

# Combining Generative and Discriminative Approaches to Unsupervised Dependency Parsing via Dual Decomposition\*

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 parser from unannotated sentences. Existing work focuses on either learning generative models using the expectation-maximization algorithm and its variants, or learning discriminative models using the discriminative clustering algorithm. In this paper, we propose a new learning strategy that learns a generative model and a discriminative model jointly based on the dual decomposition method. Our method is simple and general, yet effective to capture the advantages of both models and improve their learning results. We tested our method on the UD treebank and achieved a state-of-the-art performance on thirty languages.

## 1 Introduction

Dependency parsing is an important task in natural language processing. It identifies dependencies between words in a sentence, which have been shown to benefit other tasks such as semantic role labeling (Lei et al., 2015) and sentence classification (Ma et al., 2015). Supervised learning of a dependency parser requires annotation of a training corpus by linguistic experts, which can be time and resource consuming. Unsupervised dependency parsing eliminates the need for dependency annotation by directly learning from unparsed text.

Previous work on unsupervised dependency parsing mainly focuses on learning generative models, such as the dependency model with valence (DMV) (Klein and Manning, 2004) and combinatory categorial grammars (CCG) (Bisk and Hockenmaier, 2012). Generative models have

many advantages. For example, the learning objective function can be defined as the marginal likelihood of the training data, which is typically easy to compute in a generative model. In addition, many types of inductive bias, such as those favoring short dependency arcs (Smith and Eisner, 2006), encouraging correlations between POS tags (Cohen et al., 2008; Cohen and Smith, 2009; Berg-Kirkpatrick et al., 2010; Jiang et al., 2016), and limiting center embedding (Noji et al., 2016), can be incorporated into generative models to achieve better parsing accuracy. However, due to the strong independence assumption in most generative models, it is difficult for these models to utilize context information that has been shown to benefit supervised parsing.

Recently, a feature-rich discriminative model for unsupervised parsing is proposed that captures the global context information of sentences (Grave and Elhadad, 2015). Inspired by discriminative clustering, learning of the model is formulated as convex optimization of both the model parameters and the parses of training sentences. By utilizing language-independent rules between pairs of POS tags to guide learning, the model achieves state-of-the-art performance on the UD treebank dataset.

In this paper we propose to jointly train two state-of-the-art models of unsupervised dependency parsing: a generative model called LC-DMV (Noji et al., 2016) and a discriminative model called Convex-MST (Grave and Elhadad, 2015). We employ a learning algorithm based on the dual decomposition (Dantzig and Wolfe, 1960) inference algorithm, which encourages the two models to influence each other during training.

We evaluated our method on thirty languages and found that the jointly trained models surpass their separately trained counterparts in parsing accuracy. Further analysis shows that the two models

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

positively influence each other during joint training by implicitly sharing the inductive bias.

## 2 Preliminaries

### 2.1 DMV

The dependency model with valence (DMV) (Klein and Manning, 2004) is the first generative model that outperforms the left-branching baseline in unsupervised dependency parsing. In DMV, a sentence is generated by recursively applying three types of grammar rules to construct a parse tree from the top down. The probability of the generated sentence and parse tree is the probability product of all the rules used in the generation process. To learn the parameters (rule probabilities) of DMV, the expectation maximization algorithm is often used. Noji et al. (2016) exploited two universal syntactic biases in learning DMV: restricting the center-embedding depth and encouraging short dependencies. They achieved a comparable performance with state-of-the-art approaches.

### 2.2 Convex-MST

Convex-MST (Grave and Elhadad, 2015) is a discriminative model for unsupervised dependency parsing based on the first-order maximum spanning tree dependency parser (McDonald et al., 2005). Given a sentence, whether each possible dependency exists or not is predicted based on a set of handcrafted features and a valid parse tree closest to the prediction is identified by the minimum spanning tree algorithm.

For each sentence  $\mathbf{x}$ , a first-order dependency graph is built over the words of the sentence. The weight of each edge is calculated by  $\mathbf{w}^T \mathbf{f}(\mathbf{x}, i, j)$ , where  $\mathbf{w}$  is the parameters and  $\mathbf{f}(\mathbf{x}, i, j)$  is the handcrafted feature vector of the dependency from the  $i$ -th word to the  $j$ -th word in sentence  $\mathbf{x}$ . For sentence  $\mathbf{x}$  of length  $n$ , we can represent it as matrix  $\mathbf{X}$  where each row is a feature vector. The parse tree  $\mathbf{y}$  is a spanning tree of the graph and can be represented as a binary vector with length  $n \times n$  where each element is 1 if the corresponding arc is in the tree and 0 otherwise.

Learning is based on discriminative clustering with the following objective function:

$$\frac{1}{N} \sum_{\alpha=1}^N \left( \frac{1}{2n_{\alpha}} \|\mathbf{y}_{\alpha} - \mathbf{X}_{\alpha} \mathbf{w}\|_2^2 - \mu \mathbf{v}^T \mathbf{y}_{\alpha} \right) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

where  $\mathbf{X}_{\alpha}$  is a matrix where each row is a feature representation  $\mathbf{f}(\mathbf{x}_{\alpha}, i, j)$  of an edge in the depen-

dency graph of sentence  $\mathbf{x}_{\alpha}$ ,  $\mathbf{v}$  represents whether each dependency arc in  $\mathbf{y}_{\alpha}$  satisfies a set of pre-specified linguistic rules, and  $\lambda$  and  $\mu$  are hyper-parameters. The Frank-Wolfe algorithm is employed to optimize the objective function.

### 2.3 Dual Decomposition

Dual decomposition (Dantzig and Wolfe, 1960), a special case of Lagrangian relaxation, is an optimization method that decomposes a hard problem into several small sub-problems. It has been widely used in machine learning (Komodakis et al., 2007) and natural language processing (Koo et al., 2010; Rush and Collins, 2012).

Komodakis et al. (2007) proposed using dual decomposition to do MAP inference for Markov random fields. Koo et al. (2010) proposed a new dependency parser based on dual decomposition by combining a graph based dependency model and a non-projective head automata. In the work of Rush et al. (2010), they showed that dual decomposition can effectively integrate two lexicalized parsing models or two correlated tasks.

### 2.4 Agreement based Learning

Liang et al. (2008) proposed agreement based learning that trains several tractable generative models jointly and encourages them to agree on certain latent variables. To effectively train the system, a product EM algorithm was used. They showed that the joint model can perform better than each independent model on the accuracy or convergence speed. They also showed that the objective function of the work of Klein and Manning (2004) is a special case of the product EM algorithm for grammar induction. Our approach has a similar motivation to agreement based learning but has two important advantages. First, while their approach only combines generative models, our approach can make use of both generative and discriminative models. Second, while their approach requires the sub-models to share the same dynamic programming structure when performing decoding, our approach does not have such restriction.

## 3 Joint Training

We minimize the following objective function that combines two different models of unsupervised

dependency parsing:

$$J(\mathbf{M}_F, \mathbf{M}_G) = \sum_{\alpha=1}^N \min_{\mathbf{y}_\alpha \in \mathcal{Y}_\alpha} (F(\mathbf{x}_\alpha, \mathbf{y}_\alpha; \mathbf{M}_F) + G(\mathbf{x}_\alpha, \mathbf{y}_\alpha; \mathbf{M}_G))$$

where  $N$  is the size of training data,  $\mathbf{M}_F$  and  $\mathbf{M}_G$  are the parameters of the first and second model respectively,  $F$  and  $G$  are their respective learning objectives, and  $\mathcal{Y}_\alpha$  is the set of valid dependency parses of sentence  $\mathbf{x}_\alpha$ . While in principle this objective can be used to combine many different types of models, here we consider two state-of-the-art models of unsupervised dependency parsing, a generative model LC-DMV (Noji et al., 2016) and a discriminative model Convex-MST (Grave and Elhadad, 2015). We denote the parameters of LC-DMV by  $\Theta$  and the parameters of Convex-MST by  $\mathbf{w}$ . Their respective objective functions are,

$$\begin{aligned} F(\mathbf{x}_\alpha, \mathbf{y}_\alpha; \Theta) &= -\log(P_\Theta(\mathbf{x}_\alpha, \mathbf{y}_\alpha)f(\mathbf{x}_\alpha, \mathbf{y}_\alpha)) \\ G(\mathbf{x}_\alpha, \mathbf{y}_\alpha; \mathbf{w}) &= \frac{1}{2n_\alpha} \|\mathbf{y}_\alpha - \mathbf{X}_\alpha \mathbf{w}\|_2^2 + \frac{\lambda}{2N} \|\mathbf{w}\|_2^2 - \mu \mathbf{v}^T \mathbf{y} \end{aligned}$$

where  $P_\Theta(\mathbf{x}_\alpha, \mathbf{y}_\alpha)$  is the joint probability of sentence  $\mathbf{x}_\alpha$  and parse  $\mathbf{y}_\alpha$ ,  $f$  is a constraint factor, and the notations in the second objective function are explained in section 2.2.

### 3.1 Learning

We use coordinate descent to optimize the parameters of the two models. In each iteration, we first fix the parameters and find the best dependency parses of the training sentences (see section 3.2); we then fix the parses and optimize the parameters. The detailed algorithm is shown in Algorithm 1.

Pretraining of the two models is done by running their original learning algorithms separately. When the parses of the training sentences are fixed, it is easy to show that the parameters of the two models can be optimized separately. Updating the parameters  $\Theta$  of LC-DMV can be done by simply counting the number of times each rule is used in the parse trees and then normalizing the counts to get the maximum-likelihood probabilities. The parameters  $\mathbf{w}$  of Convex-MST can be updated by stochastic gradient descent. After updating  $\Theta$  and  $\mathbf{w}$  at each iteration, we additionally train each model separately for three iterations, which we find further improves learning.

---

### Algorithm 1 Parameter Learning

---

**Input:** Training sentence  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$   
 Pre-train  $\Theta$  and  $\mathbf{w}$   
**repeat**  
   Fix  $\Theta$  and  $\mathbf{w}$  and solve the decoding problem to get  $\mathbf{y}_\alpha, \alpha = 1, 2, \dots, N$   
   Fix the parses and update  $\Theta$  and  $\mathbf{w}$   
**until** Convergence

---



---

### Algorithm 2 Decoding via Dual Decomposition

---

**Input:** Sentence  $\mathbf{x}$ , fixed parameters  $\mathbf{w}$  and  $\Theta$   
 Initialize vector  $\mathbf{u}$  of size  $n \times n$  to  $\mathbf{0}$   
**repeat**  
    $\hat{\mathbf{y}} = \arg \min_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y}; \Theta) + \mathbf{u}^T \mathbf{y}$   
    $\hat{\mathbf{z}} = \arg \min_{\mathbf{z} \in \mathcal{Y}} G(\mathbf{x}, \mathbf{z}; \mathbf{w}) - \mathbf{u}^T \mathbf{z}$   
   **if**  $\hat{\mathbf{y}} = \hat{\mathbf{z}}$  **then**  
     return  $\hat{\mathbf{y}}$   
   **else**  
      $\mathbf{u} = \mathbf{u} + \tau (\hat{\mathbf{y}} - \hat{\mathbf{z}})$   
   **end if**  
**until** Convergence

---

## 3.2 Joint Decoding

Given a training sample  $\mathbf{x}$  and parameters  $\mathbf{w}, \Theta$ , the goal of decoding is to find the best parse tree:

$$\hat{\mathbf{y}} = \arg \min_{\mathbf{y} \in \mathcal{Y}} \frac{1}{2n} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 - \mu \mathbf{v}^T \mathbf{y} - \log P_\Theta(\mathbf{x}, \mathbf{y})$$

We employ the dual decomposition algorithm to solve this problem (shown in Algorithm 2), where  $\tau$  represents the step size.

The most important part of the algorithm is solving the two separate decoding problems:

$$\hat{\mathbf{y}} = \arg \min_{\mathbf{y} \in \mathcal{Y}} -\log(P_\Theta(\mathbf{x}, \mathbf{y})f(\mathbf{x}, \mathbf{y})) + \mathbf{u}^T \mathbf{y}$$

$$\hat{\mathbf{z}} = \arg \min_{\mathbf{z} \in \mathcal{Y}} \frac{1}{2n} \|\mathbf{z} - \mathbf{X}\mathbf{w}\|_2^2 - \mu \mathbf{v}^T \mathbf{z} - \mathbf{u}^T \mathbf{z}$$

The first decoding problem can be solved by a modified CYK parsing algorithm that takes into account the information in vector  $\mathbf{u}$ . The second decoding problem can be solved using the same algorithm of Grave and Elhadad (2015) (we use the projective version in our approach).

## 4 Experiments

### 4.1 Setup

We use UD Treebank 1.4 as our datasets. We sorted the datasets in the treebank by the number

of training sentences of length  $\leq 15$  and selected the top thirty datasets, which is similar to the setup of Noji et al. (2016). For each dataset, we trained our method on the training data with length  $\leq 15$  and tested our method on the testing data with length  $\leq 40$ . We tuned the hyper-parameters of our method on the dataset of the English language and reported the results on the thirty datasets without any further parameter tuning. We compared our method with four baselines. The first two baselines are Convex-MST and LC-DMV that are independently trained. To construct the third baseline, we used the independently trained Convex-MST baseline to parse all the training sentences and then used the parses to initialize the training of LC-DMV. This can be seen as a simple method to combine two different approaches. On the other hand, we did not use the LC-DMV baseline to initialize Convex-MST training because the objective function of Convex-MST is convex and therefore the initialization does not matter.

## 4.2 Results

In Table 1, we compare our jointly trained models with the four baselines. We can see that with joint training and independent decoding, LC-DMV and Convex-MST can achieve superior overall performance than when they are separately trained with or without mutual initialization. Joint decoding with our jointly trained models performs worse than independent decoding. We made the same observation when applying joint decoding to the separately trained models (not shown in the table). We believe this is because unsupervised parsers have relatively low accuracy and forcing them to reconcile would not lead to better parses. On the other hand, joint decoding during training helps propagate useful inductive biases between models and thus leads to better trained models.

## 4.3 Analysis of Parsing Results

We analyze the parsing results from the two models to see how they benefit each other with joint training. Note that LC-DMV limits the depth of center embedding and encourages shorter dependency length, while Convex-MST encourages dependencies satisfying pre-specified linguistic rules. Therefore, we would like to see whether the jointly-trained LC-DMV produces more dependencies satisfying the linguistic priors than its separately-trained counterpart, and whether the jointly-trained Convex-MST produces parse trees

Language	M	D	D-I	M-J	D-J	DD
A.Greek	43.4	33.1	38.8	44.2	<b>44.9</b>	38.9
A.Greek-P	50.4	43.0	44.7	50.8	<b>52.9</b>	44.9
Basque	50.0	45.4	54.2	52.1	<b>55.7</b>	50.2
Bulgarian	61.6	62.4	60.3	64.7	<b>73.8</b>	64.8
Czech	48.6	17.4	53.9	48.7	<b>54.0</b>	53.5
Czech-CAC	50.4	53.0	53.9	55.6	<b>62.3</b>	50.2
Dutch	45.3	34.1	<b>56.7</b>	48.2	43.5	40.7
Dutch-LS	42.4	27.0	16.4	<b>43.2</b>	41.2	36.3
English	54.0	56.0	49.8	57.3	<b>60.1</b>	53.4
Estonian	<b>49.4</b>	31.8	47.5	48.7	44.0	44.4
Finnish	<b>44.7</b>	26.9	39.0	44.2	43.5	31.2
Finnish-FTB	<b>49.9</b>	31.0	47.9	47.7	48.0	36.5
French	<b>62.0</b>	48.6	57.0	54.5	57.0	55.5
German	51.4	50.5	54.1	49.3	<b>55.7</b>	48.6
Gothic	52.7	49.9	47.3	<b>59.6</b>	56.4	58.0
Hindi	56.8	54.2	48.4	52.1	<b>60.0</b>	49.1
Italian	69.1	<b>71.1</b>	67.4	62.8	70.3	64.5
Japanese	44.8	43.8	43.8	42.8	<b>45.8</b>	41.0
Latin-ITTB	38.8	38.6	42.3	<b>47.0</b>	42.2	40.3
Latin-PROIEL	44.3	34.8	38.7	<b>46.8</b>	41.8	42.9
Norwegian	55.3	45.5	51.4	57.4	<b>60.8</b>	46.6
Old_Church_S	56.4	26.6	51.3	58.3	<b>58.6</b>	42.0
Polish	63.4	63.7	61.5	70.7	<b>74.2</b>	68.9
Portuguese	57.9	<b>67.2</b>	60.1	56.1	62.9	57.4
Portuguese-BR	59.3	63.1	62.0	65.5	<b>68.8</b>	58.3
Russian-STR	47.6	51.7	56.5	52.1	<b>64.4</b>	52.6
Slovak	57.4	59.3	51.9	61.7	<b>65.9</b>	58.7
Slovenian	54.0	49.5	56.3	65.5	<b>69.6</b>	56.1
Spanish	61.9	61.9	60.3	57.4	<b>68.0</b>	60.2
Spanish-AC	59.4	59.5	56.4	56.8	<b>65.2</b>	57.6
Average	52.7	47.2	50.3	54.2	<b>56.5</b>	49.6
Average $\leq 15$	55.4	48.9	54.9	57.3	<b>60.2</b>	53.8

Table 1: Directed dependency accuracy on thirty datasets with test sentences of length  $\leq 40$ . The last row indicates the average directed accuracy on sentences of length  $\leq 15$ . M (Convex-MST) and D (LC-DMV) are the independently trained baselines. D-I is the third baseline in which the LC-DMV training is initialized by the parses produced from the trained Convex-MST model. With our jointly trained models, M-J and D-J denote separate decoding and DD denotes joint decoding.

with less center embedding and shorter dependencies than its separately-trained counterpart.

Figure 1 shows the percentages of dependencies satisfying linguistic rules when using the separately and jointly trained LC-DMV to parse the test sentences in the English dataset. As we can see, with joint training, LC-DMV is indeed influenced by Convex-MST and produces more dependencies satisfying linguistic rules.

Table 2 shows the average dependency length when using the separately and jointly trained Convex-MST to parse the English test dataset. The dependency length can be seen to decrease with joint training, showing the influence from LC-DMV. As to center embedding depth, we find that separately trained Convex-MST already produces very few center embeddings of depth 2 or

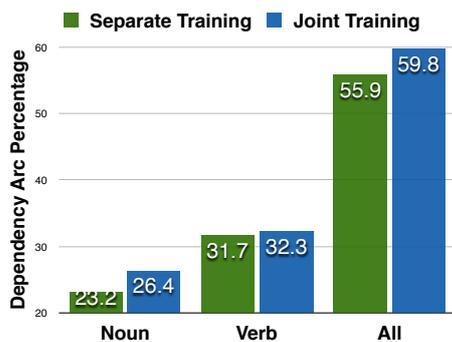


Figure 1: Percentages of dependencies satisfying linguistic rules in the LC-DMV parses of the English test dataset. Noun and Verb denote dependencies headed by nouns and verbs.

Methods	Average Dependency Length
Separate Training	1.673
Joint Training	<b>1.627</b>

Table 2: Average dependency length in the Convex-MST parses of the English test dataset.

more, so the influence from the center embedding constraint of LC-DMV during joint training is not obvious. We note that the influence on Convex-MST from LC-DMV during joint training is relatively small, which may contribute to the much smaller accuracy improvement (1.5%) of Convex-MST with joint training in comparison with the 9.3% improvement of LC-DMV. We conducted an additional experiment that scaled down the Convex-MST objective in joint training in order to increase the influence of LC-DMV. The results show that LC-DMV indeed influences Convex-MST to a greater degree, but the parsing accuracies of the two models decrease.

## 5 Conclusion

In this paper, we proposed a new learning strategy for unsupervised dependency parsing that learns a generative model and a discriminative model jointly based on dual decomposition. We show that with joint training, two state-of-the-art models can positively influence each other and achieve better performance than their separately trained counterparts.

## 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.

Yonatan Bisk and Julia Hockenmaier. 2012. Simple robust grammar induction with combinatory categorical grammars.

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.

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.

George B Dantzig and Philip Wolfe. 1960. Decomposition principle for linear programs. *Operations research*, 8(1):101–111.

Edouard Grave and Noémie Elhadad. 2015. [A convex and feature-rich discriminative approach to dependency grammar induction](#). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1375–1384, Beijing, China. 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*, page 478. Association for Computational Linguistics.

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. 2007. Mrf optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

Terry Koo, Alexander M Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1298. Association for Computational Linguistics.

Tao Lei, Yuan Zhang, Lluís Màrquez, Alessandro Moschitti, and Regina Barzilay. 2015. [High-order low-rank tensors for semantic role labeling](#). In *Proceedings of the 2015 Conference of the North*

*American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1150–1160, Denver, Colorado. Association for Computational Linguistics.

Percy S Liang, Dan Klein, and Michael I. Jordan. 2008. [Agreement-based learning](#). In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 913–920. Curran Associates, Inc.

Mingbo Ma, Liang Huang, Bing Xiang, and Bowen Zhou. 2015. Dependency-based convolutional neural networks for sentence embedding. *arXiv preprint arXiv:1507.01839*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 91–98. Association for Computational Linguistics.

Hiroshi Noji, Yusuke Miyao, and Mark Johnson. 2016. [Using left-corner parsing to encode universal structural constraints in grammar induction](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 33–43, Austin, Texas. Association for Computational Linguistics.

Alexander M. Rush and Michael Collins. 2012. [A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing](#). *J. Artif. Int. Res.*, 45(1):305–362.

Alexander M Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11. Association for Computational Linguistics.

Noah A Smith and Jason Eisner. 2006. Annealing structural bias in multilingual weighted grammar induction. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 569–576. Association for Computational Linguistics.