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

Abstract

We study the impact of big models (in terms of the degree of lexicalization) and big data (in terms of the training corpus size) on dependency grammar induction. We experimented with L-DMV, a lexicalized version of Dependency Model with Valence (Klein and Manning, 2004) and L-NDMV, our lexicalized extension of the Neural Dependency Model with Valence (Jiang et al., 2016). We find that L-DMV only benefits from very small degrees of lexicalization and moderate sizes of training corpora. L-NDMV can benefit from big training data and lexicalization of greater degrees, especially when enhanced with good model initialization, and it achieves a result that is competitive with the current state-of-the-art.

1 Introduction

Grammar induction is the task of learning a grammar from a set of unannotated sentences. In the most common setting, the grammar is unlexicalized with POS tags being the tokens, and the training data is the WSJ10 corpus (the Wall Street Journal corpus with sentences no longer than 10 words) containing no more than 6,000 training sentences (Cohen et al., 2008; Berg-Kirkpatrick et al., 2010; Tu and Honavar, 2012).

Lexicalized grammar induction aims to incorporate lexical information into the learned grammar to increase its representational power and improve the learning accuracy. The most straightforward approach to encoding lexical information is full lexicalization (Pate and Johnson, 2016; Spitkovsky et al., 2013). A major problem with full lexicalization is that the grammar becomes much larger and thus learning is more data demanding. To mitigate this problem, Headden et al. (2009) and Blunsom and Cohn (2010) used partial lexicalization in which infrequent words are replaced by special symbols or their POS tags. Another straightforward way to mitigate the data scarcity problem of lexicalization is to use training corpora larger than the standard WSJ corpus. For example, Pate and Johnson (2016) used two large corpora containing more than 700k sentences; Marecek and Straka (2013) utilized a very large corpus based on Wikipedia in learning an unlexicalized dependency grammar. Finally, smoothing techniques can be used to reduce the negative impact of data scarcity. One example is Neural DMV (NDMV) (Jiang et al., 2016) which incorporates neural networks into DMV and can automatically smooth correlated grammar rule probabilities.

Inspired by this background, we conduct a systematic study regarding the impact of the degree of lexicalization and the training data size on the accuracy of grammar induction approaches. We experimented with a lexicalized version of Dependency Model with Valence (L-DMV) (Klein and Manning, 2004) and our lexicalized extension of NDMV (L-NDMV). We find that L-DMV only benefits from very small degrees of lexicalization and moderate sizes of training corpora. In comparison, L-NDMV can benefit from big training data and lexicalization of greater degrees, especially when it is enhanced with good model initialization. The performance of L-NDMV is competitive with the current state-of-the-art.

2 Methods

2.1 Lexicalized DMV

We choose to lexicalize an extended version of DMV (Gillenwater et al., 2010). We adopt a sim-

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

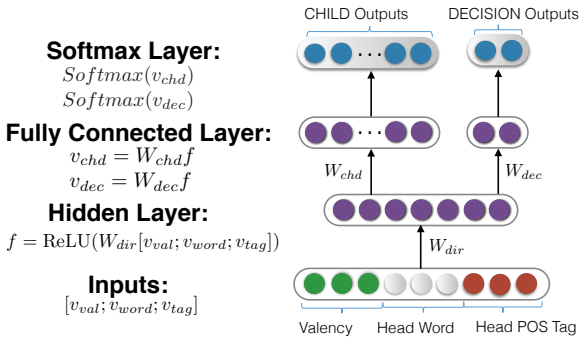


Figure 1: The structure of the neural networks in the L-NDMV model. It predicts the probabilities of the CHILD rules and DECISION rules.

ilar approach to that of Spitkovsky et al. (2013) and Blunsom and Cohn (2010) and represent each token as a word/POS pair. If a pair appears infrequently in the corpus, we simply ignore the word and represent it only with the POS tag. We control the degree of lexicalization by replacing words that appear less than a cutoff number in the WSJ10 corpus with their POS tags. With a very large cutoff number, the grammar is virtually unlexicalized; but when the cutoff number becomes smaller, the grammar becomes closer to be fully lexicalized. Note that our method is different from previous practice that simply replaces rare words with a special "unknown" symbol (Headden III et al., 2009). Using POS tags instead of the "unknown" symbol to represent rare words can be helpful in the neural approach introduced below in that the learned word vectors are more informative.

2.2 Lexicalized NDMV

With a larger degree of lexicalization, the grammar contains more tokens and hence more parameters (i.e., grammar rule probabilities), which require more data for accurate learning. Smoothing is a useful technique to reduce the demand for data in this case. Here we employ a neural approach to smoothing. Specifically, we propose a lexicalized extension of neural DMV (Jiang et al., 2016) and we call the resulting approach L-NDMV.

Extended Model: The model structure of L-NDMV is similar to that of NDMV except for the representations of the head and the child of the CHILD and DECISION rules. The network structure for predicting the probabilities of CHILD rules $[p_{c_1}, p_{c_2}, ..., p_{c_m}]$ (m is the vocabulary size; c_i is the i-th token) and DECISION rules $[p_{stop}, p_{continue}]$ given the head word, head POS tag, direction and valence is shown in Figure 1. We denote the input continuous representations of the head word, head POS tag and valence by v_{word}, v_{tag} and v_{val} respectively. By concatenating these vectors we get the input representation to the neural network: $[v_{val}; v_{word}; v_{tag}]$. We map the input representation to the hidden layer f using the direction-specific weight matrix W_{dir} and the ReLU activation function. We represent all the child tokens with matrix $W_{chd} = [W_{word}, W_{tag}]$ which contains two parts: child word matrix $W_{word} \in \mathbb{R}^{m \times k}$ and child POS tag matrix $W_{tag} \in \mathbb{R}^{m \times k'}$, where k and k' are the prespecified dimensions of output word vectors and tag vectors respectively. The i-th rows of W_{word} and W_{tag} represent the output continuous representations of the i-th word and its POS tag respectively. Note that for two words with the same POS tag, the corresponding POS tag representations are the same. We take the product of f and the child matrix W_{chd} and apply a softmax function to obtain the CHILD rule probabilities. For DECISION rules, we replace W_{chd} with the decision weight matrix W_{dec} and follow the same procedure.

Extended Learning Algorithm: The original NDMV learning method is based on hard-EM and is very time-consuming when applied to L-NDMV with a large training corpus. We propose two improvements to achieve significant speedup. First, at each EM iteration we collect grammar rule counts from a different batch of sentences instead of from the whole training corpus and train the neural network using only these counts. Second, we train the same neural network across EM iterations without resetting. More details can be found in the supplementary material. Our algorithm can be seen as an extension of online EM (Liang and Klein, 2009) to accommodate neural network training.

2.3 Model Initialization

It was previously shown that the heuristic KM initialization method by Klein and Manning (2004) does not work well for lexicalized grammar induction (Headden III et al., 2009; Pate and Johnson, 2016) and it is very helpful to initialize learning with a model learned by a different grammar induction method (Le and Zuidema, 2015; Jiang et al., 2016). We tested both KM initialization and the following initialization method: we first learn

an unlexicalized DMV using the grammar induction method of Naseem et al. (2010) and use it to parse the training corpus; then, from the parse trees we run maximum likelihood estimation to produce the initial lexicalized model.

3 Experimental Setup

For English, we used the BLLIP corpus[1] in addition to the regular WSJ corpus in our experiments. Note that the BLLIP corpus is collected from the same news article source as the WSJ corpus, so it is in-domain and is ideal for training grammars to be evaluated on the WSJ test set. In order to solve the compatibility issue as well as improve the POS tagging accuracy, we used the Stanford tagger (Toutanova et al., 2003) to retag the BLLIP corpus and selected the sentences for which the new tags are consistent with the original tags, which resulted in 182244 sentences with length less than or equal to 10 after removing punctuations. We used this subset of BLLIP and section 2-21 of WSJ10 for training, section 22 of WSJ for validation and section 23 of WSJ for testing. We used training sets of four different sizes: WSJ10 only (5779 sentences) and 20k, 50k, and all sentences from the BLLIP subset. For Chinese, we obtained 4762 sentences for training from Chinese Treebank 6.0 (CTB) after converting data to dependency structures via Penn2Malt (Nivre, 2006) and then stripping off punctuations. We used the recommended validation and test data split described in the documentation.

We trained the models with different degrees of lexicalization. We control the degree of lexicalization by replacing words that appear less than a cutoff number in the WSJ10 or CTB corpus with their POS tags. For each degree of lexicalization, we tuned the dimension of the hidden layer of the neural network on the validation dataset. For English, we tested nine word cutoff numbers: 100000, 500, 200, 100, 80, 70, 60, 50, and 40, which resulted in vocabulary sizes of 35, 63, 98, 166, 203, 226, 267, 306, and 390 respectively; for Chinese, the word cutoff numbers are 100000, 100, 70, 50, 40, 30, 20, 12, and 10. Ideally, with higher degrees of lexicalization, the hidden layer dimension should be larger in order to accommodate the increased number of tokens. For the neural network of L-NDMV, we initialized the word and tag vectors in the neural network by learning a CBOW model using the Gensim package (Řehůřek and Sojka, 2010). We set the dimension of input and output word vectors to 100 and the dimension of input and output tag vectors to 20. We trained the neural network with learning rate 0.03, mini-batch size 200 and momentum 0.9. Because some of the neural network weights are randomly initialized, the model converges to a different local minimum in each run of the learning algorithm. Therefore, for each setup we ran our learning algorithm for three times and reported the average accuracy. More detail of the experimental setup can be found in the supplementary material.

[1] Brown Laboratory for Linguistic Information Processing (BLLIP) 1987-89 WSJ Corpus Release 1

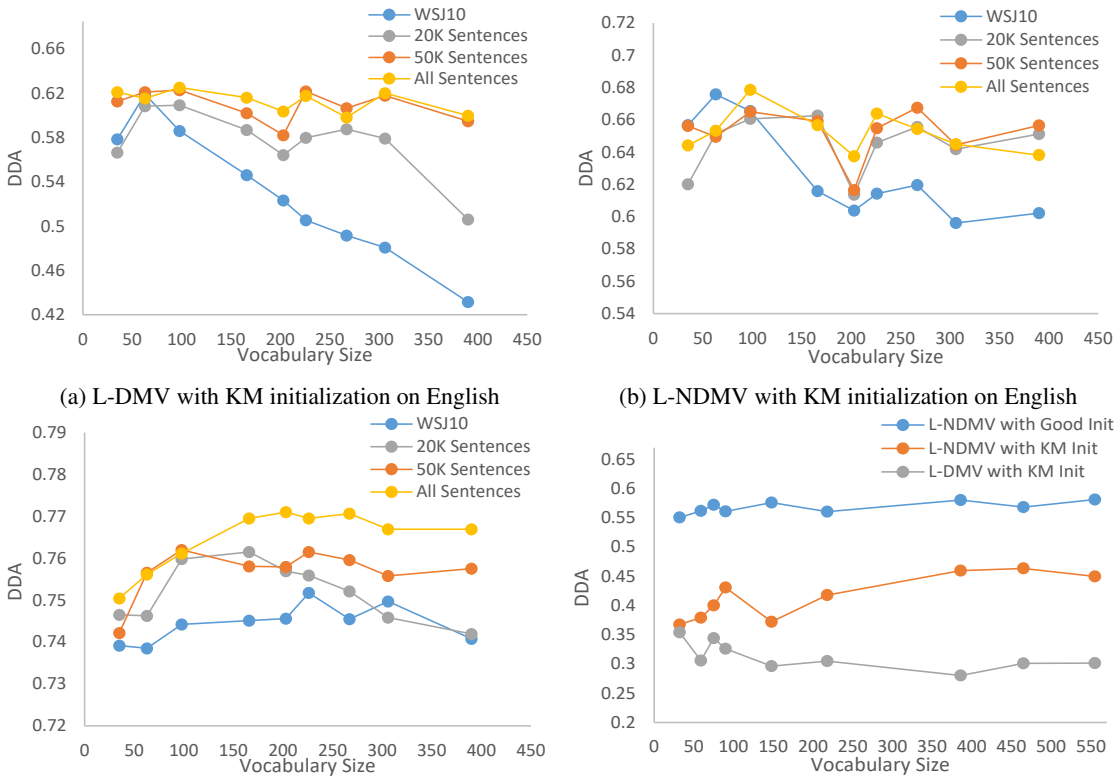
4 Experimental Results

4.1 Results on English

Figure 2(a) shows the directed dependency accuracy (DDA) of the learned lexicalized DMV with KM initialization. It can be seen that on the smallest WSJ10 training corpus, lexicalization improves learning only when the degree of lexicalization is small; with further lexicalization, the learning accuracy significantly degrades. On the three larger training corpora, the impact of lexicalization on the learning accuracy is still negative but is less severe. Overall, lexicalization seems to be very data demanding and even our largest training corpora could not bring about the benefit of lexicalization. Increasing the training corpus size is helpful regardless of the degree of lexicalization, but the learning accuracies with the 50K dataset are almost identical to those with the full dataset, suggesting diminishing return of more data.

Figure 2(b) shows the results of L-NDMV with KM initialization. The parsing accuracy is improved under all the settings, showing the advantage of NDMV. The range of lexicalization degrees that improve learning becomes larger, and the degradation in accuracy with large degrees of lexicalization becomes much less severe. Diminishing return of big data as seen in the first figure can still be observed.

Figure 2(c) shows the results of L-NDMV with the initialization method described in section 2.3. It can be seen that lexicalization becomes less data demanding and the learning accuracy does not decrease until the highest degrees of lexicalization. Larger training corpora now lead to significantly better learning accuracy and support lexicalization



(a) L-DMV with KM initialization on English

(b) L-NDMV with KM initialization on English

(c) L-NDMV with good initialization on English

(d) L-DMV and L-NDMV on Chinese

Figure 2: The impact of the training corpus size and the degree of lexicalization on L-DMV and L-NDMV with different initialization methods on English and Chinese.

of greater degrees than smaller corpora. Diminishing return of big data is no longer observed, which implies further increase in accuracy with even more data.

Table 1 compares the result of L-NDMV (with the largest corpus and the vocabulary size of 203 which was selected on the validation set) with previous approaches to dependency grammar induction. It can be seen that L-NDMV is competitive with previous state-of-the-art approaches. We did some further analysis of the learned word vectors in L-NDMV in the supplementary material.

4.2 Results on Chinese

Figure 2(d) shows the results of the three approaches on the Chinese treebank. Because the corpus is relatively small, we did not study the impact of the corpus size. Similar to the case of English, the accuracy of lexicalized DMV degrades with more lexicalization. However, the accuracy with L-NDMV increases significantly with more lexicalization even without good model initialization. Adding good initialization further boosts the performance of L-NDMV, but the benefit of lexicalization is less significant (from 0.55 to 0.58).

Methods	**WSJ10**	**WSJ**
Unlexicalized Approaches, with WSJ10		
EVG (Headden III et al., 2009)	65.0	-
TSG-DMV (Blunsom and Cohn, 2010)	65.9	53.1
PR-S (Gillenwater et al., 2010)	64.3	53.3
HDP-DEP (Naseem et al., 2010)	73.8	-
UR-A E-DMV (Tu and Honavar, 2012)	71.4	57.0
Neural E-DMV(Jiang et al., 2016)	72.5	57.6
Systems Using Lexical Information and/or More Data		
LexTSG-DMV (Blunsom and Cohn, 2010)	67.7	55.7
L-EVG (Headden III et al., 2009)	68.8	-
CS (Spitkovsky et al., 2013)	72.0	64.4
MaxEnc (Le and Zuidema, 2015)	73.2	**65.8**
L-NDMV + WSJ	75.1	59.5
L-NDMV + Large Corpus	**77.2**	63.2

Table 1: Comparison of recent grammar induction systems.

5 Effect of Grammar Rule Probability Initialization

We compare four initialization methods to L-NDMV: uniform initialization, random initialization, KM initialization (Klein and Manning, 2004), and good initialization as described in section 2.3 in Figure 3. Here we trained the L-NDMV model on the WSJ10 corpus with the same experimental setup as in section 3.

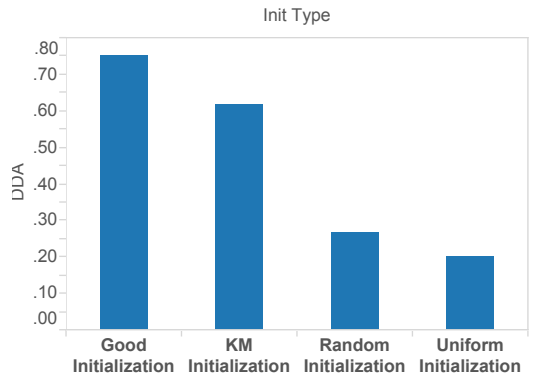


Figure 3: Comparison of four initialization methods to L-NDMV: uniform initialization, random initialization, KM initialization and good initialization.

Again, we find that good initialization leads to better performance than KM initialization, and both good initialization and KM initialization are significantly better than random and uniform initialization. Note that our results are different from those by Pate and Johnson (2016), who found that uniform initialization leads to similar performance to KM initialization. We speculate that it is because of the difference in the learning approaches (we use neural networks which may be more sensitive to initialization) and the training and test corpora (we use news articles while they use telephone scripts).

6 Conclusion and Future Work

We study the impact of the degree of lexicalization and the training data size on the accuracy of dependency grammar induction. We experimented with lexicalized DMV (L-DMV) and our lexicalized extension of Neural DMV (L-NDMV). We find that L-DMV only benefits from very small degrees of lexicalization and moderate sizes of training corpora. In contrast, L-NDMV can benefit from big training data and lexicalization of greater degrees, especially when enhanced with good model initialization, and it achieves a result that is competitive with the state-of-the-art.

In the future, we plan to study higher degrees of lexicalization or full lexicalization, as well as even larger training corpora (such as the Wikipedia corpus). We would also like to experiment with other grammar induction approaches with lexicalization and big training data.

References

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590. Association for Computational Linguistics.

Phil Blunsom and Trevor Cohn. 2010. Unsupervised induction of tree substitution grammars for dependency parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1204–1213. Association for Computational Linguistics.

Shay B Cohen, Kevin Gimpel, and Noah A Smith. 2008. Logistic normal priors for unsupervised probabilistic grammar induction. In *Advances in Neural Information Processing Systems*, pages 321–328.

Jennifer Gillenwater, Kuzman Ganchev, Joao Graça, Fernando Pereira, and Ben Taskar. 2010. Sparsity in dependency grammar induction. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 194–199. Association for Computational Linguistics.

William P Headden III, Mark Johnson, and David McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 101–109. Association for Computational Linguistics.

Yong Jiang, Wenjuan Han, and Kewei Tu. 2016. Unsupervised neural dependency parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 763–771, Austin, Texas. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

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.

David Marecek and Milan Straka. 2013. Stop-probability estimates computed on a large corpus improve unsupervised dependency parsing. In *ACL (1)*, pages 281–290.

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.

Joakim Nivre. 2006. *Inductive dependency parsing*. Springer.

John K Pate and Mark Johnson. 2016. Grammar induction from (lots of) words alone.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Valentin I Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2013. Breaking out of local optima with count transforms and model recombination: A study in grammar induction. In *EMNLP*, pages 1983–1995.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics.

Kewei Tu and Vasant Honavar. 2012. Unambiguity regularization for unsupervised learning of probabilistic grammars. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1324–1334. Association for Computational Linguistics.