PAPER Special Section on Data Engineering and Information Management

A Linguistics-Driven Approach to Statistical Parsing for Low-Resourced Languages

Prachya BOONKWAN[†a)], ***Nonmember and*** **Thepchai SUPNITHI**[†b)], ***Member***

SUMMARY Developing a practical and accurate statistical parser for low-resourced languages is a hard problem, because it requires large-scale treebanks, which are expensive and labor-intensive to build from scratch. Unsupervised grammar induction theoretically offers a way to overcome this hurdle by learning hidden syntactic structures from raw text automatically. The accuracy of grammar induction is still impractically low because frequent collocations of non-linguistically associable units are commonly found, resulting in dependency attachment errors. We introduce a novel approach to building a statistical parser for low-resourced languages by using language parameters as a guide for grammar induction. The intuition of this paper is: most dependency attachment errors are frequently used word orders which can be captured by a small prescribed set of linguistic constraints, while the rest of the language can be learned statistically by grammar induction. We then show that covering the most frequent grammar rules via our language parameters has a strong impact on the parsing accuracy in 12 languages.

key words: *statistical parsing, grammar induction, language parameters, Universal Grammar, treebank*

1. Introduction

Statistical parsing is a crucial part in natural language processing. A statistical parser assigns to an input sentence the most likely syntactic derivation with respect to the parsing model automatically learned from a large amount of data explicitly annotated with syntactic structures [1].

For the training process, available supervised parsers demand large amounts of hand-labeled data, such as Penn Treebank [2] or CCGbank [3]. However, producing such rich linguistic resources is labor- and time-intensive, requiring well trained linguists to define and annotate syntactic categories and resolve inconsistent annotations. This practice becomes impractical for a low-resourced language whose linguists are scarce. As a result, there are few treebanks available for training supervised parsers.

The notion of *grammar induction* was introduced to remedy this issue by automatic learning of linguistic structures from raw text. This reduces the labor of constructing a treebank from scratch and offers a possibility of building practical statistical parsers. Pioneering work in this area is based on unsupervised learning; namely, the Constituent-Context Model (CCM) [4], the Dependency Model with Valence (DMV), and the mixture model of CCM+DMV [5]. These models are essentially modeled as PCFGs and approximated by the Expectation-Maximization Algorithm [6]. This algorithm computes the model's approximation with a polynomial-timed dynamic programming, yet limited by local optima, data sparsity, and model's complexity.

Several approaches to improving the DMVs were proposed such as structure search techniques [7]–[11], training strategies [12], substructure reranking [13], [14], and the use of punctuation marks [15]. The others avert these issues by employing Markov Chain Monte Carlo methods which approximate the models from syntactic structures sampled from the posterior distributions. These inferred structures include probabilistic CFGs [16], [17], Tree Substitution Grammars [18], and Adaptor Grammars [19], [20].

Despite their efficiency, all of these unsupervised techniques have one serious drawback for practical statistical parsing: they are prone to *dependency attachment errors*. Boonkwan and Steedman [21], [22] pointed out that frequent collocation can sometimes cause unexpected mistakes in parsing. For example, a verb (`VBZ`) and a determiner (`DT`) tend to co-occur in a verb phrase, resulting in grammatically incorrect bracketing `[[VBZ DT] NN]` instead of grammatically correct one `[VBZ [DT NN]]`.

In this paper, we effectively solve this problem by using a small, novel set of cross-linguistic language parameters as a guide for grammar induction. Our intuition is: most dependency attachment errors are, in fact, frequently used word orders (e.g. verb+noun, determiner+noun, adjective+noun), which can be captured by a prescribed small set of linguistic constraints. Our method can be seen as an integration of the two contrasting schools of thought towards natural language processing:

1. Chomsky's Theory of *Principles and Parameters*: there are strong constraints on possible grammars children can learn from the noisy linguistic inputs [23], and
2. *Empiricist Learning*: the children have a general-purpose learning mechanism that allows them to draw complex inferences of linguistic structures from the noisy linguistic inputs [20], [24].

We mainly follow the indirect method of language parameter elicitation and supplement missing parts by the direct method if necessary. We design these language parameters and corresponding example sentences to capture frequent grammar rules in the Zipf's distribution (power

Manuscript received July 28, 2014.
Manuscript revised November 28, 2014.
Manuscript publicized January 21, 2015.

[†]The authors are with Language and Semantic Technology Laboratory, National Electronics and Computer Technology Center, Thailand.
a) E-mail: prachya.boonkwan@nectec.or.th
b) E-mail: thepchai.supnithi@nectec.or.th

DOI: 10.1587/transinf.2014DAP0024

law) of dependency types in natural languages and easily elicit from non-linguist informants and machine translation (Sect. 2). Then we statistically induce the rest of the grammar from unlabeled data (Sect. 3). We show that covering the most frequent grammar rules via our language parameters has a strong impact on the parsing accuracy in 12 languages (Sect. 4). Our error analysis suggests that remaining uncaptured dependency types can still be recovered if we fine-tune our language parameters according to the treebanks' annotation guideline (Sect. 5).

2. Language Parameters

In this section we present the main contributions of this paper: the design of our language parameters and corresponding example sentences that are used as a guide for grammar induction. There are three requirements taken into account in our design.

1. It maximizes the cross-linguistic coverage by choosing the most frequent word order schemes with a great impact.
2. It is feasible to elicit the language parameters from various sources ranging from syntax textbooks to non-linguist informants and machine translation.
3. Its language parameters can be converted into a compact syntactic representation.

We devise a linguistic questionnaire that allows linguistic experts to code language parameters on their own, as well as facilitates a short interview with non-linguist informants. Such questionnaire contains two parts: language parameters (frequent word orders) and a mapping table from our cross-linguistically frequent category classes to the guideline's tagset.

The most important part, *language parameters*, characterizes languages with their basic word orders as shown in Table 1; for example, the order of subject and predicate, etc. Frequent word orders are well studied in *The World Atlas of Language Structures* [25]. We compile the book's chapters 81–138 and 143–144, each of which describing frequent word orders, phrase structures, and clause structures. We classify these parameters into eight groups of questions that constitute the questionnaire. Table 1 lists our language parameters, where each grammatical unit separated by the plus signs '+' can be rearranged to capture the corresponding word order. For example, Rule 1 can be changed to subject+indirect object+object+verb for head-final Japanese. The syntactic head specified by underlining can also be customized for a specific treebank annotation guideline if necessary; e.g. from modal+VP to modal+VP in Rule 3.2. Rules with brackets (i.e. Rules 4.2 and 6) allows only local reordering, such as changing from [owner+possessivizer]+ownee to ownee+[possessivizer+owner] in Rule 4.2.

To help formalize these language parameters obtained from non-linguist informants, we also introduce an interview dialog for indirect parameter elicitation as shown in Fig. 1. The dialog is designed based on the language parameters, where the parameters and the questions in the dialog correspond to each other, question against question. In most questions, an informant is asked to translate these linguistic units into his own language and provide word alignment between the source and target languages. In Questions 4.5 and 7.2, the informant is asked whether there are particles and nominalizing affixes in his language, respectively.

With this dialog, human and machine translation become beneficial sources of language parameters. The translation of each example sentence and its word alignment information are used to induce basic word orders on the target language. Regarding human translation, we conduct short interviews with native speakers of Arabic, Chinese, English, Japanese, and German. Each informant is asked to translate each example sentence and provide its word alignment table. We then induce the basic word orders from these alignment tables and validated them with the grammar textbooks. This process takes up to 2–4 hours per language.

Machine translation is the other source of language parameters where native speakers for a language are scarce.

Table 1 Our language parameters. The syntactic head of each constituent is underlined. NP and VP stand for 'noun phrase' and 'verb phrase', respectively.

	Groups		Parameters
1.	Sentence		subject+verb+indirect object +direct object
2.	Simple modifiers	2.1	adjective+noun
		2.2	adverb+VP
		2.3	adverb+adjective
		2.4	negator+verb negator+adjective negator+adverb
3.	Complex verbs	3.1	Does a copulative verb exist?: *YES*
		3.2	modal+VP
		3.3	subject+verb+complementing VP
		3.4	subject+verb+object+complementing VP
4.	Complex modifiers	4.1	preposition+NP
		4.2	[owner+possessivizer]+ownee
		4.3	relative pronoun+VP
		4.4	NP+long modifier VP+long modifier gerund+long modifier sentence+long modifier
		4.5	sentence+particle
		4.6	number+noun classifier Numeral units can be used as: *adjective/adverb/NP modifier/VP modifier*
5.	Gerunds		Can perform as an NP?: *YES* Can perform as an NP modifier?: *YES* and word order is: NP+gerund Can perform as a VP modifier?: *YES* and word order is: VP+gerund
6.	Subordinate conjunctions		main clause+[conjunction+subclause]
7.	Transformation	7.1	infinitive marker+VP
		7.2	nominalizing affix+NP nominalizing affix+VP
8.	Relocation and dropping	8.1	Is dative shift allowed?: *YES*
		8.2	These can be omitted from the sentence: *subject*, *object*, *indirect object*

Q1 Translate: *Mary gives John a flower.*
Q2.1 Translate: *small kittens*
Q2.2 Translate: *Mary sits quietly.*
Q2.3 Translate: *strongly bitter tea*
Q2.4 Translate: (a) *The car does not work.* (b) *a not complex exercise*
Q3.1 Translate: (a) *John is a student.* (b) *John is tall.* (c) *John is in the classroom.*
Q3.2 Translate: *Mary can swim.*
Q3.3 Translate: *Mary wants to swim.*
Q3.4 Translate: *John asks Mary to hold the door for him.*
Q4.1 Translate: (a) *a gift in the box* (b) *Mary walks into the classroom.*
Q4.2 Translate: *John's car*
Q4.3 Translate: *John lifts the box that contains many books.*
Q4.4 Translate: (a) *John is the man on the beach.* (b) *John walks on the shore.* (c) *John is the man running on the shore.* (d) *On Monday, John will hand in his homework.*
Q4.5 Ask the informant if there are any adverb-like words which seem to modify the verb; e.g. *over* in 'Mary starts the process *over*'.
Q4.6 Translate: three cars, and notice if a noun classifier is used.
Q5 Translate: (a) *Running is good.* (b) *a running man* (c) *John is running.*
Q6 Translate: (a) *If you press this button, the door will open.* (b) *The door will open if you press this button.*
Q7.1 Translate: *Mary carefully reads her draft to identify the inconsistency.*
Q7.2 Ask the informant if there are any bound morphemes that transform any of these into a noun phrase: (a) noun phrase (b) a verb phrase
Q8.1 Translate: *John introduces to Mary his long-time friends from high school.*
Q8.2 Translate: *Mary gives John a flower*, and validate the grammaticality of the omission of any of these (a) subject (b) direct object and (c) indirect object.

Fig. 1 Dialog for indirect parameter elicitation.

Table 2 Our cross-linguistic tagset.

n	adj	v	vi	vt
vd	vcomp	vicomp	vtcomp	modal
copula	gerund	adv	part	neg
prep	postp	relpro	conj	subconj
cl	poss	inf	npnom	vpnom

Most modern MT systems are based on phrase-based translation [26], which is essentially equivalent to a probabilistic finite-state transducer [27]. Although our grammar rules are as expressive as context-free grammars (to be explained in Sect. 3.1), we are still able to elicit a correct set of language parameters using example sentences simple enough to correctly cover them. The cons of using MT is it provides only 1-best translation and word alignment according to the model, thus likely to be less accurate than human informants. We run Google Translate for Bulgarian, Danish, Dutch, Portuguese, Spanish, Swedish, and Turkish, and manually analyze the language parameters for these languages. The whole process including validation with the grammar textbooks also takes up to 2 hours per language.

Once we obtain the language parameters, we construct a table that maps each POS tag in the annotation guideline for treebanks to our cross-linguistic tagset (Table 2). These tags, along with the language parameters given previously, are used to construct a headed context-free grammar used as a guide for grammar induction. The POS-mapping process depends on the complexity of POS tags described in the annotation guideline, taking 4–6 hours to finish.

Table 3 An excerpt of rule conversion.

	Parameters	Headed CFG Rules
1.	subject+verb+indirect object+direct object	S → NP VP VP → vi VP → vt NP VP → vd NP NP
2.	subject+verb +complementing VP	VP → vicomp VP
3.	adjective+noun	NP → n NP → adj NP
4.	modal+VP	VP → modal VP
5.	prep+NP	PP → prep NP

To summarize, the entire process of building a set of language parameters for a previously unseen language takes up to 10 hours in total.

3. Grammar Induction

3.1 Language Parameters as Hard Constraints

The language parameters extracted from our linguistic questionnaire plays an key role as a guide for grammar induction. They help determine how many possibilities of phrases and sentences that can be formed in the language.

Once we obtain the language parameters and the POS-tag mapping, we convert them into a headed context-free grammar [28], which are in the following forms:

$$A \rightarrow \underline{B}C \qquad A \rightarrow B\underline{C} \qquad A \rightarrow \underline{w} \tag{1}$$

where A, B, C are nonterminals and w is a cross-linguistic tag, and the syntactic head of the constituent is underlined.

We formulate universal conversion rules for each language parameter into a headed CFG by closely following Boonkwan's [22] conversion method for his language parameters. We replace each phrase label in the parameters with the corresponding tags in Table 2. For example, 'subject', 'direct object', and 'indirect object' are replaced by NP. An excerpt of the conversion rules is illustrated in Table 3. We finally convert these grammar rules into the Chomsky Normal Form to achieve a binarized headed CFG. If any cross-linguistic tag is left unmatched in the POS mapping, all rules that make use of such tag will be eliminated by transitive closure.

To achieve statistical parsing, we approximate the parsing model from all possible parses of each sentence in the data set, while the produced headed CFG is used as hard constraints. We parse each input sentence with the headed CFG with the CKY Algorithm and store all parses in a packed chart. If a packed chart is incomplete, we apply wildcard combination to largest partially parsed subtrees so as to achieve a complete one.

3.2 Parsing Model

A parsing model assigns a probability to a syntactic anal-

ysis of a string of a language. In practice, we cannot directly measure the probability of a whole tree by counting tree frequency due to data sparsity. We therefore factorize the tree probability into a product of the probabilities of non-overlapping substructures.

We employ the probabilistic context-free grammar [29], [30]. The probability of a syntactic tree t with respect to a PCFG G is given by:

$$P(t|G) = \pi(r|G) \prod_{t_i \in \mathrm{dtrs}(t)} P(t_i|G) \tag{2}$$

where R is the set of grammar rules, $r \in R$ is the grammar rule used in the topmost derivation from the root node to its immediate daughters, and each $t_i \in \mathrm{dtrs}(t)$ is an immediate subtree of the root node of t. The probability of a grammar rule r, denoted by $\pi(r|G)$, is called a *parameter* of the parsing model π. If G is in the Chomsky Normal Form, the probability of t is reduced into the following form.

$$P(t|G) = \begin{cases} \pi(A \to w) & \text{preterminal} \\ \pi(A \to BC)P(t_1|G)P(t_2|G) & \text{branching} \end{cases} \tag{3}$$

where t_1 and t_2 are immediate subtrees of t and have labels B and C, respectively.

For a headed PCFG G, the probability of t is given by:

$$P(t|G) = \begin{cases} \pi(A \to \underline{w}) & \text{preterminal} \\ \pi(A \to \underline{B}C)P(t_1|G)P(t_2|G) & \text{left-headed} \\ \pi(A \to B\underline{C})P(t_1|G)P(t_2|G) & \text{right-headed} \end{cases} \tag{4}$$

where the headedness is preserved. We will estimate the parsing model in Eq. (4) with the input data set.

3.3 Parameter Estimation

We approximate the parsing model with the Variational Bayesian EM Algorithm [31], [32]. We follow the approach of Kurihara and Sato's [33] variational version of the Inside-Outside Algorithm for approximating the model parameters π in Eq. (4), because it was shown to be less data-overfitting than the standard Inside-Outside one.

Let us summarize the variational Inside-Outside Algorithm as follows. This algorithm approximates the model parameters with an input data set and a set of hyper-parameters $\mathbf{u}^{\mathrm{prior}} = \{u_r^{\mathrm{prior}} \geq 0 | r \in R\}$. Each u_r^{prior} determines the sensitivity to noise of rule r — the less, the more sensitive. The algorithm is as follows.

1. **Initialization:** The posterior hyper-parameters $\mathbf{u}^{(0)}$ are set to the prior hyper-parameters $\mathbf{u}^{\mathrm{prior}}$, while the model parameters $\pi^{(0)}$ is initialized in some way such as randomization and biased preferences.
2. **VBE Step:** For each node n_i having label A, we precompute the outside score $\mathbf{f}(n_i^A)$. If node n is a branching $n_i \Rightarrow d_{i1}d_{i2}$, where d_{i1} and d_{i2} have labels B and C, respectively, also precompute the inside scores $\mathbf{e}(d_{i1}^B)$ and $\mathbf{e}(d_{i2}^C)$. Compute the latent variables $\mathbf{q}$ by:

$$q_i(A \to \underline{B}C) = \sum_{n_i^A \Rightarrow d_{i1}^B d_{i2}^C} \mathbf{f}(n_i^A)\mathbf{e}(d_{i1}^B)\mathbf{e}(d_{i2}^C) \tag{5}$$

$$q_i(A \to B\underline{C}) = \sum_{n_i^A \Rightarrow d_{i1}^B d_{i2}^C} \mathbf{f}(n_i^A)\mathbf{e}(d_{i1}^B)\mathbf{e}(d_{i2}^C) \tag{6}$$

$$q_i(A \to \underline{w}) = \sum_{n_i^A \Rightarrow w} \mathbf{f}(n_i^A) \tag{7}$$

   The scores $\mathbf{e}(\cdot)$ and $\mathbf{f}(\cdot)$ can be computed by the standard Inside-Outside Algorithm.
3. **VBM Step:** Each model parameter π_r is estimated by:

$$\hat{\pi}_{A\to\alpha} \propto \exp\left[\psi(\phi_{A\to\alpha}) - \psi\left(\sum_{A\to\alpha'} \phi_{A\to\alpha'}\right)\right] \tag{8}$$

   where ψ is the digamma function, α and α' are strings of symbols, and

$$\phi_{A\to\alpha} = u_{A\to\alpha}^{\mathrm{prior}} + \sum_i \mathbb{E}_{q_i(A\to\alpha)}\pi_{A\to\alpha} \tag{9}$$

4. Repeat the VBE and VBM steps until the posterior probability converges.

4. Experiments

4.1 Data Sets, Accuracy Metrics, and Settings

We compare our method with other state-of-the-art techniques, most of which are assessed using the Wall Street Journal part of Penn Treebank [2]. We used WSJ10, WSJ15, and WSJ20, the standard collection of trees whose sentence lengths do not exceed 10, 15, and 20 words, respectively, after eliminating punctuation marks and empty elements. We automatically converted PTB into dependency structures with the LTH Conversion Tool [34][†]. The program is trained and tested using POS tag sequences from WSJ10, WSJ15 and WSJ20 as the terminal symbols (rather than strings of words) to minimize data sparsity.

For multilingual experiments, we use available dependency corpora from the CoNLL-X Shared Task 2006 [35] including Danish [DA] [36], Dutch [DU] [37], Portuguese [PO] [38], and Swedish [SV] [39], all of which are Indo-European. To investigate grammar induction in other language families, we also evaluate our method against Arabic [AR] [40], Bulgarian [BU] [41], Chinese [CH] [42], German [DE] [43], Japanese [JA] [44], Spanish [ES] [45], and Turkish [TU] [46]. We followed the same data preparation procedure as for WSJ, where punctuation marks are taken out and only POS tag sequences are used instead of word strings. To enhance the granularity of the tagsets, we fused CoNLL-X's CPOSTAG and POSTAG fields to become one fine-grained tag, and distinguished verb transitivity in the tagsets when indicated in the attribute field. For all inflected languages (Bulgarian and Turkish), we excluded morphological attributes from the POS tags to minimize implicit

[†] Configuration: `-splitSlash=false -qmod=true -deepenQP=true -whAsHead=true`.

Table 4 Results of multilingual experiments with corpora 10. We compare our results against three baselines: #1 Naseem et al. [49]; #2 Gillenwater et al. [10]; #3 Boonkwan and Steedman [21].

Languages	Baselines (Directed F1)			Our Method	
	#1	#2	#3	Directed F1	NED
PO10	71.5	49.5	–	**77.17**	90.03
EN10	71.9	64.4	**75.47**	73.34	86.66
ES10	64.8	57.9	–	**71.21**	86.83
SV10	63.3	41.4	–	**70.37**	84.54
JA10	–	59.4	68.55	**69.52**	88.30
CH10	–	35.8	62.25	**66.09**	81.76
AR10	–	–	–	64.83	85.17
BU10	–	59.8	–	**62.05**	79.08
TU10	–	56.9	–	**62.00**	77.12
DE10	–	45.7	56.71	**60.86**	77.68
DA10	51.9	–	–	**59.34**	81.86
DU10	–	38.8	–	**53.89**	77.11

supervision from the morphology. We partially fine-tune dependency annotation for coordinate structures, NPs, and VPs by speculating from 20 first trees in each corpora. This topic will be revisited in Sect. 5.

We measure the performance of our system by the directed dependency accuracy metric [5]. We count a directed dependency of a word pair to be correct if it exists in the gold standard. All accuracy numbers are reported in terms of F_1 scores. We also reported neutralized edge distance scores (NED) [47] which are more neutral to the different guidelines for treebank annotation.

For the prior hyper-parameters, we set $u_r^{\text{prior}} = 0$ for all rules r produced by the language parameters. If r is produced from combinations of subtrees in unparsable sentences, we set $u_r^{\text{prior}} = 9$ to make estimation more noise-tolerant. We use Viterbi Algorithm [48] for decoding.

4.2 Multilingual Experiments

This section presents experiment results for grammar induction over 12 languages. We compare our results with three prototype-driven parsers: Naseem et al. [49], Gillenwater et al. [10], and Boonkwan and Steedman [21]. The accuracy comparison is shown in Table 4.

Our method significantly outperforms the state-of-the-art techniques on 10 out of 12 languages (except English and Arabic). There is no prior work on Arabic so it is presented without any compared baselines. For English, our method performs almost as well as Boonkwan and Steedman [21], and it outperforms Bisk and Hockenmaier's [50] unsupervised CCG parser ($F_1 = 71.5\%$) and Cohn et al.'s [18] unsupervised TSG parser ($F_1 = 65.9\%$).

4.3 Learning from Longer Sentences

Next we investigate the learning capability of grammar induction from longer sentences (up to 15 and 20 words, respectively), as shown in Fig. 2.

The accuracy trends of most languages are: the accuracy decreases and saturates as the input sentences get longer. This is because basic word orders are frequently used in shorter sentences and they can be captured by our language parameters. Similar to supervised parsing, complex syntactic structures are used more frequently as the sentence length increases. These phenomena lead to syntactic ambiguity and ultimately decrease the parsing accuracy.

This decrease of accuracy also implies the Zipf's distribution of dependency types for each language. For the languages whose parsing accuracy decreases within 10% F_1 score (Danish, Dutch, English, Japanese, Portuguese, Spanish, and Swedish), their Zipf's distributions tend to have a shorter tail, resulting in better coverage of our language parameters.

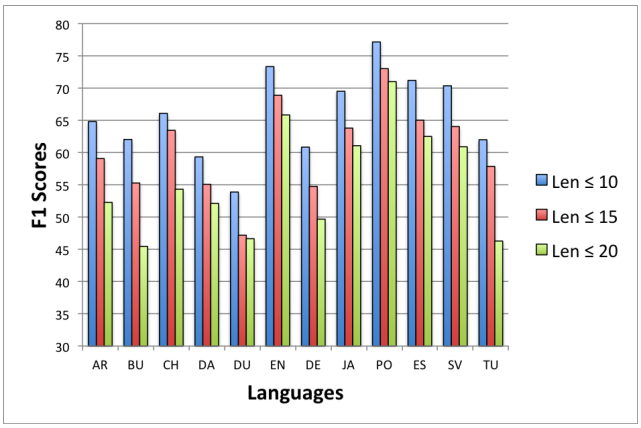

Fig. 2 Parsing accuracies on long sentences.

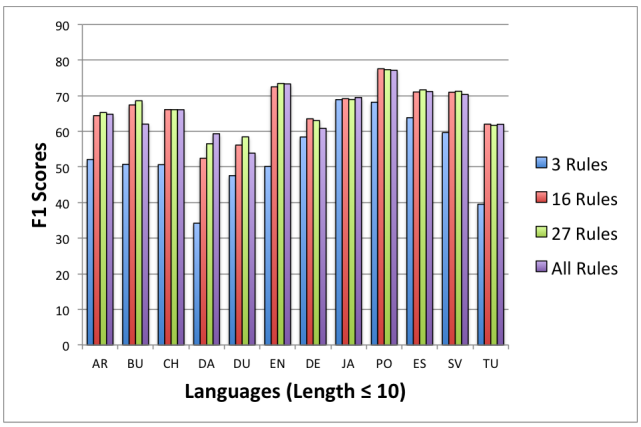

Fig. 3 Effects of language parameters.

4.4 Effects of Language Parameters

Finally we study the effects of language parameters in capturing word orders. Our language parameters are sorted with respect to the number of languages that exhibit each of them as examined by Haspelmath et al. [25]. We vary the number of language parameters used in guiding grammar induction and plot the parsing accuracies as shown in Fig. 3.

The accuracy trends of most languages are: the parsing accuracy dramatically increases when we increase the number of language parameters from the first 3 rules to the first

16 rules. For some languages (Bulgarian, Chinese, Danish, English, and Turkish), the accuracy increases 15% F_1 by doing so. However, the accuracy marginally increases in Japanese because we fine-tune its language parameters with respect to the annotation guideline, producing some additional constraints.

Also note that the accuracy does not always increase when more language parameters are incorporated. In Bulgarian, Dutch, and German, the accuracy slightly decreases when all language parameters are used. We suspect that less-frequent word orders in our language parameters for these languages may be mistakenly encoded, resulting in these decreases.

5. Discussion

We examine the causes of errors in our experiments by comparing the output trees with the gold standard ones. We found several discrepancies between the dependency annotation schemes used in some of the treebanks and those prescribed in our language parameters.

Three types of annotation discrepancies were particularly frequent in the corpora: coordinate structures, NP structures, and VP structures. There are six annotation schemes for coordinate structure (Fig. 4), three annotation schemes for noun phrases (Fig. 5), and two annotation schemes for verb phrase (Fig. 6). Our parser is capable of generating coordinate structures C1 and C2, NP structure N1, and VP structures V1 and V2.

By thorough observation, each corpus is annotated with a different scheme as summarized in Table 5. With respect to coordinate structure, the majority of the corpora are annotated with coodinate structure type C2 (Fig. 4 (b)) that assigns the conjunction as the head of the coordinate structure. The majority of the corpora use the NP structure type 1 (Fig. 5 (a)) that assigns the true core noun as the head of the NP. The majority of the corpora use the VP structure type 1 (Fig. 6 (a)) that assigns the auxiliary as the head of the VP if it is present.

Not surprisingly, the accuracy of grammar induction depends on the dependency annotation schemes. First, there are some annotation schemes that our syntactic prototypes cannot produce (i.e. coordinate structure types 3-6 and NP structure types 2 and 3) because their existence is beyond our initial expectation.

6. Related Work

The use of language parameters in grammar induction has been of general interest during the past decade. Language parameters are encapsulated in a *syntactic prototype*, a small set of fundamental linguistic knowledge that can be used to guide unsupervised grammar induction, thus improving the parsing accuracy.

Regarding its use, a syntactic prototype can be either *universal* (i.e. it is built once and then used for all languages) or *ad hoc* (i.e. it can be rapidly built for one language). It can be used as either *soft* or *hard constraints* for various unsupervised grammar induction techniques. We compare four types of existing syntactic prototypes against our language parameters in Table 6. Our language parameters are phrase structure rules with dependency attachment, built *ad hoc* and used as hard constraints for grammar induction. Note that the size of Naseem et al.'s [49] rules does not include language-specific dependency rules.

It should also be noted that these syntactic prototypes were constructed in different ways. Haghighi and

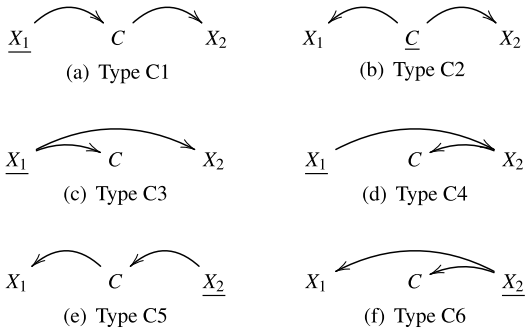

Fig. 4 Discrepant annotation schemes of coordinate structures, where C is a conjunction, X_1 and X_2 are conjunctions, and the heads are underlined.

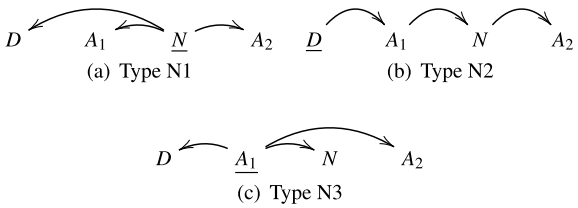

Fig. 5 Discrepant annotation schemes of NP structures, where N is a noun, A_1 and A_2 are nominal modifiers, D is a determiner, and the heads are underlined.

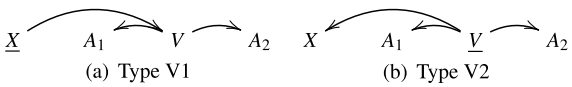

Fig. 6 Discrepant annotation schemes of VP structures, where V is a verb, X is an auxiliary, A_1 and A_2 are adverbs, and the heads are underlined.

Table 5 Dependency annotation schemes of the corpora.

Languages	Coordinate Structures	NP Structures	VP Structures
AR	C2	N2	V2
BU	C2	N1	V2
CH	C2	N1	V2
DA	C1	N3	V1
DU	C2	N1	V1
EN	C1	N1	V1
DE	C3	N1	V1
JA	C6	N1	V1
PO	C1	N1	V1
ES	C3	N3	V2
SV	C4	N1	V1
TU	C5	N1	V2

Table 6 Comparison of existing syntactic prototypes and our language parameters.

Prototypes	Rule Types	Sizes	Builds	Usages
Haghighi and Klein [51]	bracketing	20	ad hoc	soft constr.
Druck et al. [52]	dependency (with expected values)	20	ad hoc	soft constr.
Naseem et al. [49]	dependency	13	universal	soft constr.
Bisk and Hockenmaier [50]	CCG	9	universal	hard constr.
Our language parameters	phrase structure with dependency attachment	33	ad hoc	hard constr.

Klein's [51] bracketing rules and Druck et al.'s [52] are automatically extracted from the corpora and used as soft constraints. Naseem et al.'s [49] dependency rules and Bisk and Hockenmaier's [50] CCG rules are handcrafted. In particular, Naseem et al. [49] also make use of dataset-specific constraints in addition to the universal ones, e.g. word's offset, non-recursive phrase structures, etc. to boost the parsing accuracy significantly. This set of constraints are unsystematically designed and laborious to achieve when applying to low-resourced languages.

On the other hand, our language parameters are carefully designed with this spirit, where the information of frequent word orders is adequate to characterize languages. These parameters can be elicited from non-linguist informants (especially for less studied languages) via the questionnaire or extracted from syntax compendiums, grammar textbooks, or treebank annotation guidelines.

7. Conclusions and Future Work

We have introduced a novel, effective approach to building a statistical parser for low-resourced languages by using language parameters as a guide for grammar induction. A set of 33 parameters of basic word orders as well as a linguistic questionnaire, which are easy to acquire from non-linguist informants and machine translation systems, capture frequent grammar rules in the Zipf's distribution of natural languages. The rest of the grammar can be learned from large unlabeled data.

For future work, we plan to improve our language parameters for a wider coverage of dependency annotation schemes. We also plan to investigate the negative effects of annotation discrepancy on the parsing performance. Second, we plan to automatically learn the mapping between the treebank's tagset and our cross-linguistic tagset, thus shortening the time of constructing a new parser. Third and last, we plan to use our language parameters as a guide for inducing a synchronous CFG in hierarchical phrase-based MT [53]. It is challenging to learn an SCFG from a language pair whose language parameters are entirely different.

Acknowledgements

The authors are grateful for the two anonymous reviewers and the associate editors who give us beneficial and insightful comments and suggestions for the final manuscript. We would also like to sincerely thank Prof. Mark Steedman (The University of Edinburgh), Sharon Goldwater (The University of Edinburgh), Stephen Clark (Cambridge University) for suggestions of the research direction.

References

[1] E. Black, J. Lafferty, and S. Roukos, "Development and evaluation of a broad-coverage probabilistic grammar of English-language computer manuals," Proc. 30th Annual Meeting on Association for Computational Linguistics, pp.185–192, 1992.
[2] M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," Computational Linguistics, vol.19, pp.313–330, 1993.
[3] J. Hockenmaier and M. Steedman, "CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank," Computational Linguistics, vol.33, no.3, pp.355–396, 2007.
[4] D. Klein and C.D. Manning, "A generative constituent-context model for improved grammar induction," Proc. 40th Associations for Computational Linguistics, pp.128–135, 2002.
[5] D. Klein, The Unsupervised Learning of Natural Language Structure, Ph.D. thesis, Stanford University, March 2005.
[6] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," J. Royal Statistical Society, vol.39, no.1, pp.1–38, 1977.
[7] N.A. Smith, Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text, Ph.D. thesis, Department of Computer Science, John Hopkins University, 2006.
[8] S.B. Cohen, K. Gimpel, and N.A. Smith, "Logistic normal priors for unsupervised probabilistic grammar induction," Advances in Neural Information Processing Systems 21, pp.321–328, 2008.
[9] W.P. Headden III, M. Johnson, and D. McClosky, "Improving unsupervised dependency parsing with richer contexts and smoothing," Proc. Conference of the NAACL, pp.101–109, Boulder, Colorado, June 2009.
[10] J. Gillenwater, K. Ganchev, J. Graça, F. Pereira, and B. Taskar, "Sparsity in dependency grammar induction," Proc. ACL-2010 Short Papers, pp.194–199, 2010.
[11] V.I. Spitkovsky and H. Alshawi, "Lateen EM: Unsupervised training with multiple objectives applied to dependency grammar induction," Proc. EMNLP-2011, pp.1269–1280, 2011.
[12] V.I. Spitkovsky, H. Alshawi, and D. Jurafsky, "From baby steps to leapfrog: How "less is more" in unsupervised dependency parsing," Proc. NAACL-HLT 2010, 2010.
[13] R. Reichart and A. Rappoport, "Automatic selection of high quality parses created by a fully unsupervised parser," Proc. 13th Conference on CoNLL, pp.156–164, 2009.
[14] F. Dell'Orletta, G. Venturi, and S. Montemagni, "ULISSE: An unsupervised algorithm for detecting reliable dependency parses," Proc. 15th Conference on CoNLL 2011, pp.115–124, 2011.
[15] V.I. Spitkovsky and H. Alshawi, "Punctuation: Making a point in unsupervised dependency parsing," Proc. 15th Conference on CoNLL 2011, pp.19–28, 2011.
[16] M. Johnson, T.L. Griffiths, and S. Goldwater, "Bayesian inference for PCFGs via Markov chain Monte Carlo," Proc. NAACL 2007, pp.101–109, 2007.
[17] P. Liang, S. Petrov, M.I. Jordan, and D. Klein, "The infinite PCFG using hierarchical dirichlet processes," Proc. 2007 Joint Conference on EMNLP and CoNLL, pp.688–697, 2007.
[18] T. Cohn, P. Blunsom, and S. Goldwater, "Inducing tree-substitution grammars," J. Machine Learning Research, vol.9999, pp.3053–3096, 2010.
[19] M. Johnson, T.L. Griffiths, and S. Goldwater, "Adaptor grammars: A framework for specifying compositional nonparametric

Bayesian models," Advances in Neural Information Processing Systems, vol.19, 2007.
[20] S. Goldwater, Nonparametric Bayesian Models of Lexical Acquisition, Ph.D. thesis, Department of Cognitive and Linguistic Sciences, Brown University, Providence, Rhode Island, May 2007.
[21] P. Boonkwan and M. Steedman, "Grammar induction from text using small syntactic prototypes," Proc. 5th IJCNLP, pp.438–446, 2011.
[22] P. Boonkwan, Scalable Semi-Supervised Grammar Induction using Cross-Linguistically Parameterized Syntactic Prototypes, Ph.D. thesis, School of Informatics, University of Edinburgh, 2014.
[23] N. Chomsky, Aspects of the Theory of Syntax, MIT Press, 1965.
[24] J. Elman, E. Bates, M.H. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plunkett, Rethinking Innateness: A Connectionist Perspective on Development, MIT Press/Bradford Books, Cambridge, Massachusetts, 1996.
[25] M. Haspelmath, M.S. Dryer, D. Gil, and B. Comrie, The World Atlas of Language Structures, Oxford University Press, http://wals.info/, July 2005.
[26] P. Koehn, F.J. Och, and D. Marcu, "Statistical phrase based translation," Proc. HLT/NAACL, pp.48–54, 2003.
[27] A. Lopez, "Statistical machine translation," ACM Computing Surveys, vol.40, no.3, pp.1–49, Aug. 2008.
[28] M. Collins, Head-Driven Statistical Models for Natural Language Parsing, Ph.D. thesis, University of Pennsylvania, 1999.
[29] E. Charniak, "Statistical parsing with a context-free grammar and word statistics," Proc. 14th National Conference on Artificial Intelligence, Menlo Park, CA, AAAI Press/MIT Press, 1997.
[30] M. Johnson, "PCFG models of linguistic tree representations," Computational Linguistics, vol.24, no.4, pp.613–632, Dec. 1998.
[31] H. Attias, "A variational Bayesian framework for graphical models," Advances in Neural Information Processing Systems (NIPS 2000), pp.209–215, 2000.
[32] Z. Ghahramani and M.J. Beal, "Variational inference for Bayesian mixtures of factor analyses," Advances in Neural Information Processing Systems (NIPS 2000), pp.449–455, 2000.
[33] K. Kurihara and T. Sato, "Variational Bayesian grammar induction for natural language," International Colloquium on Grammatical Inference, pp.84–96, 2006.
[34] R. Johansson and P. Nugues, "Extended constituent-to-dependency conversion for english," Proc. NODALIDA 2007, pp.105–112, 2007.
[35] S. Buchholz and E. Marsi, "CoNLL-X shared task on multilingual dependency parsing," Proc. CoNLL-2006, pp.149–164, 2006.
[36] M.T. Kromann, L. Mikkelsen, and S.K. Lynge, "Danish dependency treebank," Proc. TLT, pp.217–220, 2003.
[37] L. van der Beek, G. Bouma, R. Malouf, and G. van Noord, "The Alpino dependency treebank," Language and Computers, pp.1686–1691, 2002.
[38] S. Afonso, E. Bick, R. Haber, and D. Santos, "Floresta Sinta(c)tica: A treebank for Portuguese," Proc. LREC, pp.1698–1703, 2002.
[39] J. Nilsson, J. Hall, and J. Nivre, "MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity," NODALIDA Special Session on Treebanks, pp.119–132, 2005.
[40] O. Smrž, J. Šnaldauf, and P. Zemánek, "Prague dependency treebank for Arabic: Multi-level annotation of Arabic corpus," Proc. International Symposium on Processing of Arabic, pp.147–155, 2002.
[41] K. Simov, P. Osenova, M. Slavcheva, S. Kolkovska, E. Balabanova, D. Doikoff, K. Ivanova, A. Simov, E. Simov, and M. Kouylekov, "Building a linguistically interpreted corpus of Bulgarian: The Bultreebank," Proc. LREC, pp.1729–1736, 2001.
[42] C. Keh-Liann and Y.M. Hsieh, "Chinese treebanks and grammar extraction," Proc. IJCNLP-2004, pp.560–565, 2004.
[43] S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith, "The TIGER treebank," Proc. Workshop on Treebanks and Linguistic Theories, pp.24–41, 2002.
[44] Y. Kawata and J. Bartels, "Stylebook for the Japanese treebank in VERBMOBIL," Tech. Rep., Eberhard-Karls-Universität Tübingen, 2000.
[45] M. Civit and M.A. Martí, "Bulding Cast3lb: A Spanish treebank," Research on Language & Computation, pp.549–574, 2004.
[46] K. Oflazer, B. Say, D.Z. Hakkani-Tür, and G. Tür, "Building a Turkish treebank," Treebanks: Building and Using Syntactically Annotated Corpora, 2003.
[47] R. Schwartz, O. Abend, R. Reichart, and A. Rappoport, "Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation," Proc. ACL-2011, pp.663–672, 2011.
[48] A.J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," IEEE Trans. Inf. Theory, vol.13, no.2, pp.260–269, 1967.
[49] T. Naseem, H. Chen, R. Barzilay, and M. Johnson, "Using universal linguistic knowledge to guide grammar induction," Proc. EMNLP-2010, pp.1234–1244, 2010.
[50] Y. Bisk and J. Hockenmaier, "Simple robust grammar induction with combinatory categorial grammar," Proc. 26th National Conference on Artificial Intelligence (AAAI), pp.1643–1648, 2012.
[51] A. Haghighi and D. Klein, "Prototype-driven grammar induction," Proc. 44th Annual Meeting of the Association for Computational Linguistics, pp.881–888, 2006.
[52] G. Druck, G. Mann, and A. McCallum, "Semi-supervised learning of dependency parsers using generalized expectation criteria," Proc. 47th Annual Meeting of the Association of Computational Linguistics and the 4th IJCNLP of the AFNLP, Suntec, Singapore, pp.360–368, Aug. 2009.
[53] D. Chiang, "Hierarchical phrase-based translation," Computational Linguistics, vol.33, no.2, pp.201–228, June 2007.

Prachya Boonkwan received B.Eng. and M.Eng. degrees in Computer Engineering from Kasetsart University in 2002 and 2005, respectively. He received a Ph.D. degree in Informatics from the University of Edinburgh, UK, in 2014. Since 2005, he has been with Language and Semantic Technology Lab at NECTEC in Thailand. His topics of interest include: grammar induction, statistical parsing, statistical machine translation, natural language processing, machine learning, and formal syntax.

Thepchai Supnithi received a B.S. degree in Mathematics from Chulalongkorn University in 1992. He received M.S. and Ph.D. degrees in Engineering from Osaka University, Japan, in 1997 and 2001, respectively. Since 2001, he has been with Language and Semantic Technology Lab at NECTEC in Thailand. His topics of interest include: statistical machine translation, natural language processing, machine learning, ontology, and knowledge engineering.