Towards More Natural Artificial Languages

Mark Hopkins
Department of Computer Science
Williams College
mh24@williams.edu

Abstract

A number of papers have recently argued in favor of using artificially generated languages to investigate the inductive biases of linguistic models, or to develop models for low-resource languages with underrepresented typologies. But the promise of artificial languages comes with a caveat: if these artificial languages are not sufficiently reflective of natural language, then using them as a proxy may lead to inaccurate conclusions. In this paper, we take a step towards increasing the realism of artificial language by introducing a variant of indexed grammars that draw their weights from hierarchical Pitman-Yor processes. We show that this framework generates languages that emulate the statistics of natural language corpora better than the current approach of directly formulating weighted context-free grammars.

1 Introduction

In the World Atlas of Linguistic Structures, Dryer (2013) reports that the plurality of world languages follow a subject-object-verb (SOV) word order. However, relatively few SOV languages (Japanese, Turkish, Persian) have a significant Internet footprint. Today, the Internet is dominated by subject-verb-object (SVO) languages like English, Spanish, and Chinese. The resulting paucity of non-SVO data makes it difficult to study whether linguistic models have an inductive bias towards particular word orders, or to develop models that perform well on low-resource languages from underrepresented linguistic families. In recent work, Wang and Eisner (2016), Ravfogel et al. (2019) and White and Cotterell (2021) argue that artificial languages could be an effective tool for addressing challenges like these, enabling researchers to create large corpora that manifest targeted linguistic phenomena.

An obvious objection presents itself: what if the models aren't realistic enough? If not, then conclusions drawn from artificial languages may not transfer to natural languages. One response to this objection would be to abandon the entire enterprise, and with it the potential advantages of simulated data. An alternative is to follow the tradition of other disciplines who model natural systems (e.g. physics, geology, meteorology) and iterate on these models until they are sufficiently good predictors of observed phenomena.

In this spirit, this paper builds upon the framework of White and Cotterell (2021), who used weighted context-free grammars to construct artificial languages for studying the inductive biases of neural language models towards particular word orders. Observing that their framework did not account for selectional preference (the linguistic phenomenon that head words and their syntactic dependents are not probabilistically independent), we generalize weighted context-free grammars by introducing the *weighted random-access indexed grammar*, which facilitates the development of artificial languages that manifest selectional preference. We also present a methodology for building grammars that emulate statistical relationships observed in natural language corpora. Inspired by Teh (2006), we use hierarchical Pitman-Yor processes (Pitman and Yor, 1997) as the token-generating distributions for open-class categories (like noun, verb, and adjective). We set the hyperparameters by matching the statistics of the produced artificial languages with natural language corpora. As a pilot experiment for our framework, we partially replicate an experiment performed by White and Cotterell (2021) that studied the inductive bias of transformer and LSTM-based language models towards languages featuring various syntactic parameter configurations (Chomsky, 1981; Baker, 2008).

Finally, we accompany this paper with a Python package called `testperanto`[1], to allow researchers to use and refine our framework for further linguis-

[1] `https://github.com/Mark-Hopkins-at-Williams/testperanto` (Apache 2.0 license)

Proceedings of the 26th Conference on Computational Natural Language Learning (CoNLL), pages 85 - 94
December 7-8, 2022 ©2022 Association for Computational Linguistics

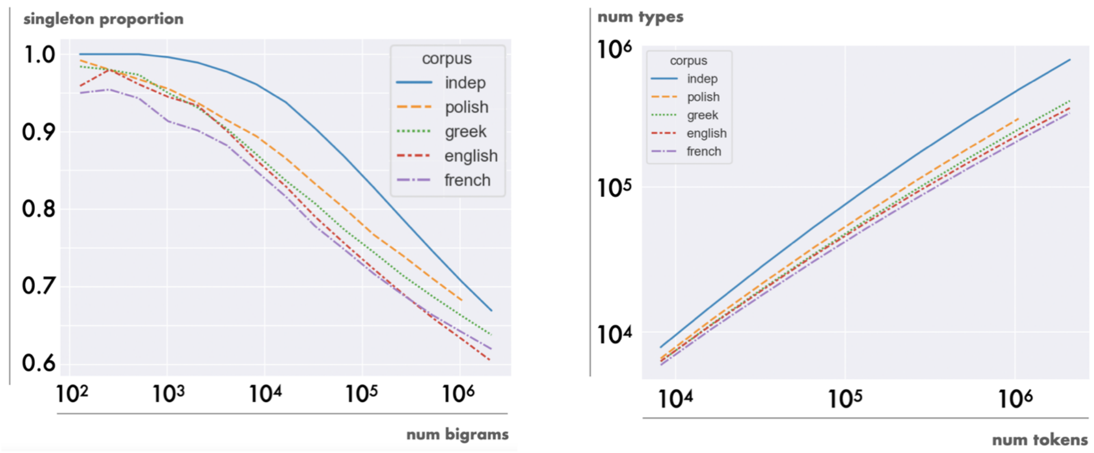

Figure 1: A comparison of the singleton proportion curves of adjective-noun bigrams in the Europarl corpus with bigrams generated using independent adjective and noun distributions.

tic studies.

2 Related Work

Both Wang and Eisner (2016) and Ravfogel et al. (2019) constructed artificial languages by manipulating sentences from existing natural language corpora. Both approaches made use of a dependency parser (or a gold parsed corpus) to inform these manipulations, altering syntactic constituent order (Wang and Eisner, 2016; Ravfogel et al., 2019) or token morphology (Ravfogel et al., 2019).

White and Cotterell (2021) argued that manipulated natural language corpora have downsides. Based on a series of negative results (Cotterell et al., 2018; Mielke et al., 2019), they suggested that it may not be possible to remove confounding linguistic features from an existing corpus, making it difficult to isolate typological features for study. To maximize the ability to run a controlled experiment, they generated fully artificial languages from hand-built weighted context-free grammars. However, although their grammars modeled certain syntactic dependencies (e.g. conjugating a verb with its subject), they did not model semantic dependencies. We assert that it is prohibitively difficult to directly formulate weighted context-free grammars that model semantic dependencies (e.g. selectional preference), motivating our extension – the weighted random-access indexed grammar.

3 Motivation

White and Cotterell (2021) generated artificial language using a *weighted context-free grammar* (WCFG). A WCFG augments a context-free grammar (CFG) with a function q that assigns a non-negative weight $q(r)$ to each grammar rule r. This induces a weight for each derivation: the product of the weights of the rules used in the derivation. More formal details can be found in Collins (2013).

WCFGs produce terminal symbols (words) according to probability distributions that depend exclusively on the grammar nonterminals. Consider the following CFG:

$$\begin{array}{rcl} \mathsf{S} & \rightarrow & \mathsf{NN\ VP} \\ \mathsf{VP} & \rightarrow & \mathsf{VB\ NN} \\ \mathsf{VB} & \rightarrow & \texttt{drank} \mid \texttt{ate} \\ \mathsf{NN} & \rightarrow & \texttt{you} \mid \texttt{it} \mid \texttt{water} \mid \texttt{food} \end{array}$$

By using plain nonterminals like VB and NN, the respective probabilities of sentences `it drank water` and `it drank food` depend only on the probability of the rules VB $\rightarrow$ `water` and VB $\rightarrow$ `food`. Crucially, the verb choice does not differentiate the sentence probabilities. This is unrealistic – it is more common to drink water than to drink food, whereas it is more common to eat food than to eat water. This phenomenon (that linguistic arguments are not independent of their predicates) is known as *selectional preference*.

One way to detect selectional preference (Teh, 2006) is to collect dependency relationships from a parsed natural language corpus (e.g. `amod`, `nsubj`, `dodj`) and extract the dependency bigrams (e.g. for `amod`, the first three dependency bigrams in Europarl are `internal market`, `European citizens`, and `cultural exception`). Then, as we stream through the dependency bigrams, we plot either the number of observed bigram types

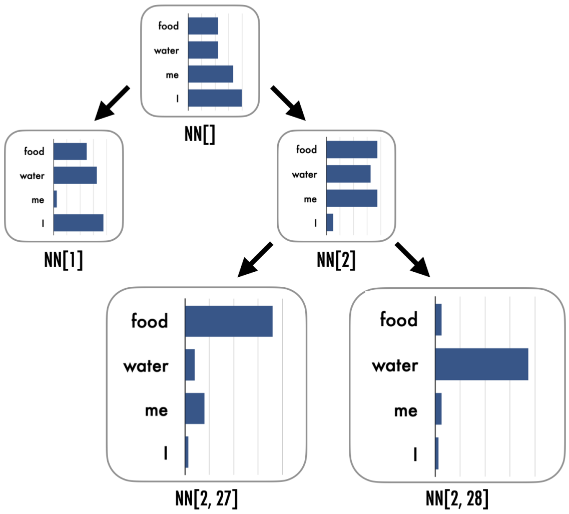

Figure 2: An example hierarchical Pitman-Yor process. NN[] is the global noun distribution. NN[1] and NN[2] respectively represent the likelihood that a noun is the subject or object of a verb. NN[2, 27] and NN[2, 28] respectively represent the likelihood that a noun is the object of verb 27 (eat) or verb 28 (drink) of the vocab.

(a type-token curve) or the proportion of bigrams whose type has been observed exactly once (a singleton proportion curve). In Figure 1, we contrast the curves generated[2] using four Europarl corpora (Koehn, 2005) with a bigram corpus constructed by sampling one adjective and one noun from independent distributions respectively derived from adjective and noun frequency in the English Europarl corpus. The curves generated using the independent bigram corpus are outliers. For instance, when the number of observed bigrams is plotted on a log scale, the natural corpora have roughly linear singleton proportion curves, whereas the independent corpus has a considerable bow in the curve.

We would like to generate artificial languages such that the dependencies have similar statistics to naturally observed dependencies. Rather than independently generating open-class words, Teh (2006) suggests using a hierarchical Pitman-Yor process (Pitman and Yor, 1997) – a tree-structured set of distributions over the same domain, in which child distributions are resamplings of their parents. Figure 2 shows an example. A hierarchical Pitman-Yor process allows us to model context-specific word distributions (e.g. `food` is more likely to appear as the object of the verb `eat` than `water`, `I`, or `me`) that are jointly influenced by global word frequency priors. A Pitman-Yor process $\mathsf{PY}(d, \theta, P_{\mathsf{base}})$ is characterized by a *discount* parameter $d \in [0, 1)$, a *strength* parameter $\theta \in (-d, \infty)$, and a *base distribution* P_{base} over integers $\{1, \ldots, V\}$. We follow (Teh, 2006) in describing a Pitman-Yor process as a stochastic process that generates samples $\langle x_1, x_2, \ldots\rangle$ from i.i.d. samples $\langle y_1, y_2, \ldots\rangle$ drawn from base distribution P_{base}. Intuitively, it is a "rich-get-richer" process, in which the jth sample x_j is set to either the value y_i assigned to a previous x-sample (with probability proportional to the number of previous x-samples that were assigned the value y_i), or the next y-sample in the sequence that hasn't yet been used. Formally, let $b_1 = 1$ and draw subsequent binary values b_{n+1} from a Bernoulli (coin-flip) distribution where:

$$P(b_{n+1} = 1) = \frac{\theta + d \sum\limits_{1 \leq i \leq n} b_i}{\theta + n}$$

Variable b_{n+1} determines whether the $(n+1)$th sample is set to the value of a previous assignment ($b_{n+1} = 0$) or the next unused y_i sample ($b_{n+1} = 1$). Now define $t_1 = 1$ and consider $j, n \in \mathbb{Z}^+$. If $b_{n+1} = 0$, then let $t_{n+1} = j$ with probability:

$$\frac{1}{n} \sum_{1 \leq i \leq n} \mathbb{1}(t_i = j)$$

Otherwise, if $b_{n+1} = 1$:

$$t_{n+1} = 1 + \sum_{1 \leq i \leq n} b_i$$

The nth sample drawn from the Pitman-Yor process is $x_n = y_{t_n}$. A Pitman-Yor process, for all practical purposes, can generate an "open-class" of words by using a uniform base distribution P_{unif} with a sufficiently large vocabulary size V (for our experiments, we use the space of all 32-bit integers).

A hierarchical Pitman-Yor process is simply a Pitman-Yor process that uses another Pitman-Yor process as its base distribution. For instance, we could define a global adjective distribution $P_{\mathsf{adj}} = \mathsf{PY}(0.4, 500, P_{\mathsf{unif}})$, and then for noun y_1 of our vocabulary, we could define a noun-dependent adjective distribution $P_{\mathsf{adj},\mathsf{y}_1} = \mathsf{PY}(d, \theta, P_{\mathsf{adj}})$.

4 Approach

The main challenge: how do we construct a WCFG that derives its weights from the linked distributions of a hierarchical Pitman-Yor process? Concerned with the induction of better n-gram language

[2] To generate Figure 1, we shuffled the Europarl sentences and extracted the adjective-noun dependencies using `spaCy`. The shuffling smooths irregularities caused by topic shift.

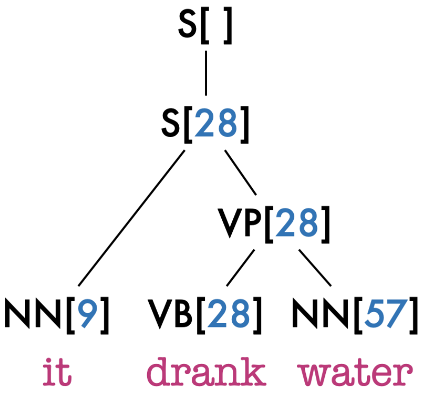

Figure 3: An example derivation, using the indexed grammar from Figure 4.

			ζ
S[]	$\rightarrow$	S[z_1]	$z_1 \mapsto$ VB[]
S[y_1]	$\rightarrow$	NN[z_1] VP[y_1]	$z_1 \mapsto$ NN[1, y_1]
VP[y_1]	$\rightarrow$	VB[y_1] NN[z_1]	$z_1 \mapsto$ NN[2, y_1]
VB[27]	$\rightarrow$	ate	
VB[28]	$\rightarrow$	drank	
NN[9]	$\rightarrow$	it	
NN[56]	$\rightarrow$	food	
NN[57]	$\rightarrow$	water	

Figure 4: An example indexed grammar. The base weight $w_0(\rho)$ of each indexed rule ρ is 1.

models, previous work (Teh, 2006; Blunsom and Cohn, 2011) mainly focused on how to incorporate hierarchical Pitman-Yor processes into sequential models like Hidden Markov Models. Here, our concern is how to incorporate these distributions into a generative syntactic model convenient for engineering artificial languages with specific linguistic typologies. There exist many syntactic models to choose from, including dependency grammars (Eisner, 1996), tree-adjoining grammars (Joshi, 1987), lexical functional grammars (Kaplan, 1985), CCGs (Steedman and Baldridge, 2011), HPSGs (Pollard and Sag, 1994) and GPSGs (Gazdar et al., 1985). In this work, we choose to extend context-free grammars, partly because of their popularity and partly to facilitate comparison with (White and Cotterell, 2021), who used WCFGs – however, our approach can be adapted to other syntactic formalisms.

4.1 Intuition

Our approach is a variation on indexed grammars (Aho, 1968; Hopcroft et al., 2001), which augment CFG nonterminals with a sequence of symbols called *indices*. Before going through the formalism, we briefly preview how it works, using a derivation (Figure 3) for an example indexed grammar (Figure 4). At the top level, it applies CFG rule S[] $\rightarrow$ S[28], which involves two choices:

1. the choice of "indexed rule": S[] $\rightarrow$ S[z_1]
2. the choice of indices to assign to its z-variables: $\{z_1 \mapsto 28\}$

Next, the derivation expands S[28] by applying the CFG rule S[28] $\rightarrow$ NN[9] VP[28]. Again, this involves two choices:

1. the choice of indexed rule: S[y_1] $\rightarrow$ NN[z_1] VP[y_1]
2. the choice of indices to assign to its z-variables: $\{z_1 \mapsto 9\}$

Note the role of the variables: y-variables match LHS indices and copy them to the RHS, whereas z-variables introduce new indices on the RHS. Each z-variable z_i of an indexed rule is associated with a key $\zeta(z_i)$ (Figure 4, right column) that references a distribution in a "distribution table" τ. The weight associated with a derivation rule (e.g. S[28] $\rightarrow$ NN[9] VP[28]) is the product of the base weight w_0 of the indexed rule (e.g. w_0(S[y_1] $\rightarrow$ NN[z_1] VP[y_1]), and the probabilities of the z-assignments (e.g. τ(NN[1, 28])(9)). As with CFGs, the weight of a derivation is the product of the derivation rules.

4.2 Random-access Indexed Grammars

Let $Y = \{y_1, y_2, ...\}$ and $Z = \{z_1, z_2, ...\}$ be reserved symbols called y- and z-variables. A *random-access indexed grammar (RIG)*[3] is a 5-tuple (N, T, F, S, R) where:

- N is a set of *nonterminal* symbols
- T is a set of *terminal* symbols
- F is a set of *index* symbols, or *indices*[4]
- $S \in N$ is the *start symbol*

[3] The standard definition of indexed grammars (Hopcroft et al., 2001) treats the indices as a stack, rather than as a random-access array. Our departure from the standard definition (introducing y- and z-variables to allow random-access matching) prioritizes the ease of grammar engineering over definitional conciseness and representational power. Moreover, since our use case is generation, we are not concerned with indexed grammar variants that prioritize efficiency of parsing or induction (e.g. (Gazdar, 1987)).

[4] In this paper, we will use the set of nonnegative 32-bit integers as our set F of indices.

- R is a finite set of *indexed rules* (to be defined shortly)

In contrast to standard CFG rules, indexed rules use *indexed nonterminals*, symbols of the form $\mathsf{A}[\phi]$, where $\mathsf{A} \in N$ and $\phi \in (F \cup Y \cup Z)^*$. A *grounded indexed nonterminal* is an indexed nonterminal $\mathsf{A}[\phi]$ such that $\phi \in F^*$. An *indexed rule* has the form:

$$\mathsf{A}[\phi] \rightarrow \mathsf{rhs}$$

where $\mathsf{A}[\phi]$ is an indexed nonterminal without z-variables, and rhs is a sequence of terminals and indexed nonterminals whose y-variables all appear in ϕ.

To define the semantics of a RIG, let a *substitution* be a function $\sigma : D \rightarrow F$ with domain $D \subseteq Y \cup Z$. We apply a substitution σ to a indexed nonterminal $\mathsf{A}[\phi_1, \ldots, \phi_n]$ as follows:

$$\sigma(\mathsf{A}[\phi_1, \ldots, \phi_n]) = \mathsf{A}[\bar{\sigma}(\phi_1), \cdots, \bar{\sigma}(\phi_n)]$$

where:

$$\bar{\sigma}(x) = \begin{cases} \sigma(x) & \text{if } x \in D \\ x & \text{if } x \notin D \end{cases}$$

for $x \in F \cup Y \cup Z$. We apply a substitution σ to an indexed rule ρ by applying σ to every indexed nonterminal in ρ. For example, if:

$$\begin{aligned} \sigma &= \{\mathsf{y}_1 \mapsto 52, \mathsf{z}_1 \mapsto 14\} \\ \rho &= \mathsf{S}[\mathsf{y}_1] \rightarrow \mathsf{NN}[\mathsf{z}_1]\ \mathsf{VP}[\mathsf{y}_1] \end{aligned}$$

then:

$$\sigma(\rho) = \mathsf{S}[52] \rightarrow \mathsf{NN}[14]\ \mathsf{VP}[52]$$

Each indexed rule ρ implicitly represents the set of CFG rules that can be obtained by applying a substitution to the variables of the indexed rule:

$$\mathcal{R}(\rho) = \{\sigma(\rho) \mid \sigma : V(\rho) \rightarrow F\}$$

Here, $V(\rho) \subseteq Y \cup Z$ is the set of variables that appear in indexed rule ρ. The RIG encodes a CFG consisting of the union $\bigcup_{\rho \in R} \mathcal{R}(\rho)$ of these rules.

4.3 Weighted RIGs

Next, we introduce weights from a hierarchical Pitman-Yor process. We reference the process distributions via a *distribution table* – a function τ that maps grounded indexed nonterminals to distributions (e.g. the distributions of a hierarchical Pitman-Yor process). For instance, in the distribution table τ implied by Figure 2, $\tau(\mathsf{NN}[2, 28])$ corresponds to the lower right distribution.

A weighted random-access indexed grammar (WRIG) is a tuple (G, τ, w_0, ζ) where:

- $G = (N, T, F, S, R)$ is a RIG
- τ is a distribution table
- w_0 assigns a nonnegative weight (called the *base weight*) to each indexed rule $\rho \in R$
- ζ assigns a z-*weighting* to each indexed rule $\rho \in R$. The z-weighting $\zeta(\rho)$, abbreviated ζ_ρ for clarity, is a function that assigns an indexed nonterminal (that may contain y- but not z-variables) to each z-variable of the rule.

Every WRIG encodes a WCFG. Each CFG rule $r = \sigma(\rho)$ encoded by indexed rule ρ (where $\sigma : V(\rho) \rightarrow F$ is a substitution) has weight:

$$q(r) = w_0(\rho) \cdot \prod_{\mathsf{z} \in Z(\rho)} w_{\mathsf{z}}(\sigma(\mathsf{z}))$$

where $Z(\rho) \subseteq Z$ is the set of z-variables that appear in indexed rule ρ, and $w_{\mathsf{z}} = \tau(\sigma(\zeta_\rho(\mathsf{z})))$ is the distribution associated with grounded indexed nonterminal $\sigma(\zeta_\rho(\mathsf{z}))$ in the distribution table τ.

Example: The second rule of the RIG in Figure 4 encodes (among others) the CFG rule:

$$\mathsf{S}[28] \rightarrow \mathsf{NN}[9]\ \mathsf{VP}[28]$$

The weight of this CFG rule is:

$$\begin{aligned} & w_0(\mathsf{S}[\mathsf{y}_1] \rightarrow \mathsf{NN}[\mathsf{z}_1]\ \mathsf{VP}[\mathsf{y}_1]) \\ \cdot\ & \tau(\mathsf{NN}[1, 28])(9) \end{aligned}$$

In other words, it is the base weight of the indexed rule, multiplied by the probability of word 9 (`it`) being the subject of verb 28 (`drink`).

4.4 Voiceboxes

Using a WRIG, syntax can be specified with relative ease, i.e. without the need to manually formulate an arduous number of rules. However, terminal rules (i.e. rules that generate the lexemes) are a different story. We need an auxiliary mechanism to automatically invent lexemes from grounded indexed preterminals, i.e. a mechanism that will translate a preterminal (see Figure 4) like `VB[27]` – the 27th verb of the vocabulary – into a lexeme (e.g., `ate`). To do so, we pair the WRIG with a *voicebox*,

			ζ
$\mathsf{S}[]$	$\rightarrow$	$\mathsf{S}[z_1, z_2]$	$z_1 \mapsto \mathsf{VB}[], z_2 \mapsto \mathsf{COUNT}[]$
$\mathsf{S}[y_1, y_2]$	$\rightarrow$	$\mathsf{IC}[y_1, y_2]$, $\mathsf{DC}[z_1, z_2]$	$z_1 \mapsto \mathsf{VB}[], z_2 \mapsto \mathsf{COUNT}[]$
$\mathsf{IC}[y_1, y_2]$	$\rightarrow$	$\mathsf{NP}[z_1, y_2, 1]$ $\mathsf{VP}[y_1, y_2]$	$z_1 \mapsto \mathsf{NN}[1, y_1]$
$\mathsf{DC}[y_1, y_2]$	$\rightarrow$	`weil` $\mathsf{NP}[z_1, y_2, 1]$ $\mathsf{VPD}[y_1, y_2]$	$z_1 \mapsto \mathsf{NN}[1, y_1]$
$\mathsf{VP}[y_1, y_2]$	$\rightarrow$	$\mathsf{VB}[y_1, y_2]$ $\mathsf{NP}[z_1, z_2, 2]$	$z_1 \mapsto \mathsf{NN}[2, y_1], z_2 \mapsto \mathsf{COUNT}[]$
$\mathsf{VPD}[y_1, y_2]$	$\rightarrow$	$\mathsf{NP}[z_1, z_2, 2]$ $\mathsf{VB}[y_1, y_2]$	$z_1 \mapsto \mathsf{NN}[2, y_1], z_2 \mapsto \mathsf{COUNT}[]$
$\mathsf{NP}[y_1, y_2, y_3]$	$\rightarrow$	$\mathsf{DT}[y_2, y_3]$ $\mathsf{NN}[y_1, y_2, y_3]$	

Figure 5: A WRIG capturing simple German syntax and morphology. Each indexed rule has base weight 1.

x	$\tau(x)$	description
$\mathsf{VB}[]$	$\mathsf{PY}(0.4, 1, P_{\mathsf{unif}})$	global verb distribution
$\mathsf{NN}[]$	$\mathsf{PY}(0.4, 500, P_{\mathsf{unif}})$	global noun distribution
$\mathsf{NN}[1]$	$\mathsf{PY}(0.4, 500, \tau(\mathsf{NN}[]))$	global subject distribution
$\mathsf{NN}[1, y_1]$	$\mathsf{PY}(0.4, 10, \tau(\mathsf{NN}[1]))$	subject distribution for head verb y_1
$\mathsf{NN}[2]$	$\mathsf{PY}(0.4, 500, \tau(\mathsf{NN}[]))$	global object distribution
$\mathsf{NN}[2, y_1]$	$\mathsf{PY}(0.4, 0.1, \tau(\mathsf{NN}[2]))$	object distribution for head verb y_1
$\mathsf{COUNT}[]$	$\mathsf{Unif}(\{1, 2\})$	global count distribution (1=singular, 2=plural)

Figure 6: Distribution table for the WRIG in Figure 5. P_{unif} is a uniform distribution over all 32-bit integers.

a function that maps grounded indexed nonterminals (specifically, preterminals) to lexemes. The voicebox is then used to generate terminal rules on-the-fly. Note that the voicebox can also support morphology. For example, if the preterminal $\mathtt{VB}[27, 3, 1]$ encodes the third-person singular conjugation of verb 27, then the voicebox might produce $\beta(\mathtt{VB}[27, 3, 1]) = \mathtt{eats}$.

5 Demo: Simple German Syntax with Selectional Preference

To demonstrate how linguistic phenomena can be modeled by a WRIG, we present a small example in Figure 5, whose distribution table is given by Figure 6. It models various aspects of German syntax: word order (independent clauses are SVO, whereas dependent clauses are SOV), verb conjugation (present singular and present plural), and case roles (nominative and accusative). Figure 7 shows the first five sentences of a corpus generated by the WRIG. To interpret the indexed nonterminals, note that subject count (1=singular, 2=plural) and case (1=nominative, 2=accusative) are encoded as integer indices:

- $\mathsf{S}[y_1, y_2], \mathsf{IC}[y_1, y_2], \mathsf{DC}[y_1, y_2]$: respectively produce a sentence, independent clause, and dependent clause with subject count y_2, whose head is the ${y_1}^{\text{th}}$ verb of the vocabulary
- $\mathsf{NP}[y_1, y_2, y_3]$: produces a noun phrase with count y_2 and case y_3, whose head is the ${y_1}^{\text{th}}$ noun of the vocabulary
- $\mathsf{VP(D)}[y_1, y_2]$: produces a (dependent clause) verb phrase with subject count y_2, whose head is the ${y_1}^{\text{th}}$ verb of the vocabulary
- $\mathsf{NN}[y_1, y_2, y_3]$: produces the ${y_1}^{\text{th}}$ noun of the vocabulary, declined for count y_2 and case y_3
- $\mathsf{VB}[y_1, y_2]$: produces the ${y_1}^{\text{th}}$ verb of the vocabulary, conjugated for subject count y_2
- $\mathsf{DT}[y_1, y_2]$: produces a determiner for a noun with count y_1 and case y_2

Terminal rules for open-class nonterminals $\mathsf{NN}[y_1, y_2, y_3]$ and $\mathsf{VB}[y_1, y_2]$ are generated by a voicebox that randomly concatenates German syllables to create new words, and adds German morphological endings based on count and case. For the closed-class $\mathsf{DT}[y_1, y_2]$, the voicebox generates the German definite determiner for the specified count and case. For instance (see Figure 7), the noun hunghub[5] appears as `den hunghub` when it is accusative singular and `die hunghuben` when it is accusative plural.

[5] In this grammar, all nouns are masculine. See the `testperanto` tutorials for an example of how to model noun gender.

der zerheimherrun konzumschlage den lagfrischhan , weil der terterfin die wirnachparen kennjahre
der dungtun milchsichkeite die hunghuben , weil die vorsamrichen den tagwohn jahrkolen
der derver milchsichkeite den hunghub , weil der tiktikflach die hunghuben milchsichkeite
die kenngunhungen milchsichkeiten den milchmanmilch , weil der tiklang den frauhung telmonhane
der niedlang milchsichkeite den dichgeh , weil die frauhungrungen die langterleren samkenntelen

Figure 7: Example sentences generated by the simple German WRIG. Observe that the verb `milchsichkeiten` strongly tends to take the noun hunghub as its object – the hyperparameters of this particular WRIG have been set to encourage atypically strong selectional preference between verbs and their objects.

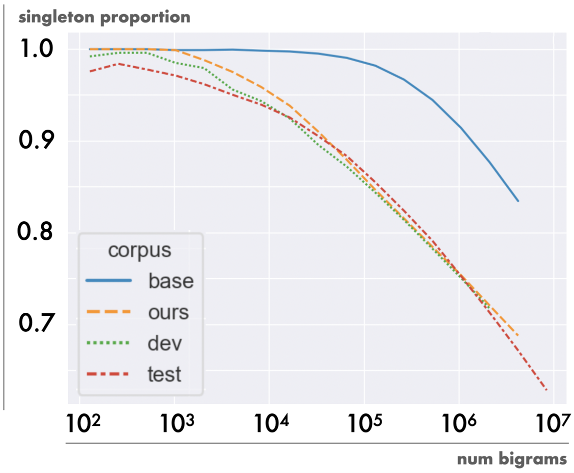

Figure 8: Singleton proportion of verb-object dependency bigrams as corpus size increases.

		singleton proportion		type-token ratio	
		`dev`	`test`	`dev`	`test`
amod	base	0.099	0.094	0.23	0.23
	ours	**0.0074**	**0.013**	**0.016**	**0.018**
nsubj	base	0.045	0.057	0.083	0.12
	ours	**0.0044**	**0.010**	**0.014**	**0.041**
dobj	base	0.081	0.088	0.18	0.22
	ours	**0.0081**	**0.014**	**0.036**	**0.054**

Figure 9: Absolute difference of singleton proportion and type-token ratio between artificial corpora (ours and base) and natural corpora (dev and test), averaged over power-of-two corpora sizes from 2^7 to 2^{22}.

By associating the noun distributions with the distributions of a hierarchical Pitman-Yor process, we also model selectional preference. By assigning a Pitman-Yor process of very low strength (0.1) to the verb-dependent object distributions, we enforce unusually strong selectional preference between verbs and objects, allowing us to see its manifestation of in just a small sample of generated sentences (Figure 7). In particular, the invented verb `milchsichkeiten` frequently takes the noun hunghub as its object.

6 Experiment: Word Order Bias

As a pilot study of our framework, we re-created an experiment performed by White and Cotterell (2021), who used WCFGs to investigate the inductive biases of neural language models for various word orders exhibited by natural language. We created a WRIG based on their WCFG description, which produces simple declarative sentences with relative clauses, prepositional phrases, and clausal complements. We used a voicebox that assigned concatenations of random syllables to each generic noun, verb, and adjective. It used English prepositions, determiners, and morphology (e.g. verbs with a singular subject were suffixed with the letter "s"). We set the parameters of our Pitman-Yor processes by specifying discount and strength parameters so that our produced sentences closely matched the type-token ratio and singleton proportion curves of the English side of the WMT 2014 German-English parallel corpus (Bojar et al., 2014; Luong et al., 2015) for the following dependency bigrams: adjective-noun (amod), verb-subject (nsubj), verb-object (dobj). Figure 8 compares the singleton proportion curves of verb-object dependencies for our generated corpus, versus the development corpus (WMT 2014 Ger-Eng) and a held-out test corpus: the English side of the JParaCrawl 3.0 Jpn-Eng corpus (Morishita et al., 2022). We also compare our corpus statistics to a baseline that attempts to replicate (White and Cotterell, 2021), using independent adjective, noun, and verb distributions rather than tied hierarchical Pitman-Yor distributions. Visual inspection shows that the independent baseline is an outlier, unrepresentative of the statistics manifested by natural corpora. We can distill these curves into a single numeric indicator by averaging the absolute difference between an artificial corpus curve (`ours` or `base`) and a natural corpus curve

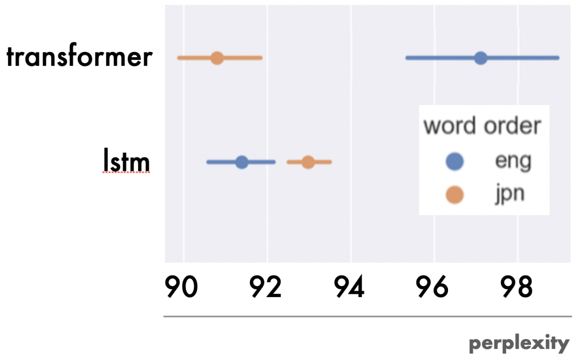

Figure 10: Visualization of experimental results using a point plot. The transformer produces lower-perplexity language models for the artificial languages that follow a Japanese word order, while the LSTM produces lower-perplexity language models for the artificial languages that follow an English word order.

(dev or test) for each power of two on the x-axis. Figure 9 presents these numbers for singleton proportion and the type-token ratio: the statistics for our generated corpus are an order-of-magnitude closer to natural corpora than the baseline.

We created two variants of the WRIG, corresponding to the standard word orders of English and Japanese. For instance, as a head-final language, the Japanese WRIG included the rule[6]:

$$\mathsf{VP}[y_1, y_2] \to \mathsf{NP}[z_1, z_2]\ \mathsf{VB}[y_1, y_2]$$

and as a head-initial language, the English WRIG included the rule:

$$\mathsf{VP}[y_1, y_2] \to \mathsf{VB}[y_1, y_2]\ \mathsf{NP}[z_1, z_2]$$

Following (White and Cotterell, 2021), the WRIGs also differed in:

- the position of the complementizer in complements, relative to the sentential component
- the position of the adposition in adpositional phrases, relative to the adpositional object
- the position of a relative clause, relative to the noun it modifies

We generated 1,000,000 sentences for each WRIG variant, and divided these into ten evenly sized corpora. Each corpus of 100,000 sentences was further divided into an 80k-10k-10k train-dev-test partition. On each train set, we trained[7] a transformer-based and an LSTM-based language model, resulting in 10 trained language models (LMs) per choice of neural architecture and WRIG. Finally, we evaluated these LMs on the respective test sets.

For each architecture (transformer and LSTM) and word order (English and Japanese), Figure 10 visualizes the test perplexity over the ten trials using a point plot[8]. For transformer LMs, we obtained lower perplexity on the languages that followed a Japanese word order. For LSTM LMs, we observed the opposite: a (statistically significant) lower perplexity on the languages that followed an English word order. While these results generally support the findings of White and Cotterell (2021), White and Cotterell (2021) did not find significant differences between the LSTM LMs. We find it encouraging that our results do not differ wildly from White and Cotterell (2021) (it would be troubling for the prospects of artificial languages if each iterative improvement dramatically reversed the conclusions of the previous iteration). At the same time, we also find it encouraging that the differences between their results and ours offer a possible reconciliation between White and Cotterell (2021) and Ravfogel et al. (2019), who reported, based on experiments with naturally-derived corpora, that LSTM LMs performed better on SVO (versus SOV) languages.

7 Conclusion

With this work, our goal is to enable researchers to more easily develop models for typologically diverse languages, and to investigate under what conditions such models perform effectively. By demonstrating that RIGs (weighted by hierarchical Pitman-Yor processes) can model realistic syntactic and semantic dependencies, we hope to provide some confidence that the framework can prove a useful proxy for real-world data, when such data is not readily available. To facilitate adoption of our framework, we are also releasing an open-source Python package called testperanto for building WRIGs, providing fellow researchers with a means to generate artificial languages that emulate the typology of the natural languages they seek to study.

[6] A brief guide to the referenced indexed nonterminals of the WRIG: $\mathsf{VP}[y_1.y_2]$ produces a verb phrase with subject count y_2, whose head is the $y_1{}^{th}$ verb of the vocabulary. $\mathsf{NP}[y_1, y_2]$ produces a noun phrase with count y_2, whose head is the $y_1{}^{th}$ noun of the vocabulary. $\mathsf{VB}[y_1, y_2]$ produces the $y_1{}^{th}$ verb of the vocabulary, conjugated for subject count y_2.

[7] Like White and Cotterell (2021), we used the fairseq implementation (Ott et al., 2019) of these language models.

[8] We used seaborn to generate the plot. A point plot shows the mean of the ten trials (the dot) and the 95% confidence interval (the line).

References

Alfred V Aho. 1968. Indexed grammars—an extension of context-free grammars. *Journal of the ACM (JACM)*, 15(4):647–671.

Mark C Baker. 2008. *The atoms of language: The mind's hidden rules of grammar*. Basic books.

Phil Blunsom and Trevor Cohn. 2011. A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 865–874, Portland, Oregon, USA. Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Noam Chomsky. 1981. Principles and parameters in syntactic theory. *Explanation in linguistics*, pages 32–75.

Michael Collins. 2013. Lexicalized probabilistic context-free grammars. *Lecture Notes*.

Ryan Cotterell, Sabrina J. Mielke, Jason Eisner, and Brian Roark. 2018. Are all languages equally hard to language-model? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 536–541, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew S. Dryer. 2013. Order of subject, object and verb. In Matthew S. Dryer and Martin Haspelmath, editors, *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Gerald Gazdar. 1987. COMIT ==> PATR II. In *Theoretical Issues in Natural Language Processing 3*.

Gerald Gazdar, Ewan Klein, Geoffrey K Pullum, and Ivan A Sag. 1985. *Generalized phrase structure grammar*. Harvard University Press.

John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. 2001. Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1):60–65.

Aravind K Joshi. 1987. An introduction to tree adjoining grammars. *Mathematics of language*, 1:87–115.

Ronald M. Kaplan. 1985. Structural correspondences and Lexical-Functional Grammar. In *Proceedings of the first Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, Hamilton, NY.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Sabrina J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4975–4989, Florence, Italy. Association for Computational Linguistics.

Makoto Morishita, Katsuki Chousa, Jun Suzuki, and Masaaki Nagata. 2022. Jparacrawl v3. 0: A large-scale english-japanese parallel corpus. *arXiv preprint arXiv:2202.12607*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Jim Pitman and Marc Yor. 1997. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, pages 855–900.

Carl Pollard and Ivan A Sag. 1994. *Head-driven phrase structure grammar*. University of Chicago Press.

Shauli Ravfogel, Yoav Goldberg, and Tal Linzen. 2019. Studying the inductive biases of RNNs with synthetic variations of natural languages. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3532–3542, Minneapolis, Minnesota. Association for Computational Linguistics.

Mark Steedman and Jason Baldridge. 2011. Combinatory categorial grammar. *Non-Transformational Syntax: Formal and Explicit Models of Grammar. Wiley-Blackwell*, pages 181–224.

Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992, Sydney, Australia. Association for Computational Linguistics.

Dingquan Wang and Jason Eisner. 2016. The galactic dependencies treebanks: Getting more data by synthesizing new languages. *Transactions of the Association for Computational Linguistics*, 4:491–505.

Jennifer C. White and Ryan Cotterell. 2021. Examining the inductive bias of neural language models with artificial languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 454–463, Online. Association for Computational Linguistics.