

Supplementary Material of “Dependency Grammar Induction with Neural Lexicalization and Big Training Data”^{*}

Wenjuan Han, Yong Jiang and Kewei Tu

{hanwj, jiangyong, tukw}@shanghaitech.edu.cn

School of Information Science and Technology
ShanghaiTech University, Shanghai, China

1 Learning Algorithm of L-NDMV

The objective function of learning L-NDMV is the log-likelihood $L(\Theta)$ of the training sentences $X = \{x_1, x_2, \dots, x_n\}$, where the parameters of the neural network are denoted by Θ .

$$L(\Theta) = \sum_{\alpha=1}^n \log P(x_\alpha; \Theta)$$

This objective function can be optimized using the hard expectation-maximization algorithm. The hard E-Step uses the current model parameterized by Θ to parse each training sentence x_i and then calculates the counts of all the grammar rules in the best parse of x_i , represented by $e_c(x_i)$ for the CHILD rule c , $e_d(x_i)$ for the DECISION rule d and $e_r(x_i)$ for the ROOT rule r respectively. Let $\phi(X)$ be the set of counts of all the grammar rules used in the best parses of all the training sentences in X . The M-Step optimizes the following objective function:

$$\begin{aligned} l(\Theta; \phi(X)) = & \sum_{i=1}^n \left(\sum_c e_c(x_i) \log p_c \right. \\ & + \sum_d e_d(x_i) \log p_d \\ & \left. + \sum_r e_r(x_i) \log p_r \right) \end{aligned}$$

This learning method when applied to L-NDMV with a large training corpus is very time-consuming. We propose two improvements to the learning method to achieve significant speedup. First, at each E-step we calculate the grammar rule counts from a different batch of sentences instead of from the whole training corpus and then train the neural network using only these counts. Second, unlike in the NDMV approach, we no longer

^{*}This work was supported by the National Natural Science Foundation of China (61503248).

Algorithm 1 Learning L-NDMV

- 1: **Input:** Corpus X
 - 2: Initialize Θ
 - 3: **repeat**
 - 4: Shuffle the sentences in X
 - 5: Sample a mini-batch \tilde{X} from X
 - 6: Run the CYK algorithm to calculate $\phi(\tilde{X})$ using the current model parameterized by Θ
 - 7: Update Θ to maximize $l(\Theta; \phi(\tilde{X}))$ using momentum based mini-batch gradient descent
 - 8: **until** Convergence
-

train a new neural network from scratch at each M-step but instead train the same network network across EM iterations. Together, these two improvements speed up training by more than 50 times with almost no drop in learning accuracy. Algorithm 1 outlines our learning algorithm of L-NDMV.

Our algorithm resembles online EM (Liang and Klein, 2009) except that at each EM iteration we utilize counts collected from only the current batch of training sentences. Because each count of grammar rule usage becomes a training sample for the neural network, training the neural network with counts collected from all the sentences, as in standard online EM, would beat the purpose of online learning and significantly slow down training.

2 Visualization of the Learned Word Vectors

To analyze what information is learned to be encoded in the input continuous representation of head words, we used the t-SNE algorithm (Van der Maaten and Hinton, 2008) to map the word vectors learned by our approach onto a 2D plane in Figure

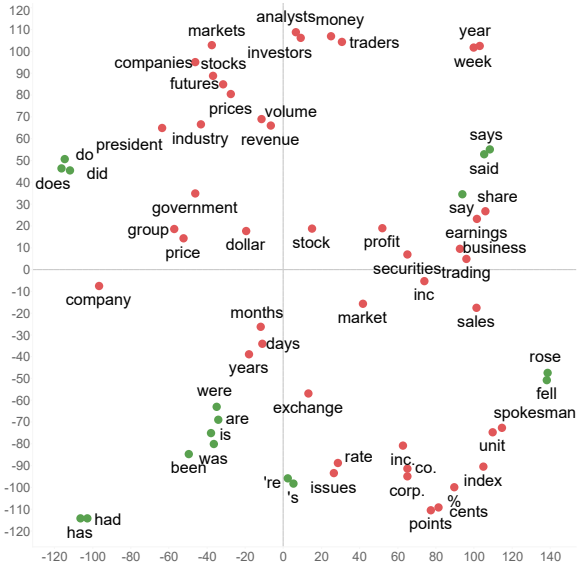


Figure 1: Visualization of the learned word vectors. Nouns are represented by small red dots and verbs are represented by small green dots.

1. In order to avoid interference of the word vector initialization method, we used random initialization instead. Interestingly, words that are close to each other in the figure (and hence have similar word vectors) are not only syntactically similar, but also semantically similar. For instance, three noun abbreviations “inc.,” “co.” and “corp.” have very similar coordinates in the figure. This is because semantically similar words typically have similar contexts in the training corpus and such similarity can be captured by our grammar induction approach. Lexical semantics, as captured here in word vectors, is beneficial to syntactic parsing when there are rare words or syntactic ambiguities (Le and Zuidema, 2015). For example, if from the training data it is learned that a dependency between “in” and “corp.” is very likely, then even if “in” and “co.” never co-occur in the training data, our approach would still learn the dependency relation between them.

3 The Dependency Rules of Chinese

When using the grammar induction method of Naseem et al. (2010) to initialize our model as described in section 2.3, we need to specify a set of prior dependency rules. The dependency rules employed by Naseem et al. include a set of universal dependency rules as well as a few English-specific rules. We replace the English-specific rules with a few Chinese-specific rules designed by ourselves.

Universal Dependency Rules	
Root \rightarrow Verb	Adjective \rightarrow Adverb
Verb \rightarrow Noun	Noun \rightarrow Adjective
Verb \rightarrow Pronoun	Noun \rightarrow Demonstrative
Verb \rightarrow Verb	Noun \rightarrow Noun
Adposition \rightarrow Noun	
Chinese-Specified Rules	
Root \rightarrow Adjective	Dec \rightarrow Verb
Noun \rightarrow Dec	Deg \rightarrow Noun
Noun \rightarrow Deg	Adjective \rightarrow Noun

Table 1: The prior dependency rules used in our experiments for Chinese.

In addition, we remove the universal rules that involve tags not used in the Chinese dataset. Table 1 shows the prior dependency rules that we use for Chinese.

4 Details of Experimental Setup

We learn the L-DMV and L-NDMV with the child valence and decision valence both set to 2. For the neural network of L-NDMV, we set the dimension of input and output word vectors to 100 and the dimension of input and output tag vectors to 20. For each degree of lexicalization, we tuned the dimension of the hidden layer of the neural network on the validation dataset, with five candidate dimensions: 18, 23, 28, 33, and 38.

References

Phong Le and Willem Zuidema. 2015. Unsupervised dependency parsing: Let’s use supervised parsers. *arXiv preprint arXiv:1504.04666*.

Percy Liang and Dan Klein. 2009. [Online em for unsupervised models](#). In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL ’09, pages 611–619, Stroudsburg, PA, USA. Association for Computational Linguistics.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85.

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1244. Association for Computational Linguistics.