Modeling Valence Effects in Unsupervised Grammar Induction

David McClosky
Brown Laboratory for Linguistic Information Processing (BLLIP)
Brown University
Providence, RI 02912
dmcc@cs.brown.edu

About this report: While the results of this paper are not entirely positive, I feel it contains some valuable ideas. In hopes that others will find these helpful or interesting, I've published this as a technical report. It is still my belief that the poor multilingual performance is due to a bug in the implementation, but I do not currently have time to investigate this. This work was done during the 2006-2007 school year (partially at Charles University in Prague).

Abstract

We extend the dependency grammar induction model of Klein and Manning (2004) to incorporate further valence information. Our extensions achieve significant improvements in the task of unsupervised dependency grammar induction. We use an expanded grammar which tracks higher orders of valence and allows each valence slot to be filled by a separate distribution rather than using one distribution for all slots. Additionally, we show that our performance improves if our grammar restricts the maximum number of attachments in each direction, forcing our system to focus on the common case. Taken together, these techniques constitute a 23.4% error reduction in dependency grammar induction over the model by Klein and Manning (2004) on English.

1 Introduction

Unsupervised dependency grammar induction aims to uncover syntactic dependency structures of surface text using no annotated examples. In this section, we will motivate why we learn dependency grammars rather than constituency grammars as well as why it is important to perform this task in an unsupervised fashion. While constituency grammars provide sets of labeled brackets, a dependency grammar produces a graph. Each node in this directed acyclic graph represents a word and directed edges between words denote governance of one word over the other. A virtual root node is connected to the head of the sentence. A more complete description of dependency grammars can be found in Nivre (2005). Dependency grammars have increased in popularity in recent years, in part because many parsing applications only require these modification structures rather than the more complex constituency information.

Our task is unsupervised. As input, our system takes only sequences of part-of-speech tags. With the exception of these part-of-speech tags, we do not use any labeled data.[1] Labeled data is harder to acquire than unlabeled data since it is expensive to annotate and often not available for less common languages or domains. As many problems in natural language processing increasingly rely on capturing statistics from corpora, there is a strong push towards the more portable semi-supervised and unsupervised techniques.

In Section 2, we describe the Dependency Model with Valence (DMV) by Klein and Manning (2004) and some of its extensions since the model is the starting point for our work. Next, we present some

[1] For a fully unsupervised approach, Klein and Manning (2004) show that tags may be induced instead, resulting in a performance penalty. For a deeper investigation of the interactions between tag and grammar induction systems, see Headden III et al. (2008).

empirical properties of valence which have influenced the design of our model (Section 3). In this paper, valence will refer to the number and type of arguments accepted by a word. Naturally, learning valence is a considerable portion of inducing correct grammars (dependency or otherwise). Our model and its estimation are described in Sections 4 and 5, respectively. In Section 6, we describe how we will evaluate and present our results on English. Our analysis of the results follows in Section 7 and we conclude in Section 8.

2 Previous Work

This work is a direct extension of the work on dependency grammar induction by Klein and Manning (2004) (see also Klein (2005)). Klein and Manning's Dependency Model with Valence (DMV) is a simplified version of some of the models used by supervised dependency parsers. The model is an example of a generative head-outward process. In this process, first we choose the head word of the sentence from the distribution P_{root}. A head chooses its arguments in one direction until it generates a special stop symbol, Δ, then generates arguments in the other direction. Each of these child arguments recursively generates its own arguments using this process. Let $D_{left}(h)$ and $D_{right}(h)$ be functions which provide the arguments of head h in the left and right directions, respectively. The probability of generating the subtree for head h is given by

$$P(h) = \prod_{d \in \{left, right\}} \left[\prod_{a \in D_d(h)} P_{\text{stop}}(\neg\Delta \mid h,\ d,\ adj) \times P_{\text{arg}}(a \mid h,\ d) P(D(a)) \right] \times P_{\text{stop}}(\Delta \mid h,\ d,\ adj)$$

where adj is an adjacency bit which is true if and only if an argument has been generated in the current direction. P_{stop} is a distribution over two symbols, Δ (indicating we should not produce further arguments in this direction) and $\neg\Delta$ (indicating the opposite). An important note for subsequent discussion is that while P_{stop} is conditioned on the adjacency bit, P_{arg} is not.

Let the vocabulary be the set of part of speech tags Σ. For each $X \in \Sigma$, we define two nonterminals. Let $\overleftrightarrow{X}$ indicate a symbol that has not generated any stops and $\overrightarrow{X}$ be the symbol after it has generated a left stop. $\forall H, A \in \Sigma$:

Dependency action	**Equivalent CFG rule**
Choose head of sentence	$Root \rightarrow \overleftrightarrow{H}$
H attaches left to A	$\overleftrightarrow{H} \rightarrow \overleftrightarrow{A}\ \ \overleftrightarrow{H}$
H generates left stop	$\overleftrightarrow{H} \rightarrow \overrightarrow{H}$
H attaches right to A	$\overrightarrow{H} \rightarrow \overrightarrow{H}\ \ \overleftarrow{A}$
H generates right stop	$\overrightarrow{H} \rightarrow H$

Figure 1: Simplified version of the grammar used by (Klein and Manning, 2004; Smith and Eisner, 2006), demonstrating corresponding CFG rules for dependency grammar actions. This grammar schema does not properly handle adjacency and the situation when $H = A$.

Klein and Manning (2004) show how each dependency action corresponds to a context-free rule and derive a corresponding probabilistic context-free grammar (PCFG). Since DMV assumes left and right attachments are independent from each other, we will follow Klein and Manning (2004) in choosing to make left attachments before right attachments. We present a simplified version of the grammar in DMV in Figure 1.

Once the dependency grammar has been converted into a context-free grammar, standard techniques for estimating PCFGs can be used, e.g. the inside-outside algorithm (Baker, 1979).

However, estimating the simplified PCFG from the schema from Figure 1 with the inside-outside algorithm does not achieve above baseline parsing accuracy (Table 4; baselines will be discussed later). Klein and Manning (2004) describe two changes which improve performance significantly. First, the model must be initialized reasonably before estimation. We discuss this in detail in Section 5. The second change is the presence of the adjacency bit: after the first attachment is made, the distribution of future attachments changes. While DMV conditions only the stop probabilities on this valence informa-

tion, we will show that there is an added benefit if the entire attachment distribution is conditioned on it.

Smith and Eisner (2006) (see also Smith (2006)) present a different type of extension of the models by Klein and Manning (2004). The authors adopt Klein and Manning's models as testbeds for better parameter estimation techniques. While their new estimators significantly outperform the commonly used Expectation-Maximization algorithm (EM), their estimation methods are minimally supervised and require a small amount of labeled held-out data to tune parameters. Our work continues to use EM since we assume the scenario of having no labeled data available. Furthermore, since we are primarily concerned with changing the model rather than the estimation procedure, our work and theirs are complementary and can be expected to yield further improvements in combination. This remains as future work.

In addition to the dependency model, Klein and Manning (2004) provide a constituency model which induces unlabeled bracketings from sentences. Furthermore, the authors demonstrate a method of combining the two models. The combined model performs better on both constituency and dependency grammar induction tasks, as the two models are sensitive to different aspects of the data. Haghighi and Klein (2006) describe a minimally-supervised extension to the constituency model where a small number of constituent prototypes are provided as additional input. As our model provides similar probabilities as DMV, it is conceivable that it could be combined with either of these constituency models in a similar fashion.

3 Valence Effects

A key part of grammar induction is learning valence,[2] i.e. the number and types of arguments accepted by a word. For example, the verb "walk" takes a subject and optionally a location as an object. Thus, "walk" has one argument slot to the left and an optional argument slot to the right. In this section, we present some empirical properties of dependencies involving valence. These properties drive several decisions of our grammar design and parameterization.

Since we deal with part of speech tags rather than words in this paper, our models of valence will be restricted in power and significantly coarser. For example, it will group all mass and count nouns together and attempt to learn some sort of average of their valence. Worse, transitive and ditransitive verbs will be grouped together. Nevertheless, even with this simplification, valence information can be very discriminating.

3.1 Number of arguments

To start, we study the distribution over the number of arguments accepted by a word. In Table 1, we show the empirical distribution of the number of arguments taken by a head. Our data comes from seven languages in the CoNLL 2006 Shared Task (Buchholz and Marsi, 2006).[3] Since the models that we consider make attachments in a single direction before considering the other direction and because we wish to ignore any asymmetries of specific languages, we group arguments by direction and then count the number of arguments in each direction. Thus, a head with three left attachments and two right attachments contributes one raw count to each of the "2" and "3+" buckets. From Table 1 we can see that these distributions are fairly similar across languages, with a standard deviation of about 3-4%. Also note that there is a relatively small amount of probability mass in the "3+" bucket — less than 3% on average. We will return to this figure later when we present our Restricted Valence Grammar (Section 4). The figures come from sentences of up to 10 words, but we see similar trends on longer sentences. If we instead consider sentences of all lengths, the "3+" bucket still only receives 5.0% probability mass on average and the variance is still small.

DMV can approximate this distribution with the help of the adjacency bit: stop probabilities are conditioned on whether any attachments have been made (as well as the head and current direction). Thus, for each head, the model can learn a first approximation of how many arguments it should take

[2] In this paper, we will use valence in its most general sense where it applies to all parts of speech.

[3] Arabic (Hajič et al., 2004), Bulgarian (Simov et al., 2005), Czech (Böhmová et al., 2003), German (Brants et al., 2002), Japanese (Kawata and Bartels, 2000), Portuguese (Afonso et al., 2002), and Swedish (Nilsson et al., 2005)

	Number of arguments			
Language	0	1	2	3+
Arabic	44.5	46.9	7.1	1.5
Bulgarian	35.6	49.2	11.4	3.8
Czech	36.1	53.0	8.6	2.3
German	34.1	44.9	16.5	4.6
Japanese	40.9	51.6	6.1	1.4
Portuguese	35.4	54.6	8.3	1.6
Swedish	33.2	51.8	12.1	2.9
Mean	37.1	50.3	10.0	2.6
Standard deviation	4.1	3.5	3.6	1.3

Table 1: Empirical distribution of the number of arguments taken by a head in both directions, first grouping arguments by direction. In other words, a head with 3 left attachments and 2 right attachments contributes one raw count to each of the "2" and "3+" buckets of this distribution.

in each direction (0, 1, or 2+). It is clear now why DMV only needs a single adjacency bit.

3.2 Type of arguments

To learn correct valence information, we also need to learn the types (parts of speech, in our case) of arguments accepted by a tag. Even though we are using part of speech tags instead of word forms, we can still learn some useful properties about which part of speech tags can fill each argument slot. DMV has a mechanism which can approximate the distribution over the number of arguments. However, it cannot learn which arguments fill each valence slot since the distribution of children, P_{arg}, is conditioned only on the head and direction.

To demonstrate the importance of conditioning arguments on valence, we present some empirical statistics on dependencies extracted from the Wall Street Journal (WSJ) section of the Penn Treebank (Marcus et al., 1993): the empirical distribution over arguments given the argument slot. Table 2 shows three distributions for the two most frequent parents in this corpus – left attachments for singular nouns (NN) and left and right attachments for past tense verbs (VBD). In each table, we show the probability of attaching to a tag given which slot we are filling. The "All" row shows the overall distribution ignoring argument slots. Since the most common number of attachments is one, the overall distribution is fairly close to the distribution of the first argument. However, note that after making the first attachment, the distributions change drastically: nouns increase their preference for determiners (Table 2a), verbs making left attachments lose their preference for nouns in favor of a more uniform distribution (Table 2b), and verbs making right attachments shift from preferring nouns and adverbs to prepositions, (Table 2c). Together, these tables should demonstrate the need for learning different distributions per argument slot.

Slot	CD	DT	JJ	NN	NNP	Others
1st	6.6	**38.7**	23.9	13.5	4.3	13.0
2nd	3.4	**55.1**	15.6	8.6	4.7	12.5
3rd+	3.5	**51.6**	14.4	7.8	8.0	14.6
All	5.2	**45.5**	20.1	11.2	5.0	13.1

(a) Distribution of left attachments from NN

Slot	CC	NN	NNP	NNS	PRP	Others
1st	0.1	**27.7**	18.2	21.6	19.2	13.3
2nd	**11.2**	10.1	7.4	8.8	8.1	54.4
3rd+	**20.9**	14.2	5.2	12.7	11.9	35.1
All	3.6	**23.1**	15.1	18.3	16.3	23.6

(b) Distribution of left attachments from VBD

Slot	IN	NN	RB	TO	VBN	Others
1st	8.2	**19.3**	12.9	8.8	9.2	41.6
2nd	**30.1**	8.3	6.6	19.1	10.9	25.1
3rd+	**20.6**	9.9	11.3	5.0	6.4	46.8
All	14.6	**16.0**	11.2	11.3	9.5	37.5

(c) Distribution of right attachments from VBD

Table 2: Empirical probability of attaching a tag conditioned on the valence slot. "All" shows these probabilities marginalizing over all slots. Bold face indicates the most likely attachment for each slot. Note how "All" ignores important shifts in the distribution as arguments are attached.

4 Model

Our generative process is based on the head-outward process used by DMV. As in DMV, we pick the head of a sentence from a distribution, P_{root}. We first pick the number of argument slots to the left conditioned

on the head.[4] These valence numbers come from a distribution called P_{val}. Next, we fill each slot with an argument conditioned on the head, direction, and the index of the slot. Each argument recursively generates its own arguments according this process. Finally, we repeat the process to generate arguments to the right. We present the generative process as a series of context free rewrites in Figure 2. Because this grammar places a hard limit on the number of attachments in a particular direction, we call this the *Restricted Valence Grammar* (RVG).

There are two differences between the Restricted Valence Grammar and the grammar used in DMV. First, while DMV conditions arguments on the head and direction, we also condition on the slot index. Second, we draw from the valence distribution, P_{val}, before making any attachments rather than drawing stop probabilities from P_{stop}.

In fact, it was not our original intention to restrict the maximum valence of our grammar. The intended implementation allowed symbols to make any number of attachments. The change is that $\overleftarrow{\overrightarrow{H}}^N$ is not required to decrease its valence after making attachments.[5] When it is finished making attachments at the N level, it can rewrite as $\overleftarrow{\overrightarrow{H}}^{N-1}$. We will refer to this grammar as the *Unrestricted Valence Grammar* (URVG).

The Unrestricted Valence Grammar with a valence of one is very similar to DMV's grammar. The DMV stop probabilities correspond to two rules in our grammar. The probability of a head taking no leftward arguments ($P_{\text{stop}}(\Delta \mid H,\ dir = left,\ adj = false)$) in DMV is the probability of the rule $\overleftarrow{\overrightarrow{H}} \rightarrow \overleftarrow{\overrightarrow{H}}^0$ in our model. The probability of a head taking additional arguments to the left ($P_{\text{stop}}(\Delta \mid H,\ dir = left,\ adj = true)$) in DMV is the probability of the rule $\overleftarrow{\overrightarrow{H}}^1 \rightarrow \overleftarrow{\overrightarrow{H}}^0$ in our model. The same holds for arguments to the right.

Let N be the maximum valence allowed.
$\forall H, A \in \Sigma, n \in 1, \ldots, N$:

Description	**Equivalent CFG rule**
Choose head of sentence	$Root \rightarrow \overleftarrow{\overrightarrow{H}}$
Left valence selection	$\overleftarrow{\overrightarrow{H}} \rightarrow \overleftarrow{\overrightarrow{H}}^n$
Left attachment	$\overleftarrow{\overrightarrow{H}}^n \rightarrow \overleftarrow{\overrightarrow{A}}\ \overleftarrow{\overrightarrow{H}}^{n-1}$
Left stop	$\overleftarrow{\overrightarrow{H}}^0 \rightarrow \overrightarrow{H}$
Right valence selection	$\overrightarrow{H} \rightarrow \overrightarrow{H}^n$
Right attachment	$\overrightarrow{H}^n \rightarrow \overrightarrow{H}^{n-1}\ \overleftarrow{\overrightarrow{A}}$
Right stop	$\overrightarrow{H}^0 \rightarrow H$

Figure 2: Schema for the Restricted Valence Grammar used in our experiments.

5 Model Estimation

Following Klein and Manning (2004), we estimate the parameters in our model, using the Inside-Outside algorithm (Baker, 1979), a specific version of the Expectation-Maximization algorithm (EM) (Dempster et al., 1977) for learning PCFGs.[6]

It is well known that setting a proper initial state when using the EM algorithm is critical (Carroll and Charniak, 1992). Since EM can only find local maxima, the initial state given to EM essentially determines which peak EM will discover (modulo any noise introduced by the estimation procedure). Techniques such as adding noise and random restarts will help EM find different and sometimes better peaks. However, we will often achieve an additional benefit from domain-specific information. In our case, this takes the form of general trends in the statistics of dependency attachments.

Klein and Manning (2004) and Smith and Eisner (2006) start their system with the preference for short attachments over long ones. They express this by initializing their system before the M step with a distribution over trees rather than before the E step with initial rules on probabilities. A somewhat generalized version of their initializers follows:

$$\forall t \in T(W), u(W,t) = \prod_{p=1}^{|W|} \prod_{c \in C_t(p)} \frac{1}{|p-c|} + \lambda$$

where $T(\cdot)$ is function which returns all possi-

[4] As in Klein and Manning (2004), our decision to go left first is arbitrary.

[5] N indicates the highest valence we keep track of instead of the maximum valence.

[6] Inside-outside estimation code from `http://www.cog.brown.edu/~mj/Software.htm`

ble parse trees over the tags in its input, $u(W, t)$ gives the weight of parse tree t over tags W, $C_t(\cdot)$ gives the children of word w in tree t, and λ is a constant ($\lambda = 0$ for Klein and Manning's, $\lambda = 1$ for Smith and Eisner's "Local" initializer). Broadly speaking, their initializer assigns weight to a parse inversely proportional to the distance between parents and their children under that parse. In other words, the initializer starts EM with a tendency towards trees with shorter dependencies. We will refer to this as the Harmonic Tree Initializer (HTI).

Unlike (Klein and Manning, 2004; Smith and Eisner, 2006), we start the EM algorithm before the E step with initial weights on dependency rules. Abstractly, our initializer uses the same idea, where the initial weight of a rule is instead inversely proportional to the average distance between the parent and children as seen in the training data.

Let $u(p, c, dir)$ be the initial weight of the rule for parent p attaching to child c in direction dir:

$$u(p, c, dir) = \sum_{\substack{s \in \mathbf{s} \\ p_i, c_i \in S(p,c,dir,s)}} \frac{1}{|p_i - c_i| + k}$$

where $S(p, c, dir, s)$ returns the indices of all occurrences of p and c in sentence s where the parent follows the child if $dir = left$ and precedes the child if $dir = right$, $\mathbf{s}$ is a set of all training sentences, and k is a constant that determines the uniformity of the distribution. Because we initialize rule weights in a harmonic fashion, we will refer to our initializer as the Harmonic Rule Initializer (HRI).

6 Evaluation

6.1 Experimental Setup and Metrics

The input of our task is sequences of part of speech tags of length 10 or less after punctuation has been removed. While the task appears simple, it has proved to be a difficult unsupervised problem. We use the Wall Street Journal corpus (Marcus et al., 1993) for English and four languages from the CoNLL 2006 Shared Task on Dependency Parsing (Buchholz and Marsi, 2006): Bulgarian (Simov et al., 2005), Portuguese (Afonso et al., 2002), Swedish (Nilsson et al., 2005), and German (Brants et al., 2002). The $_{10}$ suffix indicates a corpus after length restrictions.

To evaluate our system, we compare against gold dependencies. For English, the gold dependencies are given to us by the Collins head finder (Collins, 1999) as in (Klein and Manning, 2004; Smith and Eisner, 2006). Our metrics are directed and undirected accuracy. Directed accuracy is the percentage of dependencies predicted with the correct parent, child, and direction. Undirected accuracy is the same, but ignoring direction. As our baselines, we present a subset of the baselines and results from (Klein and Manning, 2004) in Table 4. In this table, "Left branching" denotes a structure where we always branch left (i.e. the head is the rightmost word of a constituent) and "Right branching" is the symmetric case.

6.2 Results

Table 3 shows the results of our system on English with different grammars and maximum valences. "(U)RVG" refers to our (Un)restricted Valence Grammar (Figure 2).

To make our model more similar to DMV, we also present a version of our model where P_{arg} is not conditioned on the index of the argument slot. This can be achieved via parameter tying in the EM algorithm, and we refer to this model as "(U)RVG (tied)".

Finally, we show the results of picking rule weights in a supervised fashion in "URVG (oracle)". We use maximum likelihood estimates of rule weights with simple add-λ smoothing and evaluate on each section of WSJ in a round-robin fashion. Our point is not to compete with supervised dependency models, but to give an upper-bound on the performance of our unsupervised parameter estimation.

Our results on English are quite strong. In directed accuracy, we get 56.5% dependencies correct, 22.9% more than the left branching baseline and 13.3% more than DMV. We see similar results in Swedish with a directed accuracy of 45.4%, 17.1% better than the left branching baseline of 28.3%. Given these results, we were surprised to learn that our system performs worse than the baselines for German, Portuguese, and Bulgarian. We are not certain of the cause of this. Our current theory is that the Harmonic Rule Initializer is not as portable as we had hoped. Given that it is the only part of DMV which do not implement, we feel it is likely

Model	Directed	Undirected
Left branching	33.6	56.7
Right branching	24.0	55.9
Klein and Manning (2004)	43.2	64.5
This paper	56.5	69.7

Table 4: Performance of Klein and Manning (2004) and our model versus baselines on English (WSJ_{10}). Directed and undirected scores refer to the percentage of directed and undirected dependencies correct, respectively.

the source of the problem. While the Harmonic Rule Initializer is useful for induction on English, it fails to provide a useful starting state for other languages. We believe this is because the Harmonic Rule Initializer is based on overly local information or that there is a bug in our implementation.

7 Discussion

Our best models result from restricting our maximum valence. RVG's performance is always at least that of URVG. Even with a maximum valence of one, RVG performs better than DMV in the directed case and only slightly worse in the undirected case. It is best to set the maximum valence to two, which can potentially model almost all dependencies and has far fewer rules than higher valences. If we give EM a model with higher valences, it seems that we find worse estimates under the load of additional parameters (6,880 vs. 10,160 initial rules). From Table 1, we know that a maximum valence of three can only improve performance by about 3%. Given our current levels of accuracy, it is better to focus on getting the more common first and second attachments correctly. It may be possible to use a model with lower maximum valence to initialize a model of higher valence, though we leave this for future work.

Table 3 also confirms our belief that tying parameters is too restrictive. When maximum valence is set to 2, parameter tying hurts performance by about 4%.

Our improvements appear to come mostly from limiting the maximum valence and learning parameters untied but not from our initializer. URVG with a valence of one is nearly DMV, but with HRI it performs significantly worse. Much of this damage is alleviated by switching to RVG, but this begs the question of how our system would perform if properly initialized by the Harmonic Tree Initializer. For now, we leave this to future work.

8 Conclusions and Future Work

We have presented several techniques which improve unsupervised dependency grammar induction. First, we have shown that there is a benefit in restricting the maximum valence of the grammar. Since approximately 97% of the time heads only make 0 to 2 attachments crosslinguistically, it is beneficial to limit the degrees of freedom in this respect (at least, as a first pass to grammar induction). Second, we have shown that parameter tying, i.e. forcing each argument slot to have the same distribution, is overly restrictive. Finally, we have shown that the distribution over the number of arguments taken by a head is fairly constant across languages. This may be useful to both supervised and unsupervised dependency parsing. We are not aware of any previous work which shows this. Unfortunately, while our model performs very well on English and Swedish, it is not competitive crosslinguistically in its current form. Given that the main difference is our use of a different initializer, our working hypothesis is that switching to a better initializer will allow our techniques to function across languages. Testing this hypothesis is at the heart of our ongoing work.

We believe that a promising extension of this work would be to use the Bayesian equivalent of the Inside-Outside algorithm (Johnson et al., 2007), which would allow us to incorporate priors into this task. Priors can encourage the resulting grammar to obey crosslinguistic statistics of dependencies, for example, those seen in Table 1 or ensuring that the grammar is mostly left- or right-branching. Additionally, a Bayesian framework may be more amenable to the joint learning of tags and syntax. With access to words and their tags, this may allow us to induce more fine-grained valence information.

References

S. Afonso, E. Bick, R. Haber, and D. Santos. 2002. "Floresta sintá(c)tica": a treebank for Portuguese. In *Proc. of the 3rd Intern. Conf. on Language Resources and Evaluation (LREC)*, pages 1698–1703.

	Valence					
	1		2		3	
Model	Dir	Undir	Dir	Undir	Dir	Undir
URVG	26.0	48.4	54.1	68.4	52.4	67.7
URVG (tied)			50.1	65.7	51.4	66.4
RVG	46.4	63.8	**56.5**	**69.7**	52.5	67.8
RVG (tied)			51.4	66.5	51.5	66.5
URVG (oracle)	72.6	75.2	81.7	84.1	82.7	84.9

Table 3: Directed and undirected accuracy on English (WSJ_{10}). Our best performance results from using the Restricted Valence Grammar with a maximum valence of 2 and initialization via the Harmonic Rule Initializer. Using different amounts of valence, the Unrestricted Valence Grammar, or parameter tying hurts performance.

J.K. Baker. 1979. Trainable grammars for speech recognition. In Jared J. Wolf and Dennis H. Klatt, editors, *Speech Communication Papers presented at the 97th Meeting of the Acoustical Society of America*, pages 547–550, MIT, Cambridge, Massachusetts.

A. Böhmová, J. Hajič, E. Hajičová, and B. Hladká. 2003. The PDT: a 3-level annotation scenario. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, volume 20 of *Text, Speech and Language Technology*, chapter 7. Kluwer Academic Publishers, Dordrecht.

S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. 2002. The TIGER treebank. In *Proc. of the 1st Workshop on Treebanks and Linguistic Theories (TLT)*.

S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *CoNLL-X*. SIGNLL.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. Technical Report CS-92-16, Department of Computer Science, Brown University, March.

Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, The University of Pennsylvania.

A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38.

Aria Haghighi and Dan Klein. 2006. Prototype-driven grammar induction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 881–888, Sydney, Australia, July. Association for Computational Linguistics.

J. Hajič, O. Smrž, P. Zemánek, J. Šnaidauf, and E. Beška. 2004. Prague Arabic dependency treebank: Development in data and tools. In *Proc. of the NEMLAR Intern. Conf. on Arabic Language Resources and Tools*, pages 110–117.

William P. Headden III, David McClosky, and Eugene Charniak. 2008. Evaluating unsupervised part-of-speech tagging for grammar induction. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*, Manchester, UK, August.

Mark Johnson, Tom L. Griffiths, and Sharon Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proceedings of NAACL 2007* (to appear).

Y. Kawata and J. Bartels. 2000. Stylebook for the Japanese treebank in VERBMOBIL. Verbmobil-Report 240, Seminar für Sprachwissenschaft, Universität Tübingen.

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 478–485, Barcelona, Spain, July.

Dan Klein. 2005. *The Unsupervised Learning of Natural Language Structure*. Ph.D. thesis, Stanford University, March.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Comp. Linguistics*, 19(2):313–330.

J. Nilsson, J. Hall, and J. Nivre. 2005. MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. In *Proc. of the NODALIDA Special Session on Treebanks*.

Joakim Nivre. 2005. Dependency grammar and dependency parsing. Technical Report 05133, Vaxjo University: School of Mathematics and Systems Engineering.

K. Simov, P. Osenova, A. Simov, and M. Kouylekov. 2005. Design and implementation of the Bulgarian HPSG-based treebank. In *Journal of Research on Language and Computation – Special Issue*, pages 495–522. Kluwer Academic Publishers.

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 44th Annual Meeting of the Association for Computational Linguistics*, pages 569–576, Sydney, Australia, July. Association for Computational Linguistics.

Noah A. Smith. 2006. *Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text*. Ph.D. thesis, Department of Computer Science, Johns Hopkins University, October.