Deep Learning for Chinese Word Segmentation and POS Tagging

Xiaoqing Zheng, Hanyang Chen and Tianyu Xu, EMNLP 2013 [PDF]
Summarized by Zhengyuan Liu, Master Student of Computer Science, UCLA
UID: 604945064, Email: zyliu@cs.ucla.edu

Introduction and Related Work

Over the past few decades, statistical approaches are dominant for Chinese word segmentation (CWS). Statistical methods usually treat CWS as a sequence labeling problem where each character is assigned a label indicating its position in a word, which can be handled with structured machine learning algorithms such as Conditional Random Fields (CRF). However, traditional statistical approaches usually involve a great number of features, which leads to several limitations. For example, the number of features could be too large for practical use and the trained models are prone to overfit on training corpus. Moreover, the effectiveness of features is so critical that feature selection becomes a major factor for the performance of these systems. Since feature selection is mainly based on human ingenuity and linguistic intuition and then trial and error, much time and labor is devoted to task-specific feature engineering.

In contrast, neural network models are designed to avoid task-specific feature engineering, which first proposed by (Bengio et al., 2003) for a probabilistic language model, and reintroduced by (Collobert et al., 2011) for multiple NLP tasks. Following above work, this paper first introduced deep learning to CWS and POS tagging, and used multilayer neural network to extract features from input sentences and then conduct word segmentation and POS tagging. This paper used two methods to train the neural network, i.e. sentence-level log-likelihood and perceptron-style training algorithm. Two kinds of character representations are used, i.e. vectors of random values and character embeddings trained by large unlabeled data. This paper achieved close to state-of-the-art performance with less computational cost by using pre-trained character embeddings and perceptron-style training algorithm.

The Neural Network Architecture

The neural network architecture proposed in this paper consists multiple layers including feature extraction for each Chinese character and a window of characters, classical neural network layers and a tag inference process based on Viterbi algorithm.

The first layer maps Chinese characters to feature vectors by a lookup operation of a character embedding matrix $\mathcal{M} \in \mathbb{R}^{d \times |D|}$, where d is the dimension of the vector (a hyper-parameter) and $|D|$ is size of the dictionary of all characters from the training corpus (Figure 1). These feature vectors are initialized with random values and can be automatically trained by back propagation algorithm, or they can be pre-trained by large unlabeled dataset.

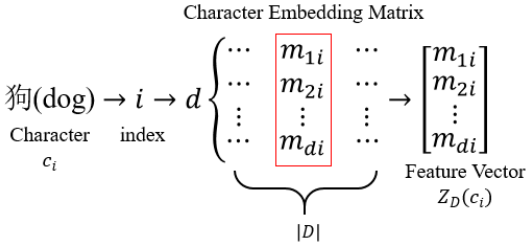

Figure 1: Mapping characters to feature vectors

In order to handle variable length of input sentence, this paper use a window approach to extract information from a character's neighboring characters. The feature vectors of the central character c_i and its neighboring characters within the window of size w (a hyper-parameter) are concatenated to form the feature vector of the window, which consist the input of the next neural network layer $f_\theta^1(c_i)$:

$$f_\theta^1(c_i) = \begin{pmatrix} Z_D(c_{i-w/2}) \\ \vdots \\ Z_D(c_i) \\ \vdots \\ Z_D(c_{i+w/2}) \end{pmatrix} \tag{1}$$

The characters with indices exceeding the sentence boundaries are mapped to one of two special symbols, namely 'start' and 'stop' symbols.

The following layers are classical neural network layers (Figure 2), which consists of two linear transform f_θ^2, f_θ^3 and one non-linear transform $g(\cdot)$ (a sigmoid function), mapping input feature vector of the window to a vector of scores for all tags in the tag set T:

$$f_\theta(c_i) = f_\theta^3(g\left(f_\theta^2 f_\theta^1(c_i)\right)) = W_3 g(W_2 f_\theta^1(c_i) + b_2) + b_3 \tag{2}$$

where $W_2 \in \mathbb{R}^{H \times wd}, b_2 \in \mathbb{R}^H, W_3 \in \mathbb{R}^{|T| \times H}, b_3 \in \mathbb{R}^{|T|}$ are the parameters to be trained. The hyper-parameter H is actually the number of hidden units.

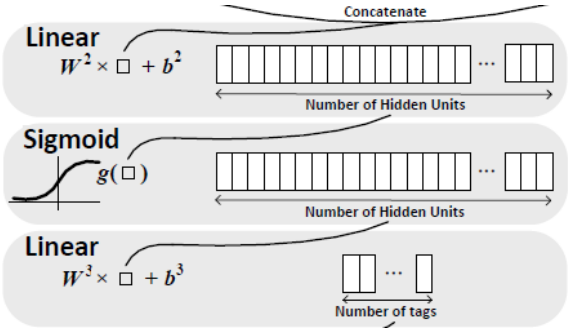

Figure 2: Classical neural network layers

Since we have got the scores for every word with all tags, the next step is tag inference based on Viterbi algorithm (Figure 3). Besides the network output scores for all characters in a sentence, this paper introduced a transition score A_{ij} for a transition from i-th tag to j-th tag in successive characters, and an initial scores A_{0i} for a sentence starting with the i-th tag. Therefore, the score of a sentence $c_{[1:n]}$ with a path of tags $t_{[1:n]}$ is given by the sum of transition scores and network scores:

$$s\left(c_{[1:n]}, t_{[1:n]}, \theta\right) = \sum_{i=1}^{n} (A_{t_{i-1}t_i} + f_\theta(t_i|i)) \tag{3}$$

The inference of CWS and POS tagging is finding the best tag path by maximizing the sentence score, and Viterbi algorithm is used for this tag inference.

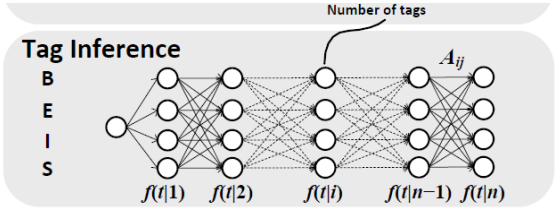

Figure 3: Tag Inference

The training process is to determine all the parameters of this network $\theta =$

$(\mathcal{M}, W_2, b_2, W_3, b_3, A)$ by training data. Two methods are used to train the neural network, first is sentence-level log-likelihood by maximizing likelihood of true path *t* for all sentences in the training set with respect to θ:

$$\theta = \arg\max_{\theta} \sum \log p(t|c, \theta) \tag{4}$$

The conditional probability of true path *t* is given by:

$$p(t|c, \theta) = \frac{\exp\{s(c,t,\theta)\}}{\sum_{\forall t'} \exp\{s(c,t',\theta)\}} \tag{5}$$

Classical gradient ascent and back propagation algorithm are used to adapt the network parameters until the character embedding layer and maximize the log-likelihood:

$$\theta \leftarrow \theta + \lambda \frac{\partial \log p(t|c, \theta)}{\partial \theta} \tag{6}$$

where λ is learning rate (a hyper-parameter).

The second method is a perceptron-style training algorithm, which is proposed by this paper. We define $L_\theta(t, t'|c)$ as the difference between the score of the true tag path *t* for a sentence *c* and the output path *t'* (i.e. the highest scoring path under current network parameters θ):

$$L_\theta(t, t'|c) = s(c, t, \theta) - \max_{t'} s(c, t', \theta) \tag{7}$$

If $t \neq t'$, then we need to alter network parameters in order to maximize $L_\theta(t, t'|c)$, i.e. increase the score of t ($f_\theta(t_i|i)$ and $A_{t_{i-1}t_i}$), and decrease the score of t' ($f_\theta(t'_i|i)$ and $A_{t_{i-1}t_i}$). Intuitively, we can realize this adjustment by altering the derivatives since derivatives determine the update of parameters. For every character c_i where $t_i \neq t'_i$ we set:

$$\frac{\partial L_\theta(t, t'|c)}{\partial f_\theta(t_i|i)} + +, \frac{\partial L_\theta(t, t'|c)}{\partial f_\theta(t'_i|i)} - -, \frac{\partial L_\theta(t, t'|c)}{\partial A_{t_{i-1}t_i}} + +, \frac{\partial L_\theta(t, t'|c)}{\partial A_{t'_{i-1}t'_i}} - - \tag{8}$$

and then update parameters of the network by gradient ascent and back propagation algorithm with the gradients computed by (8).

Experiments

Two NLP tasks are addressed in this paper: Chinese word segmentation (SEG) and joint word segmentation and POS tagging (JWP). For SEG task, the boundary tag set is {B (begin), I (inside), E (end), S (single)}. The label fashion of JWP task directly expands from SEG labels, for example, verb phrases can be labeled by four different tags: 'B_VP', 'I_VP', 'E_VP' and 'S_VP', which easily expands the framework for SEG to JWP task. Both sentence-level log-likelihood (SSL) and perceptron style training algorithm (PSA) are implemented in Java to train the neural network. In experiments PSA speeds up the training process with negligible loss in performance and can to be implemented easier.

Three sets of experiments are conducted. The first experiment is the selection of hyper-parameters (Table 1). This experiment was ran on part of Chinese Treebank 4 (CTB-4).

Table 1: Hyper-parameters of the network

Hyper-parameter	**Notation**	**Value**
dimension of character feature vector	*d*	50
window size	*w*	5
Number of hidden la	*H*	300
learning rate	λ	0.02

The second experiment is a closed test without any extra knowledge on Chinese Treebank (CTB) data set from Bakeoff-3 for both SEG and JWP tasks. The aim of this experiment is to compare this neural model with other models in the literature (Results in Table 2).

Table 2: Comparison of F-scores

Approach		F_{word}	R_{oov}	F_{pos}
SEG	(Zhao et al., 2006)	93.30	70.70	–
	(Wang et al., 2006)	93.00	68.30	–
	(Zhu et al., 2006)	92.70	63.40	–
	(Zhang et al., 2006)	92.60	61.70	–
	(Feng et al., 2006)	91.70	68.00	–
	PSA	92.59	64.24	–
	PSA + LM	94.57	70.12	–
JWP	(Ng and Lou, 2004)	95.20	–	–
	(Zhang and Clark, 2008)	95.90	–	91.34
	(Jiang et al., 2008)	97.30	–	92.50
	(Kruengkrai et al., 2009)	96.11	–	90.85
	PSA	93.83	68.21	90.79
	PSA + LM	95.23	72.38	91.82

The third experiment uses Sina news corpus (a large unlabeled data set) to train a language model and obtain character embeddings carrying more syntactic and semantic information. These embeddings are used to initialize the character embedding matrix, and the matrix will not be modified at the supervised training stage. The combination of language model improved the performance of the neural model and achieved close to state-of-the-art results (Table 2).

Conclusion and Future Work

This paper applied deep neural network to Chinese word segmentation and POS tagging task, and two methods were used to train the network, i.e. maximum-likelihood and perceptron-style algorithm. This model achieved close to state-of-the-art performance by using character representations pre-trained by large unlabeled data set. Three potential ways may be used to further improve the performance: specific linguistic features, common techniques such as cascading or voting and ad-hoc network architecture for tasks of interest.

Reference

Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. Journal of machine learning research, 2003, 3(Feb): 1137-1155.

Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research, 2011, 12(Aug): 2493-2537.

Zheng X, Chen H, Xu T. Deep Learning for Chinese Word Segmentation and POS Tagging[C] //EMNLP. 2013: 647-657.