Attention-Based Recurrent Neural Network for Sequence Labeling

Bofang Li[1,2], Tao Liu[1,2(✉)], Zhe Zhao[1,2], and Xiaoyong Du[1,2]

[1] School of Information, Renmin University of China, Beijing, China
{libofang,tliu,helloworld,duyong}@ruc.edu.cn
[2] Key Laboratory of Data Engineering and Knowledge Engineering, MOE, Beijing, China

Abstract. Sequence labeling is one of the key problems in natural language processing. Recently, Recurrent Neural Network (RNN) and its variations have been widely used for this task. Despite their abilities of encoding information from long distance, in practice, one single hidden layer is still not sufficient for prediction. In this paper, we propose an attention architecture for sequence labeling, which allows RNNs to selectively focus on every useful hidden layers instead of irrelative ones. We conduct experiments on four typical sequence labeling tasks, including Part-Of-Speech Tagging (POS), Chunking, Named Entity Recognition (NER), and Slot Filling for Spoken Language Understanding (SF-SLU). Comprehensive experiments show that our attention architecture provides consistent improvements over different RNN variations.

1 Introduction

Nowadays, analyzing and extracting useful information from plain text (especially web content) is one of the most important research areas. For many applications, sequence labeling is a fundamental pre-processing step. It is also one of the most well-studied tasks in natural language processing. As shown in Table 1, sequence labeling tasks aim at automatically assigning words in texts with labels.

Traditionally, Hidden Markov Models (HMM), Conditional Random Fields (CRFs), and Support Vector Machine (SVM) has been widely used for sequence labeling tasks [9,10,14,15]. Compared with these models, Recurrent Neural Networks (RNNs) are able to capture information from a fairly long distance. Recently, with the help of extra resources and feature engineering, the combination of RNN and other models achieves state-of-the-art results [3,8,12,13].

For sequence labeling, each target word and its corresponding label are explicitly aligned. Previous RNNs predict label solely based on each hidden layer of the corresponding target word. However, in practice, using one single hidden layer is not sufficient for prediction, even with sophisticated variations like Bi-directional Recurrent Neural Network (Bi-RNN) [16], Long Short-Term Memory (LSTM) [7], and Gated Recurrent Unit (GRU) [4].

Y. Cai et al. (Eds.): APWeb-WAIM 2018, LNCS 10987, pp. 340–348, 2018.
https://doi.org/10.1007/978-3-319-96890-2_28

Table 1. An example of sequence labeling tasks.

Words	Flight	from	Boston	to	New	York
POS	NN	IN	NNP	TO	NNP	NNP
Chunking	B-NP	B-PP	B-NP	B-PP	B-NP	I-NP
NER	O	O	B-loc	O	B-loc	I-loc
SF-SLU	O	O	B-dept	O	B-arr	I-arr

This paper proposes an Attention-based Recurrent Neural Network for Sequence Labeling (ARNN-SL), which allows RNNs to "focus" not only on the aligned hidden layer, but other informative hidden layers as well.

Different from other tasks such as machine translation [1], image caption [19], and speech recognition [5], where attention mechanism has been successfully applied, sequence labeling has its own characteristics for deciding which hidden layer is informative or not. Intuitively, the closer a hidden layer is to target word, the more information it contains. Moreover, the aligned hidden layer is always most important to this end. A windowing technique is introduced by limiting our model to selectively focus on hidden layers in a small window size, instead of irrelative hidden layers far way. ARNN-SL explicitly leverages the information from the aligned and attention-focused hidden layers for prediction.

2 Model

2.1 Simple RNN for Sequence Labeling

Formally, sequence labeling aims at finding the most probable label sequence $\mathbf{y} = \{y_1, ..., y_T\}$ for a given input word sequence $\mathbf{w} = \{w_1, ..., w_T\}$, where T is the sequence length.

The overall architecture of simple RNN (sRNN) for sequence labeling is depicted in Fig. 1. In this figure, w_t represents word at time step t and $x_t \in \mathbb{R}^n$ is w_t's word embedding. $y_t \in \mathbb{R}^L$ is the probability over L labels of word at position t and is defined as[1]:

$$y_t = softmax(Vh_t) \tag{1}$$

where $h_t \in \mathbb{R}^m$ is the hidden state at time step t. h_t encodes the information in previous time steps and is computed as:

$$h_t = \sigma(Wx_t + Uh_{t-1}) \tag{2}$$

where V, W and U are weight matrices. σ is active function and is often set to *sigmoid*.

[1] For simplicity, we omit bias terms in all the equations.

2.2 RNN Variations

Most of the variations of RNN focus on modifying the way of calculating hidden layer h_t in Eq. 2. For example, Long Short-Term Memory (LSTM) [7] and Gated Recurrent Unit (GRU) [4] introduces additional memory cells and gates to depict the long-range dependency information much better. For local context window technique [13], instead of using x_t for computing h_t, it uses a weighted sum of all x_j in a context window of wn as input: $\sum_{i=-wn}^{wn} U_i x_{t+i}$, which allows RNN to consider more local context dependency information.

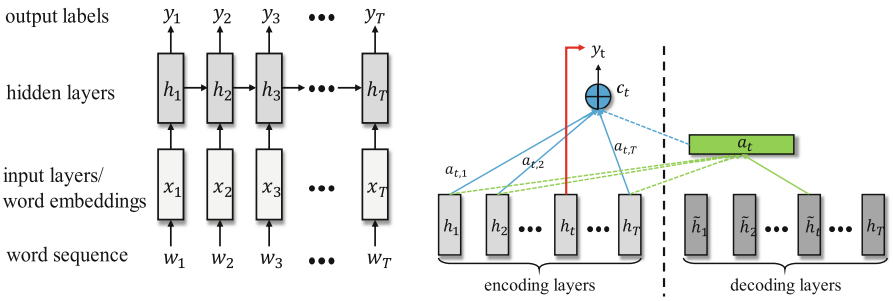

Fig. 1. Illustration of sRNN and ARNN-SL for sequence labeling.

Bi-directional Recurrent Neural Network [16] is also commonly used for improving the performance of sRNN. It computes the forward hidden layer h_t^f and backward hidden layer h_t^b using h_{t-1}^f and h_{t+1}^b respectively, and concatenates these two types of layers to form the final hidden layer h_t. In this way, both past and future information is preserved.

2.3 Proposed Attention Architecture

In practice, h_t alone is still not sufficient for encoding all the information needed for predicting y_t, even with sophisticated variations like Bi-RNN, LSTM, and GRU. In this paper, we propose Attention-based Recurrent Neural Network for Sequence Labeling (ARNN-SL), which allows RNN to selectively use multiple hidden vectors' information instead of using h_t alone.

As shown in Fig. 1, ARNN-SL has two types of hidden layers: encoding layer h_t and decoding layer $\tilde{h}_t$. The same as most encoder-decoder frameworks, the last encoding layer is used as the input of the first decoding layer, two sets of parameters are used for encoder and decoder respectively.

If we ignore the attention component c_t and all decoding layers, the architecture is exactly the same as sRNN and its variations. Attention component c_t is used to selectively gather information from encoding layers for prediction, which is computed as:

$$c_t = \sum_{j=1}^{T} a_{t,j} h_j \tag{3}$$

where $a_{t,j}$ is the weight for h_j at time step t:

$$a_{t,j} = \frac{\exp(e_{t,j})}{\sum_{k=1}^{T} \exp(e_{t,k})} \tag{4}$$

where $e_{t,j}$ is attention score. The larger $e_{t,j}$ is, the larger $a_{t,j}$ becomes and the more h_j contributes to c_t and y_t. For sequence labeling task, y_t is mainly decided by encoding layer h_t. In order to make full use of its information, h_t is directly used for prediction and $e_{t,t}$ should be set to 0. Since encoding layers which are far from current time step may be noisy, we pre-define a window size wn and directly set $e_{t,j}$ to 0 when j is outside the window. To summarize, $e_{t,j}$ is calculated as:

$$e_{t,j} = \begin{cases} score(\widetilde{h}_t, h_j) & t - wn < j < t + wn \text{ and } j \neq t \\ 0 & \text{else} \end{cases} \tag{5}$$

the input of the *score* function is the current time step's decoding layer $\widetilde{h}_t$ and h_j. As proposed in [11], there are mainly two types of *score* functions which can be used:

$$score(\widetilde{h}_t, h_j) = \begin{cases} \widetilde{h}_t^T W_e h_j & general \\ V_e \tanh(\widetilde{W}_e \widetilde{h}_t + W_e h_j) & concat \end{cases} \tag{6}$$

Finally, the calculation of the output layer y_t in Eq. 1 is defined as:

$$y_t = softmax(V_h h_t + V_c c_t) \tag{7}$$

encoding hidden layer h_t is most informative for predicting y_t and is explicitly leveraged with attention component c_t.

Since our attention architecture does not change the way of computing hidden layers, it can be directly built upon sRNN and LSTM. When ARNN-SL is built upon bi-directional RNNs, two sets of weights and variables are used for forward and backward directions respectively. For example, two attention components c_t^f and c_t^b selectively focus on h_j^f and h_j^b. They are then concatenated to form a new attention component c_t for the final prediction.

3 Related Work

Recently, attention mechanism leads to state-of-the-art results on many complex tasks such as machine translation [1,11], image caption task [19], and speech recognition [2,5]. However, due to the characteristics such as align strategy, directly applying the same mechanism to sequence labeling is not feasible.

The main difference of our specially designed ARNN-SL with attention architectures in other tasks is the way of calculating the attention component c_t, as shown in Fig. 2. In machine translation [1], a translated word could be aligned with a word at any position of the sentence. Attention architecture should selectively focus on hidden layers at every position. In image caption task [19], in order to generate current caption word, hidden layers corresponding to all image segments should be focused on for the same reason.

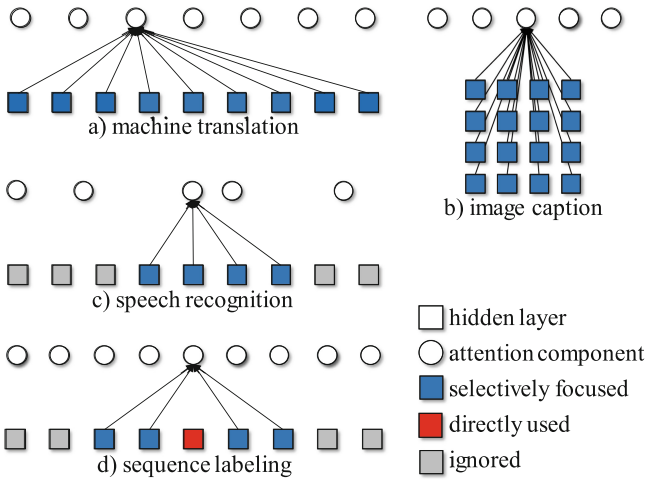

Fig. 2. Attention architectures used in different tasks.

The most similar attention architecture to ARNN-SL is used in speech recognition [2,5]. Instead of using all hidden layers, only the layers corresponding to the most probable consecutive k acoustic frames are focused on. The same idea is also implemented in machine translation [11], but it performs worse than attention for all layers. Compared with these architectures, ARNN-SL does not dynamically decide which consecutive hidden layers should be used for attention (Eq. 5). The useful hidden layers are always close to the current position for sequence labeling.

For sequence labeling, the label is mostly affected by the word and hidden layer in current time step. To the best of our knowledge, ARNN-SL is the first attention architecture that explicitly leverages the contribution from the current hidden layer and attention component (Eq. 7).

4 Experiments

4.1 Datasets and Experimental Setup

ARNN-SL is evaluated on four commonly used tasks for sequence labeling: Part-Of-Speech Tagging (POS), Chunking[2], Named Entity Recognition (NER)[3], and Slot Filling for Spoken Language Understanding (SF-SLU) [6,17,18]. Since there is no pre-defined development data for the datasets of Chunking and SF-SLU tasks, we randomly choose 20% of training data for validation. AdaDelta is used to control learning rate [21]. We use the same dropout strategy as that in [20] and the dropout rate is set to 0.5. Word embedding size and hidden layer size

[2] CoNLL 2000 shared task: http://www.cnts.ua.ac.be/conll2000/chunking.
[3] CoNLL 2003 shared task: http://www.cnts.ua.ac.be/conll2003/ner.

are set to 500. Word embeddings are either randomly initialized or pretrained using Word2Vec toolkit[4] on English Wikipedia (August 2013 dump). Models are trained for 25 epochs, and we report the results on epoch which achieves the highest performance on development data. We do not use features which are derived from lexical resources or other NLP systems. The only pre-processing we use is lowercasing.

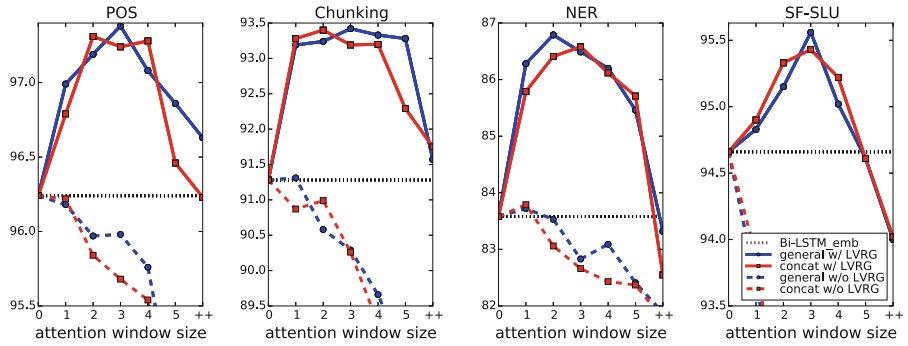

Fig. 3. Illustration of the impact of attention window sizes and score functions. ARNN-SL is built upon Bi-LSTM. Pretrained word embeddings are used. **Attention window size**: "0" indicates that attention architecture is not used, "++" indicates that windowing technique is not used and attention selectively focuses on all encoding layers. **w/o LVRG**: (dotted line) traditional attention mechanism. **w/LVRG**: (solid line) ARNN-SL which explicitly leverage the contribution from the current hidden layer and attention component (Eq. 7).

4.2 Main Results

As shown in Fig. 3, compared to traditional attention mechanism (w/o LVRG), models that explicitly leverage the information from the aligned and attention-focused hidden layers (w/LVRG) perform consistently better. In most cases, models without LVRG actually perform worse than Bi-LSTM baselines, especially on SF-SLU task and when the window size is large. The weighted sum of all encoding layers (Eq. 3) is likely to bring noises. The current encoding layer should always be directly used for prediction.

Our proposed attention architecture consistently improves the performance of Bi-LSTM on all tasks. The best performance is usually achieved at window size 2 or 3. The performance drops when the window size is bigger than 3 and reaches the minimum when attention focuses on all encoding layers. Windowing technique is indispensable for the good performance on sequence labeling.

The trend of the curve for the *concat* score function is similar to that of *general*. However, *general* score function often performs better than *concat*.

[4] http://code.google.com/p/word2vec/.

We highly recommend using *general* score function in practice, since it's also easier to implement and is 2–3 times faster.

Overall, ARNN-SL obtains 1.14%, 2.14%, 3.21% and 0.90% improvement on POS, Chunking, NER and SF-SLU respectively compared to sophisticated bi-direction Bi-LSTM baselines.

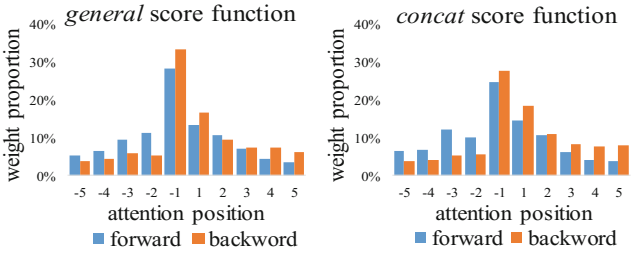

Fig. 4. Illustration of which position of encoding layers ARNN-SL focus on. ARNN-SL is built upon Bi-LSTM and is tested on SF-SLU task. For each window position j, we sum up variable $a_{t,j}$ in all training examples at all time steps. Each bar represents the percentage of this summed value over corresponding positions.

4.3 Attention Visualization

Since attention window size is crucial for ARNN-SL's performance, it's worth to explore which position of encoding layers the attention really focused on. As shown in Fig. 4, *general* and *concat* score functions both tend to focus on positions near the center. This explains the performance's decline when attention window size is bigger than 3 in the above section. This also further strengthens our claim: encoding layers which are far from the current time steps are noisy.

Another interesting phenomenon from this figure is that the forward direction of ARNN-SL mainly focuses on both position -1 and position 1. While the backward direction mainly focuses only on position -1. This may caused by that backward RNN's hidden layers contains only future information, so hidden layer at position 1 has no information about word at position -1. While word at this position may contains most useful information.

5 Conclusion and Future Work

This paper presents a novel attention architecture called ARNN-SL, designed for sequence labeling. We demonstrate its effectiveness on POS, Chunking, NER, and SF-SLU tasks. More precisely, we conclude that for sequence labeling tasks: (1) it's crucial to explicitly leverage the contribution from the current hidden layer and attention component, (2) *general* score function is a better choice than *concat*, (3) using windowing technique to restrict the attention is indispensable and the window size should be small.

The aim of this paper is investigating the impact of the attention architecture. We keep our model as simple and reproducible as possible. Note that the state-of-the-art results on POS, Chunking and NER tasks are all obtained by models combination, extra resources and feature engineering [3,8,12]. In the future, it's promising to implement ARNN-SL under the same sophisticated configurations for further improvements on these tasks (e.g. combining ARNN-SL and CRF/CNN, make using of different features and DBpedia knowledge).

Acknowledgments. This work is supported by the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China, National Natural Science Foundation of China with grant No. 61472428.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2014)
2. Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., Bengio, Y.: End-to-end attention-based large vocabulary speech recognition. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, pp. 4945–4949. IEEE (2016)
3. Chiu, J.P.C., Nichols, E.: Named entity recognition with bidirectional LSTM-CNNs. TACL **4**, 357–370 (2016)
4. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: EMNLP (2014)
5. Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., Bengio, Y.: Attention-based models for speech recognition. CoRR abs/1506.07503 (2015)
6. Hemphill, C.T., Godfrey, J.J., Doddington, G.R.: The ATIS spoken language systems pilot corpus. In: DARPA Speech and Natural Language Workshop, pp. 96–101 (1990)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
8. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
9. Kudo, T., Matsumoto, Y.: Chunking with support vector machines. In: NAACL, pp. 1–8. ACL (2001)
10. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: ICML, pp. 282–289 (2001)
11. Luong, T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: EMNLP (2015)
12. Ma, X., Hovy, E.H.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In: ACL, pp. 147–155. ACL (2016)
13. Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., et al.: Using recurrent neural networks for slot filling in spoken language understanding. IEEE/ACM Trans. Audio Speech Lang. Process. **23**(3), 530–539 (2015)
14. Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: ACL, pp. 147–155. ACL (2009)

15. Raymond, C., Riccardi, G.: Generative and discriminative algorithms for spoken language understanding. In: INTERSPEECH, pp. 1605–1608 (2007)
16. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**(11), 2673–2681 (1997)
17. Tur, G., Hakkani-Tur, D., Heck, L.: What is left to be understood in ATIS? In: Spoken Language Technology Workshop, pp. 19–24. IEEE (2010)
18. Wang, Y.Y., Acero, A., Mahajan, M., Lee, J.: Combining statistical and knowledge-based spoken language understanding in conditional models. In: COLING/ACL, pp. 882–889. ACL (2006)
19. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A.C., Salakhutdinov, R., Zemel, R.S., Bengio, Y.: Show, attend and tell: neural image caption generation with visual attention. In: ICML (2015)
20. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. CoRR abs/1409.2329 (2014)
21. Zeiler, M.D.: Adadelta: an adaptive learning rate method. CoRR abs/1212.5701 (2012)