Feature Induction for Online Constraint-based Phonology Acquisition

Jason Naradowsky

Abstract

Log-linear models provide a convenient method for coupling existing machine learning methods to constraint-based linguistic formalisms like optimality theory and harmonic grammar. While the learning methods themselves have been well studied in this domain, the question of how these constraints originate is often left unanswered. We present a novel, error-driven approach to constraint induction that performs lightweight decisions based on local information. When evaluated on the task of reproducing human gradient phonotactic judgements, a model trained with this procedure can sometimes nearly match the performance of state-of-the-art methods that rely on global information and individual assessment of all possible constraints. We conclude by discussing methods for incorporating context and linguistic bias into the induction scheme to produce more accurate grammars.

1 Introduction

Log-linear models, also known as maximum entropy models in NLP, have been successfully applied to many linguistic and language-related problems (in part-of-speech tagging [10], named entity recognition [2], and parsing [13]). They have also enjoyed a long history of use as the basis for models of phonological grammar, being first applied to phonology by Goldwater & Johnson [3], and subsequently [14], [5], amongst others.

Since these models consist only of a set of weighted features, they are inherently similar to Harmonic Grammar (HG; see overviews in [12], [8]) and a natural candidate for coupling established learning methods to this linguistic formalism. Indeed, previous work has examined the behavior of standard learning algorithms in this domain [1], and have incorporated additional machine learning concepts, like regularization [7], into the learning of phonological grammar with hidden metrical structures. However, the more fundamental question of where these constraints originate is often ignored.

A counterexample is the work of Hayes and Wilson (H&W; [14]), in which a constraint induction procedure [1] is used to bootstrap, from an empty constraint set, a phonological grammar capable of capturing the gradience of human phonotactic predictions. In this approach the learner is only exposed to a set of observed onset clusters, and is aware of the natural classes of phonetic features associated with each phoneme. Induction occurs by iterating over all the observed clusters, and measuring the usefulness of each possible constraint on the whole of the data. A small set of constraints that are deemed to be most appropriate are selected, based on multiple thresholds, and added to the constraint set. The constraints themselves are composed of logical combinations of the natural classes of their contained phonemes. This procedure is similar in spirit to much of the work on feature induction for similar models in the statistical literature [9], and for conditional-random fields [6].

This is a very effective learning strategy, but it gains that effectiveness at the cost of computational efficiency, and the number of hypothesis that must be simultaneously considered is very large. As it is unclear if the human learning mechanism could cope with these large computational burdens, this approach may be ill-suited as a cognitive model of the human language acquisition process. And it is important to note that while this approach takes a very top-down, or global strategy, it is not guaranteed to be optimal, as it does not consider interactions between constraints at each induction step, and to reason over all possible subsets of all possible constraints would be absolutely infeasible. Therefore both this induction strategy and the one we will introduce below are both, to one degree or another, approximations of the optimal learner.

We propose an alternative learning strategy in which constraint induction is a fundamentally error-driven process that occurs at a comparably local level, removing a great deal of the computational overhead of more global approaches. Under this strategy both constraint induction and the adjustment of the constraint weights can be explained by the same contrastive mechanism: when a form is observed a neighborhood of phonetically similar forms are produced. If the model rates a neighboring form as more acceptable than the observed form, the constraint weights are updated. Optionally, a constraint induction process is performed in which the differences between the two forms are exploited to limit the scope of the constraint induction decision, as the distinctive features that distinguish the winning form from the observed form must form the basis of the appropriate constraints.

[1] Where a phonological constraint in the grammar corresponds directly to a feature in the log-linear model.

Hence this method does not unambiguously select optimal constraints, but instead determines with high precision where to start the search, and can easily reduce the number of hypothesized constraints by orders of magnitude. We show that this method can, although with higher variance, often provide gradient phonotactic judgements competitive with the more global, state-of-the-art approach of H&W. We introduce a method for determining when to update feature weights and when to induce constraints, and find that together with a small history, and the use of non-uniform priors, we are able to mitigate the larger amount of variance in the model's solutions. The benefits gained from caching a small history support the notion that learning might occur in brief clusters of particularly helpful examples [11].

2 Log-linear Models of Phonological Grammars

Log-linear models, as we will discuss here, can be useful for efficient estimation of conditional parameters, and classification of a piece of data based on its characteristics. These characteristics are known as features in the machine learning literature, and as constraints when phrased as a model of phonotactic grammar. During training the weights of these constraints are adjusted such that a function over them will maximize the probability of placing the correct label to each example.

The form of the model is:

$$P(x) = \frac{e^{-h(x)}}{\sum_{n}^{N} e^{-h(n)}} \tag{1}$$

Where $h(x)$ is a score function, x is an example form, and $n \in N$ represents an enumeration over all the possible forms. This formula describes the probability of the example form x as the negated an exponentiated score $h(x)$ of the form, normalized by the sum of all such scores. We can largely ignore the normalization because we in learning we are focused only on the unnormalized scores, known as *harmony* in harmonic grammar:

$$h(x) = \sum_{i=1}^{F} w_i C_i(x) \tag{2}$$

For example x, with constraints C_1 to C_F, where F is the size of the constraint set, the score is the sum of each constraint weight, w_i, multiplied by its count C_i - the number of times the constraint is violated in example x.

2.1 Learning with the Perceptron Update

In this section we describe, primarily through example, how the model assigns probability mass to examples, and how the model can learn via the perceptron update method.

Iteration 1:

/pa/	*-voice ($w1 = 2.0$)	*+voiced ($w_2 = 1.5$)	*+lab ($w_3 = 1.0$)	score, $h(x)$	$e^{-h(x)}$
/pa/	1 (+2.0)	1 (+1.5)	1 (+1.0)	4.5	.011
→ /ba/	0 (+0.0)	2 (+3.0)	1 (+1.0)	4.0	.018

The model selects /ba/ as the optimal candidate by assigning it a lower score $h(x)$, or a higher $e^{-h(x)}$, based on the current feature weights. Because our preferred form is /pa/, the model has made an error, and this erroneous classification triggers a perceptron update step.

The perceptron update itself is quite simple. First, we compute the vector containing the difference between the counts of the winning example and the counts of the preferred example:

	*-voice	*+voiced	*+lab
/pa/ - /ba/	0 - 1 = - 1	2 - 1 = 1	1 - 1 = 0

The weight parameter is then updated by incrementing each weight by the count difference in this vector [2]. Alternatively the perceptron update can . In this example the weight of *[+voice] is updated from its original value of 1.5, to 2.5, based on the count vector difference of 1. When we reevaluate the candidates with the updated model, we find that the model now selects the correct form:

Iteration 2:

/pa/	*-voice ($w1 = 1.0$)	*+voiced ($w_2 = 2.5$)	*+lab ($w_3 = 1.0$)	score, $h(x)$	$e^{-h(x)}$
→ /pa/	1 (+1.0)	1 (+2.5)	1 (+1.0)	4.5	.011
/ba/	0 (+0.0)	2 (+5.0)	1 (+1.0)	6.0	.002

[2] The update is often phrased as the product of the difference counts with a learning rate, τ, with this example representing a special case where $\tau = 1.0$.

In contrast to H&W, and the gradient descent / stochastic gradient descent approaches to log-linear training, it is important to note that the perceptron update is fundamentally local in nature: it operates over small contrastive neighborhoods, and therefore does not have the same convergence guarantees as that have a more global optimization function.

3 Plausibility of Global Decisions

Constraint induction, the process by which constraints are added to the model, can be a deceiving term. While the nomenclature may conjure the notion that these constraints are carefully constructed to improve the model, in domains where the space of constraints are small induction is much more appropriately phrased as a search: *In what order will we explore the space.* When we restrict the notion of a phonological grammar to focus only on English onset clusters, we find ourselves in such a domain; though the space is still extremely large, it is feasibly enumerable.

This leads us to the global, "bird's-eye view" approach, where, given some measure of how useful a constraint is, we can enumerate all constraints and choose to add all constraints whose usefulness is above some threshold. The problem is not entirely that simple, as there are many metrics we could use to gauge the usefulness of each constraint, and this usefulness might depend on the weights of the current features in the grammar, which are constantly changing.

Prominent work in constraint induction for log-linear models takes this approach [14] and we take this work as a constant point of comparison throughout this paper. The particular measure used, for a constraint C_i is the observed number of violations in the corpus, $O[C_i]$, minus the expected number, $E[C_i]$. For the observed counts we can simply enumerate over the corpus and collect this statistic exactly. For the expected counts we must sum the number of constraint violations over all possible forms. A more feasible approximation is to cap the number of possible forms to those of length n and under, where n is the length of the longest form in the observed data (see [4] for similar approximation to the global partition function).

3.1 Exploring the Constraint Space: Formalization & Bounds

Even with these restricted bounds, the constraint space is large, forcing the calculation $O[C_i] - E[C_i]$ over all constraints $i \in |C|$ to be a very burdensome measure both in terms of complexity and space. Examining the size of these spaces explicitly, our task begins with a sound inventory of 24 distinct phones. Each phone marks a subset of 15 unique binary (+/-) articulatory features (often referred to as distinctive features, in this work these features are *consonantal*, *approximate*, *sonorant*, *continuant*, *nasal*, *voice*, *spread*, *labial*, *coronal*, *anterior*, *strident*, *lateral*, *dorsal*, *high*, and *back*). A constraint formed over a single phone (we will sometime refer to this as a *slice*) will consist of subsets of these values. In practice a maximum subset size of 3 is capable of capturing many phonological phenomena.

Assuming a constraint marks all distinctive features the possible combinations would already be quite large, but on average each phone in the data marks about seven distinctive features (as some features are considered to be not applicable to some phones, and are not marked). We compute the size of all possible single-slice constraints as: $\binom{7}{3} + \binom{7}{2} + 7 = 35 + 21 + 7 = 63$

We will complicate this further by adding some additional complexity to the feature representation. The above calculation treats all subsets as conjunctions of distinctive features, but it is very natural to use negation to express phonological constraints (represented here as the ˆsymbol). There are subtle differences in when these constraints will fire[3]. This doubles the number of single-silce constraints, and we will also add one additional option to express wildcard (which will be denoted by an underscore). This is useful for expressing the semantics like "anything followed by a sibilant". Our final number of single-slice constraint possibilities is 127.

But single-slice constraints would have little explanatory power: it's unlikely that many, if any, grammatical phones would be unobserved. Only when we move to clusters of phones is there an appeal to latent constraint-based grammars over simple unigram counts of observed statistics. Constraints composed of two to three slices are typical and powerful enough to explain many phenomena in the English data. With a max slice size of three, the final number of possible constraints becomes $127 + (127)^2 + (127)^3 = 127 + 16,129 + 2,048,383 = 2,064,639$ constraints.

It is unclear what spaces are reasonably searchable by the human learner, and what methods can be considered cognitively plausible. However, we will attempt to claim that the method of constraint induction we propose has merits as a more cognitively plausible model given the drastic reduction in size of the spaces the learner makes decisions from, and the natural way it extends both online learning methods and traditional harmonic grammar (part of this approach can be viewed as providing an implementation of the GEN function and exploiting the relationships among the competing candidates it produces).

[3] Note that this differs from simply negating the signs of the distinctive features

4 Error-Driven Constraint Induction

4.1 Constructing Neighborhoods: Minimally-distinguishable Pairs

The notion of minimal pairs is well-established in phonology, where they are defined as a pair of words which differ in only one phonological aspect but possess distinct meanings. Because these words are distinguishable by speakers of the language, the phonological aspect that differs between them is identified as being salient, and the two sounds treated as separate phonemes.

We define the notion of a *contrastive pair* in a somewhat analogous manner. We define the set of observed/grammatical forms O[4], and the set of unobserved/ungrammatical forms U. A contrastive pair comprises an observed form $o \in O$ and an unobserved form $u \in U$, such that u is the unobserved form most similar to o, e.g. when $o = N$, and $U = \{NG, ZH\}$, a contrastive pair is formed when $u = NG$ since it shares more distinctive features than with ZH. When we know two sounds are represented as different phonemes, there must be some distinctive feature that distinguishes them. Similarly when we have a contrastive pair there must be some distinctive feature not shared between the two forms that can form the basis of a constraint, allowing the model to distinguish between them.

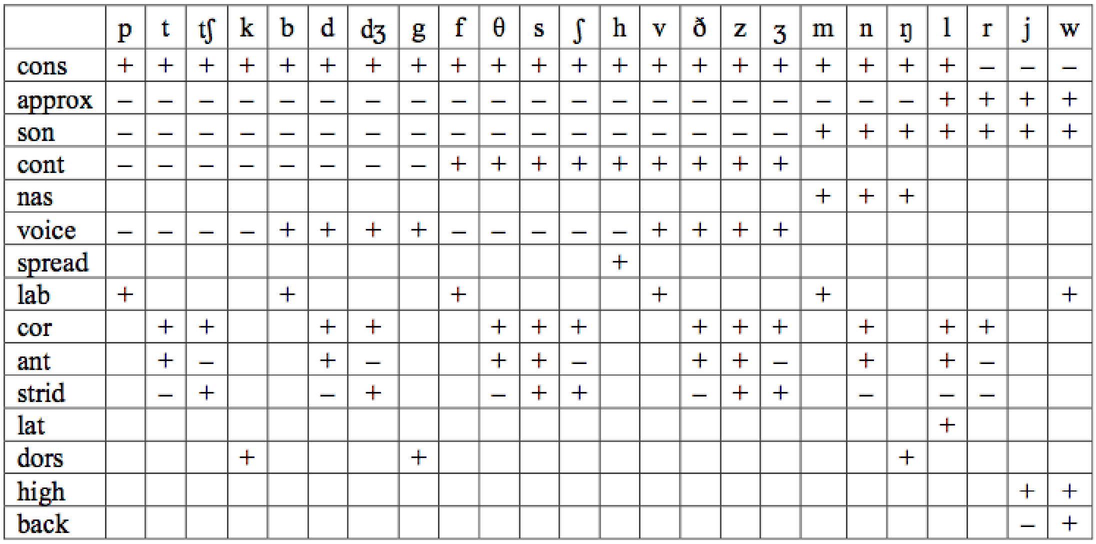
Central to our hypothesis is that phonotactic competition exists in neighborhoods, and not globally across all possible forms, and that these neighborhoods are determined by phonetic similarity. Thus N is in competition with NG, and its relation to other unobserved forms, like ZH is not of direct concern: the constraint most appropriate for establishing ZH as ungrammatical should come from an observed form with which it can form a contrastive pair, like Z.

4.1.1 Phonetic Distance

Formulating a measure of phonetic similarity strictly from sets of distinctive features is nontrivial. In the naive approach, we could simply count the number of binary "flips' required to turn the distinctive features of one phone to equal the other [5]. We denote distinctive feature j of the phone at position i in the word w by $p_{i,j}(w)$:

[4] In this paper all problem setups will assume that the learner will, if given enough examples, observe all grammatical forms, i.e., there is no held-out data. We will find that this can lead to troublesome behavior when as the model finds increasingly more specific analyses of the data, a problem known as *over-fitting*.

[5] The Hamming distance between the binary feature vectors.



	p	t	tʃ	k	b	d	dʒ	g	f	θ	s	ʃ	h	v	ð	z	ʒ	m	n	ŋ	l	r	j	w
cons	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	–	–	–
approx	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	+	+	+	+
son	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	+	+	+	+	+	+	+
cont	–	–	–	–	–	–	–	–	+	+	+	+	+	+	+	+	+							
nas																		+	+	+				
voice	–	–	–	–	+	+	+	+	–	–	–	–	–	+	+	+	+							
spread													+											
lab	+				+				+					+				+						+
cor		+	+			+	+			+	+	+			+	+	+		+		+	+		
ant		+	–			+	–			+	+	–			+	+	–		+		+	–		
strid		–	+			–	+			–	+	+			–	+	+		–		–	–		
lat																					+			
dors				+				+												+				
high																							+	+
back																							–	+

Figure 1: Feature set for English consonants, reproduced from Hayes & Wilson 2008

$$\text{phon-distance}(w_1, w_2) = \sum_{f}^{F} i \in |w_1| \begin{cases} 1, & \text{if } p_{i,f}(w_1) \neq p_{i,f}(w_2) \\ 0, & \text{if } p_{i,f}(w_1) = p_{i,f}(w_2) \end{cases}$$

This metric will turn out to be poorly suited to matching speaker intuitions on perceived similarity, but this is in part due to the distinctive features' "optional" binary representation. For example, sounds that are coronal (made with the blade of the tongue) may optionally be anterior (the point of articulation lying forward of the alveolar ridge). Compared to a sound that is not coronal, the sound will be doubly penalized since the the compared sound is not even in the proper category to mark this.

A second objection to the binary feature representation is that it does not easily allow for expressing real-valued relationships amongst distinctive features. For instance, place of articulation features are drawn from a continuum starting from the front of the mouth, and ending in the throat at the glottis (Fig. 4.1.1. While the spatial relationship between discrete places may not be useful to the model, removing that information with a binary feature representation deprives the model of learning this.

To allow us to compare to previous research the input of the learning algorithm must remain constant, but we propose a modified phonetic distance metric that is underlyingly more inspired by a multivalued and tiered representation of phonology

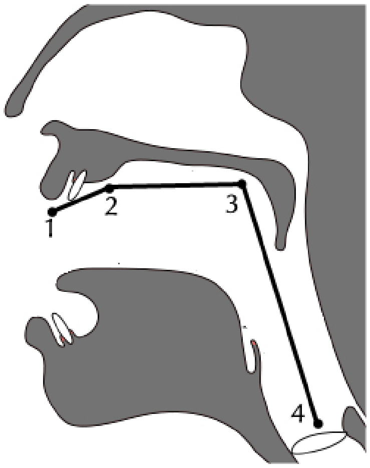


Figure 2: A graphical depiction showing the inherently continual places of articulation. Here we highlight four particularly prominent places: 1. Labial, 2. Alveolar, 3. Velar, 4. Glottal. Finer-grained distinctions are often posited to provide an account for a linguistic phenomenon, so while the mapping from this continual, analog physical space to discrete features can be arbitrary, the order remains faithful. This illustrates our motivation for real-valued features: it captures the intuition that, all other features remaining constant, a velar sound (3) will be more dissimilar to a labial sound (1), than an alveolar (2).

(see univalent or privative feature representations). Under this metric subcategories like anteriority are weaker contributors to the distance than their containing category, and are only included if both phones are within categories that mark it [6]

This means for sounds that differ further in place, labial vs dorsal, the distance will be further than place values that are closer, like labial vs. coronal. Subcategories like anterior are weaker contributors to the calculated distance. Fig 4.2 shows the nearest five neighbors for each phone as used in our experiments[7].

4.2 Projecting Constraints from Contrastive Pairs

Back to the topic of neighborhood construction, consider the observed form N. Using the distance function we can generate a distribution over forms in the neighborhood by exponentiating the inverse of distances and normalizing. Forms which are not in close phonetic proximity becomes exponentially less likely to be incorporated into a target form's neighborhood. We then sample a single form from this distribution and treat it as a proper contrastive pair. In our example, N will have the neighborhood distribution consisting primarily of M, NG, TH, T, and S. Naturally many of the forms in the neighborhood will be positive examples, and constructing constraints to discern between two positive examples has proven to be problematic, in our own experience, to learning suitable feature weights using perceptron updates.

We address this concern by preceding learning with a number of burn-in iterations, in which we draw from the English input data distribution (simulating the act of observing a form) and add it to a history queue. As we perform learning / constraint induction iterations, each observed form is added to the queue, and all forms in the queue are excluded from being in a contrastive pair. In our experiments we use a large number (3,000) of burn-in iterations to guarantee that the learner has likely been exposed to all common forms in the input data.

And because we construct neighborhoods of forms with the same length as the observed form, this would narrow the neighborhood down to the unobserved, single-phone forms: NG, and ZH, where NG, having the shorter distance, would be much more likely to be sampled.

We phrase constraint induction as an error-driven process (Algorithm 1 describes the procedure). When the contrastive pair (o,u) is scored using the model's current

[6] For reproducibility, disagreement between manner and approximates contributes 3 to the distance, place and consonant 2, and anteriority contributing 1 if both phones are coronal.

[7] These neighborhoods are generally in line with the authors' intuition as speakers of English, but they have not been objectively compared with studies of human perception measuring the confusability of these forms.

Phone	Neighborhood	Phone	Neighborhood
K	G (1), T (2), P (2), NG (3), D (3)	R	L (1), Y (1), W (1), T (6), K (7)
D	T (1), G (2), B (2), K (3), DH (3)	S	TH (0), SH (1), DH (1), Z (1), F (2)
M	N (2), NG (2), F (3), P (3), B (4)	P	B (1), K (2), T (2), M (3), G (3)
B	P (1), D (2), G (2), K (3), T (3)	S T	TH T (0), SH T (1), Z T (1), TH T (1), DH T (1)
HH W	HH Y (1), HH R (1), HH L (1), HH L (2), HH Y (2)	L	R (1), Y (1), W (1), TH (6), T (6)
F	V (1), SH (2), S (2), TH (2), M (3)	S W	S L (1), TH W (1), TH L (1), TH W (1), TH R (1)
HH	SH (3), TH (3), DH (3), F (3), ZH (3)	JH	CH (1), ZH (3), D (4), DH (4), SH (4)
T	D (1), P (2), K (2), S (3), G (3)	S P	TH P (0), DH P (1), TH P (1), TH B (1), Z P (1)
W	L (1), R (1), Y (1), P (5), B (6)	S M	TH M (0), DH M (1), TH M (1), Z M (1), SH M (1)
N	M (2), NG (2), TH (3), T (3), S (3)	T W	T Y (1), T R (1), T L (1), D W (2), D L (2)
V	F (1), DH (2), ZH (2), Z (2), HH (3)	S L	S R (1), TH W (1), TH L (1), TH L (1), S Y (1)
G	K (1), D (2), B (2), T (3), P (3)	S K	TH K (0), TH G (1), Z K (1), S G (1), SH K (1)
T R	T W (1), T Y (1), T L (1), T Y (2), D W (2)	B R	B W (1), B Y (1), B L (1), B Y (2), P R (2)
K R	K W (1), K L (1), K Y (1), K Y (2), G Y (2)	G R	G Y (1), G L (1), G W (1), G Y (2), K R (2)
SH	TH (1), ZH (1), S (1), DH (2), F (2)	TH R	TH W (1), S R (1), S Y (1), TH L (1), S L (1)
CH	JH (1), SH (3), S (4), N (4), TH (4)	S N	TH N (0), DH N (1), TH N (1), Z N (1), SH N (1)
K L	K W (1), K R (1), K Y (1), K Y (2), K R (2)	F L	F Y (1), F R (1), F W (1), V L (2), F Y (2)
Y	L (1), R (1), W (1), K (7), T (7)	G W	G L (1), G Y (1), G R (1), K W (2), K W (2)
F R	F Y (1), F L (1), F W (1), V R (2), F Y (2)	P R	P Y (1), P L (1), P W (1), P Y (2), B L (2)
B L	B W (1), B Y (1), B R (1), B Y (2), P R (2)	P L	P R (1), P Y (1), P W (1), P Y (2), B L (2)
D R	D W (1), D L (1), D Y (1), D Y (2), T R (2)	K W	K Y (1), K R (1), K L (1), G Y (2), K L (2)
TH W	TH R (1), TH Y (1), S R (1), S Y (1), TH L (1)	TH	S (0), SH (1), DH (1), Z (1), F (2)
G L	G Y (1), G R (1), G W (1), G Y (2), K L (2)	Z	DH (0), S (1), ZH (1), TH (1), SH (2)
D W	D R (1), D Y (1), D L (1), T W (2), D L (2)	DH	Z (0), S (1), ZH (1), TH (1), SH (2)

Figure 3: Observed phones and their nearest five neighbors, calculated by the phonetic distance described in Section 4.1.1. In practice these neighborhoods are greatly filtered down by removing observed words, and single phone "clusters" will only have a couple neighbors, mostly shared across the entire set of single phones. Future work may look to not remove these forms outright, but to down weight them based on how recently they had been seen.

feature weights, if the unobserved form ranks higher than the preferred observed form, an update is triggered. The difference between the distinctive feature sets of the two forms is identified and projected out as the basis for new constraints. For the pair (o=N,u=NG), this difference is that NG marks [+dors], therefore any constraint that fires on NG and not on N, precisely the type of constraint the model could use to downweight NG with respect to N, must contain [+dors].

This observation, in conjunction with the reasonable settings for the parameters governing maximum distinctive features-per-constraint slice and maximum slices per constraint, drastically restricts the space of possible constraints to a more feasible size. In the case of $(o = N, u = NG)$, there is a first implicit induction from the space of all single-slice constraints (a number we computed in Section 3.1 as 127, though we will concentrate first only on positive, non-universal constraint slices, starting from a space of size 63), down to only those relative to the contrastive pair: $\binom{5}{3} + \binom{5}{2} + 5 = 25$ given that NG marks 5 distinctive features. Employing the heuristic, we consider only the constraints which contain [+dors], reducing the space to $\binom{5}{2}$(ways of constructing 3-feature constraints) +5 (ways of constructing 2-feature constraints) = 15. There are similar reductions when considering the negated constraints, except that we project off the observed form. Thus we have quartered the local constraint space down to a size that is considerably more manageable. As we move into constructing constraints composed of multiple slices these reductions become far more valuable, as there is exponential growth over the size of the single-slice constraints.

In order to construct these larger constraints we construct a lattice where each array in the lattice is the array of possible single-slice constraints (each cell is one of all possible subsets of binary feature values), and the length of the lattice is the desired length of the constraint. Enumerating all paths through the lattice yields all possible constraints of this size.[8]

Once the set of all possible constraints that meet the heuristic requirements has been constructed, a small number of constraints are sampled, typically uniformly, from this pool and added to the model's constraint set. The augmentation of the model's constraint set is immediately followed by a perceptron update.

[8] This is mostly an implementational detail, and may raise some criticism as to our claim that this model is less computationally burdensome than Hayes & Wilson when we enumerate a large number of constraints in each local decision. Because the actual set of constraints induced are sampled from the lattice-produced collection of constraints, this process could be alternatively implemented using a weighted finite-state machine as a more lightweight model of the selection process.

Algorithm 1 Learning Procedure

```
function LEARN
    model ← []
    while learning do
        $o \leftarrow draw(D)$
        $u \leftarrow draw(neighborhood(o)$
        if model.$score(u)$ > model.$score(o)$ then
            $diff \leftarrow u.features \setminus o.features$        ▷ Constraint Induction
            $cspace \leftarrow []$
            for $c \leftarrow allconstraints$ do        ▷ Filter Space
                if c.contains(diff) then
                    $cspace+ = c$
                end if
                model.$features+ = draw(space)$        ▷ Add constraint
            end for
            for $j \leftarrow 0$ until model.$size$ do        ▷ Perceptron Update
                model[$j$] = $f.violations(u) - f.violations(o) * rate$
            end for
        end if
    end while
end function

function NEIGHBORHOOD($o$)
    $plattice \leftarrow [o.size]$
    for $i \leftarrow o.size$ do
        $plattice[i] \leftarrow o[i].phones$        ▷ (Phone, Distance) pairs
    end for
    return $draw(plattice.allPaths)$        ▷ Sum weight in each path
end function
```

5 Experiments

Evaluating the contributions of this paper poses a difficult problem, as a clear evaluation should decouple the learning mechanism from the constraint induction procedure – two aspects of the learner that are inherently linked in our work due to the error-driven nature of the constraint induction process. In one set of evaluations we will focus purely on constraint induction, examining a set of constraints induced by the model, and judge the induction process indirectly by its ability to reduce errors and improve likelihood. In a second set of evaluations we assess the use of the joint induction and training on the task of gradient phonological decisions against a collection of human judgements.

5.1 Constraint Induction

With no gold standard constraints to compare to, we have to focus our attention toward evaluating the "symptoms" of good constraint evaluation. The mechanism is error-driven, so if both the constraint induction and weight updates are functioning properly the frequencies of triggering constraint induction and weight update procedures should decrease. Additionally, as we update the feature weights our intention is to move probability mass away from the ungrammatical forms - pulling more from the least likely and less from the somewhat more acceptable. A useful measure of this is log likelihood, the sum of the log probabilities of each observed form. If the model is improving the probability of the observed forms should increase. In Fig. 5.1 we plot this information over 500 iterations of learning.

Of course, this reduction of errors occurs between contrastive pairs, and so there is no guarantee that optimizing this function is equivalent to inducing constraints that are relevant from a phonologist's perspective. We list the twenty constraints with the highest weights after 200 iterations of training in Fig. 5.1. [9]

5.2 Gradient Predictions

To lend additional experimental support to the constraint evaluations, which have no objective, quantitative point of comparison, we also examine the grammar's ability to reproduce the gradience of human phonotactic judgements. Our gold standard data is gleaned from an early experiment by Scholes (1966) in which the grammaticality of 66 non-words were scored on a yes/no basis by native English speakers. The percentage of participants that responded yes is taken as a measure of gradient grammaticality,

[9] It's hard to assess this section myself

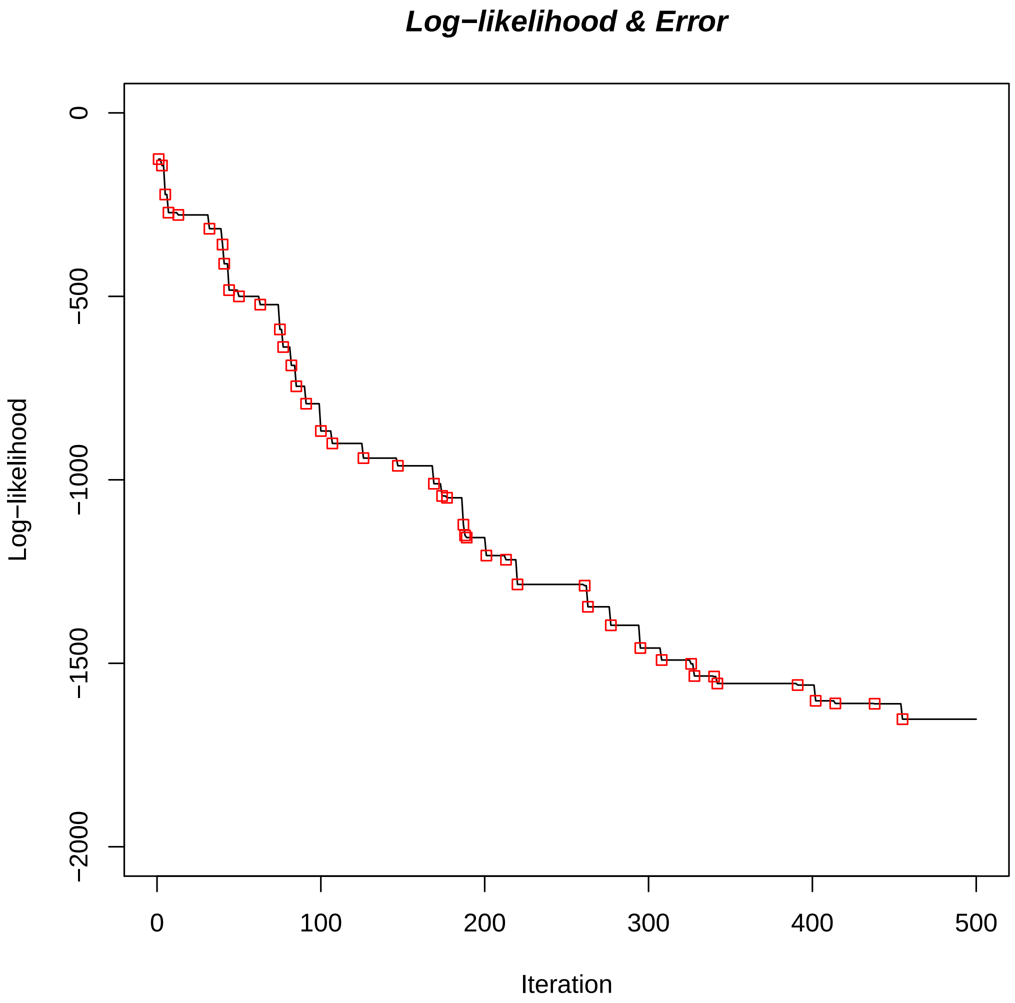


Figure 4: Plot of log-likelihood and erroneous predictions. Lower log-likelihood is better. Naturally the model makes many errors in its initial predictions, but towards the end of training there are large contiguous stretches where the model make the correct prediction. While this model posits new constraints for each error, augmented models may explore the trade-off between weight updates and constraint induction with alternate update strategies.

Weight	**Feature**	**Description**
1.34	[_] [ˆ-strid,+cor]	S T vs. TH S
1.28	[ˆ+cons,+ant]	S vs ZH
1.26	[ˆ-approx,+lab]	M vs. NG
1.26	[_] [ˆ-cons,+cor]	G R vs. P W
1.26	[-approx,-voice] [ˆ-ant,+cor]	G R vs. CH L
1.22	[ˆ-son,-voice] [_]	S K vs. Z P
1.22	[ˆ+strid][-strid,+son]	S L vs TH L
1.20	[ˆ-son,+voice][+lab]	G R vs. P W
1.20	[ˆ+dorsal,+cons][ˆ+approx,-ant]	G R vs. P W
1.20	[ˆ+dorsal][ˆ-strid,-ant]	G R vs. CH L
1.18	[ˆ+cont,-voice]	S vs. ZH
1.18	[+strid,+cor][ˆ-strid]	S T vs. S S
1.18	[-son,-voice][_]	G R vs. P W
1.18	[_][ˆ+approx,+cor]	G R vs. P Y
1.16	[-son,+ant][-ant]	S W vs. S R
1.16	[_][ˆ+high,+approx]	S W vs. S R
1.16	[_][ˆ+high,+son]	S W vs. S R
1.16	[ˆ-voice][_]	T R vs. V R
1.14	[ˆ+strid,-voice][_]	S L vs. TH L
1.14	[-son][ˆ+ant,+lat]	S L vs. S R

Figure 5: Top 20 constraints from a run achieving an .804 Spearman correlation with human judgements. Parameters for the run restricted constraints to a max window of 2, max slice size of 2, 500 iterations with perceptron training, with a constraint induction step at each iteration (if the observed form is not the winning candidate in the neighborhood) and sampling three constraints. The sample size accounts for a few constraints targeting the same contrastive pair.

with the results generally accepted to be consistent with other gradient phonotactic measures (Frisch et al. 2000?).

We compare against the scores of Hayes & Wilson [10], who also used Scholes scores to assess their model, using Spearman nonparametric correlation. This measure assesses the relationship between two variables only monotonic functions, i.e., it is concerned with the rank of the variable values, not their particular value. Fig (5.2) shows the results.

Our model exhibits a general logarithmic trend in improvement as it bootstraps, on average, the most useful and discerning features into the model before beginning to dredge the remainder of the space and the less probably tails of neighborhood distributions. Our model rarely, if ever, exceeds the Scholes correlation of the published Hayes and Wilson result, but it does come quite close (-.829 vs. -.859), sometimes reaching nearly identical scores at rare points during learning.

Though we cannot claim to have improved upon the correlation performance of Hayes & Wilson, the intention of this work was not to improve upon global decisions with only local information, but to provide comparable enough performance to be treated as an alternative theory. Exactly where this performance threshold lies is hard to determine, but we feel that the performance of this model indicates that in the right circumstances the model can come within just a few points of the performance of the global strategy. Further research exploring exactly what those circumstances are, and modifications to steer the model toward them, may lend further credence to the model through consistently higher performance.

6 Smarter, Less Aggressive Induction

In the previous section we presented a series of experiments illustrating the performance of the induction procedure, and while comparison to human judgements generally reached state-of-the-art performance at some point during training, the performance of the final models had often degraded by a significant margin. There are many potential factors that may have contributed to this behavior, in this section we turn to the constraint induction criteria.

In previous experiments each example that was misclassified causing a feature update was then subject to reconstruction of the neighborhood and reassessment by the model. If the preferred example was still not deemed optimal by the model, a constraint induction step was performed each time. This is one of the more ag-

[10] In the run distributed with the software, for which we find a $\rho = -0.859$.

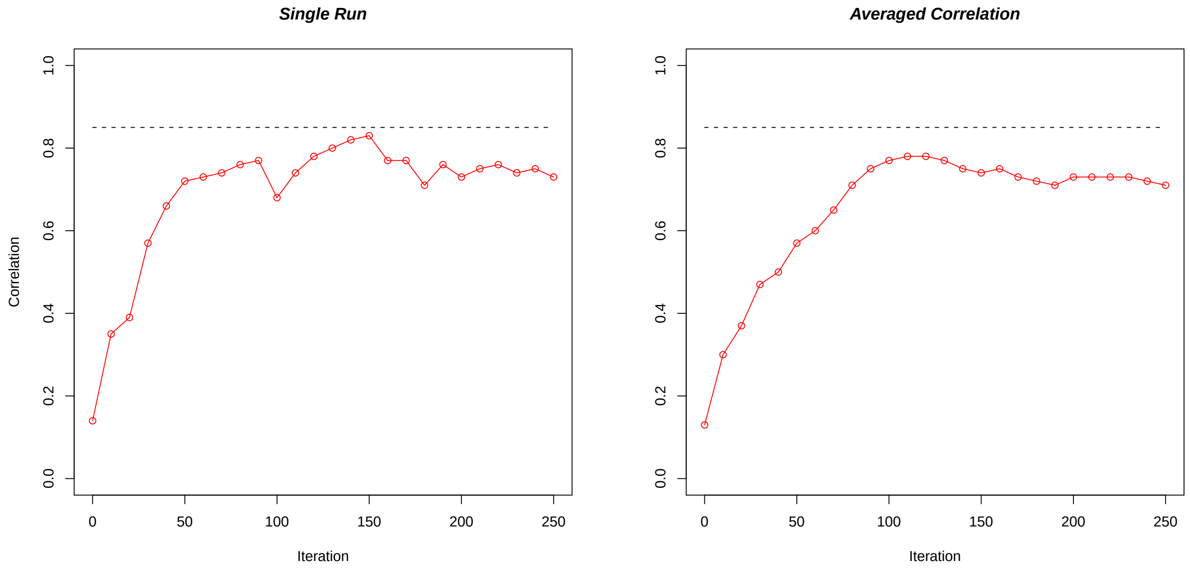


Figure 6: Performance of the learner in comparison to human judgements. Plot is a measure as Spearman correlation against Scholes values, the horizontal line represents the performance of the learner presented in H&W. Performance in single runs (left) occasionally becomes quite competitive with state-of-the-art, despite using only local information. In the averaged run (right), combining 10 separate runs, the curve illustrates a consistent behavior of over-fitting and eventual performance degradation.

gressive possible scenarios[11]. Alternatively we could delay induction by an arbitrary factor, inducing one in every three times we would have previously added constraints. Instead we turn to what we feel might be a somewhat more principled model of induction.

Classification errors fall mainly into three categories: (1) the feature weights are inappropriate. Ultimately this is the cause of all errors in a correctly setup problem, but errors may also occur due to (2) the absence of discerning constraints. A final potential cause of errors in our domain, where neighborhoods are constructed probabilistically, is (3) that the neighborhoods may be constructed improperly. Filtering out observed forms goes a long way to prevent this, but forming a contrastive pair between very dissimilar forms can certainly introduce unusual constraints into the model, which can in turn contribute to the model over-fitting the data in unfavorable ways.

In this section we turn out attention primarily toward errors of the third type, as continued constraint induction explores forms that lie further toward the tails of the distributions that are being sampled from. These forms are inherently less likely, and yet simply stopping at a given earlier iteration only sidesteps the problem. We want the model to come closer to converging at its best configuration.

To that end we introduce the notion of a model history, with the intention of establishing a measure of feature weight convergence. When an error occurs and the feature weights relevant to the example (present in the count difference vector) have not been updated in the history, we will bias the model slightly toward updating. If these weights have seen frequent updates, we assume that perhaps they are not discerning properly on the basis of not having the correct constraint in the model.

Keeping second-order statistics over more fine grained events, like partitioning histories into sets for each example, could make this approach much more effective, but perhaps at the cost of cognitive plausibility. In this initial presentation we refrain from seeking greater performance at the cost of backpedaling from our presentation of this as a generally local, computationally less burdensome approach to constraint induction. A simple, but still quite memory-consuming technique would be to simply keep counts of how frequently errors are made on a particular observed form, or more specifically on a particular contrastive pair, and make a constraint induction step more likely if the model continues to err on the example using only feature updates.

[11] However, constraints are not sampled without replacement, so we rely on the redundancy of re-sampling favorable constraints to mitigate the instability of constantly introducing new constraints.

7 Incorporating Linguistic Bias

In our previous exposition we have refrained from included much a priori knowledge of what type of constraints should be induced, leaving the model to draw constraints solely on the basis of what it deems most useful. Being noncommittal in this regard is mathematically realized by setting uniform priors over the constraint distribution that induced constraints are sampled from. However, we can just as easily apply a bias by forcing the constraint distribution to fit standard statistical distributions that favor certain characteristics in an ideal constraint.

Applying some a priori knowledge to the learning procedure is not uncommon: in fact our standard base of comparison throughout this paper, H&W, include a preference for constraints of a particular type in their search heuristics. Separate from constraint accuracy, the model presented in H&W will prefer constraints that are smaller, and that have more general features.

As a first exploration into steering the model toward more linguistically-acceptable solutions, we choose to also apply a bias based on constraint complexity. Our lattice approach to constructing the distribution over new constraints provides a simple method for incorporating this bias, as we can simply keep a running tally of the size of the constraints as the distribution is constructed, yielding a constraint and its sampling weight. We begin by centering the distribution over constraints of size 2, where size denotes the number of binary feature values, and applying a geometric penalty as the size of the constraint is further removed from the optimal.

7.1 Further Constraint Refinement with Histories

We can also proxy some of the accuracy and generality preferences of H&W with an additional strategy, though it comes at the cost of keeping a memory of the observed forms the model has been exposed to. While we do keep a memory of observed forms to filter out of neighborhoods, the intent of that history is to rule out trivially common forms from being considered as negative evidence. As a cognitive model this additional history is much more in spirit to short term memory than to common knowledge.

Less universal constraints can then be ruled out by ensuring that they are not violated by the previously seen observed forms in this history. This strategy, though effective, may immediately raise some concerns from phonologists aware that this would indeed rule out many of the useful constraints posited by linguists. There are two points of rebuttal. While we implement this as a hard constraint, there is no doubt that this intuition would be properly phrased by a statistical model more sensitive to precisely what forms are being considered, and what constraints have been posited.

Secondly, this does not necessarily rule out these constraints from ever being considered, but only from being considered in a particular context. Reasonably short histories (we find a history of size 5 is often adequate on this small data set) both refine constraint induction decisions while simultaneously allowing for more variation in contexts, and more opportunities for truly useful constraints that would otherwise be excluded to find themselves in the constraint set.

We conclude with a return to our experiments matching human gradient phonotactic judgements (Fig. 7.1).

Together these augmentations to the model provide a near universal improvement. There are gains in both the maximum correlation, the correlation of the final model on both single and averaged runs, and the constraints induced show signs of migration toward more linguistically acceptable analyses. We present these results separately because they do raise concerns, especially given the training and testing data are one in the same, of probing the data and tailoring the model to this particular problem set. However, we feel that each augmentation has been grounded in a reasonably plausible motivation. We defer to future work for more principled implementations of these methods, and a more piece-wise analysis of each component's contribution to the final performance gains.

8 Conclusions & Future Work

In this paper we present a novel, error-driven technique for constraint induction for log-linear models of phonological grammar. This error driven approach reduces the decision spaces relative to global or batch induction procedures, and the marriage of an error-driven induction procedure with error-driven weight updates provides a unified framework for how phonological grammars can be bootstrapped from little prior information. We show that this approach succeeds in reducing classification errors, and provides gradient phonotactic judgements that are competitive with state-of-the-art global models at replicating human performance.

One unsolved problem is the nuisance of the many free parameters governing the constraint induction process, while having little data to provide a principled way of tuning them. Partitioning the observed set into a group of held-out clusters, with Scholes scores, to test against is most analogous to how this might be done in the NLP/ML community, but it may not be reflective of the environment of the human learning where a small set of observed forms are seen many, many times. It is very unlikely that any observable cluster goes unheard by an infant throughout the prime

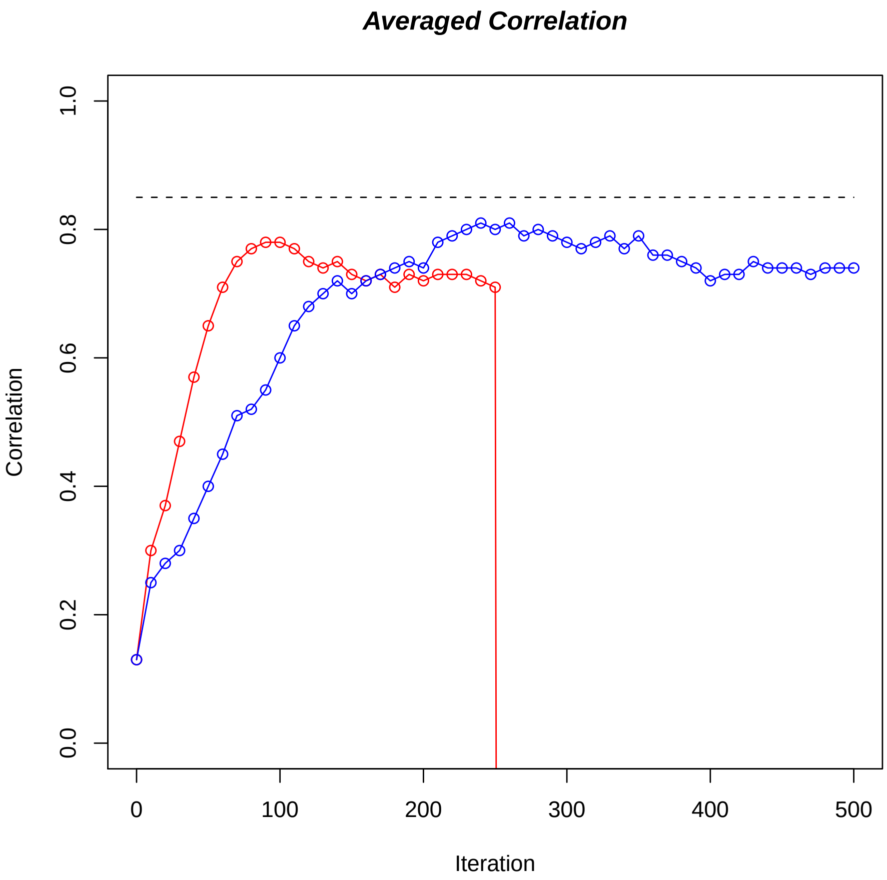


Figure 7: Revised gradient phonotactic judgement results. Having included the more relaxed constraint induction protocol, priors over the constraint sampling distribution, and a history-based filtering heuristic, the performance curve illustrates slower learning, but improved performance. Training of the original model ends at iteration 250, but we allow the revised model to continue given the less frequent constraint induction.

years of language acquisition, and the relatively small set of clusters in onsets means that each example that is purposely withheld carries with it greater value than in the NLP tasks where this strategy is most employed, and where counts of some events are expected to be sparse.

Rephrasing this as a Bayesian model conditioned on the current features and weights, observed and contrastive pair, and history of seen examples may allow for a more concise model representation – with less of a heuristic focus – and allow for easier integration of rich priors over these aspects of the model. Rather than passing on the question of external tinkering from "where does the constraint set come from", to "how do these parameters governing constraint induction get set?", and on yet again to "how do these hyperparameters get set?", the hyperparameters may be explainable through more principled means from the data.

These initial explorations are promising and a solid proof of concept for online, purely-local constraint induction. We belief that addressing these additional concerns, incorporating prior beliefs, and additional history-based filtering will produce a more powerful model that induces more linguistically ideal constraints. Though because these augmentations may raise criticism as being less cognitively plausible, we present this model as a barebones approach to contrastive, online constraint induction.

References

[1] Paul Boersma and Joe Pater. Convergence properties of a gradual learning algorithm for harmonic grammar. 2008.

[2] Hai Leong Chieu and Hwee Tou Ng. Named entity recognition: A maximum entropy approach using global information. In *In Proceedings of COLING02*, pages 190–196, 2002.

[3] Sharon Goldwater and Mark Johnson. Learning ot constraint rankings using a maximum entropy model. In *In Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, pages 111–123, 2003.

[4] Aria Haghighi and Dan Klein. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA, June 2006. Association for Computational Linguistics.

[5] Gerhard Jger. Maximum entropy models and stochastic optimality theory. 2007.

[6] Andrew McCallum. Efficiently inducing features of conditional random fields. In *Conference on Uncertainty in AI (UAI)*, 2003.

[7] Jason Naradowsky, Joe Pater, David A. Smith, and Robert Staubs. Learning hidden metrical structure with a log-linear model of grammar. In *Workshop on Computational Modelling of Sound Pattern Acquisition*, 2010.

[8] Joe Pater. Weighted constraints in generative linguistics. 33:999–1035.

[9] S. Della Pietra, V.J. Della Pietra, and John D. Lafferty. Inducing features of random fields. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997.

[10] Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP)*, 1996.

[11] Brandon C. Roy, Michael C. Frank, and Deb Roy. Exploring word learning in a high-density longitudinal corpus. In *Proceedings of the 31st Annual Meeting of the Cognitive Science Society*, 2009.

[12] Paul Smolensky and Graldine Legendre. 2006.

[13] Ben Taskar, Dan Klein, Michael Collins, Daphne Koller, and Christopher Manning. Max-margin parsing. In *In Proceedings of EMNLP*, 2004.

[14] Colin Wilson and Bruce Hayes. A maximum entropy model of phonotactics and phonotactic learning. 2008.