Grammar Induction from Text Using Small Syntactic Prototypes

Prachya Boonkwan[†,‡]
p.boonkwan@sms.ed.ac.uk
prachya.boonkwan@nectec.or.th

Mark Steedman[†]
steedman@inf.ed.ac.uk

† School of Informatics
University of Edinburgh
10 Crichton Street
Edinburgh EH8 9AB, UK

‡ National Electronics
and Computer Technology Center
112 Phaholyothin Road
Pathumthani 12120, Thailand

Abstract

We present an efficient technique to incorporate a small number of cross-linguistic parameter settings defining default word orders to otherwise unsupervised grammar induction. A syntactic prototype, represented by the integrated model between Categorial Grammar and dependency structure, generated from the language parameters, is used to prune the search space. We also propose heuristics which prefer less complex syntactic categories to more complex ones in parse decoding. The system reduces errors generated by the state-of-the-art baselines for WSJ10 (1% error reduction of F1 score for the model trained on Sections 2–22 and tested on Section 23), Chinese10 (26% error reduction of F1), German10 (9% error reduction of F1), and Japanese10 (8% error reduction of F1), and is not significantly different from the baseline for Czech10.

1 Introduction

Unsupervised grammar induction has gained general interest for several decades, offering the possibility of building practical syntactic parsers by reducing the labor of constructing a treebank from scratch. One approach is to exploit the Inside/Outside Algorithm (Baker, 1979; Carroll and Charniak, 1992), a variation of EM algorithm for PCFG, to estimate the parameters of the parser's language models. More recent advances in this approach are the constituent-context model (CCM) (Klein and Manning, 2001; Klein and Manning, 2002), dependency model with valence (DMV) based on Collin's head dependency model (1999), and the CCM+DMV mixture (Klein and Manning, 2004; Klein, 2005). Several search techniques and models have been added to CCM+DMV for dealing with local optima and data sparsity (Smith, 2006; Cohen et al., 2008; Headden III et al., 2009). Spitkovsky et al. (2010) proposed a training strategy where the model fully trained on shorter sentences and roughly trained on longer sentences tend to outperform the model fully trained on the entire dataset. Recently, Gillenwater et al. (2010) proposed the use of posterior regularization in EM in which the posterior distribution of parent-child POS tags are regulated to an expected distribution.

However, purely unsupervised learning still does not perform well because the parameter estimation can be misled by unexpected frequent cooccurrence. A common example of it is the collocation of a verb (VBZ) and a determiner (DT) in a verb phrase. This collocation results in incorrect trees such as ((VBZ DT) NN).

To avoid this problem, the use of syntactic prototypes has been proposed. Instead of enumerating every possibility, syntactic structures are cautiously constructed regarding some syntactic constraints. Haghighi and Klein (2006) proposed the use of bracketing rules extracted from WSJ10 in CCM and considerably improved accuracy. Druck et al. (2009) used dependency formation rules handcrafted by linguists to improve the accuracy of DMV. Snyder et al. (2009) do semi-supervised grammar induction from bilingual text with the help of a supervised parser on one side and word alignment. However, bilingual corpora are not available for many language pairs. Naseem et al. (2010) proposed the use of cross-linguistic knowledge represented as a set of allowable head-dependent pairs. However, this method still requires provision of language-specific rules to boost accuracy. If language-specific rules are necessary to achieve accuracy, we need more efficient ways to encode this knowledge.

This paper proposes a method for inducing language-specific word order regularities captur-

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 438–446,
Chiang Mai, Thailand, November 8 – 13, 2011.

ing cross-linguistically frequent constructions to constrain unsupervised grammar induction. We use the notion of syntactic prototype, a set of grammar rules automatically generated for such constructions. Categorial Dependency Grammar (CDG), which combines rules of constituency and dependency, is used to represent syntactic prototypes. We also propose a novel category penalty score for use in decoding, which defines the most probable parse according to a preference for less complex categories.

The paper is organized as follows. §2 details the method of encoding linguistic prior knowledge as a syntactic prototype. §3 explains an overview of our approach. §4 shows experiment results and discusses the errors. We conclude in §5.

2 Syntactic Prototypes

A *syntactic prototype* is a set of grammar rules representing default language parameters (such as word order for the most cross-linguistically frequent linguistic constructions, following Naseem et al. (2010)'s notion of cross-linguistic knowledge. This section shows how CDG rules are derived from a set of word order constraints.

2.1 Categorial Dependency Grammar

Categorial Dependency Grammar (CDG) is an extension of pure Categorial Grammar (CG) (Ajdukiewicz, 1935; Bar-Hillel, 1953) used for defining language-specific prototypes to be discovered. Its syntactic derivations define constituency and dependency in parallel. In CG, each constituent is assigned one or more syntactic categories, defined as an atomic category or a function category. For example, the proper name 'John' is assigned the atomic category np. If X and Y are categories of either kind, then X/Y and $X\backslash Y$ are function categories that map constituents of type Y respectively on the right and on the left into those of type X. For example, the intransitive verb 'walks' is assigned the function $S\backslash NP$.

We extended CG construct dependency structure alongside the syntactic derivation by encoding the direction of dependency in slashes. Using the head-outward notation for dependency of (Collins, 1999), the slash is subscripted $<$ ($>$) if the corresponding dependency is to be linked from the head on the right to its dependent on the left (the head on the left to its dependent on the right). For example, an English adjective (e.g. 'big') can be assigned the category $np/_{>}np$, while a transitive verb can be assigned $s\backslash_{>}np/_{<}np$. CDG differs from PF-CCG (Koller and Kuhlmann, 2009) in that dependency direction is specified independently from the order of function and argument, while theirs is determined by slash directionality. For them such an adjective has the implicit category $np/_{>}np$ and acts as the head of the noun phrase. The derivation rules for context-free CDG are listed below:

$$\begin{array}{llll} X/_{<}Y:d_1 & Y:d_2 & \Rightarrow & X:h(d_1)\rightarrow h(d_2) \\ X/_{>}Y:d_1 & Y:d_2 & \Rightarrow & X:h(d_1)\leftarrow h(d_2) \\ Y:d_1 & X\backslash_{<}Y:d_2 & \Rightarrow & X:h(d_1)\rightarrow h(d_2) \\ Y:d_1 & X\backslash_{>}Y:d_2 & \Rightarrow & X:h(d_1)\leftarrow h(d_2) \end{array} \quad (1)$$

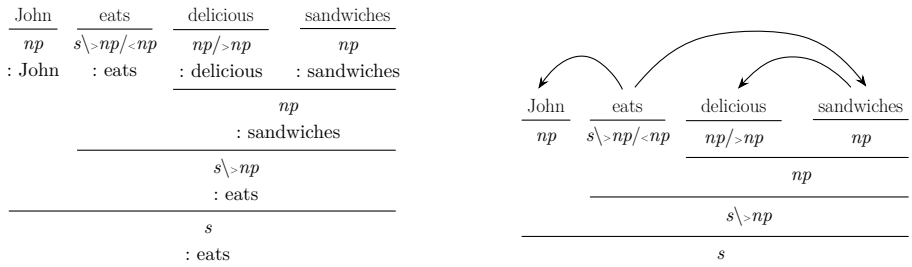
where the notations $h(d_1)\rightarrow h(d_2)$ and $h(d_1)\leftarrow h(d_2)$ mean a dependency linking from the head of the dependency structure d_1 to the head of d_2, and that linking from the head of d_2 to the head of d_1, respectively. Let us denote a constituent type with $C:w$, where C is a syntactic category and w is the head word of the constituent.

Given the CDG in (2), we obtain the syntactic derivation of the string 'John eats delicious sandwiches' in Figure 1.

$$\begin{array}{rcl} \text{John, sandwiches} & \vdash & np \\ \text{delicious} & \vdash & np/_{>}np \\ \text{eats} & \vdash & s\backslash_{>}np/_{<}np \end{array} \quad (2)$$

Figure 1(a) shows dependency-driven derivation, in which the heads of constituents are propagated. Figure 1(b) reflects the formation of the dependency structure corresponding to the dependency-driven derivation.

The attraction of using CDG for grammar induction is the integration of the constituent model and the dependency model. As shown in figure 1, the syntactic derivation defines the dependency structure, because we can directly construct a dependency structure from any head-driven syntactic derivations using the annotated directions. CDG can boost the accuracy of grammar induction by modeling rules of both constituent formation and dependency. However, the search space would become impossibly large if we had to enumerate all possible syntactic categories, including all possible arguments and dependency direction. This danger can be avoided by using small amounts of hand-crafted prior linguistic knowledge.



(a) Dependency-driven derivation.

(b) Equivalent dependency structure.

Figure 1: Syntactic derivation of 'John ate delicious sandwiches' based on CDG. Each constituent type is denoted by $C : w$, where C is a syntactic category and w is the head word, such as $s\backslash_{>}np/_{<}np$: eats.

However, a simple parametric syntactic prototype will give rise to parsing failures when faced with parametrically exceptional items, which occur in most if not all languages. We allow for such exceptions to be accommodated by defining an additional wildcard category $\star$ which combines with any syntactic category to yield the wildcard itself, according to the following additional combinatory rules:

$$\begin{aligned} \star : d_1 \quad X : d_2 \quad &\Rightarrow \quad \{\star : h(d_1) \leftarrow h(d_2), \\ &\qquad \star : h(d_1) \rightarrow h(d_2)\} \\ X : d_1 \quad \star : d_2 \quad &\Rightarrow \quad \{\star : h(d_1) \leftarrow h(d_2), \\ &\qquad \star : h(d_1) \rightarrow h(d_2)\} \end{aligned} \tag{3}$$

The wildcard is assigned to unknown words and large irreducible constituents so as to allow complete parses of otherwise unparsable sentences. As shown in (3), each wildcard derivation generates two possible dependency structures; i.e. d_1 and d_2 can be the head of a phrase. The wildcard will be revisited in §3.1.

2.2 Language Parameterization

We generate the CDG for each language automatically from language parameters. To facilitate this process, we have devised a questionnaire consisting of 30 questions concerning word orders for constructions that occur in most languages. Sorted by their importance, the questions can be grouped into the following categories:

1. The orders of subject, verb, direct object, and optional indirect object (1 question)
2. The argument orders of subject- and object-control verbs (2 questions)
3. The orders of adjectives, adverbs, and auxiliary verbs (4 questions)
4. The use of cardinal numbers and noun classifiers (2 questions)
5. The argument orders of adpositions, nominal modifiers, adverbials, possessive markers, relative pronouns, and subordinate conjunctions. (7 questions)
6. The orders of gerunds, infinitive markers, nominalizers, and sentential modifiers (6 questions)
7. The orders of particles, the existence of a copula, the usages of gerunds, the order of negative markers, the use of dative shifts, and the omission of discourse-given subject and object (8 questions)

These language parameters are used to automatically generate a CDG representing a syntactic prototype including language-specific types of cross-linguistically frequent categories[1] will be generated. For example, if a language has the word order SVIO, the syntactic category $s\backslash_{>}np$, $s\backslash_{>}np/_{<}np$, and $s\backslash_{>}np/_{<}np/_{<}np$ are generated and assigned by default to intransitive, transitive, and ditransitive verbs, respectively. Each slash in all syntactic categories is assigned with dependency directions according to Collins (1999)'s head percolation heuristics. All questions are optional; i.e. if any of the questions are left blank, all possible categories for that question will be generated.

Once the cross-linguistic category classes are generated, we then map them to the POS tags in a particular corpus. This part is an engineering task where the mapping should be best fitted to the

[1] E.g. intransitive verb, transitive verb, ditransitive verb, subject- and object-control verb, adjective, adverb, preposition, relative pronoun, gerund, copula, subordinate conjunction, noun classifier, infinitive marker, cardinal number, etc.

corpus. However, we will show in the experiment section that the preparation process for syntactic prototypes is quantifiable and reasonable in comparison to the improvement in accuracy attained.

3 Grammar Induction

3.1 Structure Enumeration

The first step in grammar induction is to enumerate all possible parses for each sentence. We use a table mapping from POS tags to language-specific categories to define the lexicon, and build a parse chart for each sentence with CKY Algorithm. A packed chart is used for both speed and space compactness. We apply a right-branching preference to eliminate spurious ambiguity caused by coordination and nominal compounding. In the event of a sentence yielding no parse using that lexicon, we assign the wildcard category '$\star$' to all unknown words and maximal irreducible constituents, and reparse the sentence.

3.2 Parsing Model

We extend the probabilistic context-free grammar with role-emission probabilities, defined as the product of the probability of each daughter category performing as a head or a dependent in a derivation. This model was motivated by Collins (1999)'s head-outward dependency model and Hockenmaier (2003)'s generative model for parsing CCG. Given a CDG G, we define the probability of a tree t having the constituent type $C : w$ by:

$$P(t|s,G) = \frac{1}{Z} \prod_{\substack{C:w\to\alpha \\ \in R(t)-L(t)}} \begin{array}{l} \pi_{\text{exp}}(\alpha|C:w,G) \\ \times \pi_{\text{head}}(H:w|G) \\ \times \pi_{\text{dep}}(D:w'|G) \end{array} \times \prod_{C:w\in N(t)} \pi_{\text{HE}}(w|C,G)^{\#_t(C:w)} \quad (4)$$

where Z is a normalization constant and each production α contains $H : w$ and $D : w'$, and $H : w$ and $D : w'$ are the head and the dependent, respectively. There are four types of parameters as follows.

1. $\pi_{\text{exp}}(\alpha|C : w, G)$: probability of the type $C : w$ generating a production α.

2. $\pi_{\text{head}}(C : w|G)$: probability of the type $C : w$ performing as a head.

$$\pi_{\text{head}}(C:w|G) = \frac{\sum_{C:w'} \#(C:w' \to \alpha_{\text{head}}^{C:w})}{\sum_{C':w'} \#(C':w' \to \alpha^{C:w})} \quad (5)$$

3. $\pi_{\text{dep}}(C : w|G)$: probability of the type $C : w$ performing as a dependent.

$$\pi_{\text{dep}}(C:w|G) = \frac{\sum_{C':w'} \#(C':w' \to \alpha_{\text{dep}}^{C:w})}{\sum_{C':w'} \#(C':w' \to \alpha^{C:w})} \quad (6)$$

4. $\pi_{\text{HE}}(w|C, G)$: probability of a category C generating the head w.

$$\pi_{\text{HE}}(w|C,G) = \frac{\sum_{t\in Q} \#_t(C:w)}{\sum_{t\in Q}\sum_{w'} \#_t(C:w')} \quad (7)$$

where $\alpha^{C:w}$ is a production that contains $C : w$, and $\alpha_{\text{head}}^{C:w}$ and $\alpha_{\text{dep}}^{C:w}$ have $C : w$ as the head and the dependent, respectively. $\#_t(C : w)$ is the frequency count of the category $C : w$ in the tree t. $N(t)$ is the set of all nonterminal nodes.

3.3 Parameter Estimation

Learning is achieved using the Variational Bayesian EM Algorithm (VB-EM) (Attias, 2000; Ghahramani and Beal, 2000) to estimate the parameters $\pi_{\text{exp}}, \pi_{\text{head}}, \pi_{\text{dep}}$, and π_{HE}. We followed the approach of Kurihara and Sato (2006) for training PCFGs with the VB-EM. This approach places Dirichlet priors over the multinomial grammar rule distributions. We set the Dirichlet hyperparameters to 1.0 for all rules containing the wildcard category and 5.0 for all others. In all other regards, we followed Kurihara and Sato (2006). The VB-EM algorithm iterates two processes of expectation calculation and parameter maximization. It is favored for the present purpose because it is less data-overfitting than the standard Inside/Outside Algorithm regarding its free-energy criteria for model selection. We calculate expected counts using Dynamic Programming (Baker, 1979; Lari and Young, 1990).

To further avoid the data over-fitting issue, we smoothed the probability of each substructure with the additive smoothing technique (Lidstone, 1920; Johnson, 1932; Jeffreys, 1948). An approximated parameter $\hat{\pi}(\tau)$ is calculated by

$$\hat{\pi}(\tau) = \frac{\pi(\tau)+\epsilon}{1+\epsilon} \quad (8)$$

where ϵ is a small constant value. In our experiments, we chose $\epsilon = 10^{-25}$.

3.4 Decoding with Category Penalty

By using prototypical syntactic categories in derivation, the system can be misled by complex

categories. A parse containing more complex categories is preferred to one containing less complex categories, because both simple and complex syntactic categories have the same chance of occurrence in parse enumeration. When learning the parameters, rules containing complex categories tend to have relatively excessive probability, as opposed to the Zipfian distribution of syntactic categories in which less complex categories are more frequently found in CCGbank. We therefore introduce a *category penalty* score.

The category penalty score is motivated by the observation that, in practical use of language, simpler categories tend to be used more frequently than the more complex ones. The penalty score $v(c)$ of the category c is defined as follows:

$$v(c) \quad = \quad k^{S(c)} \tag{9}$$

where $S(c)$ is the count of all forward and backward slashes in c and k is the penalty constant.

We weight each tree by the product of the penalty scores of the syntactic category on each node; i.e.

$$P(t|s,G) \quad = \quad \begin{cases} v(A) \cdot \pi(A \rightarrow w) & \text{lexicon} \\ v(A) \cdot \pi(A \rightarrow \alpha) & \text{branching} \\ \quad \cdot \prod_{i=1}^{|\alpha|} P(t_i|s,G) & \end{cases} \tag{10}$$

We use Viterbi decoding to find the most probable parse from a packed chart.

4 Experiments and Discussion

4.1 Datasets and Accuracy Metrics

In order to evaluate the method in comparison to the state of the art, we chose WSJ10, the standard collection of trees from WSJ part of PTB (Marcus et al., 1993) whose sentence length does not exceed ten words after taking out punctuation marks and empty elements. In stead of surface forms, we used a set of POS sequences taken from all WSJ10's trees to avoid the data sparsity issue. We converted the Penn Treebank into dependency structures with Collins (1999)'s head percolation heuristics. Following the literature, we trained the system with sections 2–22 and evaluated the resultant model on section 23.

For multilingual experiments, we made use of dependency corpora from the 2006 CoNLL X Shared Task (Buchholz and Marsi, 2006). Shown in Table 1, Chinese (Keh-Liann and Hsieh, 2004), Czech (Bohomovà et al., 2001), German (Brants et al., 2002), and Japanese (Kawata and Bartels, 2000) were chosen for the sake of language typology variety. We also chose sentences whose length does not exceed ten words after taking out punctuation marks. As a free-word-ordered, inflectional language, the Czech dataset was particularly prepared by augmenting the POS tags with inflectional information, resulting in significantly much more granularity. However, Czech's syntactic prototype does not make use of such syntactic information to restrain the search space.

Table 1: Sizes and granularity of POS of multilingual corpora

Languages	Sentences	POS Tags
WSJ10	7,422	36
Chinese10	52,424	28
Czech10	27,375	1,149
German10	13,473	51
Japanese10	12,884	77

We measured the capability of our system by two metrics: directed dependency accuracy, and undirected dependency accuracy (Klein, 2005). For directed dependency accuracy, we count a directed dependency of a word pair to be correct if it exists in the gold standard. For undirected dependency accuracy, we neglect the direction of the dependency. All accuracy numbers are reported in terms of precision, recall, and F1 scores.

4.2 Construction of Syntactic Prototypes

In order to construct syntactic prototypes for each language, we conduct an interview with a non-linguist native speaker. We ask him/her each question in the questionnaire mentioned in §2.2 by giving them a sample sentence and letting them build up the corresponding sentence in their language. We then ask questions to elicit word alignment and analyze the answers. Normally it takes up to two hours per previously unseen language to complete the questionnaire.

We then study the manual of the treebank's POS tags and mapped each to one or more language-specific category classes. Normally this process takes around four to six hours to thoroughly scrutinize the usage of each POS tag and assign them to appropriate classes. It therefore takes six to ten hours to build a syntactic prototype for each language.

4.3 Results

This section presents results of English and multilingual experiments using syntactic prototypes as a guide to grammar induction.

4.3.1 Experiments on English

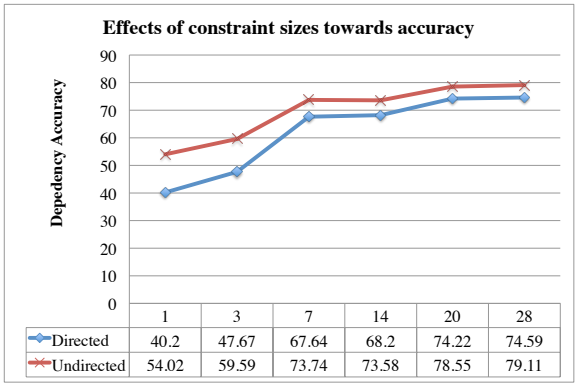
Table 2 shows experiment results of English grammar induction on WSJ10. In the beginning, we compared the produced trees against the gold standard produced from Collins's parser. We trained the system with Sections 2–22 of WSJ10 and tested it on Section 23. In decoding, we set the category penalty constant to 10^{-15}. The F1 score outperforms the baseline set by (Naseem et al., 2010). To exhibit the stability of the approach, we also ran ten-fold cross validation on English; i.e. we divided WSJ10 into ten parts and, for each fold, we chose nine parts for as a training set and the other one as a test set. We attained higher F1 score, as expected for cross-validation, which effectively tests on the development set.

4.3.2 Effects of Numbers of Constraints and Category Penalties

Figure 2 shows the effects of numbers of constraints towards directed and undirected F1 scores. We varied the number of constraints in English syntactic prototypes. The category class 4 (the usages of cardinal numbers and noun classifiers) was neglected, because we can treat cardinal numbers as adjectives or nouns and there are no true noun classifiers in English. Therefore there are 28 constraints in total for English. We again trained on Sections 2–22 of WSJ10 and tested on Section 23. In decoding, we set the category penalty constant to 10^{-15}. We then evaluated the accuracy against the gold standard produced by Collins's parser.

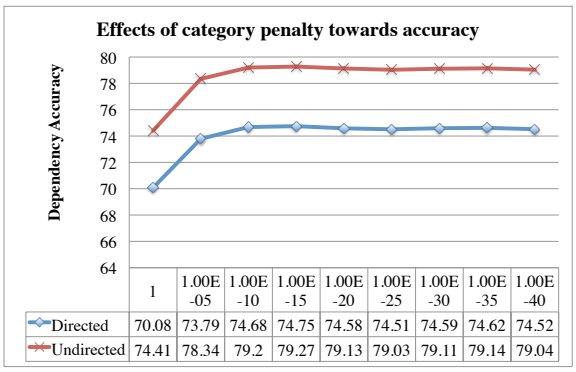
As we increased the number of constraints in syntactic prototypes, we found that both directed and undirected F1 scores increase and start to saturate after the first 20 rules. In keeping with the Zipfian distribution, the first 20 rules cover frequent linguistic phenomena. When we pruned the search space with linguistic constraints, the directed accuracy starts to approach the undirected accuracy. We also note that errors generated by the system reflect the same attachment ambiguity errors as supervised parsing.

Figure 3 shows the effects of category penalty constants on accuracy. We again trained on Sections 2–22 of WSJ10 and tested on Section 23. We then evaluated the accuracy against the gold standard produced by Collins's parser. We notice that both accuracy scores saturate at the penalty constant of 10^{-15} and slightly decay afterwards.



	1	3	7	14	20	28
Directed	40.2	47.67	67.64	68.2	74.22	74.59
Undirected	54.02	59.59	73.74	73.58	78.55	79.11

Figure 2: Effects of numbers of constraints on the directed and undirected dependency accuracy. The category penalty constant is fixed at 10^{-15}.



	1	1.00E -05	1.00E -10	1.00E -15	1.00E -20	1.00E -25	1.00E -30	1.00E -35	1.00E -40
Directed	70.08	73.79	74.68	74.75	74.58	74.51	74.59	74.62	74.52
Undirected	74.41	78.34	79.2	79.27	79.13	79.03	79.11	79.14	79.04

Figure 3: Effects of category penalty on the directed and undirected dependency accuracy.

4.3.3 Multilingual Experiments

We also conducted multilingual experiments on Chinese, Czech, German, and Japanese to show the stability of the approach. We ran ten-fold cross validation on each language and calculated the average F1 scores. Our baseline systems are as follows: (Naseem et al., 2010) for English, (Snyder et al., 2009) for Chinese, and (Gillenwater et al., 2010) for Czech, German, and Japanese. In Table 3, our system significantly outperforms almost all the baselines, except in the Czech experiment. We believe that the under-performance of our system on Czech is caused by the data sparsity issue. Designed based on rather fixed word ordered languages, the syntactic prototype needs to generate almost all possible syntactic categories to capture Czech's free word orderedness. Although its POS

Table 2: Undirected and directed dependency accuracy of grammar induction on English Penn Treebank. The baseline for English is (Naseem et al., 2010).

	Undirected			Directed			Baseline
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Directed F1**
WSJ10 (Sect. 23)	79.24	79.29	79.27	74.72	74.77	**74.75**	73.80
WSJ10 (10X)	79.59	79.65	79.62	75.44	75.50	75.47	—

tags are grouped to easily map to cross-linguistic category classes, each class still contains a lot of syntactic categories. Because we do not use inflectional information in restraining the search space, the data sparsity becomes significant in Czech and therefore deteriorates the accuracy.

4.4 Error Analysis

We counted erroneous dependency pairs generated in the English experiment (10X) in §4.3.1, and we classified errors into two types: over-generation and under-generation. From Table 4, we can notice that the majority of errors are caused by adverbial and prepositional attachment (e.g. RB > VB, CD < IN, and NN < IN), and NP structural ambiguity (e.g. NN > NNP, DT > NN, and NNP > NNP).[2] These errors are common in supervised parsing. There is also under-generation of adverbial preposition phrases. We believe that the category penalty score accounts for this issue, resulting in the NP-modifying preposition (such as $np\backslash_{<}np/_{>}np$) being preferred to the adverbial one (such as $s\backslash_{>}np\backslash_{<}(s\backslash_{>}np)/_{<}np$).

Table 4: Top-10 over-generation and under-generation in the English experiment (10X) when compared against Collins's gold standard

Over-generation		**Under-generation**	
Errors	**Counts**	**Errors**	**Counts**
RB > VB	402	VBD < IN	364
CD < IN	200	DT > NN	331
NN < IN	197	VBD < TO	283
RB > VBN	188	VBD < RB	275
NN < TO	181	VBZ < RB	244
NNP > CD	180	IN < NN	219
MD > VB	166	JJ > NN	203
NNS < IN	149	MD < RB	194
NNP > NN	145	MD < VB	185
NN > NNP	141	NNP > NNP	179

5 Conclusion

We have demonstrated an efficient approach to grammar induction using linguistic prior knowledge encoded as a prototype or lexical schema. This prior knowledge was used to capture frequent linguistic phenomena. To integrate the strength of constituent and dependency models, Categorial Dependency Grammar was used as the backbone formalism. We also proposed a category penalty score preferring less complex categories, based on the observation of the Zipfian distribution of category types in CCGbank.

Syntactic prototypes can capture the most frequent constructions and improve accuracy on almost all of the selected languages. We found that dependency accuracy correlates with the Zipfian distribution as the number of constraints increases, as the increase in accuracy saturates after the first 20 rules. Error analysis suggests that the main sources of error are in adverbial and prepositional attachment, and NP structural ambiguity, which are also problematic for supervised parsing.

Future work remains as follows. First, we are looking forward to improving the capability of our syntactic prototype to also handle free word ordered languages by generating syntactic categories with more flexible combination and restraining the search space with inflectional information. Second, we plan to experiment on grammar induction from untagged words by decomposing the model into tagging and parsing subproblems (Ganchev et al., 2009; Rush et al., 2010; Auli and Lopez, 2011). Third and finally, we will experiment on longer sentences to show the scalability of our approach in dealing with larger data.

Acknowledgement

We would like to thank Tom Kwiatkowski, Michael Auli, Christos Christodoulopoulos, Alexandra Birch, Mark Granroth-Wilding, and

[2] Similar to the dependency directions in §2.1, the notations < and > are pointers to the syntactic head of the phrase. For example, DT > NN means that NN is the head and DT is its dependent.

Table 3: Undirected and directed dependency accuracy of grammar induction for English, Chinese, Czech, German, and Japanese. Our baseline systems are as follows: †(Naseem et al., 2010) for English, ‡(Snyder et al., 2009) for Chinese, and ⋆(Gillenwater et al., 2010) for Czech, German, and Japanese.

	Undirected			**Directed**			**Baseline**
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**directed F1**
WSJ10 (10X)	79.59	79.65	79.62	75.44	75.50	**75.47**	73.80†
Chinese10 (10X)	68.80	68.88	68.84	62.21	62.29	**62.25**	35.77‡
Czech10 (10X)	59.04	61.94	60.46	53.27	55.88	54.54	**54.70**⋆
German10 (10X)	65.13	65.20	65.17	56.68	56.74	**56.71**	47.40⋆
Japanese10 (10X)	75.65	78.97	77.27	67.11	70.05	**68.55**	60.80⋆

Emily Thomforde (University of Edinburgh), Adam Lopez (Johns Hopkins University), and Michael Collins (Columbia University) for useful comments and discussion related to this work, and the three anonymous reviewers for their useful feedback. This research was funded by the Royal Thai Government Scholarship to Prachya Boonkwan and EU ERC Advanced Fellowship 249520 GRAMPLUS to Mark Steedman.

References

Kazimierz Ajdukiewicz. 1935. Die Syntaktische Konexität. *Polish Logic*, pages 207–231.

Hagai Attias. 2000. A variational Bayesian framework for graphical models. In *Advances in Neural Information Processing Systems (NIPS 2000)*.

Michael Auli and Adam Lopez. 2011. A comparison of loopy belief propagation and dual decomposition for integrated CCG supertagging and parsing. In *Proceedings of ACL-2011*, June.

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

Yehoshua Bar-Hillel. 1953. A Quasi-Arithmetical Notation for Syntactic Description. *Language*, 29:47–58.

A. Bohomovà, J. Hajic, E. Hajicova, and B. Hladka. 2001. The Prague dependency treebank: Three-level annotation scenario. In Anne Abeillé, editor, *Treebanks: Building and Using Syntactically Annotated Corpora*.

T. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. 2002. The TIGER Treebank. In *Proceedings Workshop on Treebanks and Linguistic Theories*.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL-2006*, pages 149–164.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic depedency grammars from corpora. In C. Weir, S. Abney, R. Grishman, and R. Weischedel, editors, *Working Notes of the Workshop Statistically-Based NLP Techniques*, pages 1–13. AAAI Press, Menlo Park, CA.

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 21*.

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

Gregory Druck, Gideon Mann, and Andrew McCallum. 2009. Semi-supervised learning of dependency parsers using generalized expectation criteria. In *Proceedings of 47th Annual Meeting of the Association of Computational Linguistics and the 4th IJCNLP of the AFNLP*, pages 360–368, Suntec, Singapore, August.

Kuzman Ganchev, Joao Graca, Jennifer Gillenwater, and Ben Taskar. 2009. Posterior regularization for structured latent variable models. Technical Report MS-CIS-09-16, University of Pennsylvania Department of Computer and Information Science.

Zoubin Ghahramani and Matthew J. Beal. 2000. Variational inference for Bayesian mixtures of factor analyses. In *Advances in Neural Information Processing Systems (NIPS 2000)*.

Jennifer Gillenwater, Kuzman Ganchez, Joao Graça, Fernando Pereira, and Ben Taskar. 2010. Sparsity in dependency grammar induction. In *Proceedings of ACL-2010 Short Papers*, pages 194–199.

Aria Haghighi and Dan Klein. 2006. Prototype-driven grammar induction. In *Proceedings of 44th Annual Meeting of the Association for Computational Linguistics*, pages 881–888.

William P. Headden III, Mark Johnson, and David McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, Colorado, June.

Julia Hockenmaier. 2003. Parsing with generative models of predicate-argument structure. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 359–366, Sapporo, Japan.

H. Jeffreys. 1948. *Theory of Probability*. Clarendon Press, Oxford, second edition.

W. E. Johnson. 1932. Probability: deductive and inductive problems. *Mind*, 41:421–423.

Y. Kawata and J. Bartels. 2000. Stylebook for the Japanese Treebank in VERBMOBIL. Technical report, Eberhard-Karls-Universität Tübingen.

Chen Keh-Liann and Yu-Ming Hsieh. 2004. Chinese treebanks and grammar extraction. In *Proceedings of IJCNLP-2004*, pages 560–565.

Dan Klein and Christopher D. Manning. 2001. Natural language grammar induction using a constituent-context model. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems (NIPS 2001)*, volume 1, pages 35–42. MIT Press.

Dan Klein and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of the 40th Associations for Computational Linguistics*, pages 128–135.

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 of the Association for Computational Linguistics*.

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

Alexander Koller and Marco Kuhlmann. 2009. Dependency trees and the strong generative capacity of ccg. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 460–468, April.

Kenichi Kurihara and Taisuke Sato. 2006. Variational Bayesian grammar induction for natural language. In *International Colloquium on Grammatical Inference*, pages 84–96.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

G. J. Lidstone. 1920. Note on the general case of the Bayes-Laplace formula for inductive or *a posteriori* probabilities. *Transactions of the Faculty of Actuaries*, 8:182–192.

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

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of EMNLP-2010*.

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 EMNLP-2010*.

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

Benjamin Snyder, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *Proceedings of the Joint Conference of the 47th ACL and the 4th IJCNLP*.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010. From baby steps to leapfrog: How "less is more" in unsupervised dependency parsing. In *Proceedings of NAACL-HLT 2010*.