Supplemental Materials for "Universals of word order result from optimization of grammars for efficient communication"

Michael Hahn
Department of Linguistics
Stanford University

Daniel Jurafsky
Department of Linguistics
Stanford University

Richard Futrell
Department of Language Science
University of California, Irvine

January 24, 2019

1 UD Formalization

2 Results on all UD Relations

Table 2 shows the predicted prevalence of correlations between the *obj* dependency and all UD dependency types, along with the expected prevalence according to typological studies.

3 Results on Best Optimized Grammars

In the main part, we reported the percentage of optimized grammars supporting correlations. Here we report these numbers for the grammars that optimize each objective most successfully. That is, for each language and objective, we chose among the eight optimized grammars the one that achieves the best value for the objective function on the validation partition of the

Correlates with... verb	object	Operationalization	Real	DepL	Pred	Pars	Efficiency
adposition	NP	lifted_case	86	**81*****	47	**76*****	**68*****
copula	NP	lifted_cop	94	**81*****	53	**79*****	**61****
auxiliary	VP	aux	88	**74*****	**84*****	55	**69****
noun	genitive	nmod	80	**82*****	55	**74*****	**70*****
noun	relative clause	acl	80	**85*****	48	**77****	**73*****
complementizer	S	lifted_mark	76	**85*****	**59****	**80*****	**74****
verb	PP	obl	88	**78*****	**72*****	59	**69****
want	VP	xcomp	88	**90*****	**78****	**92*****	**92*****
verb	subject	nsubj	33	**29****	51	**8*****	**13*****
verb	manner adverb	advmod	35	51	**21*****	51	**32*****

Significance levels: *: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$

Table 1: Greenbergian Correlations. Following **?**, each correlation is stated in terms of a pair of a 'verb patterner' and an 'object patterner', whose relative order correlate with that of verbs and objects. For each correlation, we give our operationalization in terms of UD. For each correlation we report what percentage of the languages in our sample satisfied it ('Real'). We then report, for each correlation and each objective function, how many (in %) of the optimized grammars satisfy the correlation, with the significance level in a logistic mixed-effects analysis across language families.

Relation	Real	DepL	Pred	Pars	Efficiency	Expected Prevalence
acl	80	**85*****	43	**82*****	**71*****	$> 50\%$ (?)
aux	12	**26*****	**19*****	49	**35****	$< 50\%$ (?)
lifted_case	86	**81*****	**40****	**77*****	**67*****	$> 50\%$ (?)
lifted_cop	94	**80*****	59	**72*****	**63****	$> 50\%$ (?)
lifted_mark	76	**85*****	**57***	**76*****	**72*****	$> 50\%$ (?)
nmod	80	**82*****	49	**71*****	**68*****	$> 50\%$ (?)
obl	88	**78*****	**71*****	46	**68*****	$> 50\%$ (?)
xcomp	88	**90*****	**76*****	**89*****	**85*****	$> 50\%$ (?)
nsubj	33	**29*****	53	**7*****	**23*****	$> 50\%$ (?)
advmod	35	51	**18*****	47	**39****	$> 50\%$ for manner adverbs; $\approx 50\%$ for negation, TAM, intensifiers (?)
advcl	86	**84*****	51	**67*****	**67***	$> 50\%$ (??)
amod	51	**84*****	42	**53***	51	$\approx 50\%$ (?)
nummod	37	**74****	49	52	52	$\approx 50\%$ (?, 89A vs. 83A)
appos	77	**77*****	49	**67*****	**62****	Unknown
lifted_cc	71	**83*****	51	**81*****	**71*****	Unknown
ccomp	86	**88*****	**70****	**77*****	**77****	Unknown (cf. ?)
csubj	79	**78*****	60	**66*****	**68****	Unknown (cf. ?)
expl	12	36	62	40	53	Unknown
iobj	74	49	**78*****	**38***	62	Unknown
vocative	46	**42***	59	45	**45***	Unknown
compound	66	**61***	48	**58***	52	Uninterpretable
det	31	**71*****	**56***	48	54	Uninterpretable (conflates articles and others)
dislocated	47	61	**68****	55	58	Uninterpretable
dep	56	**64***	52	58	**58***	Uninterpretable
conj	72	**82*****	**42****	**62*****	**59***	UD artifact
discourse	24	**36****	**41***	49	46	UD artifact
fixed	78	**32*****	45	47	44	UD artifact
flat	73	**66****	45	**59****	54	UD artifact
goeswith	61	57	44	57	52	UD artifact
list	94	59	50	54	52	UD artifact
orphan	82	**70****	50	**55***	59	UD artifact
parataxis	66	56	43	57	59	UD artifact
reparandum	18	53	57	43	48	UD artifact

Significance levels: *: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$

Table 2: Predictions on all UD relations occurring in more than one language. Numbers are the fraction of languages where the direction of the given relation agrees in direction with the *obj* relation. In the last column, we indicate what direction would be expected typologically. "Uninterpretable" UD relations are those which collapse so many different linguistic relationships that they are not linguistically meaningful. "UD artifact" relations are those whose order is determined strictly by UD parsing standards, such that their order is not linguistically meaningful: these include dependencies such as the connection between two parts of a word that have been separated by whitespace inserted as a typo (*goeswith*).

Correlates with... verb	object	Operationalization	Real	DepL	Pred	Pars	Efficiency
adposition	NP	lifted_case	86	98	69	88	83
copula	NP	lifted_cop	94	98	58	85	74
auxiliary	VP	aux	88	84	90	61	73
noun	genitive	nmod	80	98	75	88	91
noun	relative clause	acl	80	100	69	88	90
complementizer	S	lifted_mark	76	100	76	82	87
verb	PP	obl	88	98	88	75	91
want	VP	xcomp	88	100	92	94	96
verb	subject	nsubj	33	25	53	10	21
verb	manner adverb	advmod	35	61	27	55	36

Table 3: Percentages of grammars supporting correlations, among grammars that best optimize each objective on treebanks of a given language.

treebanks – that is, the one where the optimization procedure was most successful.

Numbers for these languages are shown in Table 3. As expected, the correlations mostly have stronger prevalences than when considering all optimized languages. For Dependency Length, the correlations that it predicts mostly have prevalences close to 100 %, whereas both the UD languages and efficiency-optimized languages show somewhat lower percentages. That is, dependency length mostly overshoots for the correlations that it predicts correctly.

4 Further Statistical Analyses

In the main results table, we reported the fraction of optimized languages satisfying a correlation. We show that essentially the same predicted fractions are obtained when controlling for the imbalance in distribution of language families.

To this end, we used the logistic mixed-effects regressions that were used to estimate significance to compute the expected prevalence of a correlation controlling for language families. To obtain predictions controlling for languages and language families, we plot the logit-transformed posterior of α in

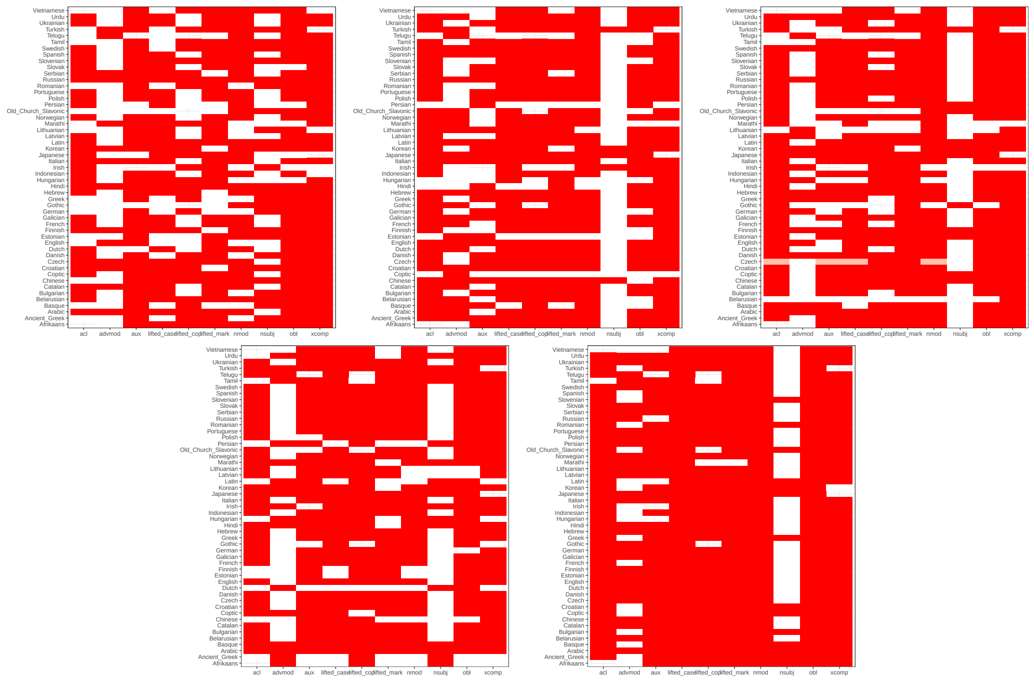

Figure 1: Correlations satisfied by the best word order grammars for predictability, parseability, efficiency, by the UD languages, and by the best grammars for dependency length.

Figure 2.

5 Relation to the Information Bottleneck

Our efficiency objective (Equation **??**), to be maximized for word order grammars L, is restated below.

$$J_T(L) = I[L(T); T] - \lambda H[L(T)]. \tag{1}$$

This equation can be seen as a special case of the objective function proposed in **?** for general lossy compression, based on rate-distortion theory (**??**). In that theory, the goal is to derive an encoding $\hat{X} = L(X)$ of some variable X which preserves the information in X relevant to some third variable Y, while minimizing irrelevant information. The three variables Y,X, and $\hat{X}$ are assumed to form a Markov chain $Y \to X \underset{\rightarrow}{L} \hat{X}$. Given those variables, the information bottleneck objective to be maximized is given in Eq. 2.

$$J_{X,Y}(L) = I[\hat{X}; Y] - \lambda I[\hat{X}; X]. \tag{2}$$

In this equation, the joint distribution of X and Y is seen as fixed, and the conditional distribution $L = \hat{X}|X$ is optimized.

In our setting we consider a language L expressing trees T as utterances U: $T \underset{\rightarrow}{L} U$. To make our formulation more parallel to the information bottleneck, we assume the utterances U are passed through a channel from producer to comprehender, and received as $\hat{U} = U$. We have the Markov chain $T \underset{\rightarrow}{L} U \to \hat{U}$, where we are assuming the conditional channel distribution $\hat{U}|U$ is fixed and the conditional distribution $L = U|T$ is to be optimized. Then our objective is parallel to Eq. 2:

$$J_T(L) = I[U; T] - \lambda I[U; \hat{U}]. \tag{3}$$

Assuming the channel $U \to \hat{U}$ is noiseless, we have $U = \hat{U}$, and thus $I[U; \hat{U}] = I[U; U] = H[U]$. Then Eq. 3 reduces to a form of Eq. 1:

$$J_T(L) = I[U; T] - \lambda H[U].$$

Our efficiency objective is thus closely related to the information bottleneck; the difference is that we are optimizing the mapping from underlying meanings to representations (strings) assuming that these strings will be transmitted in a fixed noiseless channel.

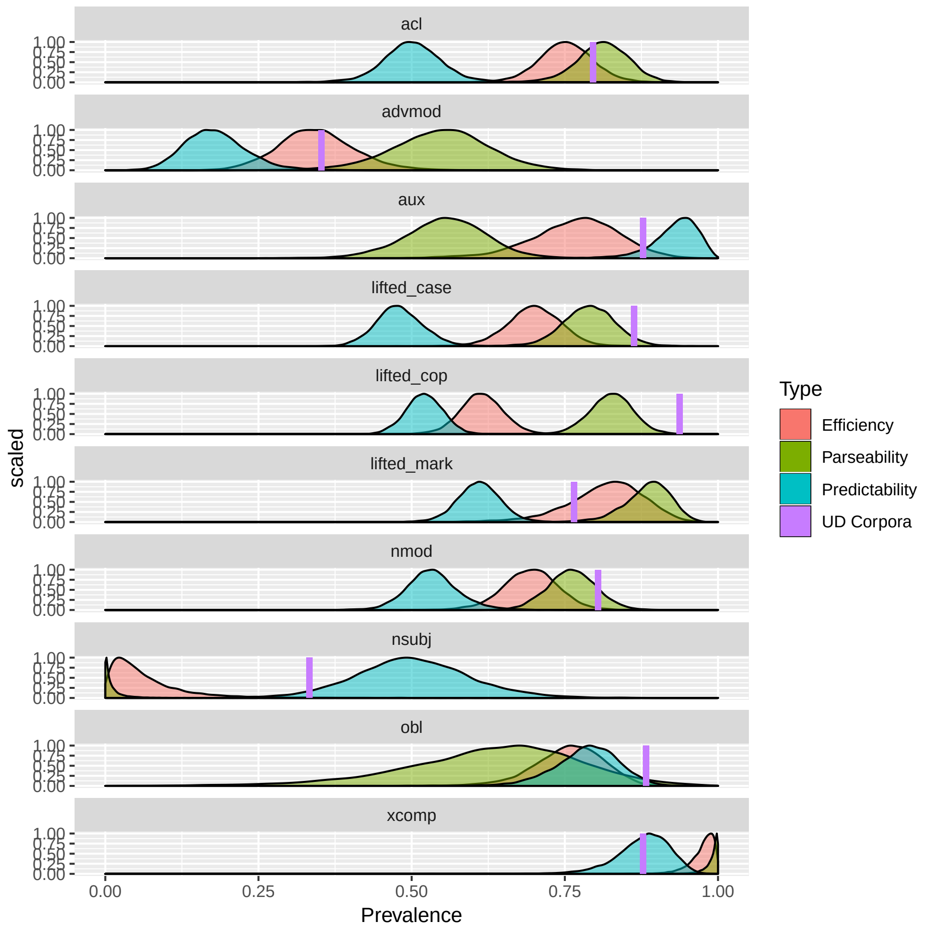

Figure 2: Posteriors of the prevalences of the correlations among languages optimized for Predictability (red), parseability (green), and efficiency (blue), controlling for languages and language families, with prevalence in the UD languages.

6 Possible values of λ

In the efficiency objective (1), the value of λ is constrained to be in $[0, 1)$. This means, surprisal must be weighted less strongly than parsability.

Greater values of λ mathematically result in degenerate solutions. To show this, note that the following inequality always holds:

$$I[L(T); T] \leq H[L(T)]. \tag{4}$$

Therefore, if $\lambda \geq 1$, the efficiency objective satisfies

$$J_T(L) = I[L(T); T] - \lambda H[L(T)] \leq 0. \tag{5}$$

and it takes the maximally possible value zero if $L(T)$ maps all trees to a single utterance, in which case both $I[L(T); T]$ and $H[L(T)]$ are zero. This is a degenerate language with only a single utterance, which is simultaneously used to convey all meanings. While the design of our word order grammars precludes a collapse of all trees to a single utterance, this shows that an objective with $\lambda \geq 1$ cannot be a generally applicable objective for language efficiency.

In our experiments, we chose $\lambda = 0.9$ as a mathematically valid value that puts similar weight on both predictability and parsability. We previously carried out language optimization for $\lambda = 1$, and obtained essentially identical predictions for word order correlations and relative optimality of language.

7 Joint optimization of grammar, parser, and language model parameters

For each of our objective functions, we seek parameters θ of a word order grammar L_θ that maximize the average utility of sentences in the language obtained by linearizing unordered trees $t \sim T$ according to L_θ. The calculation of the predictability and parseability objectives also requires fitting language models parameterized by ϕ and parsers parameterized by ψ, respectively; and the calculation of the efficiency objective requires joint optimization of the word order grammar, language models, and parsers. The optimization problems come out to:

$$\max_{\theta} \mathbb{E}_{t \sim T}\, \mathbb{E}_{\mathbf{w} \sim L_\theta(t)}\, R_{DepL}(t; \mathbf{w}) \tag{6}$$

$$\max_{\theta,\phi} \mathbb{E}_{t \sim T}\, \mathbb{E}_{\mathbf{w} \sim L_\theta(t)}\, R^{\phi(\theta)}_{Pred}(t; \mathbf{w}) \tag{7}$$

$$\max_{\theta,\psi} \mathbb{E}_{t \sim T}\, \mathbb{E}_{\mathbf{w} \sim L_\theta(t)}\, R^{\psi(\theta)}_{Pars}(t; \mathbf{w}) \tag{8}$$

$$\max_{\theta,\phi,\psi} \mathbb{E}_{t \sim T}\, \mathbb{E}_{\mathbf{w} \sim L_\theta(t)}\, R^{\psi(\theta),\phi(\theta)}_{Efficiency}(t; \mathbf{w}). \tag{9}$$

Note that the original word orders from the treebank never enter this objective—in particular, they do not influence the language model and the parser, which are entirely determined by θ.

The case of dependency length in Eq. **??** is the simplest, requiring only coordinate descent on the parameters $a_\tau, b_\tau \in \theta$. A similar optimization problem was solved by hill-climbing in **?** and **?**. We optimize this objective by stochastic gradient descent with a batch size of one unordered dependency tree.

The case of predictability, parseability, and efficiency are more challenging, as we need to simultaneously look for an ordering model optimized for the loss of a language model and/or a parser, and for a language model and/or parser that are optimized for that given ordering model. We solve Eqs. 7–9 by performing stochastic gradient descent over all sets of parameters θ, ϕ, and/or ψ simultaneously. Below we discuss the process only for the predictability objective (Eq. 7); the other objectives are optimized analogously. In each step, we sample a dependency tree t from the treebank, then sample an ordering from the current setting of θ to obtain a linearized sentence $\mathbf{w} \sim P_{L_\theta}(\cdot|t)$. Then we do a gradient descent step using the estimator

$$\begin{pmatrix} \partial_\theta \left(\log P_{L_\theta}(\mathbf{w}|t)\right) \cdot R^{\phi}_{Pred}(t; \mathbf{w}) \\ \partial_\phi R^{\phi}_{Pred}(t; \mathbf{w}) \end{pmatrix} \tag{10}$$

for the gradient

$$\begin{pmatrix} \partial_\theta \\ \partial_\phi \end{pmatrix} \mathbb{E}_{\mathbf{w}' \sim L_\theta(t)}\, R^{\phi}_{Pred}(t; \mathbf{w}').$$

This unbiased estimator is the ordinary gradient estimator for ϕ, and the REINFORCE estimator for θ (**?**).

In addition, we estimated maximum-likelihood ordering grammars on the original ordered dependency trees with SGD. For nonprojective trees, we ignore discontinuities.

A different commonly considered information-theoretic objective function for word order is Uniform Information Density (**?**), formalized as minimizing a superlinear function of per-word surprisal (**?**). Note that optimizing grammars for this objective faces additional challenges, as the optimization problem cannot be rewritten as joint optimization in the form (7). We thus do not evaluate this objective function here.

8 Implementation Details

Locality In Equation **??**, we assume that $head(w_i) = i$ iff w_i is the root of the sentence, so that the root does not contribute to dependency length.

Language Model We choose a standard LSTM (**?**) language model with vocabulary size restricted to the most frequent 50,000 words in the treebanks for a given language. Given the small size of the corpora, this limit is only attained only for few languages. In each time step, the input is a concatenation of embeddings for the word, for language-specific POS tags, and for universal POS tags. The model predicts both the next word and its POS tags in each step. Using POS tags is intended to prevent overfitting on small corpora. These decisions were made before evaluating word order properties.

Parser We use a biaffine attention parser architecture (**???**). This architecture is remarkably simple: the words of a sentence are encoded into context-sensitive embeddings using bidirectional LSTMs, then a classifier is trained to predict the head for each work. The classifier works by calculating a score for every pair of word embeddings (w_i, w_j), indicating the likelihood that the jth word is the head of the ith word. This is a highly generic architecture for recovering graph structures from strings, and is a simplification of graph-based parsers which reduce the parsing problem to a minimal spanning tree problem (**?**).

To reduce overfitting on small corpora, we choose a delexicalized setup, parsing only from POS tags. Preliminary experiments showed that a parser incorporating word forms overfitted long before the ordering model had converged. Parsing from POS tags reduces early overfitting. This decision was made before evaluating word order properties.

Hyperparameters For locality, predictability, and parseability, we first selected hyperparameters on the respective objectives for selected languages on the provided development partitions. Then, for each language and each objective function, we created eight random combinations of these selected hyperparameter values, and selected the setting that yielded the best value of the respective objective function on the language. We then used this setting for creating optimized word order grammars.

All word and POS embeddings are randomly initialized with uniform values from $[-0.01, 0.01]$. We do not use pretrained embeddings (**?**): these could improve performance of language models and parsers, they would introduce confounds from the languages' actual word orders as found in the unlabeled data.

Optimization details We employ two common variance reduction methods to improve the estimator (10), while keeping it unbiased. For dependency length and predictability, note that the loss incurred for a specific word only depends on ordering decisions made up to that word (and its head, in the case of dependency length). We represent the process of linearizing a tree as a dynamic stochastic computation graph, and use these independence properties to apply the method described in **?** to obtain a version of (10) with lower variance. Second, we use a word-dependent moving average of recent per-word losses as control variate **?**. These two methods reduce the variance of the estimator and thereby increase the speed of optimization and reduce training time, without biasing the results. For numerical stability, we represent $a_\tau \in [0, 1]$ via its logit $\in \mathbb{R}$. Furthermore, to encourage exploration of the parameter space, we add an entropy regularization term (**?**) for each Direction Parameter a_τ, which penalizes a_τ values near 0 or 1. The weight of the entropy regularization was chosen together with the other hyperparameters.

These techniques for improving (10) are well-known in the machine learning literature, and we fixed these before evaluating optimized grammars for word order properties.

We update word order grammar parameters θ using SGD with momentum. For the language model parameters ϕ, we use plain SGD. For the parser parameters ψ, we use Adam (**?**), following **?**. The learning rates and other optimization hyperparameters were determined together with the other hyperparameters.

For each language and objective function, we apply Early Stopping (**?**)

as follows. We compute a Monte-Carlo estimate of the objective on the UD development set in intervals of 50,000 training steps, and stop optimization once the value of the objective deteriorates compared to the last previous estimate. This technique, well known in the machine learning literature, helps prevent overfitting to the specific corpora used.

The choice of optimization methods and the stopping criterion were fixed before we investigated word order correlations.

Recall that, for each language, we created 8 optimized languages for each optimization criterion. We enforced balanced distribution of object-verb and verb-object ordering among optimized languages by fixing a_τ for the *obj* dependency to be 0.0 in four of these languages, and 1.0 in the other four. This maximizes statistical power for detecting and quantifying correlations between the *obj* relation and other relations.

9 Robustness to different language models and parsers

Here we take up the question of the extent to which our results are dependent on the particular parser and language model used in the optimization process. We want to know: when we optimize a word order grammar for parseability, have we produced a language which is highly parseable *in general*, or one which is highly parseable *for a specific parser*? We wish to argue that natural language syntax is optimized for parseability in general, meaning that syntactic trees are highly recoverable from word orders in principle. If it turns out that our optimized languages are only optimal for a certain parser from the NLP literature, then we run the risk of circularity: it may be that the reason this parser was successful in the NLP literature was because it implicitly encoded word order universals in its inductive biases, and thus it would be no surprise that languages which are optimized for parseability also show those universals.

In this connection, we note that the parser and language model architectures we use are highly generic, and do not encode any obvious bias toward natural-language-like word orders. The LSTM language model is a generic model of sequence data which is also been used to model financial time series (**?**) and purely theoretical chaotic dynamical systems (**?**); the neural graph-based parser is simply solving a minimal spanning tree problem (**?**).

Nevertheless, it may be the case that a bias toward word order universals is somehow encoded implicitly in the hyperparameters and architectures of these models.

Here we address this question by demonstrating that our languages optimized for prediction and parseability are also optimal under a range of different language models and parsers. These results show that our optimization process creates language in which strings are generally predictable and informative about trees, without dependence on a particular prediction and parsing algorithms.

9.1 CYK Parsers

We constructed simple Probabilistic Context-Free Grammars (PCFGs) from corpora and word order grammars, using a simplified version of the models of Collins 1997 (Model 1). In our PCFGs, each head independently generates a set of left and right dependents. We formulate this as a PCFG where each rule has the form:

$$\mathrm{POS}_H \rightarrow \mathrm{POS}_H\ \mathrm{POS}_D$$

for head-initial structures, and

$$\mathrm{POS}_H \rightarrow \mathrm{POS}_D\ \mathrm{POS}_H$$

for head-final structures, where each symbol is a POS tag. Thus, POS tags act both as terminals and as nonterminals.

We estimated probabilities by taking counts in the training partition, and performing Laplace smoothing with a pseudocount $\alpha = 1$ for each possible rule of this form. For such a PCFG, exact parsing is possible using Dynamic Programming, and specifically the CYK algorithm.

This parsing strategy is very different from the neural graph-based parser: While the graph-based parser solves a minimum spanning tree problem, the CKY algorithm uses dynamic programming to compute the exact probabilities of trees given a sentence, as specified by the generative model encoded in the PCFG. Second, while the graph-based neural parser uses machine learning to induce syntactic knowledge from data, the CKY parser performs exact probabilistic inference.

We used the CKY algorithm to compute the parsing loss $H[T|L_\theta(T)]$ on the validation partition of the English UD corpora, for random and optimized ordering grammars. Results (Figure 3) show that optimized grammars are more parseable for exact parsing of a simple PCFG.

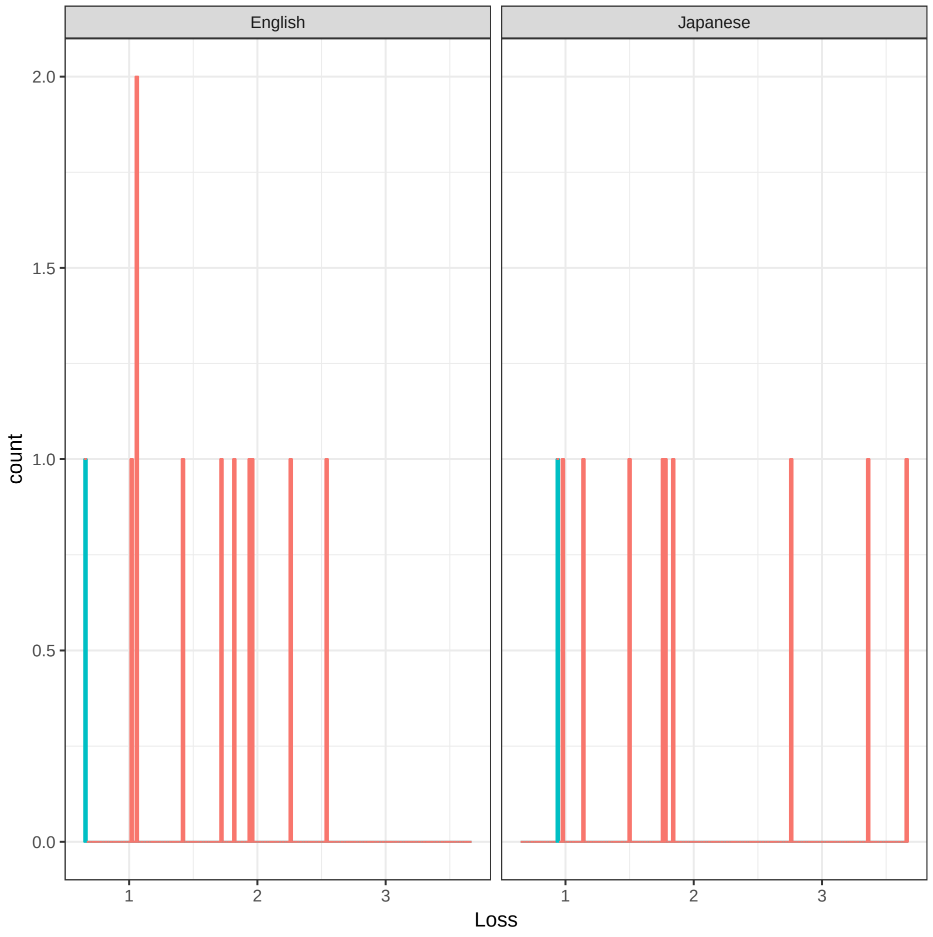

Figure 3: Parsability loss $H[T|L_\theta(T)]$ (lower is better) computed by a simple CKY parser, for random word order grammars (red) and word order grammars optimized for efficiency (blue).

9.2 Distorted graph-based parsers

Many of the word order biases explained by parseability in Table **??** are also explained by dependency length minimization. Therefore, we address here the idea that the graph-based parser might have a built-in bias toward parses involving short dependencies, which we call a **locality bias**.

A simple way to eliminate or alter a locality bias for any parser would be to change the order in which the parser sees words. Suppose that a parser P has a bias toward positing a dependency between a word and the immediate previous word.

Even–odd order. A sequence of n words originally ordered as $w_1w_2w_3w_4\cdots w_n$ is reordered by separating the even and odd indices: $w_2w_4w_6\cdots w_{n-1}w_1w_3w_5\cdots w_n$ (assuming n odd). Therefore all words that are adjacent in the original order will be separated by a distance of $\approx n/2$ in the distorted order, while all words of distance 2 in the original order will become adjacent.

Interleaving order. In interleaving ordering, a sequence originally ordered as $w_1w_2w_3\cdots w_n$ is split in half at the middle (index $m = \lceil n/2 \rceil$), and the two resulting sequences are interleaved, yielding $w_1w_mw_2w_{m+1}w_3w_{m+3}\cdots w_n$. Thus all words that were originally adjacent will have distance 2 in the distorted order, with the intervening word coming from a very distant part of the sentence.

Inwards order. A sequence originally ordered as $w_1w_2w_3\cdots w_{n-1}w_n$ is ordered from the edges of the string inwards, as $w_1w_nw_2w_{n-1}\cdots w_{\lceil n/2 \rceil}$. This corresponds to folding the string in on itself once, or equivalently, splitting the sequence in half at the middle, then interleaving the two resulting sequences after reversing the second one. The result is that the most non-local possible dependencies in the original order become the most local dependencies in the distorted order.

Sorted order. A sequence is reordered by sorting by POS tags, and randomizing the order within each block of identical POS tags. To each word, we then add a symbol encoding the original position in the sequence. For instance

PRON VERB PRON

may be reordered as

PRON 1 PRON 3 VERB 2

or

PRON 3 PRON 1 VERB 2

The numbers are provided to the parser as atomic symbols from a vocabulary ranging from 1 to 200; numbers greater than 200 (which may occur in extremely long sentences) are replaced by an out-of-range token.

Experiments Using English and Japanese data, we trained parsers for the ten random word order grammars and for the best grammar optimized for efficiency, with the input presented in each of the distorted orderings. Resulting parsing accuracy scores are shown in Figure 4. In all settings, the language optimized for efficiency achieved higher parsing accuracy than random ordering grammars, showing that the parser's preference for optimized languages cannot be attributed to a locality bias.

9.3 n-gram language models

We constructed a Bigram model with Kneser-Ney smoothing. A bigram model predicts each word taking only the previous word into account. This contrasts with LSTMs, which take the entire context into consideration. Thus, bigram models and LSTMs stand on opposing ends of a spectrum of language models taking more and more aspects of the context into account.

We estimated language models on the training partitions, and used the validation partitions to estimate surprisal. We conducted this for the ten random and the best optimized ordering grammars on English and Japanese data. Results (Figure 5) show that languages optimized for efficiency are optimal for a bigram language model.

9.4 Discussion

It is always a logical possibility in studies such as this one that the results are dependent on the particular probabilistic models used.

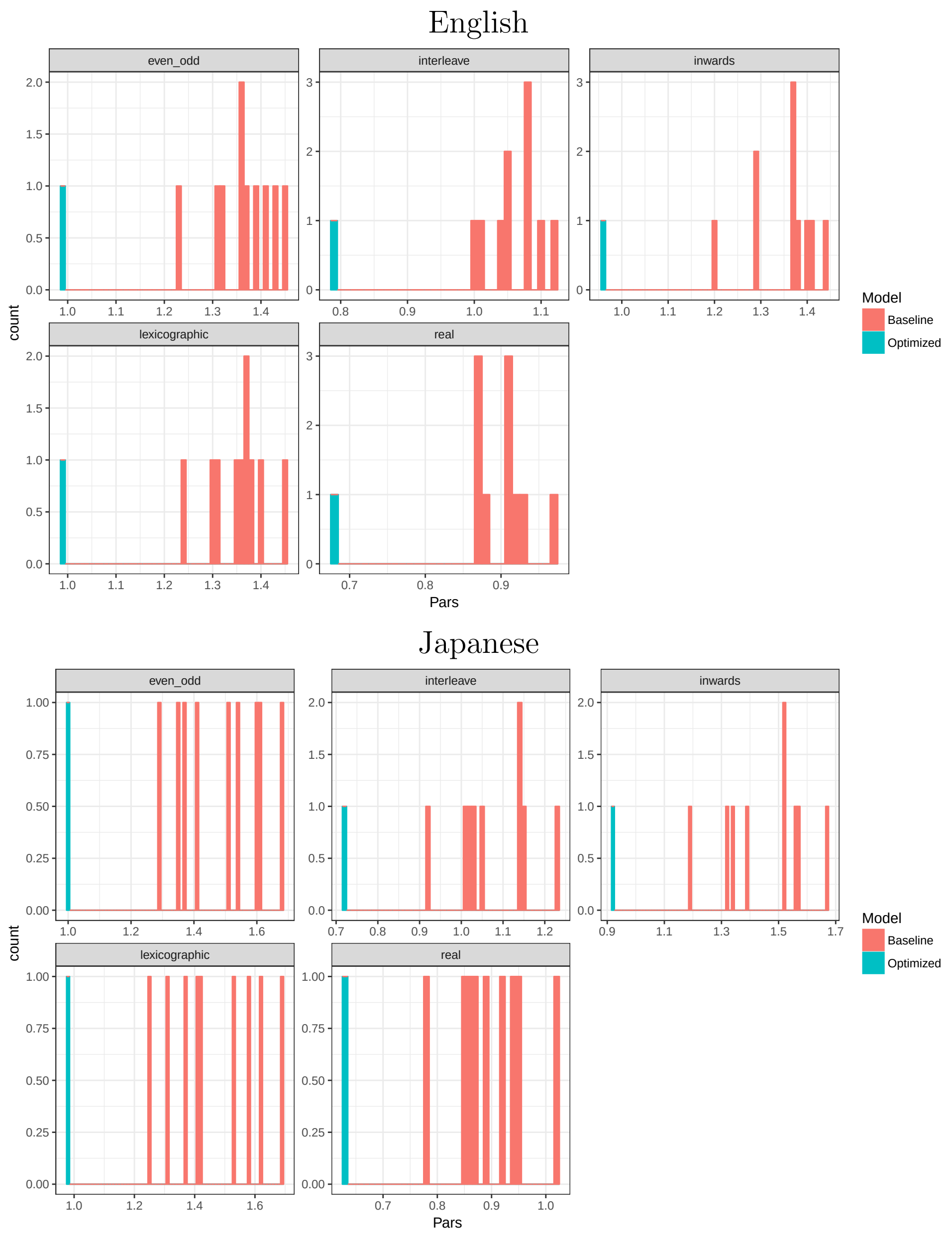

Figure 4: Parseability of baseline grammars and grammars optimized for efficiency, in English (top) and Japanese (bottom), measured by parsing loss $H[T|L_\theta(T)]$ (lower is better), for the four distorted orderings, and the actual orderings ('real').

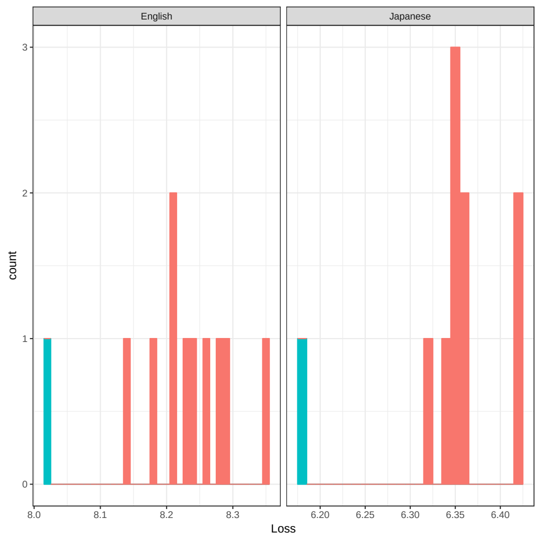

Figure 5: Surprisal (lower is better) computed from Bigram model, on English and Japanese data ordered according to random ordering grammars (red) and ordering grammars optimized for efficiency (blue).

10 Effects of data sparsity

Our efficiency objective comes down to

If the difference between random and optimized languages is due to data sparsity, we expect that it decreases as the amount of training data is increased. If, on the other hand, there is an inherent difference in efficiency between random and optimized languages, we expect that the difference persists as training data is increased.

We considered Czech, the UD language with the largest amount of available treebank data (approx. 2.2 Million words), up to $\approx$ 300 times more data than is available for some other UD languages. We cosidered both a random ordering grammar, and the best ordering grammar optimized for parseabaility. For both of these ordering grammars, we trained the parser on successively larger portions of the training data (0.1 %, 1 %, 5%, 10%, 20 %, ..., 90 %, 100 %) and recorded parsing accuracy. Furthermore, for the random grammar, we varied the number of neurons in the BiLSTM (200, 400, 800) to test whether results depend on the capacity of the network.

The resulting curves are shown in Figure 6. A gap in parsing accuracy of

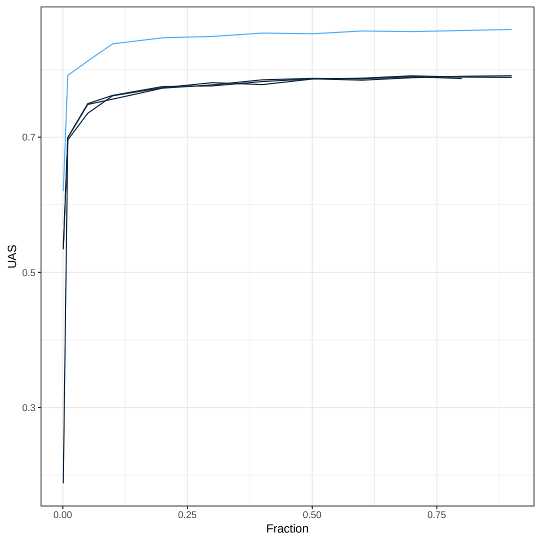

Figure 6: Parsing accuracy (measured in UAS) for optimized (light blue) and random (black) ordering grammar on Czech data, as a function of the fraction of total training data provided.

about 0.07-0.1 appears already at 0.01 % of the training data (2000 words), and persists for larger amounts of training data.

11