hyper-parameters which can be tuned on the development set. For the class O, we only calculate the second summand of equation 6. By doing this, we do not learn a pattern for class O but nevertheless increase its difference to the best competitive label. During testing, the model will predict class O if the score for all the other classes is lower than 0.

One of the advantages of this loss function over the softmax function is efficiency. Since only two classes are computed at every training iteration, the network can be trained quite fast even with a large number of classes. Furthermore, a ranking loss function is suitable for tasks like slot filling because it does not force the network to learn a pattern for the O class which in fact may not exist.

5. EXPERIMENTAL RESULTS

5.1. Data

To compare with previously studied methods, we report results on the widely used ATIS dataset [24, 25]. This dataset is on the air travel domain, and consists of audio recordings of people making travel reservations. All the words are labeled with a semantic label in a BIO format (B: begin, I: inside, O: outside). For example, 'New York' contains two words 'New' and 'York' and therefore can be tagged with 'B-fromloc.city_name' and I-fromloc.city_name' respectively. Words which does not have semantic labels are tagged with O. In total, the number of semantic labels is 127, including the label of the class O. The training data consists of 4,978 sentences and 56,590 words. Test data contains 893 sentences and 9,198 words. To evaluate our models, we used the script provided in the text chunking CoNLL shared task 2000[1] as other related works.

5.2. Model training

We used the Theano library [26] to implement the model. To train the model, stochastic gradient descent (SGD) was applied. The following table shows the hyper-parameters which we used for all the RNN models.

Parameters	Value
activation function	sigmoid
regularization	L2
L2 weight	1e-7
mini batch size	1
initial learning rate	0.02
hidden layer size	100

For tuning the hyper-parameters, we performed a 5-fold cross-validation. The training starts with the initial learning rate in the first ten epochs. Afterwards, we halved the learning rate and finished the training after 25 epochs. We also experimented with more advanced techniques like AdaGrad [27] and AdaDelta [28] but did not achieve improvements over SGD with the described simple learning rate schedule. Since the learning schedule does not need a cross-validation set, we train the final best model with all the training data.

5.3. Results

5.3.1. Uni- vs. bi-directional

In the first experiment, we compare the uni-directional and bi-directional RNN trained with the cross-entropy loss function. Since the information of the future words can be used to predict the semantic label of the target word, we also report the F1-score by using only the backward pass. The results listed at the top of Table 1 reveal that the bi-directional RNN outperforms the uni-directional one. Interestingly, the result of a uni-directional RNN with backward pass is not so different from the uni-directional RNN with forward pass. This shows that the information of the future words is also important to predict the semantic label of the target word. Combining (by addition and concatenation) the hidden layers of forward and backward pass in a bi-directional RNN structure provides consistent improvements over the uni-directional RNN. Using addition is, however, more accurate than using concatenation. It provides the best performance with a F1-score of 94.92%.

[1] http://www.cnts.ua.ac.be/conll2000/chunking/

Methods	F1-score
Uni-directional RNN w. forward pass	94.11
Uni-directional RNN w. backward pass	93.78
Bi-directional RNN (add)	94.92
Bi-directional RNN (concat)	94.85
Bi-directional RNN (add) with ranking loss	**95.47**
Bi-directional RNN (concat) with ranking loss	95.40

Table 1. top: uni- vs. bi-directional RNN
with cross-entropy loss function, bottom: ranking loss

5.3.2. Effect of objective function

Instead of applying a softmax layer and training the network with the cross entropy loss function, we used the ranking loss function described in Section 4. The hyper-parameters of the ranking loss function are optimized with a 5-fold cross validation. The best parameters are: $\gamma = 2, m^+ = 3, m^- = 0.5$. We obtain a F1-score of 95.47% on the ATIS test set which corresponds to 0.55% absolute improvement compared to the cross entropy loss function (see bottom part of Table 1).

To have a better understanding of the effect of omitting the second term in the objective function 6 for the artificial class 'O', we also trained a model using both terms. This variation of the objective function can be interpreted as cross entropy loss function with a negative sample of size one. The F1-score drops to 94.85% which is quite close to the results of the model using softmax layer trained with cross entropy loss. This result indicates that it is important to not force the model to learn a pattern for class 'O'.

To provide some insight information how the parameters of the ranking loss function effect the final F1-score, we present the effect of different positive margins m^+ (see Table 2) and different scaling factors γ (see Table 3) on the results. In both experiments, m^- is fixed to 0.5.

Positive margin	2	2.5	3	3.5	4
F1-score	95.05	95.14	**95.47**	95.08	94.89

Table 2. F1-scores with different positive margins

Table 2 shows that the variance of the results is quite small when the positive margin ranges from 2 to 4. The best performance is observed for $m^+ = 3$.

Scaling factor	1	1.5	2	2.5	3
F1-score	95.04	95.15	**95.47**	95.17	95.11

Table 3. F1-scores with different scaling factors

The results in Table 3 reveal the same trend as the results in Table 2: The F1-score is improved by increasing the scaling factor up to 2 and decreased afterwards.

5.4. Comparisons with state-of-the-art

Table 4 lists several previous results on the ATIS data set including our best results. The previous best result was achieved using LSTM and recently improved by using recurrent neural networks with memory [21]. The results in Table 4 show that our bi-directional ranking RNN (R-biRNN) outperforms the previous best models. Finally, we trained five different models with different random initializations of parameters and combined them with a voting process. In case of a tie, we picked one of the most frequent classes randomly. The combination achieves an F1 score of 95.56 which is the new state-of-the-art result on this benchmark dataset. Note that we did not use any additional (linguistic) features or data sources.

Methods	F1-score
CRF [18]	92.94
simple RNN [17]	94.11
CNN [19]	94.35
LSTM [20]	94.85
RNN-EM [21]	95.25
biRNN	94.92
R-biRNN	95.47
5xR-biRNN	**95.56**

Table 4. F1 score(%) on ATIS data

6. CONCLUSIONS

In this paper, we presented a bi-directional Elman-type recurrent neural network for the slot filling task of spoken language understanding. This network takes the information not only from past but also from future contexts to predict the semantic label of the target word. The results revealed that integrating future information is important for this task. Furthermore, using a ranking loss function to train the model improved the performance over the cross entropy loss function. On the ATIS spoken language understanding task, we achieved new state-of-the-art results with 95.56% macro F1-score without using any additional features or data sources.

One possible future work is to extend the ranking loss function to include not only one negative sample but also the top k closest negative samples during training.

7. REFERENCES

[1] G. Tur and L. Deng. Intent Determination and Spoken Utterance Classification, in Chapter 4, Spoken Language Understanding: Systems for Extracting Semantic Information from Speech, pp. 81-104, Wiley, 2011.

[2] S. Yaman, L. Deng, D. Yu, Y. Wang, and A. Acero. An integrative and discriminative technique for spoken ut-

terance classification, IEEE Trans. Audio, Speech, and Language Processing, vol. 16, no. 6, pp. 12071214, 2008.

[3] Y. Wang, L. Deng, and A. Acero. Spoken Language Understanding An Introduction to the Statistical Framework, IEEE Signal Processing Magazine, vol. 22, no. 5, pp. 16-31, 2005.

[4] J. Lafferty, A. McCallum, and F. P ereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in Proc. of ICML, 2001.

[5] Y. Wang, L. Deng, and A. AceroSemantic Frame Based Spoken Language Understanding, in Chapter 3, Spoken Language Understanding: Systems for Extracting Semantic Information from Speech, pp. 35-80, Wiley, 2011.

[6] G. Dahl, D. Yu, L. Deng, and A. Acero. Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition, in IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 1, pp. 30-42, 2012.

[7] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition, in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, 2012.

[8] L. Deng, G. Tur, X. He, and D. Hakkani-Tur, Use of Kernel Deep Convex Networks and End-To-End Learning for Spoken Language Understanding, in Proc. of SLT, 2012.

[9] G. Tur, L. Deng, D. Hakkani-Tur, and X. He, Towards Deeper Understanding Deep Convex Networks for Semantic Utterance Classification, in Proc. of ICASSP, 2012.

[10] A. Mnih and G. Hinton, A scalable hierarchical distributed language model in Proc. of NIPS, 2008.

[11] Socher, R., Lin, C., Ng, A., and Manning, C. Learning continuous phrase representations and syntactic parsing with recursive neural networks, Proc. of ICML, 2011.

[12] T. Mikolov, Stefan Kombrink, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur, Extensions of recurrent neural network based language model, in Proc. of ICASSP, 2011.

[13] Y. Bengio, R. Ducharme and P. Vincent, A Neural Probabilistic Language Model, in Proc. of NIPS, 2000.

[14] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In Proc. of Workshop at ICLR, 2013.

[15] R. Collobert and J. Weston, A unified architecture for natural language processing: deep neural networks with multitask learning, in Proc. of ICML, 2008.

[16] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, Natural language processing (almost) from scratch, in Journal of Machine Learning Research, vol. 12, 2011.

[17] K. Yao, G. Zweig, M. Hwang, Y. Shi, and D. Yu, Recurrent neural networks for language understanding, in Prof. of Interspeech, 2013.

[18] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, Using recurrent neural networks for slot filling in spoken language understanding, IEEE/ACM Trans. on Audio, Speech, and Language Processing, vol. 23, no. 3, pp. 530539, 2015.

[19] P. Xu and R. Sarikaya, Convolutional neural network based triangular CRF for joint intent detection and slot filling, in Prof. of ASRU, 2013.

[20] K. Yao, B. Peng, Y. Zhang, D. Yu, G. Zweig, and Y. Shi, Spoken language understanding using long short-term memory neural networks, in Proc. of SLT, 2014.

[21] B. Peng, K. Yao. Recurrent Neural Networks with External Memory for Language Understanding, in arXiv, 2015.

[22] Elman, J. L. Finding structure in time. Cognitive science, 1990.

[23] Dos Santos, C. N.; Xiang, B.; and Zhou, B. Classifying relations by ranking with convolutional neural networks. In Proc. of ACL, 2015.

[24] C. Hemphill, J. Godfrey, and G. Doddington, The ATIS spoken language systems pilot corpus, in Proc. of the DARPA speech and natural language workshop, 1990.

[25] P. Price, Evaluation of spoken language systems: The ATIS domain, in Proc. of the Third DARPA Speech and Natural Language Workshop. Morgan Kaufmann, 1990.

[26] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I.J. Goodfellow, A. Bergeron, N. Bouchard, Y. and Bengio, Y. Theano: new features and speed improvements. In Proc. of Deep Learning and Unsupervised Feature Learning NIPS Workshop, 2012.

[27] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization, Journal of Machine Learning Research, vol. 12, pp. 21212159, 2010.

[28] M.D. Zeiler. ADADELTA: An Adaptive Learning Rate Method . CoRR, abs/1212.5701, 2012.