# Unsupervised Neural Dependency Parsing*

**Yong Jiang, Wenjuan Han** and **Kewei Tu**
{jiangyong,hanwj,tukw}@shanghaitech.edu.cn
School of Information Science and Technology
ShanghaiTech University, Shanghai, China

## Abstract

Unsupervised dependency parsing aims to learn a dependency grammar from text annotated with only POS tags. Various features and inductive biases are often used to incorporate prior knowledge into learning. One useful type of prior information is that there exist correlations between the parameters of grammar rules involving different POS tags. Previous work employed manually designed features or special prior distributions to encode such information. In this paper, we propose a novel approach to unsupervised dependency parsing that uses a neural model to predict grammar rule probabilities based on distributed representation of POS tags. The distributed representation is automatically learned from data and captures the correlations between POS tags. Our experiments show that our approach outperforms previous approaches utilizing POS correlations and is competitive with recent state-of-the-art approaches on nine different languages.

## 1 Introduction

Unsupervised structured prediction from data is an important problem in natural language processing, with applications in grammar induction, POS tag induction, word alignment and so on. Because the training data is unannotated in unsupervised structured prediction, learning is very hard. In this paper, we focus on unsupervised dependency parsing, which aims to identify the dependency trees of sentences in an unsupervised manner.

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

Previous work on unsupervised dependency parsing is mainly based on the dependency model with valence (DMV) (Klein and Manning, 2004) and its extension (Headden III et al., 2009; Gillenwater et al., 2010). To effectively learn the DMV model for better parsing accuracy, a variety of inductive biases and handcrafted features have been proposed to incorporate prior information into learning. One useful type of prior information is that there exist correlations between the parameters of grammar rules involving different POS tags. Cohen and Smith (2009; 2010) employed special prior distributions to encourage learning of correlations between POS tags. Berg-Kirkpatrick et al. (2010) encoded the relations between POS tags using manually designed features.

In this work, we propose a neural based approach to unsupervised dependency parsing. We incorporate a neural model into the DMV model to predict grammar rule probabilities based on distributed representation of POS tags. We learn the neural network parameters as well as the distributed representations from data using the expectation-maximization algorithm. The correlations between POS tags are automatically captured in the learned POS embeddings and contribute to the improvement of parsing accuracy. In particular, probabilities of grammar rules involving correlated POS tags are automatically smoothed in our approach without the need for manual features or additional smoothing procedures.

Our experiments show that on the Wall Street Journal corpus our approach outperforms the previous approaches that also utilize POS tag correla-



*Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 763–771,
Austin, Texas, November 1-5, 2016. 

tions, and achieves a comparable result with recent state-of-the-art grammar induction systems. On the datasets of eight additional languages, our approach is able to achieve better performance than the baseline methods without any parameter tuning.

## 2 Related work

### 2.1 Dependency Model with Valence

The dependency model with valence (DMV) (Klein and Manning, 2004) is the first model to outperform the left-branching baseline in unsupervised dependency parsing of English. The DMV model is a generative model of a sentence and its parse tree. It generates a dependency parse from the root in a recursive top-down manner. At each step, a decision is first made as to whether a new child POS tag shall be generated from the current head tag; if the decision is yes, then a new child POS tag is sampled; otherwise, the existing child tags are recursively visited. There are three types of grammar rules in the model: `CHILD, DECISION` and `ROOT`, each with a set of multinomial parameters $\mathrm{P}_{CHILD}(c|h, dir, val)$, $\mathrm{P}_{DECISION}(dec|h, dir, val)$ and $\mathrm{P}_{ROOT}(c|root)$, where $dir$ is a binary variable indicating the generation direction (left or right), $val$ is a boolean variable indicating whether the current head POS tag already has a child in the current direction or not, $c$ indicates the child POS tag, $h$ indicates the head POS tag, and $dec$ indicates the decision of either STOP or CONTINUE. A `CHILD` rule indicates the probability of generating child $c$ given head $h$ on direction $dir$ and valence $val$. A `DECISION` rule indicates the probability of STOP or CONTINUE given the head, direction and valence. A `ROOT` rule is the probability of a child $c$ generated by the root. The probability of a dependency tree is the product of probabilities of all the grammar rules used in generating the dependency tree. The probability of a sentence is the sum of probabilities of all the dependency trees consistent with the sentence.

The basic DMV model has the limitation of being oversimplified and unable to capture certain linguistic structures. Headden et al. (2009) incorporated more types of valence and lexicalized information in the DMV model to increase its representation power and achieved better parsing accuracy than the basic DMV model.

### 2.2 DMV-based Learning Algorithms for Unsupervised Dependency Parsing

To learn a DMV model from text, the Expectation Maximization (EM) algorithm (Klein and Manning, 2004) can be used. In the E step, the model calculates the expected number of times each grammar rule is used in parsing the training text by using the inside-outside algorithm. In the M-step, these expected counts are normalized to become the probabilities of the grammar rules.

There have been many more advanced learning algorithms of the DMV model beyond the basic EM algorithm. In the work of Cohen and Smith (2008), a logistic normal prior was used in the DMV model to capture the similarity between POS tags. In the work of Berg-Kirkpatrick et al. (2010), features that group various morphological variants of nouns and verbs are used to predict the `DECISION` and `CHILD` parameters. These two approaches both utilize the correlations between POS tags to obtain better probability estimation of grammar rules involving such correlated POS tags. In the work of Tu and Honavar (2012), unambiguity of parse trees is incorporated into the training objective function of DMV to obtain a better performance.

### 2.3 Other Approaches to Unsupervised Dependency Parsing

There are many other approaches to unsupervised dependency parsing that are not based on DMV. Daumé III (2009) proposed a stochastic search based method to do unsupervised Shift-Reduce transition parsing. Rasooli and Faili (2012) proposed a transition based unsupervised dependency parser together with "baby-step" training (Spitkovsky et al., 2010) to improve parsing accuracy. Le and Zuidema (2015) proposed a complicated reranking based unsupervised dependency parsing system and achieved the state-of-the-art performance on the Penn Treebank dataset.

### 2.4 Neural based Supervised Dependency Parser

There exist several previous approaches on using neural networks for supervised dependency parsing. Garg and Henderson (2011) proposed a Temporal Restricted Boltzmann Machine to do transition

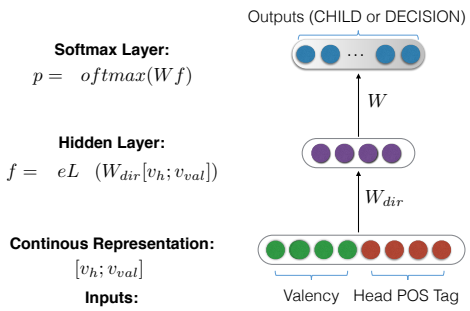


**Figure 1:** Structure of the neural network. Both CHILD and DECISION use the same architecture for the calculation of distributions.

based dependency parsing. Stenetorp (2013) applied recursive neural networks to transitional based dependency parsing. Chen and Manning (2014) built a neural network based parser with dense features instead of sparse indicator features. Dyer et al. (2015) proposed a stack long short-term memory approach to supervised dependency parsing. To our knowledge, our work is the first attempt to incorporate neural networks into a generative grammar for unsupervised dependency parsing.

## 3 Neural DMV

In this section, we introduce our neural based grammar induction approach. We describe the model in section 3.1 and the learning method in section 3.2.

### 3.1 Model

Our model is based on the DMV model (section 2.1), except that the CHILD and DECISION probabilities are calculated through two neural networks. We do not compute the ROOT probabilities using a neural network because doing that complicates the model while leads to no significant improvement in the parsing accuracy. Parsing a sentence using our model can be done in the same way as using DMV.

Below we show how the CHILD rule probabilities are computed in our neural based DMV model. Denote the set of all possible POS tags by $T$. We build a neural network to compute the probabilities of producing child tag $c \in T$ conditioned on the head, direction and valence $(h, dir, val)$.

The full architecture of the neural network is shown in Figure 1. First, we represent each head tag $h$ as a $d$ dimensional vector $v_h \in \mathbb{R}^d$, represent each value of valence $val$ as a $d'$ dimensional vector $v_{val} \in \mathbb{R}^{d'}$. We concatenate $v_h$ and $v_{val}$ as the input embedding vector. Then we map the input layer to a hidden layer with weight matrix $W_{dir}$ through a ReLU activation function. We have two versions of weight matrix $W_{dir}$ for the direction $dir$ being left and right respectively.

$$f(h, dir, val) = \text{ReLU}(W_{dir}[v_h; v_{val}])$$

We then take the inner product of $f$ and all the child POS tag vectors and apply a softmax function to obtain the rule probabilities:

$$[p_{c_1}, p_{c_2}, ..., p_{c_{\|T\|}}] = \text{Softmax}(W^{\text{T}} f)$$

where $W = [v_{c_1}, v_{c_2}, ..., v_{c_{\|T\|}}]$ is an embedding matrix composed of all the child POS tag vectors.

We use the same neural architecture to predict the probabilities of DECISION rules. The difference is that the neural network for DECISION has only two outputs (STOP and CONTINUE). Note that the two networks share parameters such as head POS tag embeddings and direction weight matrices $W_{left}$ and $W_{right}$. Valence embeddings are either shared or distinct between the two networks depending on the variant of DMV we use (i.e., whether the maximal valences for CHILD and DECISION are the same).

The parameters of our neural based model include the weights of the neural network and all the POS and valence embeddings, denoted by a set $\Theta = \{v_h, v_c, v_{val}, v_{dec}, W_{dir}; h, c \in T, val \in \{0, 1, ...\}, dir \in \{left, right\}, dec \in \{\text{STOP}, \text{CONTINUE}\}\}$.

### 3.2 Learning

In this section, we describe an approach based on the EM algorithm to learn our neural DMV model. To learn the parameters, given a set of unannotated sentences $x_1, x_2, ..., x_N$, our objective function is the log-likelihood function.

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

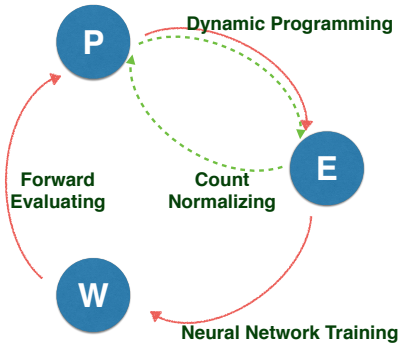


**Figure 2:** Learning procedure of our neural based DMV model. Green dashed lines represent the EM algorithm for learning traditional DMV. Red solid lines represent the learning procedure of our model. $P$ represents the rule probabilities of DMV, $E$ represents the expected counts of rules, and $W$ represents the parameters of the neural networks. In the traditional EM algorithm, the expected counts are directly used to re-estimate the rule probabilities. In our approach, parameter re-estimation is divided into two steps: training the neural networks from the expected counts and forward evaluation of the neural networks to produce the rule probabilities.

The approach is visualized in the Figure 2. The E-step computes the expected number of times each grammar rule used in parsing each training sentence $x_i$, denoted by $e_c(x_i)$ for CHILD rule $c$, $e_d(x_i)$ for DECISION rule $d$, and $e_r(x_i)$ for ROOT rule $r$. In the M-step of traditional DMV learning, these expected counts are normalized to re-estimate the parameters of DMV. This maximizes the expected log likelihood ($ELL$) with respect to the DMV model parameters.

$$ELL(\Theta) = \sum_{\alpha=1}^{N} \left( \sum_{c} e_c(x_i) \log p_c + \sum_{d} e_d(x_i) \log p_d + \sum_{r} e_r(x_i) \log p_r \right)$$

In our model, however, we do not directly assign the optimal rule probabilities of CHILD and DECISION; instead, we train the neural networks to output rule probabilities that optimize ELL, which is equivalent to a weighted cross-entropy loss function for each neural network. Note that while the traditional M-step produces the global optimum of ELL, our neural-based M-step does not. This is because a neural network tends to produce similar outputs for correlated inputs. In our case, the neural network is able to capture the correlations between different POS tags as well as different valence values and smooth the probabilities involving correlated tags and valences. In other words, our M-step can be seen as optimizing the ELL with a regularization term taking into account the input correlations. We use momentum based batch stochastic gradient descent algorithm to train the neural network and learn all the embeddings and weight matrices.

In addition to standard EM, we can also learn our neural based DMV model based on the Viterbi EM algorithm. The difference from standard EM is that in the E-step, we compute the number of times each grammar rule is used in the best parse of a training sentence instead of considering all possible parses.

## 4 Experiments

### 4.1 Setup

We used the Wall Street Journal corpus (with section 2-21 for training, section 22 for validation and section 23 for testing) in section 4.2 and 4.3. Then we reported the results on eight additional languages in section 4.4. In each experiment, we trained our model on gold POS tags with sentences of length less than 10 after punctuation has been stripped off. As the EM algorithm is very sensitive to initializations, we used the informed initialization method proposed in (Klein and Manning, 2004).

The length of embeddings is set to 10 for both POS tags and valence. We trained the neural networks with batch size 10 and used the change of the validation set loss function as the stop criteria. We ran our model for five times and reported the averaged directed dependency accuracy (DDA) of the learned grammars on the test sentences with length less than 10 and all sentences.

### 4.2 Comparisons of Approaches based on POS Correlations

We first evaluated our approach in learning the basic DMV model and compared the results against (Cohen and Smith, 2009) and (Berg-Kirkpatrick et al., 2010), both of which have very similar motivation as ours in that they also utilize the correlation between POS tags to learn the basic DMV model. Table 1

| Methods | WSJ10 | WSJ |
|---|---|---|
| Standard EM | 46.2 | 34.9 |
| Viterbi EM | 58.3 | 39.4 |
| LN (Cohen et al., 2008) | 59.4 | 40.5 |
| Shared LN (Cohen and Smith, 2009) | 61.3 | 41.4 |
| Feature DMV (Berg-Kirkpatrick et al., 2010) | 63.0 | - |
| Neural DMV (Standard EM) | 51.3 | 37.1 |
| Neural DMV (Viterbi EM) | **65.9** | **47.0** |

**Table 1:** Comparisons of Approaches based on POS Correlations

shows the results. It can be seen that our approach with Viterbi EM significantly outperforms the EM and viterbi EM baselines and also outperforms the two previous approaches.

### 4.3 Results on the extended DMV model

We directly apply our neural approach to learning the extended DMV model (Headden III et al., 2009; Gillenwater et al., 2010) (with the maximum valence value set to 2 for both CHILD and DECISION rules). As shown in Table 2, we achieve comparable accuracy with recent state-of-the-art systems. If we initialize our model with the grammar learned by Tu and Honavar (2012), the accuracy of our approach can be further improved.

Most of the recent state-of-the-art systems employ more complicated models and learning algorithms. For example, Spitkovsky et al. (2013) take several grammar induction techniques as modules and connect them in various ways; Le and Zuidema (2015) use a neural-based supervised parser and reranker that make use of high-order features and lexical information. We expect that the performance of our approach can be further improved when these more advanced techniques are incorporated.

### 4.4 Results on other languages

We also applied our approach on datasets of eight additional languages from the PASCAL Challenge on Grammar Induction (Gelling et al., 2012). We ran our approach using the hyper-parameters from experiment 4.2 on the new datasets without any further tuning. We tested three versions of our approach based on standard EM, softmax EM (Tu and Honavar, 2012) and Viterbi EM respectively. The results are shown in Table 3 for test sentence length no longer than ten and Table 4 for all test sentences.

| Methods | WSJ10 | WSJ |
|---|---|---|
| Systems with Basic Setup | | |
| 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 |
| UR-A E-DMV (Tu and Honavar, 2012) | 71.4 | 57.0 |
| Neural E-DMV | 69.7 | 52.5 |
| Neural E-DMV (Good Init) | **72.5** | **57.6** |
| Systems Using Extra Info | | |
| 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 |

**Table 2:** Comparison of recent unsupervised dependency parsing systems. Basic setup means learning from POS tags with sentences of length $\leq$ 10 and punctuation stripped off. Extra information may contain punctuations, longer sentences, lexical information, etc. For Neural E-DMV, "Good Init" means using the learned DMV model from Tu and Honavar (2012) as our initialization.

Our neural based methods achieve better results than their corresponding baselines in $75.0\%$ of the cases for test sentences no longer than 10 and $77.5\%$ for all test sentences. The good performance of our approach without data-specific hyper-parameter tuning demonstrates the robustness of our approach. Carefully tuned hyper-parameters on validation datasets, in our experience, can further improve the performance of our approach, in some cases by a large margin.

### 4.5 Effects of Hyper-parameters

We examine the influence of hyper-parameters on the performance of our approach with the same experimental setup as in section 4.3.

**Activation function** We compare different linear and non-linear functions: ReLU, Leaky ReLU, Tanh, Sigmoid. The results are shown in Table 5. Non-linear activation functions can be seen to significantly outperform linear activation functions.

**Length of the embedding vectors** The dimension of the embedding space is an important hyper-parameter in our system. As Figure 3 illustrates, when the dimension is too low (such as $dim = 5$), the performance is bad probably because the embedding vectors cannot effectively discriminate between

| | Arabic | Basque | Czech | Danish | Dutch | Portuguese | Slovene | Swedish |
|---|---|---|---|---|---|---|---|---|
| Standard EM | | | | | | | | |
| DMV | **45.8** | 41.1 | 31.3 | 50.8 | 47.1 | **36.7** | 36.7 | 43.5 |
| Neural DMV | 43.4 | **46.5** | **33.1** | **55.6** | **49.0** | 30.4 | **42.2** | **44.3** |
| Softmax EM $\sigma = 0.25$ | | | | | | | | |
| DMV | 49.3 | 45.6 | 30.4 | 43.6 | **46.1** | 33.5 | 29.8 | **50.3** |
| Neural DMV | **54.2** | **46.3** | **36.8** | **44.0** | 39.9 | **35.8** | **31.2** | 49.7 |
| Softmax EM $\sigma = 0.5$ | | | | | | | | |
| DMV | **54.2** | 47.6 | **43.2** | 38.8 | **38.0** | 33.7 | 23.0 | 37.2 |
| Neural DMV | 44.6 | **48.9** | 33.4 | **50.3** | 37.5 | **35.3** | **32.2** | **43.3** |
| Softmax EM $\sigma = 0.75$ | | | | | | | | |
| DMV | 42.2 | **48.6** | 22.7 | 41.0 | **33.8** | 33.5 | **23.2** | 41.6 |
| Neural DMV | **56.7** | 45.3 | **31.6** | **41.3** | 33.7 | **34.7** | 22.9 | **42.0** |
| Viterbi EM | | | | | | | | |
| DMV | 32.5 | 47.1 | 27.1 | 39.1 | 37.1 | 32.3 | 23.7 | 42.6 |
| Neural DMV | **48.2** | **48.1** | **28.6** | **39.8** | **37.2** | **36.5** | **39.9** | **47.9** |

**Table 3:** DDA results (on sentences no longer than 10) on eight additional languages. Our neural based approaches are compared with traditional approaches using standard EM, softmax EM (parameterized by $\sigma$) and Viterbi EM.

| **Activation function** | **WSJ10** |
|---|---|
| ReLU | **69.7** |
| Leaky ReLU | 67.0 |
| Tanh | 66.2 |
| Sigmoid | 62.5 |
| Linear | 55.1 |

**Table 5:** Comparison between activation functions.

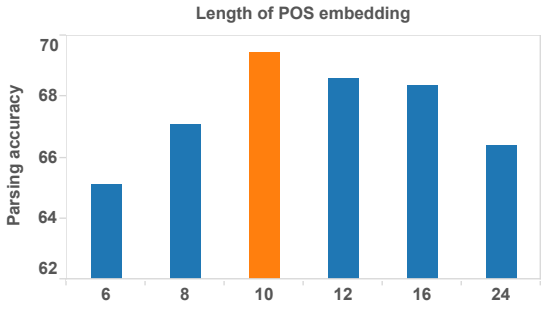


**Figure 3:** Parsing accuracy vs. length of POS embedding

different POS tags. On the other hand, when the dimension is too high (such as $dim = 30$), since we have only 35 POS tags, the neural network is prone to overfitting.

**Shared parameters** An alternative to our neural network architecture is to have two separate neural networks to compute `CHILD` and `DECISION` rule probabilities respectively. The embeddings of the head POS tag and the valence are not shared between the two networks. As can be seen in Table 6, sharing POS tags embeddings attribute to better performance.

| | **WSJ10** | **WSJ** |
|---|---|---|
| Separate Networks | 68.6 | 52.1 |
| Merged Network | **69.7** | **52.5** |

**Table 6:** Comparison between using two separate networks and using a merged network.

## 5 Model Analysis

In this section, we investigate what information our neural based DMV model captures and analyze how it contributes to better parsing performance.

### 5.1 Correlation of POS Tags Encoded in Embeddings

A main motivation of our approach is to encode correlation between POS tags in their embeddings so as to smooth the probabilities of grammar rules involving correlated POS tags. Here we want to examine whether the POS embeddings learned by our approach successfully capture such correlation.

We collected the POS embeddings learned in the experiment described in section 4.3 and visualized them on a 2D plane using the t-SNE algorithm (Van der Maaten and Hinton, 2008). t-SNE is a dimensionality reduction algorithm that maps data from a high dimensional space to a low dimensional one (2 or 3) while maintaining the distances between

| | Arabic | Basque | Czech | Danish | Dutch | Portuguese | Slovene | Swedish |
|---|---|---|---|---|---|---|---|---|
| Standard EM | | | | | | | | |
| DMV | 28.0 | 31.2 | 28.1 | 40.3 | 44.2 | **23.5** | 25.2 | 32.0 |
| Neural DMV | **30.6** | **38.5** | **29.3** | **46.1** | **46.2** | 16.2 | **36.6** | **32.8** |
| Softmax EM $\sigma = 0.25$ | | | | | | | | |
| DMV | 30.0 | 38.1 | 27.1 | 35.1 | **42.5** | **27.4** | 23.1 | **41.6** |
| Neural DMV | **31.5** | **40.5** | **32.6** | **38.0** | 35.7 | 26.7 | **24.2** | 41.3 |
| Softmax EM $\sigma = 0.5$ | | | | | | | | |
| DMV | **32.3** | 41.0 | **33.0** | 32.2 | 33.9 | 27.6 | 15.0 | 29.6 |
| Neural DMV | 22.5 | **42.6** | 30.6 | **40.8** | **37.5** | **28.6** | **25.0** | **33.7** |
| Softmax EM $\sigma = 0.75$ | | | | | | | | |
| DMV | 30.1 | **43.0** | 15.6 | 33.9 | **29.9** | 25.8 | **15.2** | 32.7 |
| Neural DMV | **34.9** | 37.4 | **24.7** | **34.2** | 29.5 | **28.9** | 15.1 | **33.3** |
| Viterbi EM | | | | | | | | |
| DMV | 23.9 | 40.9 | 20.4 | 32.6 | 33.0 | 26.9 | 16.5 | 36.2 |
| Neural DMV | **31.0** | **41.8** | **23.8** | **34.2** | **33.6** | **29.4** | **30.8** | **40.2** |

**Table 4:** DDA results (on all the sentences) on eight additional languages. Our neural based approaches are compared with traditional approaches using standard EM, softmax EM (parameterized by $\sigma$) and viterbi EM.

the data points in the high dimensional space. The "perplexity" hyper-parameter of the algorithm was set to 20.0 and the distance metric we used is the Euclidean distance.

Figure 4 shows the visualization result. It can be seen that in most cases, nearby POS tags in the figure are indeed similar. For example, VBP (Verb, non-3rd person singular present), VBD (Verb, past tense) and VBZ (Verb, 3rd person singular present) can be seen to be close to each other, and they indeed have very similar syntactic behavior. Similar observation can be made to NN (Noun, singular or mass ), NNPS (Proper noun, plural) and NNS (Noun, plural).

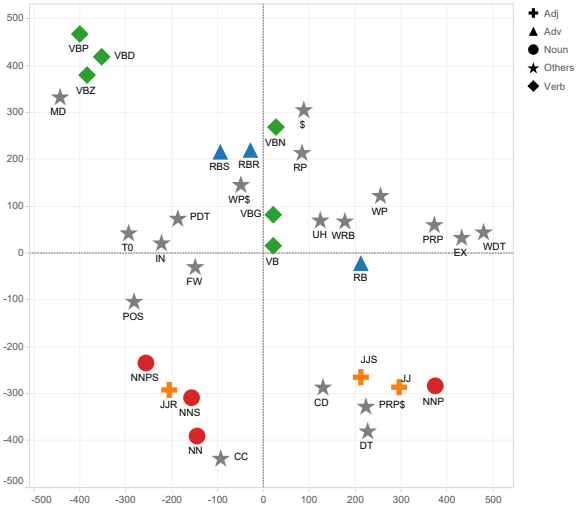


**Figure 4:** A visualization of the distances between embeddings of different POS tags.

## 5.2 Smoothing of Grammar Rule Probabilities

By using similar embeddings to represent correlated POS tags, we hope to smooth the probabilities of rules involving correlated POS tags. Here we analyze whether our neural networks indeed predict more similar probabilities for rules with correlated POS tags.

We conducted a case study on all types of verbs: VBP (Verb, non-3rd person singular present), VBZ (Verb, 3rd person singular present), VBD (Verb, past tense), VBN (Verb, past participle), VB (Verb, base form), VBG (Verb, gerund or present participle). We used the neural networks in our N-DMV model learned in the experiment described in section 4.2 to predict the probabilities of all the CHILD rules headed by a verb. For each pair of verb tags, we computed the total variation distance between the multinomial distributions of CHILD rules headed by the two verb tags. We also computed the total variation distances between CHILD rules of verb tags in the baseline DMV model learned by EM.

In Figure 5, We report the differences between the total variation distances computed from our model and from the baseline. A positive value means the distance is reduced in our model compared with that in the baseline. It can be seen that overall the distances between CHILD rules of different verb tags

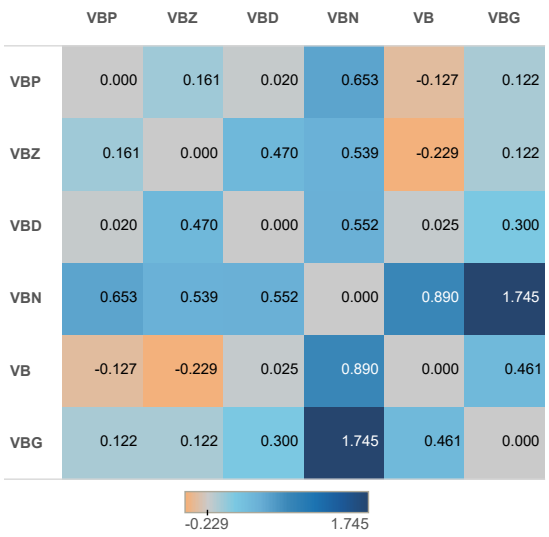


**Figure 5:** The change of the total variation distances between probabilities of CHILD rules headed by different verb tags in our model vs. the baseline. A positive value means the distance is reduced in our model compared with that in the baseline.

become smaller in our model. This verifies that our approach smooths the probabilities of rules involving correlated POS tags. From the figure one can see that the distance that reduces the most is between VBG and VBN. These two verb tags indeed have very similar syntactic behaviors and thus have similar embeddings as shown in figure 4. One the other hand, the distances between VB and VBZ/VBP become larger. This is reasonable since VB is syntactically different from VBZ/VBP in that it is very likely to generate a child tag TO to the left while VBZ/VBP always generate a subject (e.g., a noun or a pronoun) to the left.

## 6 Conclusion

We propose a neural based DMV model to do unsupervised dependency parsing. Our approach learns neural networks with continuous representations of POS tags to predict the probabilities of grammar rules, thus automatically taking into account the correlations between POS tags. Our experiments show that our approach outperforms previous approaches utilizing POS correlations and is competitive with recent state-of-the-art approaches on nine different languages.

For future work, we plan to extend our approach in learning lexicalized DMV models. In addition, we plan to apply our approach to other unsupervised tasks such as word alignment and sentence clustering.

## 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.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*, pages 740–750.

Shay B Cohen and Noah A Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 74–82. Association for Computational Linguistics.

Shay B Cohen and Noah A Smith. 2010. Covariance in unsupervised learning of probabilistic grammars. *The Journal of Machine Learning Research*, 11:3017–3051.

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.

Hal Daumé III. 2009. Unsupervised search-based structured prediction. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 209–216. ACM.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

Nikhil Garg and James Henderson. 2011. Temporal restricted boltzmann machines for dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 11–17. Association for Computational Linguistics.

Douwe Gelling, Trevor Cohn, Phil Blunsom, and Joao Graça. 2012. The pascal challenge on grammar induction. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 64–80. Association for Computational Linguistics.

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.

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*, page 478. Association for Computational Linguistics.

Phong Le and Willem Zuidema. 2015. Unsupervised dependency parsing: Let's use supervised parsers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 651–661, Denver, Colorado, May–June. Association for Computational Linguistics.

Mohammad Sadegh Rasooli and Heshaam Faili. 2012. Fast unsupervised dependency parsing with arc-standard transitions. In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 1–9. Association for Computational Linguistics.

Valentin I Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010. From baby steps to leapfrog: How less is more in unsupervised dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 751–759. Association for Computational Linguistics.

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.

Pontus Stenetorp. 2013. Transition-based dependency parsing using recursive neural networks. In *NIPS Workshop on Deep Learning*.

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.

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